The University of Edinburgh

Department of Computer Science

A Framework for Binding Operators

by

Yong Sun

CST-91-92

(also published as ECS-LFCS-92-207)

April, 1992

James Clerk Maxwell Building,
The King's Buildings,
Mayfield Road,
Edinburgh,
EH9 3JZ.

A Framework for Binding Operators

Yong SUN

Doctor of Philosophy

University of Edinburgh

1991

(Graduated in absentia February 1992)

Chinese Proverbs

- *There is no such a thing as a road, just many people walk through and make it.*
- *A thousand mile's journey has to start from the very first step, and who knows it is actually a forward step or a backward one.*
- *Is there anybody who ever makes a thorough effort to evaluate whether others' work is a contribution or a detraction to our civilization, say which (work) is which (a contribution or a detraction) and which is whose?*
- *A barrel fully-filled with water won't make a sound but the under-filled barrel makes most noisy.*
- *Ships can either be carried away by water or be sunk by water.*

Brief Abstract

Binding appears in logic, programming and concurrency, e.g, it appears in Lambda Calculus, say the λ-abstraction. However, the binding in Lambda Calculus is *unary*. Certainly, we can generalize the idea of unary binding to an arbitrary finite numbers of binding. Algebraically, we can extend the framework of Universal Algebra [23, 28,56] and take arbitrary finite bindings as primitives. Therefore, operations in the new extended signature must be of second order instead of first order, and we name them as *Binding Operators*. The resulting framework is named as a *Framework for Binding Operators*, which coincides with Shapiro's diminished second order language [138].

With a modification of Aczel's Frege Structure [4], we derive the algebras for Binding Operators, i.e. *eBAs*. The usual first order algebras, Plotkin's $\mathcal{P}\omega$-model of Lambda Calculus [105,118,133,134], and Girard's qualitative domains [47] turn out to be special cases of eBAs. And also eBAs turn out to be (i) a generalization of Kechris and Moschovakis' *suitable class of functionals* in Recursion in Higher Types [83] and (ii) a generalized Volken's λ-family [7, p.127]. Following Birkhoff [16], we would like to equationally characterize Binding Operators. Kechris and Moschovakis' Enumeration Theorem [83] suggests that an algebraic characterization of such might be possible.

Unfortunately, eBAs and the usual satisfaction $\models_{eBA}$ of *Binding Equations* over these eBAs, in Birkhoff's approach, do not work. Therefore, we have to find either a *remedy* for it or a new semantic model for Binding Operators. We will present two solutions, one for each.

(a) For a remedy, we discover a condition for Birkhoff's approach to work. This condition is necessary and sufficient, and we call it an *admissible* condition, which

turns out weaker than Plotkin's *Logical Relations* [121] in the sense that "logical" implies "admissible". An admissible equational calculus $\dot{\vdash}_{eBA}$ for Binding Equations is obtained, which is sound and complete with respect to admissible satisfaction $\dot{\models}_{eBA}$. The relationship between Completeness and Admissible Completeness (or between satisfaction $\models_{eBA}$ and admissible satisfaction $\dot{\models}_{eBA}$) is discussed, although it is not completely clear. Other problems remain open as well, say the closedness of direct products and the admissible variety problem.

(b) For a new semantic model, we will give a new binding algebra, i.e. *iBA*, which is intensional in contrast to the previous (extensional) one. Actually, an iBA is a generalization of Friedman's Prestructures [46]. A sound and complete equational calculus $\vdash_{iBA}$ (in iBAs) is established. However, the derivability of $\vdash_{iBA}$ is weaker than the one of $\dot{\vdash}_{eBA}$. In other words, to share a same proof power with $\dot{\vdash}_{eBA}$, $\vdash_{iBA}$ has to use axiomatic schemas instead of pure axioms. Also, the relations between extensional satisfaction $\models_{eBA}$ and intensional satisfaction $\models_{iBA}$, and between admissible calculus $\dot{\vdash}_{eBA}$ and calculus $\vdash_{iBA}$ are discussed.

Examples of applications with present Framework for Binding Operators and $\vdash_{iBA}$ are given. They are equationalizations of (i) First Order Logic [140,139], (ii) Lambda Calculus and Combinatory Logic [32,33,7], and (iii) Milner's Calculus of Communicating Systems [108,109] with data-dependency. These demonstrates that the Framework for Binding Operators provides a unified algebraic framework for all of logic, computation and parallel computation.

Declaration

This thesis is composed by myself and the work presented is my own (except otherwise stated), which was initiated by my supervisor G. Plotkin and developed under his guidance.

signed by

SUN, Yong

List of Publication

1. *"equational characterizations of binding"*, European workshop on Typed Lambda Calculi (Jumelage Meeting), Edinburgh, September 1989 (see [150]).

2. *"on soundness and completeness of calculi for equations, dependent equations and quasi-dependent equations"*, the 5th British Colloquium of Theoretical Computer Science, London, April 1989 (a brief abstract in Bulletin of European Association of Theoretical Computer Science (EATCS) No.39, October 1989).

3. *"equations, dependent equations and quasi-dependent equations"*, the proceedings of Logic Colloquium, Association of Symbolic Logic (ASL) abstracts, Berlin '89, Germany, July 1989.

4. *"equational logics (Birkhoff's method revisited)"*, the proceedings of the 2nd international workshop of Conditional and Typed Rewriting Systems (CTRS'90), Montreal Canada, June 1990.

5. *"specification of protocols and its independence of communication mechanisms"*, Informatik-Fachberichte (IFB), vol.130, pp.751-762, Springer-Verlag, 1987.

added after completion *"Axiomatization of Calculus of Constructions"*, preliminary proceedings of the symposium on Constructivity in Computer Science, Trinity University, San Antonio, Texas, USA, June 1991.

Acknowledgement

I would like to thank Chinese Academy of Sciences for providing me a grant to study abroad, and to thank Chaochen Zhou recommended me to come over to the Department of Computer Science of University of Edinburgh to study with Gordon Plotkin.

During my Ph.D. program, I have applied Gordon's Domain Theory [123] to semantics of Very Large Scale Integration (VLSI) circuits, and obtained some results on this application. The results are recorded in "*Logical Design of VLSI Circuit with Extension of Uncertainty (or monotonic functional completeness of Kleene ternary logic [86])*" (see [132]). While I worked on this, Rod Burstall gave me his encouragement.

Then, I turned to work on concurrency and have some results on it. They are: (a) "*Specification of Protocols and its Independence of Communication Mechanisms*" [149]; (b) "*Self-independent Petri Nets (or a dead-lock-free paradigm)*" [151].

At last, I work on a research program of systematical treatment of binding and corresponding equational characterization, which is initiated by Gordon. The results of this research is presented in this thesis, and a preliminary version of this thesis was completed by September 1988. Main results were announced in Jumelage meeting (European workshop for Typed Lambda Calculi) in Edinburgh, September 1989 [150]. Subsequent revisions are done in December 1989 and December 1990, which improve the presentation substantially. This is largely due to the comments I have received from Gordon Plotkin.

While I was working on my Ph.D program, Gordon Plotkin had provided me many precious insight and guidance. I own many thanks to him. Between October

1987 and August 1988 (inclusive), the University of Edinburgh granted me on leave for one year to the Center for Study of Language and Information (CSLI) of Stanford University (USA) to continue my research. Chinese Academy of Sciences (CAS) has a special arrangement about my grant to support the on-leave (CAS document (87) Ke Jiao No.672). CSLI kindly provided me a research office, and the Laboratory for Foundation of Computer Science (LFCS), the University of Edinburgh had contributed a fund to cover my return travel expense between Edinburgh and San Francisco (USA). Many thanks to them. LFCS and the Department of Computer Science, the University of Edinburgh provides a good working environment for doing research and some extra fund to cover my living expense in fourth year of study.

Besides previously mentioned people and organizations, I would like to thank the following people: to thank my parents for unfailing mental supports; to thank Peter Aczel for beneficial talks with him during my visits to Manchester and for his kindly supply of his work on binding; to thank Bruno Buchberger for his kindly inviting me over to RISC-Linz of Johannes Kepler University in Austria, and providing me an opportunity to present part of my work in this thesis; to thank Mike Fourman and Barry Jay for beneficial discussions of categorical aspect of binding algebras; to thank David Walker for a beneficial discussion related to complete equationalization of weak bisimulation of CCS; to thank Kevin Mitchell for a wonderful debate on the asynchronous communication in [151] as an alternative to handshaking; to thank Claire Jones for proof-reading of some part of thesis; to thank Eugenio Moggi for comments on an early draft of this thesis and for providing some relevant references; to thank Randy Pollack for a stimulating talk related to logic; to thank Xue-Zhou CAO, Mads Dam and Charlotte, Andreas Knoble and Petra, Jordi Farres-Casals, Mads Tofte, Pawel Paczkowski and Agnieszka, Farron Moller and Alice, Jo Blishen and Russ for their friendship; to thank computing officers in the Department for providing good computing facilities on which this thesis is produced.

13th December 1990

I would like to thank Peter Aczel and Robin Milner very much for their constructive comments on a previous version of this thesis. Also, they gave me confidence on

the part which was not criticized. As a result, I have changed all deduction systems from inductive definitions to logical inference rule systems. This alternation makes the presentation much more better. Furthermore, Section 3.13 is added, and the first part of Chapter 11 is rewritten (i.e. Section 11.1 and Section 11.2).

3rd October 1991

Table of Contents

1. Introduction and Brief Overview 17

1.1 Framework for Binding Operators (FBO) 20

1.1.1 Binding primitives 20

1.1.2 Signature with binding primitives 22

1.1.3 Language with finitary binding primitives 23

1.1.4 Equational examples with BOs 26

1.2 Semantics for BOs and their characterizations 28

1.2.1 Birkhoff's approach and its remedy 29

1.2.2 Friedman's approach 37

1.2.3 Core of this thesis – calculi $\vdash_{eBA}$ and $\vdash_{iBA}$ 41

1.3 Review with relevant works 45

1.3.1 Equational theories and their feasibility 45

1.3.2 Combinatory Logic 46

1.3.3 Universal Algebra 47

1.3.4 Variable Binding Term Operators 48

1.3.5 Cylindric Algebras 49

1.3.6 Monadic Algebras and Polyadic Algebras 50

1.3.7 Logicals Relation vs Admissibility 51

1.3.8 Natural Languages 52
1.4 Future developments 53
1.4.1 Higher types vs higher orders 53
1.4.2 2-category, eBAs and iBAs 54
1.4.3 Lambda Calculus vs Linear Lambda Calculus 55
1.4.4 Generalized Full Abstraction and non-Simple Algebras . . . 56
1.4.5 Generalized Institutions 56
1.5 Summary of terminology and abbreviations 57

I Many-Sorted Algebras **60**

2. Many-Sorted Universal Algebra **62**

2.1 Equational Logic $\vdash_{EQ}$ with variable indices 65
2.2 Elimination of variable indices 83
2.3 Dependent equations 86
2.4 Quasi-dependent equations 93
2.5 Universal equations 99
2.6 Discussion on related works 103
2.6.1 Equational Logic $\vdash_{EQ}$ 104
2.6.2 Calculus $\vdash_{dEQ}$ for dependent equations 106
2.6.3 Calculus $\vdash_{qEQ}$ for quasi-dependent equations 106
2.6.4 Comments 109

II Birkhoff's Approach **110**

3. Remedy to Birkhoff's Approach **112**

3.0 Well-definedness of interpretations in eBAs 113

3.1 Sub-eBAs, perfect Sub-eBAs and eBHs 116

3.2 Sub-eBAs and generated sub-eBAs 119

3.3 eBHs and their uniqueness over generated eBAs 122

3.4 Constructing term eBA 125

3.5 Generatability of term eBA 129

3.6 Universal property – free eBA 134

3.7 Binding Congruences and quotient eBAs 136

3.8 Admissible freeness 145

3.9 Admissible Binding Equations 150

3.10 Admissible Completeness of $\dot{\vdash}_{eBA}$ 153

3.11 Admissible variety problem 159

3.12 Logical Relations vs Admissibility 162

3.13 Completeness and admissible completeness 163

4. Extended Remedy **167**

4.1 Admissible dBEs 167

4.2 Admissible qBEs 170

4.3 Admissible uBEs 172

III Friedman's Approach **175**

5. Friedman's Approach **177**

5.0 Relationship between eBAs and iBAs 177

5.1 Substitutions 181

5.2 Well-definedness of interpretations 188

5.3 b-clones 192

5.4 Preservation of equality 197

5.4.1 Relationship among $\mathcal{B}_{\mathcal{Z}}$, $tr_{\mathcal{Z}}$ and $\mathcal{D}$ 197

5.4.2 Relationship among $\mathcal{B}_{\mathcal{Z}}$, $\underline{tr}$ and $\mathcal{D}$ 198

5.4.3 Intensional equality under translations 201

5.5 Preservation of derivations 203

5.5.1 Calculus $\vdash_{iBA}$ and axioms $Ax_{\flat}$ of b-clones 204

5.5.2 Preservation of derivations under $\underline{tr}$ 207

5.5.3 Preservation of derivations under $tr_{\mathcal{Z}}$ 218

5.6 Soundness and completeness of $\vdash_{iBA}$ 229

5.6.1 Soundness of $\vdash_{iBA}$ for BEs 229

5.6.2 Completeness of $\vdash_{iBA}$ 230

6. Extended Friedman's Approach **234**

6.1 Calculus $\vdash^{d}_{iBA}$ for dBEs 236

6.2 Calculus $\vdash^{q}_{iBA}$ for qBEs 238

6.3 Calculus $\vdash^{u}_{iBA}$ for uBEs 242

IV Many-Sorted FBO **247**

7. Extensions to Many-Sorted FBO **249**

7.1 Many-sorted Friedman's approach 251

7.2 Many-sorted Birkhoff's approach 253

7.3 Higher-sorted FBO 253

V Applications 255

8. Equationalization of First Order Logic 257

8.1 Equational presentation of First Order Logic 257

8.2 Sequent natural deduction 260

8.3 Complete equationalization 262

8.4 Discussion 266

9. Equational Theories of λ-algebras and of λ-models 269

9.1 Equational theories of Lambda Calculus and of Combinatory Logic . 270

9.2 Well-definedness of C-λ and of λ-C 272

9.3 Equational theory for λ-algebras 275

9.4 Equational theory for λ-models 282

9.5 Discussion 285

9.5.1 Curry's A_β and λ-algebra's A_a 285

9.5.2 Function variables and definable elements 285

9.5.3 λ-family and extensional β-algebra 286

9.5.4 Intensional β-algebra and Cartesian Closed Category 288

10. Equational Theories of Finite CCS with Data-Dependency 293

10.1 CCS 294

10.2 Equational characterization of strong bisimulation $\sim_c$ 301

10.3 Equational characterization of weak bisimulation $\approx_c$ 314

10.4 Discussion of boolean languages 330

VI Comparision and future development 335

11. Birkhoff's approach vs Friedman's approach **337**

11.1 Relation between $\models_{eBA}$ and $\models_{iBA}$ 338

11.2 Relation between $\vdash_{iBA}$ and $\vdash_{eBA}$ 339

11.3 Possible improvement on Friedman approach 342

VII Bibliography and indices **346**

1 Bibliography 347

2 Formulae and symbols 363

3 Names and organizations 385

4 Abbreviations 387

5 Definitions 388

6 Subjects 393

List of Figures

1–1 commutativity 36

2–1 interpretation or unique extension of environments 68

3–1 commutativity 145

3–2 quotient eBA and double quotient eBA 148

3–3 admissible evironment and admissibly freeness 150

3–4 binding substitution 154

3–5 quotient eBA and sub-direct embedding 161

5–1 translation $tr_{\vec{x}}[\![\bullet]\!]$ and interpretations 198

5–2 translation $\underline{\mathrm{tr}}[\![\bullet]\!]$ and interpretations 199

List of Tables

1–1 summary of terminology 58

1–2 summary of notations and abbreviations 59

1–3 comparison among first order algebras, eBAs and iBAs 59

3–1 full function space of $\{a, b, c\} \to \{a, b, c\}$ 143

Chapter 1

Introduction and Brief Overview

From 1959, Kleene published his work on recursion in higher types [87], which was soon considered to be a deep and significant extension of the theory of recursive functions on natural numbers. More recent work in this area can be found in a paper by Kechris and Moschovakis (see [83]), in a book by Fenstad (see [44]), and in a paper by Kolaitis (see [92]). Theories of finite types as well as finite orders naturally occur in many mathematical practice as demonstrated by Feferman, by Beeson and by Troelstra and Dalen in [43,13,159]. The effective aspect of these theories, which is one of our main concerns of these theories from computer science point of view, lead us to consider of incorporating Recursion Theory of finite types with them. The wide range of these theories seems to suggest a more general algebraic framework, like Universal Algebra for first order theories. The equivalent representabilities of r.e. relations and of inductive definable relations support this very idea.

In 1962, Kleene also gave an equivalence between λ-definable functionals and his generalization of Recursion Theory in [88]. Later, Scott showed that $\mathcal{P}\omega$ model of Lambda Calculus, discovered by Plotkin (see [118,105]), can be regarded as a model for Recursion Theory, see [133]. Further, Aczel exploited the above phenomenon in his paper related to Frege Structure, and apply to First Order Logic and Set Theory, see [4]. In this thesis, we proceed along this line and take arbitrary finite bindings as primitives.

For example, the binding in Lambda Calculus is *unary*. Certainly, we can generalize the idea of the unary binding to arbitrary finite numbers of binding. Alge-

braically, we can consider an extension of the framework of Universal Algebra to accommodate the new feature of binding. Therefore, the new extended signature is of second order instead of first order, and the operators in the new signature are type-2 operators. We rename them as Binding Operators (BOs). The resulting framework is named as a *Framework for Binding Operators* (FBO).

As we understand, the common practice seems in favour of *higher types* instead of *higher orders*. Recently, Shapiro argued in favour of *higher order*, i.e. extending languages from first order to second order [138]. He claimed that "diminished" second order languages are the lower bound for such an extension. His "diminished" second order language coincides with FBO.

Like the well-known Church-Rosser property of Lambda Calculus leads to equational presentation of it, we will seek an equational characterization of BOs (type-2 operators). Along this line, Aczel gives a Church-Rosser theorem for certain BOs [3], which can be generalized to all BOs. Also, with a modification of his Frege Structure [4], we can derive an algebra for BOs (extensional Binding Algebra or eBA). The derived eBA turns out to be a generalization of the following examples:

(i) first order algebras (see Subsection 1.3.3),

(ii) Kechris and Moschovakis' suitable class of functionals [83] (see comments after Definition 1.2.1.3),

(iii) Volken's λ-family (see Subsection 9.5.3 and [7, p.127]),

(iv) Plotkin's $\mathcal{P}\omega$-model of Lambda Calculus [133,134,105,118] (see comments after Definition 1.2.1.3),

(v) Girard's qualitative domain of F system [47] (see comments after Definition 1.2.1.3).

Naturally, following Birkhoff [16], we would like to characterize BOs equationally. Kechris and Moschovakis' Enumeration Theorem [83] suggests that an algebraic characterization of such might be possible.

Unfortunately, eBAs and the usual satisfaction $\models_{eBA}$ of Binding Equations (in

Birkhoff's approach) do not work. Therefore, we have to find either a remedy for it or a new semantic model for BOs. We will present two solutions, one for each.

(a) For a remedy, we discover a condition on satisfaction for Birkhoff's approach to work. This condition is necessary and sufficient, and we call it an *admissible* condition, which turns out weaker than Plotkin's *Logical Relations* [121,144] in the sense that "logical" implies "admissible". An admissible equational calculus $\dot{\vdash}_{eBA}$ for Binding Equations is supplied in Chapter 3. And the relation between satisfaction $\models_{eBA}$ and admissible satisfaction $\dot{\models}_{eBA}$ is discussed in Section 3.13, although this is not completely clear. Other problems remain open for the time being, such as the admissible variety problem.

(b) For a new semantic model, we will give a new Binding Algebra (iBA) which is intensional in contrast with the previous extensional one (eBA). Actually, an iBA is a generalization of Friedman's Prestructures [46]. A sound and complete Equational Calculus $\vdash_{iBA}$ for iBAs is established in Chapter 5. However, $\vdash_{iBA}$ is weaker than $\dot{\vdash}_{eBA}$ in proof power; and the precise difference between them is discussed in Section 11.2. Also, the relationship between extensional $\models_{eBA}$ and intensional $\models_{iBA}$ will be discussed in Section 11.1.

Examples of application of $\vdash_{iBA}$ to equationalization of First Order Logic, of Lambda Calculus and Combinatory Logic, and of Calculus of Communicating Systems (CCS) with data-dependency are subjects of Chapter 8, Chapter 9 and Chapter 10 respectively.

This chapter mainly serves as an introduction to new concepts and results. In what follows, Section 1.1 provides a Framework for BOs. Semantic issues and algebraic characterizations are classified into two approaches, i.e. Birkhoff's vs Friedman's (or Extensional vs Intensional), and summaried in Section 1.2. Section 1.3 provides the following:

Issues related to implementation and presentation such as relationship among Diophantine equations (or presentation), recursive functions and Flowchart Computability (or implementation) are in Subsection 1.3.1. Subsection 1.3.2 gives a reason for not choosing Combinatory Logic's approach in this thesis. Subsection

1.3.3 serves to verify that first order algebras are special cases of eBAs. Review of works related to binding such as Variable Binding Term Operators (VBTO), Henkin, Tarski and Monk's Cylindric Algebras, Haloms' Monadic Algebras and Polyadic Algebras are presented in Subsection 1.3.4, Subsection 1.3.5 and Subsection 1.3.6 respectively. These demonstrate that the present Framework is a generalization of them. Relationship between Logical Relations and Admissibility is discussed in Subsection 1.3.7. A potential usage of the present Framework for BOs is of dealing with Natural Languages, and this is mentioned in Subsection 1.3.8.

Some future developments closely related to Computer Science are supplied briefly in Section 1.4. They are:

(1) an introduction of higher sorts, which is accompanied by discussions

(1.a) on the distinction between higher types (or sorts) and higher orders (Subsection 1.4.1), and

(1.b) on categorical aspects of eBAs and iBAs vs 2-Categories (Subsection 1.4.2);

(2) generalized Full Abstraction (Subsection 1.4.4); and

(3) generalized Institutions (Subsection 1.4.5).

1.1 Framework for Binding Operators (FBO)

1.1.1 Binding primitives

Semantics is the essence; and languages, expressions and mathematical machinery obtain their remarkable power by *properly* capturing semantics (or meanings) to syntactic level. I substantiate this claim by picking up some handy examples, say,

(a) existential quantifier: $\exists x.P(x)$ in First Order Logic (see [160,140,113,139] and many others for references to First Order Logic),

(b) **let**-expression: **let** x **be** d **in** e in functional programming language ML [55, 62],

(c) λ-expression: $\lambda x.M$ in Lambda Calculus [26,7,71], and

(d) input command: $\alpha?x.B(x)$ in either CCS [108,109], in CSP [73,166,74], or in LOTOS [79,18].

However, abstraction is not our ultimate goal. Let us take an obvious (naive) view such that a theory A of a subject is said to be more abstract than another theory B of the same subject if expressions of A are shorter than the ones of B. Then, some basic results from Information Theory (or Kolmogorov Complexity, see [95] for a reference) show that A can do so only up to certain extent after that B would be more abstract than A if we keep the previous view; furthermore, regardless how sophisticated A is and how trivial B is, this result stands without any change. Therefore, instead of seeking arbitrary abstraction, we consider a *proper* abstraction, which means that the abstraction makes more sense in human intuition.

Now, we return to the previous examples from (a) to (d). These examples used here is not only to emphasize the importance of abstraction but also to bring in the idea of binding primitives. Observing them carefully, you would discover that there is a general representation, say $\exists$, **let**, λ and **in**-α, so-called Binding Operators (BOs for short). To put this point of BOs more explicitly, I denote a binding primitive as $\langle _ : _ \rangle$, where the first "hole" $_$ shows the bound variables in the second "hole" $_$ (term) separated by the sign ":", and alter the previous examples accordingly, i.e.

(a') existential quantifier: $\exists(\langle x : P(x) \rangle)$,

(b') **let**-expression: $\mathbf{let}(\langle x : e \rangle, d)$,

(c') λ-expression: $\lambda(\langle x : M \rangle)$, and

(d') input command: $\mathbf{in}\text{-}\alpha(\langle x : B(x) \rangle)$.

Nevertheless, $\langle _ : _ \rangle$ is a unary binding. It binds the free variable x in terms of $P(x)$, e, and M to make the function terms of $\langle x : P(x) \rangle$, $\langle x : e \rangle$, and $\langle x : M \rangle$ respectively.

It is quite easy to generalize the idea of unary binding $\langle _ : _ \rangle$ to arbitrary finite binding $\langle \underbrace{_, _, ..., _}_{n \text{ times}} : _ \rangle$ $(n \in Nat)$, even to infinite binding, say countably infinite ω $\langle \underbrace{_, _, ...}_{\omega \text{ times}} : _ \rangle$, or even uncountably infinite. But we only consider the arbitrary finite binding in this thesis, and infinite binding is deliberately left out. However, infinite binding (or more precisely closed binding) has been considered in Cylindric Algebras [66,67]. We will briefly return to Cylindric Algebras later in this chapter in Subsection 1.3.5.

The above observation lead us to consider binding in a broad sense. The wide spread examples involving binding from the above and as well as from [43,13,159] suggest that binding deserves a more general framework for investigation. A framework with great generality in our mind is Universal Algebra. We can think of extending the framework of Universal Algebra to include bindings as primitives. However, this does not mean that we do not like categorical presentation. On the contrary we would very much like to see how the distinction of syntax and semantics of Tarskian doctrine in present thesis to be presented in categorical way and the corresponding interaction between them (see Subsection 1.4.2 for more details). Nevertheless, for people who prefer categorical presentation, they should note that Universal Algebra (or Free Algebra) can be presented through an Adjunction or a Monad, see [97.70.6,128,9] for more details on this issue.

In Subsection 1.1.2, we proceed along the idea of algebraic extension and work out what is the signature with binding primitives in such a framework.

1.1.2 Signature with binding primitives

The signature with binding primitives can be thought of as an extension of ordinary signature Σ to include all arbitrary finite Binding Operators. Therefore, we have to raise the order of the language from first order to second order, i.e. the new extended signature Σ^{BO} has operations of second order.

We know that Second Order Logic is incomplete [19] and that there is a potential inconsistency in equational theories if we combine λ-abstraction with equational

theories [20]. It is very reasonable for us to accommodate the new feature of binding in some restricted way such that the new extended language will still be very rich.

We let the new extended signature Σ^{BO} be indexed by $(S^* \times S)^* \times S^* \times S$, (or more meaningful $(S^* \Rightarrow S)^* \times S^* \rightarrow S$ if you prefer a "typed" presentation) where S is $Sort(\Sigma^{BO})$ and the superscript $*$ means strings of elements from S (traditionally called the Kleene star operator). However, should you be confused by (syntactic) higher orders and (semantic) higher types, see Subsection 1.4.1 for more analysis. It is just to remind you of that $\Rightarrow$ belongs to the kind of syntactic higher orders. Such a Σ^{BO} is powerful enough to include all previous mentioned examples. Let us demonstrate this for the single-sorted case,

(a") the existential quantifier: $\exists \in \Sigma^{BO}_{(\bullet \Rightarrow \bullet) \rightarrow \bullet}$, and

(b") λ-expression: $\lambda \in \Sigma^{BO}_{(\bullet \Rightarrow \bullet) \rightarrow \bullet}$, and

(c") **let**-expression: $\mathbf{let} \in \Sigma^{BO}_{(\bullet \Rightarrow \bullet) \times \bullet \rightarrow \bullet}$, and

(d") input command: $\mathbf{in}\text{-}\alpha \in \Sigma^{BO}_{(\bullet \Rightarrow \bullet) \rightarrow \bullet}$.

Note that the meaning of $\bullet \rightarrow \bullet$ is different from $\bullet \Rightarrow \bullet$. The former shows an object being mapped from one domain to another under interpretations, and the latter intuitively is the sort for function space. This will become clearer after "interpretations" having been defined. Also, the above derived signature makes BOs coincide with type-2 operators, which are syntactic names for generalized type-2 functionals.

Having derived the signature Σ^{BO} with binding primitives, we can now turn to the language (or binding terms) for the intended framework.

1.1.3 Language with finitary binding primitives

For a start, we concentrate on single-sorted algebras of finitary Binding Operators. There are three reasons for this.

(i) Philosophically, we are thinking of some kind of type-free theory.

(ii) Notationally, this makes things more simple.

(iii) The last one is quite realistic, i.e. the distinction between higher orders and higher types as well as their interactions is not entirely clear.

Therefore, the subject of how to present many-sorted and/or higher-sorted Framework for BOs becomes delicate. It is a good idea to postpone this to a later stage of development, at least for the time being. Of course, if we do not consider higher sorts, a possible extension to such simple many-sorted Framework for BOs and their algebraic characterization can be worked out accordingly and it will be mentioned in Chapter 7, but rather briefly.

Since Σ^{BO} is single-sorted, the indices can be simplified to $Nat^* \times Nat$, where Nat is the set of natural numbers. Let V be a countably infinite set of ordinary variables with a linear order, let $x, y, z, \ldots$ range over V; and $\vec{x}, \vec{y}, \vec{z}, \ldots$, range over lists of (distinct) ordinary variables. Let FV be a family of FV_m (the set of function variables with arity $m \geq 0$), and $f, \ldots$ range over FV. For all $m \geq 0$, FV_m and V are all disjoint from each other.

Definition 1.1.3.1 (binding terms BT): Binding terms BT *is a pair of* $< T, FT >$ *where* T *is the set of* ordinary terms *and* FT *is the family of* FT_m *the set of* function terms *with arity* m *(*$m \in Nat$*). They are generated by using the formation rules:*

1.

$$\frac{x \in V}{x \in T};$$

2.

$$\frac{t_1, t_2, \ldots, t_n \in T}{f(t_1, t_2, \ldots, t_n) \in T} \; (f \in FV_n \; and \; n \in Nat);$$

3.

$$\frac{t \in T;\; x_1, x_2, \ldots, x_k \in V}{\langle x_1, x_2, \ldots, x_k : t \rangle \in FT_k} \; (x_i \neq x_j \; if \; i \neq j);$$

4. *for* $< \vec{m}, n > \in Nat^* \times Nat$ *and* $\ell = |\vec{m}|$,

$$\frac{ft_1 \in FT_{m_1}, ft_2 \in FT_{m_2}, \ldots, ft_\ell \in FT_{m_\ell}, t_1, t_2, \ldots, t_n \in T}{\sigma(ft_1, ft_2, \ldots, ft_\ell, t_1, t_2, \ldots, t_n) \in T} \; (\sigma \in \Sigma^{BO}_{<\vec{m},n>}).$$

The above definition deserves some explanation:

(1) for a conceptual reason, we keep zero binding term $< \varepsilon : t >$ different from ordinary term t;

(2) when $n = 0$, the second condition above becomes

$$\frac{f \in FV_0}{f() \in T};$$

(3) when $k = 0$, the third condition becomes

$$\frac{t \in T}{\langle \varepsilon : t \rangle \in FT_0},$$

where ε is a special symbol and denotes the empty list;

(4) when $\ell = 0$, the fourth condition becomes

$$\frac{t_1, t_2, ..., t_n \in T}{\sigma(t_1, t_2, ..., t_n) \in T} \ (\sigma \in \Sigma_{<\varepsilon,n>});$$

further if also $n = 0$, it will become

$$\frac{}{\sigma() \in T} \ (\sigma \in \Sigma_{<\varepsilon,0>});$$

(5) when $n = 0$, the fourth condition becomes

$$\frac{ft_1 \in FT_{m_1}, ft_2 \in FT_{m_2}, ..., ft_\ell \in FT_{m_\ell}}{\sigma(ft_1, ft_2, ..., ft_\ell) \in T} \ (\sigma \in \Sigma_{<\vec{m},0>}).$$

Also, we obviously have $FV_m \cap FT_m = \emptyset$ $(m \geq 0)$ and $V \subseteq T$.

Later for convenience, we introduce some abbreviations as follows.

(a) $f(t_1, t_2, ..., t_n)$ is often abbreviated as $f(\vec{t})$ such that $n = |\vec{t}|$, $\vec{t}(j) = t_j$, and $\vec{t}(j)$ means the jth element of list $\vec{t}$, where $|\vec{t}|$ means the length of list $\vec{t}$;

(b) $\langle x_1, x_2, ..., x_k : t \rangle$ is sometimes shortened as $\langle \vec{x} : t \rangle$ where $k = |\vec{x}|$;

(c) $\sigma(ft_1, ft_2, ..., ft_\ell, t_1, t_2, ..., t_n)$ is usually written as $\sigma(\vec{ft}, \vec{t})$ where $\vec{ft}(i) = ft_i$, $\vec{t}(j) = t_j$, $|\vec{ft}| = \ell$ and $n = |\vec{t}|$.

1.1.4 Equational examples with BOs

So far, we have the syntactic set-up of the Framework for BOs. With this set-up, it is possible to treat many theories in FBO.

(1) For example, the β-conversion of Lambda Calculus can be expressed as

$$(\beta) \quad \mathbf{app}(\lambda x.f(x), y) \simeq f(y)$$

where $\lambda x.f(x)$ is a conventional abbreviation for $\lambda(\langle x : f(x)\rangle)$, $\mathbf{app} \in \Sigma_{<\varepsilon,2>}$ and $\simeq$ expresses "is equal to".
Because of the well-known Church-Rosser property, Lambda Calculus may be equationally presented.

(2) For functional programming language ML, there is an equivalence relation with Lambda Calculus, simply

$$(\lambda\text{-}\mathbf{let}) \quad \mathbf{app}(\lambda x.f(x), y) \simeq \mathbf{let}\, x\, \mathbf{be}\, y\, \mathbf{in}\, f(x)$$

where $\mathbf{let}\, x\, \mathbf{be}\, y\, \mathbf{in}\, f(x)$ is a conventional abbreviation for $\mathbf{let}(\langle x : f(x)\rangle, y)$. This leads to an equational characterization of ML inherited from the one for Lambda Calculus. However, we can certainly develop the equational characterization of ML on its own. For instance,

$$\mathbf{let}(\langle x : h(f(x), z)\rangle, y) \simeq h(\mathbf{let}(\langle x : f(x)\rangle, y), z)$$

or conventionally

$$\mathbf{let}\, x\, \mathbf{be}\, y\, \mathbf{in}\, h(f(x), z) \simeq h(\mathbf{let}\, x\, \mathbf{be}\, y\, \mathbf{in}\, f(x), z)$$

expresses a property of ML that if a variable, say x, is globally declared in h but is only used locally in f then it can be declared locally instead. This property can be used to reduce the number of global variables in programming.

(3) In First Order Logic, let us take the existential quantifier $\exists$ as an example. Its role can be captured as follows.

$$(\exists\text{-}elim) \quad \frac{\{t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}}{\{\exists x.t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}}$$

where $\exists x.t$ is a conventional abbreviation for $\exists(\langle x : t\rangle)$, x is not free in u, $\hookrightarrow$ means "implies", **True** means the value *true* in truth-values, and the bar $\frac{\bullet}{\bullet}$ stands for the inference which says that the bottom $\bullet$ is inferred from the top $\bullet$.

For the top part of ($\exists$-*elim*), $\{t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}$ is a conditional (binding) equation, which is renamed as a *quasi-dependent* Binding Equation for certain reason to be presented in Section 2.6. Informally, the quasi-dependent Binding Equation expresses that the implication from the *true* value of t to the *true* value of u is not dependent on the value of the variable x. Similarly, for the bottom part of ($\exists$-*elim*), $\{\exists x.t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}$ says that the *true* value of $\exists x.t$ implies the *true* value of u.

(4) For CCS, let us consider the input command **in**-α. There is a rule helping to characterizes the relationship between an input and an **if**-statement **if** b **then** t **else** u (or $b \supset (t, u)$) i.e. (?-**then**-τ-*elim*) rule (see Lemma 10.3.8): let $\alpha?x.t$ be **in**-$\alpha(\langle x : t\rangle)$, then

$$\{\alpha?x.t \simeq \alpha?x.u\} \mapsto \alpha?x.(b \supset (t, v)) \simeq \alpha?x.(b \supset (u, v)),$$

where $\bullet \mapsto \bullet$ is another kind of implications (*dependent*) different from the previous one like $\hookrightarrow$ (*quasi-dependent*), it means either that the front $\bullet$ does not hold or that the rear $\bullet$ holds[1]. This axiom schema plays an important role in equationalizing the weak bisimulation $\approx$ of CCS with data-dependency, especially the elimination of the silent action τ in an **if**-statement prefixed by an input. For instance, this rule is needed in deriving

$$\alpha?x.(b \supset (\tau.t, v)) \simeq \alpha?x.(b \supset (t, v)).$$

All the above examples demonstrate how equations with BOs (i.e Binding Equations or BEs) can be used to express logic, programming and concurrency. This

---

[1]These two kind implications $\mapsto$ and $\hookrightarrow$ have been recognized by Henkin as $\models$ and $\models^*$ respectively. And he presents two deduction systems for natural numbers in [65] (see Section 2.6 for more).

leads us to characterize BOs by equations in general, like Equational Logic in Universal Algebra. Going along this idea, we are further interested in whether there exists an inference rule system for such equational characterization, like Birkhoff's Theorems of Equational Logic in Universal Algebra. If such an inference system does exist and if we are able to capture it precisely, then we would like to obtain the equational characterization of BOs once for all. Therefore, it is up to a user to specify which individual system he want to work with, say either in Lambda Calculus, in ML, in First Order Logic, or in CCS provided that he can give proper axioms for each. More importantly, he can work with several theories at a same time without switching frameworks. Of course, this is reliant on the availability of axioms of the corresponding theories in the Framework for BOs.

However, before we can possibly answer the questions raised above, we have to know what a *Binding Algebra* is. In other words, *what is the semantics of BOs?* Without a semantics, we can never know whether an equational characterization is sound, to say nothing of completeness.

1.2 Semantics for BOs and their characterizations

With regard to semantics (or meanings) of binding terms, we motivate the intensional aspect and the extensional aspect of the meanings of a function by the example of a polynomial, say in Number Theory. A polynomial x^2+x+1 can be viewed as a "transformation" which relates the object n to the object n^2+n+1 (say $n \in Nat$), and the same polynomial can be viewed as the object which stores the whole "transformation". The latter view is the *intension* of the polynomial and the former is the *extension* of it. So, the first "obvious" attempt to semantics is the extensional one, and it is called Birkhoff's approach. Aczel's work on Frege Structures [4] can be regarded as following this approach. For the intensional attempt to semantics for BOs, Friedman does some work on equality between functionals [46] along this line. Statman puts Friedman's work in another way and says that Friedman only treats

closed terms [143]. In what follows, we are going to present results by following each approach. Subsection 1.2.1 is devoted to Birkhoff's approach where we show that Birkhoff's method in first order algebras can not trivially applied and certain modifications are needed. Subsection 1.2.2 serves for Friedman's approach. The last part (Subsection 1.2.3) is to present the core result of the present thesis, i.e. (intensional) Completeness of $\vdash_{iBA}$ and (extensional) Admissible Completeness of $\dot{\vdash}_{eBA}$.

1.2.1 Birkhoff's approach and its remedy

This subsection will exploit the extensional aspect and establish the definition of an extensional Binding Algebra, eBA for short. First, we will give a definition for eBAs (Definition 1.2.1.3), which can be regarded as borrowed from Aczel's Frege Structures [A80] with a slight modification. The definition of interpretations in eBAs, Definition 1.2.1.4, follows after that.

Definition 1.2.1.1 (explicit closedness): *Let A be a set and let $\mathcal{F}_m(A)$ be a subset of function space $A^m \to A$ for $m \geq 0$. A family $< A', \mathcal{F}(A) >$ such that $A' \subseteq A$ and $\mathcal{F}(A) = \{\mathcal{F}_m(A) | m \in Nat\}$[2] is called explicitly closed iff*

1. *(eBA-cons) for each $m \geq 0$ and $a \in A'$, there is a unique function $C_{m,a} \in \mathcal{F}_m$ such that $C_{m,a}(\vec{a}) = a$ for $\vec{a}$ in A^m.*

2. *(eBA-proj) for each $m > 0$ and $1 \leq i \leq m$, there is a unique function, named $\pi_{m,i} \in \mathcal{F}_m$, such that $\pi_{m,i}(\vec{a}) = a_i$ for $\vec{a}$ in A^m.*

3. *(eBA-cmp) for $m > 0$, $k \geq 0$ given $g \in \mathcal{F}_m$ and $g_i \in \mathcal{F}_k$ $(1 \leq i \leq m)$, there is a unique function $h \in \mathcal{F}_k$ such that $h(\vec{a}) = g(g_1(\vec{a}), g_2(\vec{a}), ..., g_m(\vec{a}))$ or $h = g\odot < \vec{g} >$; sometimes, it is further abbreviated as $g(\vec{g})$.*

---

[2]The reason for A' being a subset of A rather than just A is to share the same concept of explicit closedness with binding subalgebras. However, the formal introduction of binding subalgebras is postponed to Chapter 3 (Definition 3.1.1).

Note that we purposely use $\odot$ for the usual compositional functional, instead of $\circ$ to avoid a potential confusion with another use of $\circ$ in the future (see Subsection 1.2.2). Also, we will write $\mathcal{F}$ and $\mathcal{F}_m$ for $\mathcal{F}(A)$ and $\mathcal{F}_m(A)$ respectively if no confusion involves.

Informally, the explicit closedness of a family means that the family is closed under constants, projections and function compositions. Such a closure is not necessarily preserved by any map over the family. We, therefore, introduce the concept of uniformity over the family.

Definition 1.2.1.2 (eBA-unif): *For $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$ ($|\vec{m}| = \ell$), an interpretation of it is in general a functional $\mathcal{A}_\sigma : \mathcal{F}_{\vec{m}} \times A^n \to A$ (or $\mathcal{F}_{m_1} \times \mathcal{F}_{m_2} \times ... \times \mathcal{F}_{m_\ell} \times A^n \to A$) of σ is* uniform *over $< A', \mathcal{F} >$ iff for any $k \geq 0$, given $g_i \in \mathcal{F}_{k+m_i}$ ($i = 1, ..., \ell$), and $g_{\ell+j} \in \mathcal{F}_k$ ($j = 1, ..., n$), the function h is in $\mathcal{F}_k$ where $h(\vec{a}) = \mathcal{A}_\sigma(\vec{h}, \vec{b})$ for all $\vec{a}$ in $(A')^k$. Here, $h_i(\vec{a^i}) = g_i(\vec{a}, \vec{a^i})$ for all $\vec{a^i}$ in A^{m_i} and $i = 1, ..., \ell$, and $b_j = g_{\ell+j}(\vec{a})$ for $j = 1, ..., n$.*

Note that h_i is in $\mathcal{F}_{m_i}$ since

$$h_i = g_i \odot < C'_{m_i,a_1}, C'_{m_i,a_2}, ... C'_{m_i,a_k}, \pi_{m_i,1}, \pi_{m_i,2}, ..., \pi_{m_i,m_i} >$$

for $1 \leq i \leq \ell$ where $a_1, a_2, ..., a_k \in A'$ and $k = |\vec{a}|$.

Now, we are able to define an extensional Binding Algebra (eBA) as following.

Definition 1.2.1.3 (eBA): *An extensional Σ^{BO}-algebra (extensional Binding Algebra or eBA) $\mathbf{A}$ is a pair of $< \mathcal{F}^{\mathbf{A}}, \mathcal{A} >$ such that*

(i) $\mathcal{F}^{\mathbf{A}} =< A', \mathcal{F}(A) >$ is an explicitly closed family and $A' = A$ and $\mathcal{F}(A) = \{\mathcal{F}_m(A) | m \in Nat\}$,

(ii) $\mathcal{A} = \{\mathcal{A}_\sigma | \sigma \in \Sigma^{BO}\}$ and for each $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$ $\mathcal{A}_\sigma$ (or $\mathbf{A}_\sigma$, $\sigma^{\mathbf{A}}$, or even σ) is uniform over $\mathcal{F}^{\mathbf{A}}$.

Sometimes, we write $\mathcal{F}$ for $\mathcal{F}^{\mathbf{A}}$ and leave $\mathbf{A}$ be decided by contexts.

Compared to Kechris and Moschovakis' suitable classes of functionals in [83], an eBA is said to be a generalized case of them in the following sense : (a) the condition (i) above corresponds to the composition of "addition of variables" (closed under

constants), "substitution of projections" (closed under projections) and "compositions" (closed under function compositions); (b) the condition (ii) above corresponds to "functional substitutions"; (c) the other conditions for a suitable class of functionals are actually limited to the standard model of natural numbers. Also, their concept of *concentration of a suitable class of functionals* is similar to the explicit closedness.

Another example of eBAs are first order algebras, see Subsection 1.3.3 for an illustration on this.

A third example of eBAs is Volken's λ-family, see Subsection 9.5.3.

A fourth example of eBAs is Plotkin's $\mathcal{P}\omega$-model of Lambda Calulus [105,118, 133,134]. To see this more clearly, let A be $\mathcal{P}\omega$ and $\mathcal{F}_m$ be the continuous function space from m product of $\mathcal{P}\omega$ to $\mathcal{P}\omega$ (apparently this $\mathcal{F}$ is explicitly closed), then given $k \in Nat$ we have that $[\![\lambda]\!]\odot < curry_{k,1}(g), h >$ and $[\![\mathbf{app}]\!]\odot < h_1, h_2 >$ are continuous for each $g \in \mathcal{F}_{k+1}$ and every $h, h_1, h_2 \in \mathcal{F}_k$. This implies that $[\![\lambda]\!]$ and $[\![\mathbf{app}]\!]$ are uniform over $\mathcal{F}$.

A fifth example of eBAs is Girard's qualitative domains in F system [47]. More specifically, let A be a qualitative domain, and $\mathcal{F}_m$ be the stable function space from A^m to A. Obviously, this $\mathcal{F}$ is explicitly closed. Furthermore, Theorem 1.11 (i) and (ii) in [47] guarantee that $[\![\lambda]\!]$ (i.e. (i)) and $[\![\mathbf{app}]\!]$ (i.e. (ii)) are uniform over stable function spaces.

Next, we turn our attention to interpretations. Let $\mathbf{A}$ be an eBA, a *valuation* $\vec{\rho}$ on $\mathbf{A}$ is a family of functions ρ (from V to A) and ρ_k (from FV_k to $\mathcal{F}_k$ for $k \in Nat$). Later, we will denote a valuation $\vec{\rho}$ as a pair of $< \rho, \vec{\varphi} >$ where $\vec{\varphi}$ is a family of φ_k ($\varphi_k = \rho_k$, $k \in Nat$), simply denote it as $< \rho, \varphi >$. Sometimes, we call a valuation $\vec{\rho}$ an *environment*. Let $< \rho, \varphi >$ be a valuation on $\mathbf{A}$. Then for any $x \in V$ and any $a \in A$, $\rho[a/x]$ can be defined by

$$\rho[a/x](y) = \begin{cases} \rho(y) & \text{if } y \neq x \\ a & \text{if } y = x. \end{cases}$$

From this, we have (1) that $\rho[a/x][b/x] = \rho[b/x]$ for all $a \in A$ and (2) that $\rho[a/x][b/y] = \rho[b/y][a/x]$ if $x \neq y$, for all $a, b \in A$. Therefore, we can let $\rho[\vec{a}/\vec{x}]$

be $\rho[a_1/x_1][a_2/x_2]...[a_{|\vec{a}|}/x_{|\vec{x}|}]$ where $|\vec{a}| = |\vec{x}|$, since the result does not depends on the order of the appearances of variables.

Definition 1.2.1.4 (interpretation in an eBA): *Let* $\mathbf{A}$ *be an eBA, and* $< \rho, \varphi >$ *be a valuation of* V *and* FV *on* $\mathbf{A}$. *An interpretation* $\mathcal{A}[\![\bullet]\!](\rho, \varphi)$ *of binding terms in* BT *over* $\mathbf{A}$ *under the environment* $< \rho, \varphi >$ *is defined inductively by*

1. $\mathcal{A}[\![x]\!](\rho, \varphi) =_{df} \rho(x)$ *for* $x \in V$;

2. $\mathcal{A}[\![f(\vec{t})]\!](\rho, \varphi) =_{df} \varphi(f)(\mathcal{A}[\![\vec{t}]\!](\rho, \varphi))$ *for* $f \in FV_n$ *and* $t_j \in T$ $(1 \leq j \leq n)$,

   where $|\vec{t}| = n$ *and* $\mathcal{A}[\![\vec{t}]\!](\rho, \varphi)$ *is the abbreviation of list* $\mathcal{A}[\![t_1]\!](\rho, \varphi), \mathcal{A}[\![t_2]\!](\rho, \varphi),$ ..., $\mathcal{A}[\![t_n]\!](\rho, \varphi)$;

3. $\mathcal{A}[\![\sigma(\vec{ft}, \vec{t})]\!](\rho, \varphi) =_{df} \sigma^{\mathbf{A}}(\mathcal{A}[\![\vec{ft}]\!](\rho, \varphi), \mathcal{A}[\![\vec{t}]\!](\rho, \varphi))$ *for* $\sigma \in \Sigma_{<\vec{m},n>}$, $ft_i \in FT_{m_i}$ $(1 \leq i \leq \ell)$ *and* $t_j \in T$ $(1 \leq j \leq n)$,

   where $|\vec{ft}| = |\vec{m}| = \ell$ *and* $\mathcal{A}[\![\vec{ft}]\!](\rho, \varphi)$ *is the abbreviation of list* $\mathcal{A}[\![ft_1]\!](\rho, \varphi)$, $\mathcal{A}[\![ft_2]\!](\rho, \varphi)$, ..., $\mathcal{A}[\![ft_\ell]\!](\rho, \varphi)$.

4. $\mathcal{A}[\![\ \langle \vec{x} : t \rangle\ ]\!](\rho, \varphi) =_{df} g$,

   where $g(\vec{a}) =_{df} \mathcal{A}[\![t]\!](\rho[\vec{a}/\vec{x}], \varphi)$ *and* $|\vec{a}| = |\vec{x}|$.

The well-definedness of the above is not obvious, and this matter will be discussed in the very beginning of Chapter 3 (Theorem 3.0.5). However, Kechris and Moschovakis' Functional Substitution Theorem in [83] may provide some confidence, since functional substitution corresponds to *uniformity* as we pointed out before.

In general, a binding term p "is equal to" another binding term q, written as $p \simeq q$, if and only if (or iff) all evaluations of the two binding terms under every environment (commonly referred to as *interpretations*) are the same, i.e. they can not be *distinguished* from one and the other. Informally, two binding terms p and q are *indistinguishable* by an eBA $\mathbf{A}$ iff all possible interpretations of p and q in $\mathbf{A}$ are the same, denoted as $\mathbf{A} \models_{eBA} p \simeq q$. Formally,

Definition 1.2.1.5 (satisfaction $\models_{eBA}$): $\mathbf{A} \models_{eBA} p \simeq q$ *(or* $\mathbf{A} \models p \simeq q$ *for short) iff* $\mathcal{A}[\![p]\!](\rho, \varphi) = \mathcal{A}[\![q]\!](\rho, \varphi)$ *for every environment* $< \rho, \varphi >$ *over* $\mathbf{A}$.

Further, we can naturally extend the above property of an individual eBA to the property of a collection of eBAs, say the collection of all possible eBAs. This universal property is written as $\models_{eBA} p \simeq q$ (or $\models p \simeq q$), and is defined by $\mathbf{A} \models_{eBA} p \simeq q$ for every eBA $\mathbf{A}$. Apparently, this kind of Indistinguishability is commonly said "equality" and it is very strong. Certainly, we can weaken it a bit and still get an interesting property, which deserves our study. For example, let γ be a set of Binding Equations (BEs) and Δ (or $p \simeq q$) be a BE, thus we have the following:

(a) *dependent Indistinguishability* $\gamma \mapsto \Delta$, so-called a *dependent Binding Equation* (dBE), i.e. if $\mathbf{A} \models \Delta'$ for each $\Delta' \in \gamma$ then $\mathbf{A} \models \Delta$; and

(b) *quasi-dependent Indistinguishability* $\gamma \hookrightarrow \Delta$, a *quasi-dependent Binding Equation* (qBE), i.e. under every environment, if the corresponding interpretations of each pair $p' \simeq q' \in \gamma$ are the same then the interpretations of the pair $p \simeq q = \Delta$ is the same. Formally,

Definition 1.2.1.6 (dBEs and qBEs): *Let γ be a set of BEs and Δ be a BE.*

1. $\mathbf{A} \models^{d}_{eBA} \gamma \mapsto \Delta$ *(or* $\mathbf{A} \models_{eBA} \gamma \mapsto \Delta$ *even* $\mathbf{A} \models \gamma \mapsto \Delta$ *) iff* $\mathbf{A} \models \Delta'$ *for each* $\Delta' \in \gamma$ *implies* $\mathbf{A} \models \Delta$*;*

2. $\mathbf{A} \models^{q}_{eBA} \gamma \hookrightarrow \Delta$ *(or* $\mathbf{A} \models_{eBA} \gamma \hookrightarrow \Delta$ *even* $\mathbf{A} \models_{eBA} \gamma \hookrightarrow \Delta$ *) iff for each environment* $< \rho, \varphi >$*, if* $\mathcal{A}[\![p']\!](\rho,\varphi) = \mathcal{A}[\![q']\!](\rho,\varphi)$ *for each pair* $p' \simeq q' \in \gamma$ *then* $\mathcal{A}[\![p]\!](\rho,\varphi) = \mathcal{A}[\![q]\!](\rho,\varphi)$ *where* $p \simeq q = \Delta$.

Later, we shall sometimes use "equality", "dependent implications" and "quasi-dependent implications" instead of Indistinguishability, dependent Indistinguishability and quasi-dependent Indistinguishability respectively.

By definition, we should know that a dBE is an equational form of an inference for a deduction of BEs. Because of this, $\gamma \mapsto \Delta$ can be viewed as either an inference rule of the deduction of BEs $(\frac{\gamma}{\Delta})$ or a $\gamma \mapsto \Delta$. This idea will play a crucial role in our complete equationalization of First Order Logic in Chapter 8.

We should also point out that the terms of "dependent equations" and "quasi-dependent equations" are "equational implications" and "conditional equations"

(or "quasi-equations") respectively. They are so chosen is to show our intention to unify and to relate the old and unconnected terms like "equational implications" and "conditional equations" (or "quasi-equations" or universal Horn Clauses). The old terms seems not serving this purpose well. The reason for such terminology is given in Section 2.6.

Further along the idea of unifying, you might notice that there is one thing in common among BEs, dBEs and qBEs. That is, the equality of the pair of binding terms in Δ is closely related to the equality of every pairs of binding terms in γ. Technically, we can bring these three forms together and combine them into one form, say $\{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$ or simply Ω. More specifically, we take its meaning as the property which is defined through the suggestion of its notation. That is, when $M = \emptyset$, Ω becomes a pure qBE, i.e. $\emptyset \mapsto (\gamma \hookrightarrow \Delta)$ or $\gamma \hookrightarrow \Delta$ for short; further if $\gamma = \emptyset$, Ω is a pure BE, i.e. $\emptyset \mapsto (\emptyset \hookrightarrow \Delta)$ or Δ for short; when all γ's are empty, Ω is a pure dBE, i.e. $\{\emptyset \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\emptyset \hookrightarrow \Delta)$ or $\{\Delta^{(m)} | m \in M\} \mapsto \Delta$ for short; of course, further if $M = \emptyset$ Ω is a pure equation, i.e. $\emptyset \mapsto \Delta$, Δ for short. Therefore, we arrive at a universal form for the three kinds of equational forms. We can simply name this equational form as *universal Binding Equations*, uBEs for short.

Definition 1.2.1.7 (uBE): *For a uBE* Ω, $\mathbf{A} \models^{u}_{eBA} \Omega$ *(or* $\mathbf{A} \models_{eBA} \Omega$ *even* $\mathbf{A} \models^{u}_{eBA} \Omega$ *) iff* $\mathbf{A} \models \gamma^{(m)} \hookrightarrow \Delta^{(m)}$ *for every* $m \in M$ *implies* $\mathbf{A} \models \gamma \hookrightarrow \Delta$.

The main aim of this thesis would be to capture the Indistinguishabilities of eBAs syntactically, and try to derive complete syntactic calculi (or inference rule systems) for BEs, dBEs, qBEs and uBEs accordingly.

As usual, the term eBA $\mathbf{T}$ is constructed from binding terms in BT. The key point of the construction is to build an explicitly closed family $\mathcal{F}^{\mathbf{T}}$ from BT as the carriers with uniform interpretations of BOs in Σ^{BO} (Theorem 3.4.9). An extensional *Binding Homomorphism* (or an eBH) from an eBA $\mathbf{A}$ to another eBA $\mathbf{A}'$ is a family of function from A to A' and functionals from $\mathcal{F}^{\mathbf{A}}_m$ to $\mathcal{F}^{\mathbf{A}'}_m$ for $m \in Nat$, simply written as $\zeta : \mathbf{A} \rightarrow \mathbf{A}'$, such that it primarily preserves constants, projections, compositions and functionals of BOs. That is, an eBH is the family which not only preserves functionality (or compositionality) but also preserves some primitive functions like

constant functions and projections (Definition 3.1.3). Then, with certain effort we would arrive at that (1) $\mathbf{A} \models p \simeq q$ iff (2) $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$ for every $\zeta : \mathbf{T} \to \mathbf{A}$, where we let $\bullet_{[p]}$ denote the element in $\mathcal{F}^{\mathbf{T}}$ which corresponds to the binding term p. This shows the importance of eBHs and implicitly demonstrates the significance of their binding kernels in capturing Indistinguishability, where a binding kernel $Ker(\zeta)$ of an eBH $\zeta : \mathbf{A} \to \mathbf{A}'$ is the least extensional Binding Congruence (eBC) containing the core ∇_ζ of ζ, where an eBC is an equivalence relation on an eBA preserving compositionality and Extensionality, and binding core ∇_ζ of an eBH $\zeta : \mathbf{A} \to \mathbf{A}'$ contains $\{< a, a' > \in A \times A | \zeta(a) = \zeta(a')\}$ and $\{< g, g' > \mathcal{F}_m \times \mathcal{F}_m | \zeta(g) = \zeta(g')\}$ for each $m \in Nat$. We further expect that a certain Birkhoff-like theorem such as the following four statements are equivalent with each other:

(1.2.1) $\mathbf{A} \models p \simeq q$,

(1.2.2) $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$ for every $\zeta : \mathbf{T} \to \mathbf{A}$,

(1.2.3) $< \bullet_{[p]}, \bullet_{[q]} > \in \bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$, and

(1.2.4) $\mathbf{T}/\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta) \models p \simeq q$.

However, Birkhoff's approach does not work as expected. It breaks down on the commutativity property, say the commutativity of $\zeta : \mathbf{A} \to \mathbf{A}'$, the natural $\nu_\zeta : \mathbf{A} \to \mathbf{A}/Ker(\zeta)$ and $\hat{\zeta} : \mathbf{A}/Ker(\zeta) \to \mathbf{A}'$[3]. That is, there does not in general exist a commutative diagram below, see *Figure 1-1*.

In other words, $\hat{\zeta}$ does not always exist as we expected. This breakdown implies that (1.2.2) and (1.2.3) above are not equivalent, although (1.2.3) and (1.2.4) are equivalent. That is, let $Mod(\Gamma)$ be the class of eBAs satisfying Γ, the equivalence between

$$Mod(\Gamma) \models_{\epsilon BA} p \simeq q$$

and

$$\mathbf{T}/\vartheta \models_{\epsilon BA} p \simeq q,$$

---

[3] $\mathbf{A}/Ker(\zeta)$ is the quotient eBA of $\mathbf{A}$ over kernel $Ker(\zeta)$. Their precise definitions can be found in Chapter 3 (Definition 3.7.7 and Definition 3.7.14).

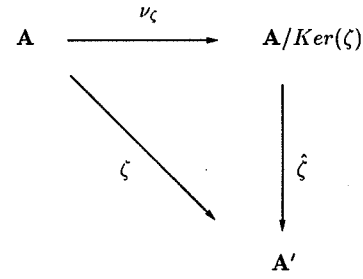

Figure 1–1: commutativity

where ϑ is $\bigcap_{\mathbf{A}\in Mod(\Gamma)} Ker(\mathbf{A})$ and $Ker(\mathbf{A})$ is $\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$, is valid iff we in general relax the satisfaction $\models_{eBA}$ to $\dot{\models}_{eBA}$ and replace $Mod(\Gamma)$ by a larger class $Adm(\Gamma)$, where $\mathbf{A} \in Adm(\Gamma)$ if $\mathbf{A}\dot{\models}_{eBA}\Gamma$; although we have that

$$\Gamma\dot{\vdash}_{eBA} p \simeq q$$

iff

$$\mathbf{T}/\dot{\vartheta} \models_{eBA} p \simeq q$$

where $\dot{\vartheta}$ is almost the same as ϑ except that it only involves admissible eBHs.

So, we are led either (*) to exploit the intensional aspect of "transformation" as explained in the beginning of Section 1.2 or (**) to find a remedy for the "obvious" extensional attempt to the semantics. Actually, we will provide two solutions in the present thesis, one of each kind.

(*) For Birkhoff's approach, we discover a *necessary and sufficient* condition on interpretations (or on eBHs), called *Admissibility* condition, of having the commutativity. This is a result from Theorem 3.8.5, Corollary 3.8.7 and Lemma 3.10.1. That is, the existence of $\hat{\zeta}$ is completely dependent on whether ζ satisfies the Admissibility condition. The essence of the Admissibility condition is Extensionality,

i.e. $\zeta : \mathbf{A} \to \mathbf{A}'$ is *admissible* iff its image of $\mathbf{A}$ is a perfect sub-eBA of $\mathbf{A}'$, where an image of an eBH is always a sub-eBA and a perfect sub-eBA is a sub-eBA with Extensionality. This Admissibility leads us to study admissible Binding Equations (admissible BEs). Informally, two binding terms p, q are *admissibly indistinguishable* by an eBE $\mathbf{A}$ iff all admissible interpretations of p and q are the same, denoted as $\mathbf{A} \dot{\models}_{eBA} p \simeq q$ (the dot $\dot{}$ on top of $\models_{eBA}$ to show the admissibility). Formally,

Definition 1.2.1.8 (Admissible BEs of eBAs): $\mathbf{A} \dot{\models}_{eBA} p \simeq q$ *(or* $\mathbf{A} \dot{\models} p \dot{\simeq} q$*) iff* $\mathcal{A}[\![p]\!](\rho, \varphi) = \mathcal{A}[\![q]\!](\rho, \varphi)$ *for every admissible environment* $< \rho, \varphi >$*, where an admissible environment means that its generated eBH is admissible.*

We succeed in proving the Admissible Completeness of Equational Calculus $\dot{\vdash}_{eBA}$. This is the best result we have in Birkhoff's approach so far. Possible improvements and the relationship between satisfaction $\models_{eBA}$ and admissible satisfaction $\dot{\models}_{eBA}$ along with the comparison between Birkhoff methods in both the first order case (Section 2.1) and in FBO (or eBAs) is provided in Section 3.13.

(**) The intensional (non-extensional) solution in Friedman's approach is the subject of the next subsection.

1.2.2 Friedman's approach

The intensional aspect of semantics was introduced in the beginning of Section 1.2, which can be viewed as to treat all free variables in binding terms BT as bound. From eBAs, we understand that an intensional Binding Algebra (iBA) $\mathbf{B}$ must contain a collection of $\vec{B}$, $pr_{k,i}(1 \leq i \leq k)$, $\circ_{m,k}(m, k \geq 0)$ and $\sigma_k^{\mathbf{B}}(\sigma \in \Sigma_{<\vec{m},n>}, k \geq 0)$ where $\vec{B}$ is a family of sets $B_k(k \geq 0)$, $pr_{k,i}$ is an element in B_k, $\circ_{m,k}$ is an operation in $B_m \times (B_k)^m \to B_k$ (or $B_m \times \underbrace{B_k \times B_k \times ... \times B_k}_{m \text{ times}} \to B_k$), and $\sigma_k^{\mathbf{B}}$ is an operation of $B_{k+\vec{m}} \times (B_k)^m \to B_k$ (or $B_{k+m_1} \times B_{k+m_2} \times ... \times B_{k+m_{|\vec{m}|}} \times \underbrace{B_k \times B_k \times ... \times B_k}_{m \text{ times}} \to B_k$). This collection is said to be a *binding pre-algebra*. Intuitively, it can be viewed as if there is a certain carrier $\mathring{A}$ such that (1) B_k is the function space of $\mathring{A}^k \to \mathring{A}$, (2) $pr_{k,i}$ is a projection map, i.e. $pr_{k,i} : \mathring{A}^k \to \mathring{A}$, and (3) $\circ_{m,k} : ((\mathring{A}^m \to \mathring{A}) \times \underbrace{((\mathring{A}^k \to \mathring{A}) \times ... \times (\mathring{A}^k \to \mathring{A}))}_{m \text{ times}}) \to (\mathring{A}^k \to \mathring{A})$ is a compositional operation, which

will be used in a more readable way $go_{m,k} < \vec{h} >$ than $o_{m,k}(g, \vec{h})$ ($|\vec{h}| = m$), and even $g(\vec{h})$ leaving the choice of k to be determined by the context. We also write $\vec{g}(\vec{h})$ for list $g_1(\vec{h}), ..., g_n(\vec{h})$ ($|\vec{g}| = n$), and write $Pr^k_{i,j}$ for list $pr_{k,i}, ..., pr_{k,j}$ ($1 \leq i \leq j \leq k$).

Definition 1.2.2.1 (iBA): *Let* **B** *be a pair of* $< \vec{B}, \mathcal{B} >$ *such that* $\vec{B} = \{B_m | m \in Nat\}$ *and* $\mathcal{B} = \{pr^{\mathbf{B}}_{k,i}, o^{\mathbf{B}}_{k,j}, \sigma^{\mathbf{B}}_k | k, j \geq 0 \text{ and } \sigma \in \Sigma_{<\vec{m},n>} \text{ and } 1 \leq i \leq k\}$ *and it is a binding pre-algebra. Thus, it is said to be an intensional Binding Algebra (iBA) iff it satisfies the following:*

1. *(iBA-assoc)*

   Suppose f *is in* B_k, $\vec{g}$ *is in* $(B_n)^k$ *(i.e.* $|\vec{g}| = k$*) and* $\vec{h}$ *is in* $(B_m)^n$ *(i.e.* $|\vec{h}| = n$*), then* $(f(\vec{g}))(\vec{h}) = f(\vec{g}(\vec{h})) : B_m$.

2. *(iBA-left-proj)*

   Suppose $\vec{f}$ *is in* $(B_k)^n$ *(i.e.* $|\vec{f}| = n$*), then* $pr_{n,i}(\vec{f}) = f_i : B_k$.

3. *(iBA-right-proj)*

   Suppose f *is in* B_k*, then,* $f(Pr^k_{1,k}) = f$.

4. *(iBA-unif)*

   Suppose σ *is in* $\Sigma_{<\vec{m},n>}$ *and* $|\vec{m}| = \ell$, $\vec{f}$ *is in* $B_{k+\vec{m}}$ *(i.e.* f_i *is in* B_{k+m_i} *for* $1 \leq i \leq \ell$*),* $\vec{g}$ *is in* $(B_k)^n$ *and* $\vec{h}$ *is in* $(B_j)^k$*. Then,*

   $$(\sigma^{\mathbf{B}}_k(\vec{f}, \vec{g}))(\vec{h}) = \sigma^{\mathbf{B}}_j(\vec{f}', \vec{g}(\vec{h})) : B_j$$

   where $f'_i = f_i(Pr^{m_i+j}_{1,m_i}, \vec{h}(Pr^{m_i+j}_{m_i+1,m_i+j}))$ *for* $i = 1, ..., \ell$.

Note that the last law of the above is to preserve functional composition of an iBA. So, its presence sounds of a technical reason. However, we will comment on its profound role with binding after defining interpretations (Definition 1.2.2.2).

iBAs can be viewed as generalized Friedman's Prestructure in [46]. And an special case of iBAs is worked out by Aczel [5]. However, their formalizations only considered λ-abstraction operator, and he does not generalize it to more general BOs as the present thesis does.

Unlike the case of eBAs, we can not go straight to interpretations of binding terms in iBAs. The reason for this comes from how to interpretate function terms. For example, intuitively we know that α-convertible terms in Lambda Calculus, which are different from each other by their bound variables, have the same denotation. This should be true for binding terms in general as well. But we deliberatively omit this kind of technical preparation, and postpone it to Chapter 5 (see Theorem 5.2.2).

Definition 1.2.2.2 (interpretation in an iBA): *Let $\vec{\psi}$ be a family of functions $\psi_m : FV_m \to B_m$ $(m \geq 0)$. Then, for $t \in T$ and $ft \in FT_m$ $(m \in Nat)$, let $\{\vec{x}\} \subseteq V$ such that $|\vec{x}| = k$, $Free(t) \cap V \subseteq \{\vec{x}\}$ and $Free(ft) \cap V \subseteq \{\vec{x}\}$[4], an interpretation $\mathcal{B}_{\vec{x}}$ over $\mathbf{B}$ under the function environments $\vec{\psi}$ on BT i.e. $\mathcal{B}_{\vec{x}}[\![t]\!]_{\vec{\psi}}$ and $\mathcal{B}_{\vec{x}}[\![ft]\!]_{\vec{\psi}}$ where $\mathcal{B}$ is a functional of $V^* \to (BT \to (Env \to \vec{B}))$, is inductively defined by:*

1. $\mathcal{B}_{\vec{x}}[\![x]\!]_{\vec{\psi}} =_{df} pr_{|\vec{x}|,i}$, *for* $x = \vec{x}(i)$;

2. $\mathcal{B}_{\vec{x}}[\![f(\vec{t})]\!]_{\vec{\psi}} =_{df} \psi_n(f) \circ_{n,k} < \mathcal{B}_{\vec{x}}[\![\vec{t}\,]\!]_{\vec{\psi}} >$,

   where $|\vec{t}| = n$ *and* $\mathcal{B}_{\vec{x}}[\![\vec{t}\,]\!]_{\vec{\psi}}$ *is the abbreviation of list* $\mathcal{B}_{\vec{x}}[\![t_1]\!]_{\vec{\psi}}$, $\mathcal{B}_{\vec{x}}[\![t_2]\!]_{\vec{\psi}}$, ..., $\mathcal{B}_{\vec{x}}[\![t_n]\!]_{\vec{\psi}}$;

3. $\mathcal{B}_{\vec{x}}[\![\sigma(\vec{ft}, \vec{t})]\!]_{\vec{\psi}} =_{df} \sigma_k(\mathcal{B}_{\vec{x}}[\![\vec{ft}]\!]_{\vec{\psi}}, \mathcal{B}_{\vec{x}}[\![\vec{t}\,]\!]_{\vec{\psi}})$,

   where $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$ *with* $|\vec{m}| = \ell$, *and* $\mathcal{B}_{\vec{x}}[\![\vec{ft}]\!]$ *is the abbreviation of list* $\mathcal{B}_{\vec{x}}[\![ft_1]\!]$, $\mathcal{B}_{\vec{x}}[\![ft_2]\!]$, ..., $\mathcal{B}_{\vec{x}}[\![ft_\ell]\!]$;

4. $\mathcal{B}_{\vec{x}}[\![(\vec{y} : t)]\!]_{\vec{\psi}} =_{df} \mathcal{B}_{\vec{x}\,\vec{z}}[\![t\vec{\imath}[\vec{y} := \vec{z}\,]\,]\!]_{\vec{\psi}}$,

   where $\vec{t}[\vec{y} := \vec{z}\,]$ *is a simultaneous substitution,* $\vec{\imath}$ *denotes an "identity" substitution map,* z_j *is the least* $z \in V$ *such that* $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$, *and* $\vec{\bullet}\lceil_k$ *containing first* k *element means the list obtained from the first* k *elements of the list* $\vec{\bullet}$.

---

[4](a) $\{\vec{x}\}$ means the set having and only having all the elements x_j of $\vec{x}$, i.e. $\{\vec{x}\} = \{y \in V | y = \vec{x}(j)$ *for some* $j\}$.

(b) $Free(t)$ and $Free(ft)$ mean all free variables in t and ft respectively (Definition 5.1.2).

Note that we will adopt the conventional omission of identity substitution map $\vec{\imath}$ in substitutions later on; e.g. we will write $t[\vec{y} := \vec{z}]$ for $t\vec{\imath}[\vec{y} := \vec{z}]$ instead (see Section 5.1 for more on substitutions).

Similar to previous subsection, the well-definedness of $\mathcal{B}_{\vec{x}}[\![\bullet]\!]_\psi$, e.g. whether $\mathcal{B}_{\vec{x}}[\![t]\!]_\psi$ is in B_k and whether $\mathcal{B}_{\vec{x}}[\![ft]\!]_\psi$ is in B_{k+m} with $|\vec{x}| = k$, is not entirely obvious; and it is postponed to Section 5.2 (see Theorem 5.2.2). However, Friedman's work [46] as well as Aczel's [5] should be able to supply enough confidence for the time being, although they only considers the λ-abstraction operator.

In iBAs, the role of binding is represented by the index on operations, say the index k in $\sigma^{\mathbf{B}}_k$. Such indices are not random, they are (semantically) inter-related with each other by the laws presented in Definition 1.2.2.1, say *(iBA-unif) law*. We should also be aware of that iBAs are not like first order algebras. That is, each operator $\sigma \in \Sigma^{BO}$ has a group of inter-related interpretations $\{\sigma^{\mathbf{B}}_k | k \in Nat\}$ rather than just one interpretation as in first order algebras.

Analogous to Definition 1.2.1.5, under the context of iBAs we define the intensional satisfaction $\models_{iBA}$ below. The central idea is to bind free variables in all arbitrary ways. informally, two binding terms p, q are *indistinguishable* by an iBA $\mathbf{A}$ iff all possible interpretations of p and q are the same. Formally,

Definition 1.2.2.3 (satisfaction $\models_{iBA}$): $\mathbf{B} \models_{iBA} p \simeq q$ *(or* $\mathbf{B} \models p \simeq q$ *for short when a confusion can be resolved by context) iff* $\mathcal{B}_{\vec{x}}[\![p]\!]_\psi = \mathcal{B}_{\vec{x}}[\![q]\!]_\psi$ *for every* ψ, $\{\vec{x}\} \subseteq V$ *and* $(Free(p) \cup Free(q)) \cap V \subseteq \{\vec{x}\}$.

On the other hand, the intuition of binding pre-algebras can be exploited because they only involve projections $pr_{k,i}$, compositions $\circ_{m,k}$ and indexed operations σ_k. And constants are missing. Furthermore, the index k of operations σ_k gives us a clue to bring down the order of languages, i.e. from second order to first order. This is formalized as *b-clones* which are certain kind of many-sorted algebras (see Definition 5.3.2 and Fact 5.3.3). The treatment of b-clones is put aside and will be resumed in Section 5.3. But the discovery of b-clones leads us to reduce Equational Calculus $\vdash_{iBA}$ (Definition 5.5.1.1) to Equational Calculus $\vdash_{EQ}$ of many-sorted first order algebras (Definition 2.1.32). The reduction is established by discovering

two translations $tr_{\vec{x}}$ and $\underline{tr}$ between binding terms BT and the terms of b-clones (Theorem 5.5.2 and Theorem 5.5.3.13). These translations preserve derivations. Actually, the proofs of preservations of derivations exemplifies the popular belief of representations of variables by projections and vice versa.

Analogous to Subsection 1.2.1 (Definitions 1.2.1.6 and 1.2.1.7), other Indistinguishabilities can also be given, which include dependent BEs, quasi-dependent BEs and the universal BEs.

Definition 1.2.2.4 (satisfactions of $\models^d_{iBA}$, $\models^q_{iBA}$ and $\models^u_{iBA}$ in iBAs): *Let* $\mathbf{B}$ *be an iBA,* γ *range over collections of BEs, and* Δ *range over BEs.*

1. $\mathbf{B} \models^d_{iBA} \gamma \mapsto \Delta$ *(or* $\mathbf{B} \models_{iBA} \gamma \mapsto \Delta$ *even* $\mathbf{B} \models \gamma \mapsto \Delta$*) iff* $\mathbf{B} \models \Delta'$ *for every* $\Delta' \in \gamma$ *implies* $\mathbf{B} \models \Delta$*;*

2. $\mathbf{B} \models^q_{iBA} \gamma \hookrightarrow \Delta$ *(or* $\mathbf{B} \models_{iBA} \gamma \hookrightarrow \Delta$ *even* $\mathbf{B} \models \gamma \hookrightarrow \Delta$*) iff for each* $\{\vec{x}\} \subseteq V$ *and* $\{\vec{x}\} \supseteq \bigcup_{p \simeq q \in \gamma \cup \{\Delta\}} (Free(p) \cup Free(q))$*, if* $\mathcal{B}_{\vec{x}}[\![p']\!]_\psi = \mathcal{B}_{\vec{x}}[\![q']\!]_\psi$ *for every* $p' \simeq q' \in \gamma$ *then* $\mathcal{B}_{\vec{x}}[\![p]\!]_\psi = \mathcal{B}_{\vec{x}}[\![q]\!]_\psi$ *where* $p \simeq q = \Delta$*;*

3. $\mathbf{B} \models^u_{iBA} \Omega$ *(or* $\mathbf{B} \models_{iBA} \Omega$ *even* $\mathbf{B} \models \Omega$*) iff* $\mathbf{B} \models \gamma^{(m)} \hookrightarrow \Delta^{(m)}$ *for every* $m \in M$ *implies* $\mathbf{B} \models \gamma \hookrightarrow \Delta$*.*

1.2.3 Core of this thesis – calculi $\dot{\vdash}_{eBA}$ and $\vdash_{iBA}$

For $\dot{\vdash}_{eBA}$ and $\vdash_{iBA}$, we shall obtain soundness and completeness theorems. Since Birkhoff's approach (extensional approach) does not work well as expected, we have to seek an alternative. The one we present is in Friedman's approach (intensional approach), which reduces Equational Calculus $\vdash_{iBA}$ to the one of many-sorted algebras through b-clones. The remedy for Birkhoff's approach is also discovered. From a set-theoretic (or model-theoretic) point of view, the remedy is more interesting. Nevertheless, the core of the present thesis composes of two parts.

The first part is Chapter 3. It follows Birkhoff's method in first order algebras but under the context of binding, and achieve a similar result as Birkhoff's Theorems in (many-sorted) first order algebras. Due to the fact that function spaces are

carriers in an eBA, there are two concepts instead of one corresponding to subalgebras in first order algebras, say *sub-eBAs* (Definition 3.1.1) and *perfect sub-eBAs* (Definition 3.1.2). The difference between sub-eBAs and perfect sub-eBAs is Extensionality. Extensionality also plays a role in extensional Binding Congruences (or eBCs) and binding kernels of eBHs (which are eBCs), see Definition 3.7.1 and Definition 3.7.14 respectively. Most significantly, Extensionality is the essence of Admissibility, where the "admissible" concept is very important because the diagram, see Figure 1-1, commutes iff ζ is admissible. Because the admissible condition is necessary and sufficient to have commutativity, we may think that *Admissible Completeness* (or completeness for Admissible BEs) of calculus $\dot{\vdash}_{eBA}$, is the best result we have in Birkhoff's approach (see Corollary 3.10.11 and Section 3.13). Directions for a possible improvement is given in Section 3.13.

About $\dot{\vdash}_{eBA}$ or simply $\dot{\vdash}$ (see Definition 3.10.9), it is almost the same as $\vdash_{iBA}$ (see Theorem 1.2.3.1 below or Definition 5.5.1.1) except for the substitution rule, where the substitution rule of $\vdash_{iBA}$ is only applicable if the involved families of substitution functions are limited to *functional* ones in contrast with arbitrary ones in $\dot{\vdash}_{eBA}$.

The second part is Chapter 5. It is to capture the equality of iBAs. The central idea is to relate this to the corresponding equality of many-sorted first order algebras. The connections between them are established

(i) by discovering two translations between binding terms and the terms of b-clones, and

(ii) by verifying that the two translations preserve the equalities and the derivations between iBAs and b-clones.

Hence, the soundness and completeness of $\vdash_{iBA}$ follows from their counterparts of $\vdash_{EQ}$ in many-sorted first order algebras. We now present the calculus $\vdash_{iBA}$, and state its completeness.

In what follows, Γ_b is assumed to be a set of BEs, i.e. each element in Γ_b is of form $p \simeq q$ such that either $p, q \in T$ or $p, q \in FT_k$ for some $k \in Nat$, where index b of Γ_b means binding (however, index b is often omitted for simplicity). We say that a family of substitution functions $\vec{\varrho}$ is *functional* iff for all $m \geq 0$,

$\forall f \in FV_m.Free(\vec{\varrho}(f)) \cap V = \emptyset$ and $\forall x \in V.\vec{\varrho}(x) \in V$ (see Section 5.1 for more on substitutions). Then, Equational Calculus $\vdash_{iBA}$ is as follows.

Theorem 1.2.3.1 (Theorem 5.6.1.1 and Theorem 5.6.2.2): *The following calculus $\vdash_{iBA}$, defined in the judgement form of*

$$\Gamma \vdash_{iBA} p \simeq q \text{ or simply } \Gamma \vdash p \simeq q,$$

is sound and complete; where Γ is a set of BEs, and $\vdash_{iBA}$ has (a) Identity rule, (b) Reflectivity rule, (c) Symmetricity rule, (d) Transitivity rule, (e) Weakening rule, and (f) cut rule (or Modus Ponens) with extra rules in the following:

1. *(α)*

   $$\vdash_{iBA} \langle \vec{x} : t \rangle \simeq \langle \vec{y} : t[\vec{x} := \vec{y}] \rangle,$$

   where $t \in T$, $\{\vec{x}\} \subseteq V$, $\{\vec{y}\} \subseteq V$, $y_j \in V$ and $y_j \notin (Free(t) - \{\vec{x}\}) \cup \{y_1, y_2, ..., y_{j-1}\}$;

2. *(ξ)*

   $$\frac{\Gamma \vdash_{iBA} t \simeq u}{\Gamma \vdash_{iBA} \langle \vec{x} : t \rangle \simeq \langle \vec{x} : u \rangle},$$

   where $t, u \in T$ and $x_j \in V$ ($1 \leq j \leq |\vec{x}|$) and $x_i \neq x_j$ ($i \neq j$);

3. *(ξ^{-1})*

   $$\frac{\Gamma \vdash_{iBA} \langle \vec{y} : t \rangle \simeq \langle \vec{z} : u \rangle}{\Gamma \vdash_{iBA} t[\vec{y} := \vec{x}] \simeq u[\vec{z} := \vec{x}]},$$

   where $\{\vec{x}\} \cap Free(\langle \vec{y} : t \rangle) = \emptyset$ and $\{\vec{x}\} \cap Free(\langle \vec{z} : u \rangle) = \emptyset$;

4. *(func-sub)*

   $$\frac{\Gamma \vdash_{iBA} p \simeq q}{\Gamma \vdash_{iBA} p\vec{\varrho} \simeq q\vec{\varrho}},$$

   where $\vec{\varrho}$ is a functional family of substitution functions;

5. *(cmp-1)*

   $$\frac{\Gamma \vdash_{iBA} t_1 \simeq u_1, t_2 \simeq u_2, ..., t_m \simeq u_m}{\Gamma \vdash_{iBA} f(t_1, t_2, ..., t_m) \simeq f(u_1, u_2, ..., u_m)},$$

   where $f \in FV_m$, $t_j, u_j \in T$ ($1 \leq j \leq m$), and $\Gamma \vdash_{iBA} t_1 \simeq u_1, t_2 \simeq u_2, ..., t_m \simeq u_m$ is the abbreviation of list $\Gamma \vdash_{iBA} t_1 \simeq u_1$, $\Gamma \vdash_{iBA} t_2 \simeq u_2$, ..., $\Gamma \vdash_{iBA} t_m \simeq u_m$;

6. (cmp-2)

$$\frac{\Gamma \vdash_{iBA} ft_1 \simeq fu_1, ft_2 \simeq fu_2, ..., ft_\ell \simeq fu_\ell, t_1 \simeq u_1, t_2 \simeq u_2, ..., t_n \simeq u_n}{\Gamma \vdash_{iBA} \sigma(ft_1, ft_2, ..., ft_\ell, t_1, t_2, ..., t_n) \simeq \sigma(fu_1, fu_2, ..., fu_\ell, u_1, u_2, ..., u_n)}$$

where $\sigma \in \Sigma_{<\vec{m},n>}$ *with* $|\vec{m}| = \ell$, $ft_i, fu_i \in FT_{m_i}$ $(1 \leq i \leq \ell)$ *and* $t_j, u_j \in T$ $(1 \leq j \leq n)$, *and* $\Gamma \vdash_{iBA} ft_1 \simeq fu_1, ft_2 \simeq fu_2, ..., ft_\ell \simeq fu_\ell, t_1 \simeq u_1, t_2 \simeq u_2, ..., t_n \simeq u_n$ *is the abbreviation of list* $\Gamma \vdash_{iBA} ft_1 \simeq fu_1$, $\Gamma \vdash_{iBA} t_2 \simeq fu_2$, ..., $\Gamma \vdash_{iBA} ft_\ell \simeq fu_\ell$, $\Gamma \vdash_{iBA} t_1 \simeq u_1, t_2 \simeq u_2, ..., t_n \simeq u_n$.

You should notice that the absence of β-conversion-like axioms in $\vdash_{iBA}$. Also, there is one interesting thing about the completeness of $\vdash_{iBA}$, i.e. (ξ^{-1}) is not needed when the given Γ does not involve function terms (see Remark 5.6.2.3). The simultaneous presence of (ξ) and (ξ^{-1}) shows an important phenomenon in programming. That is, when we declare a procedure, we break off the binding primitives and write down the body of the procedure; and when we call the procedure, binding primitives are put back and the procedure is used as an intensional object.

The equivalence between $\models_{eBA}$ and $\models_{iBA}$ is established for all semi-closed binding terms, i.e. binding terms without free ordinary variables in Section 11.1. The relation between $\dot{\vdash}_{eBA}$ and $\vdash_{iBA}$ is characterized as the equivalence between $\Gamma \dot{\vdash}_{eBA} p \simeq q$ and $cl_\alpha(\Gamma) \vdash_{iBA} p \simeq q$, where $cl_\alpha(\Gamma)$ is the closure of Γ under substitutions and α-conversions. Finally, the equivalence between $\models_{eBA}$ and $\dot{\models}_{eBA}$ is established for eBAs with up to cardinal $\aleph_0$ of ordinary carriers. Possible improvements of this result to large cardinals ($\aleph > \aleph_0$) is discussed in Section 3.13 (see [155,82,80] for references of large cardinals).

So far, we only show the calculi for pure BEs. From here, we can extend these calculi to include dBEs, qBEs and uBEs; and we obtain calculi $\dot{\vdash}^d_{eBA}$, $\vdash^d_{iBA}$, $\dot{\vdash}^q_{eBA}$, $\vdash^q_{iBA}$, and $\dot{\vdash}^u_{eBA}$ $\vdash^u_{iBA}$ respectively. Applications of these calculi are provided in Part V, which consists of Chapter 8, Chapter 9 and Chapter 10, that cover a complete equationalization of First Order Logic through uBEs, equivalences between Lambda Calculus and Combinatory Logic through dBEs and equationalizations of the weak bisimulation and strong bisimulation of CCS with data-dependency through dBEs.

1.3 Review with relevant works

We have shown that Binding Algebras (or eBAs) have close relationship with Recursion Theory in higher types. People interested in such connections can further pursue them along the line of hierarchies, see Hinman's book [72] for recursion-theoretic hierarchies. In what follows, we are going to show the other relations of the present Framework (FBO) with other relevant work.

1.3.1 Equational theories and their feasibility

Equational theories are quite powerful, e.g. Lambda Calculus and Combinatory Logic can be presented equationally. More strikingly, Matijasevic shows that recursively enumerable predicates are *Diophantine* [102] (this gave a negative answer to Hilbert's 10th problem [35]), which connects a constructive part of First Order Logic with certain equational theory (Number Theory). This can also be viewed as showing the expressive power of equational theories. Along this line, recent work on Diophantine equations emphasizes on search of natural and manageable presentations [103,81]. Others like Bergstra and Tucker demonstrate the expressive powers of equations and conditional (or quasi-dependent) equations and their differences in [14]. It is also worth to comment that their work can be regarded as showing the power of binding when you see the hidden enrichments as certain binding over signatures.

With regard to implementations of equational theories, i.e. extracting algorithms from equational theories, we just need to point out the followings.

(i) Kreisel and Tait, Kleene, Gödel show that recursive functions can be defined through collections of equations [94,85,48], which involves type-2 functionals (BOs are their syntax names);

(ii) Strong gives a bi-transformation from recursive equations to flow charts [148], which is nice from a realizational view; and

(iii) Wang and Ershov show that recursive functions are flowchart computable [163,39].

There are some people working on the issue presentations combined with implementations. For example, Hayashi [64] and Paulin-Mohring [117] pursue this issue under Intuitionistic Logic with Realizability and under Calculus of Constructions [29] respectively. Also, O'Donnell take equational logic direcctly as a programming language [115].

1.3.2 Combinatory Logic

There might be an alternative approach to ours along the idea of Combinatory Logic (CL) instead of generalizing binding from Lambda Calculus. On this sense, the present thesis can be regarded as a companion to Klop's thesis [89]. Also, Bunder provides a higher order predicate calculus through Combinatory Logic in [21]; Cylindric Algebras can be viewed as taking a similar philosophy as Combinatory Logic by introducing closed binding (see subsection 1.3.5); and Tarski and Givant give a formalization of Set Theory with similar idea of Combinatory Logic [157]. However, whether Combinatory Logic's approach can be more *algebraically generalized* to more general cases, say like the Framework for BOs (FBO) is to be seen. A simple example, which is picked up from Barendregt's book [7], can illuminate this point. In general, $CL \vdash t \simeq u$ does not necessarily imply $CL \vdash \lambda^* x.t \simeq \lambda^* x.u$ (where λ^* is the counterpart of λ-abstraction in Combinatory Logic to eliminate the need for binding), say $CL \vdash \mathbf{I}x \simeq x$, but $CL \nvdash \mathbf{S(KI)I} \simeq \mathbf{I}$. Also, This example explains why taking bound variables as meta-holes (to be filled in) can not work well. Nevertheless, Curry extends CL by a finite set A_β of closed equations such that $CL + A_\beta$ is equivalent to Lambda Calculus. A better understanding of A_β will certainly be helpful in algebraically structuralizing Combinatory Logic, and in eliminating the need of variables for substitutions. Recently, Bunder, Hindley and Seldin made a step toward this direction of understanding A_β [22].

For people who are interested in the debate between Lambda Calculus's approach

and Combinatory Logic's approach, Curry's paper [31] and Scott's paper [136] are good ones.

1.3.3 Universal Algebra

We compare eBAs with first order algebras as follows. Let us suppose that $\Sigma^{BO}_{<\vec{m},n>} = \emptyset$ if $\vec{m} \neq \varepsilon$. So, this Σ^{BO} can be viewed as an first order signature, and an extensional Σ^{BO}-algebra $\mathbf{A}$ (i.e. an eBA) consists of

(a) $\mathcal{F} =< A, \{\mathcal{F}_m(A) | m \in Nat\} >$ and

(b) for each $\sigma \in \Sigma^{BO}_{<\varepsilon,n>}$ (of every $n \in Nat$) its interpretation $\mathcal{A}_\sigma$ is uniform over $\mathcal{F}$.

Apparently, for every $m \in Nat$ if $\mathcal{F}_m(A)$ is the full function space from A^m to A, then any arbitrary interpretation $\mathcal{A}_\sigma$ of each $\sigma \in \Sigma^{BO}_{<\varepsilon,n>}$ is uniform over $\mathcal{F}$. Thus, eBAs are more general than first order algebras. In other words, the latter is a special case of the former.

However, if we do not require the existence of eBHs between these (first order) eBAs, then $\mathcal{F}_m$ can be the full function space from A^m to A; otherwise, we have to consider the least function space from A^m to A closed under constant functions, projections and compositions. The reason for this is better explained by a concrete example given after Definition 3.7.14. That is, there is no eBH ζ satisfying compositionality if the example takes the full function spaces as its function carriers. More specifically, there is a function g_0 such that no function can be its image of ζ:

$$\zeta(g_0)(a) = \zeta(g_0)(\zeta(a)) = \zeta(g_0(a)) = \zeta(a) = a,$$
$$\zeta(g_0)(a) = \zeta(g_0)(\zeta(c)) = \zeta(g_0(c)) = \zeta(b) = b.$$

Also, an eBH is no longer an arbitrary family of functionals which only preserves compositionality (functionality) as a homomorphism in first order algebras seems to be. It has to satisfy some minimal requirements, like preservations of constant functions and projections.

Categorically, we fix a Σ^{BO} such that $\Sigma^{BO}_{<\vec{m},n>} = \emptyset$ if $\vec{m} \neq \varepsilon$, let $Alg(\Sigma^{BO})$ be the class of first order algebras with signature Σ^{BO} (forgetting binding), and

$eBAlg(\Sigma^{BO})$ be the class of eBAs with signature Σ^{BO}. Then, $Alg(\Sigma^{BO})$ and $eBAlg(\Sigma^{BO})$ are categories. There is a (embedding) functor $\eta_{\Sigma^{BO}} : Alg(\Sigma^{BO}) \rightarrow eBAlg(\Sigma^{BO})$ such that for $\mathbf{D} \in Alg(\Sigma^{BO})$ ($\mathbf{D} =< D, \mathcal{D} >$), $\eta_{\Sigma^{BO}}(\mathbf{D}) = \mathbf{A} \in eBAlg(\Sigma^{BO})$ where

(1) $D = A$ and $\mathcal{D} = \mathcal{A}$,

(2) for each $m \in Nat$, $\mathcal{F}_m^{\mathbf{A}}$ contains

(2.a) $C_{m,a}$ ($a \in D$) and

(2.b) $\pi_{m,i}$ ($1 \leq i \leq m$)

and

(3) the family $\{\mathcal{F}_m^{\mathbf{A}} | m \in Nat\}$ is the least family closed under compositions of constant functions (2.a) and projection functions (2.b).

For a morphism $\iota : \mathbf{D}^1 \rightarrow \mathbf{D}^2$, $\eta_{\Sigma^{BO}}(\iota)$ is a morphism $\eta_{\Sigma^{BO}}(\eta) : \mathbf{A}^1 \rightarrow \mathbf{A}^2$ where $\eta_{\Sigma^{BO}}(\iota)$ preserves constant functions and projection functions.

1.3.4 Variable Binding Term Operators

As far as I know, BOs have not been well studied in an algebraic form regardless of whether the study is systematic or not. The work relating to Binding Operators (BOs) in literature is mostly restricted to *Variable Binding Term Operators* (VBTOs). In general, a signature Σ^{VBTO} is indexed by elements in $\bigcup_{m \in Nat}(S^m \times S)^* \times S$, where $S = Sort(\Sigma^{VBTO})$. That is, BOs are technically more primitive than VBTOs, and $\Sigma^{VBTO} \subset \Sigma^{BO}$ strictly. For example, $\forall$, λ and **in**-α are VBTOs and **let** is not a VBTO but a BO. Therefore, the work [1,2,63,34,30,125], only dealing with VBTOs, is a kind of special case in the Framework for BOs. Also, since CC-like calculi (CC is short for Calculus of Constructions) [61,29] involves only VBTO's operations (say λ, sum Σ and product Π), the same as First Order Logic, equation-

alization of them seems to be feasible in the present Framework for BOs[5]. On the other hand, whether the difference between VBTOs and BOs is significant is yet to be seen. We will touch on this point in Subsection 1.3.8 later.

1.3.5 Cylindric Algebras

Closely to eBAs, there is work on Cylindric Algebras, Monadic Algebras and Polyadic Algebras, see joint work of Henkin, Monk and Tarski on Cylindric Algebras [66,67], Halmos' work on Monadic Algebras and Polyadic Algebras [58,59,67]. This kind of work can simply be put in the spirit of taking algebras out of First Order Logic. So, the essential part is the treatment of the quantifiers. In turn, the feature of the binding being dealt with is more or less the kind of VBTO's binding.

Technically, Cylindric Algebras are a special kind of Binding Algebras, namely, *closed* Binding Algebras. To see this relation more explicitly, we observe that all terms appearing in Cylindric Algebras are closed function terms in Binding Algebras, i.e. they are all bound by $\vec{V}$ (list of all variables in V). Since every $\vec{V}$ is associated to an ordinal α, Cylindric Algebras are *α-closed* Binding Algebras. The ordinal α can be viewed as the dimension of Cylindric Algebras. In particular, let α be ω (the least limit ordinal), then we would have the following. The kth cylindrifier c_k over a function term $\langle \vec{V} : t \rangle$, say $c_k(\langle \vec{V} : t \rangle)$, behaves like an existential quantifier, say $\exists_k^{cyl}$, over the corresponding function term like $\exists_k^{cyl}(\langle \vec{V} : t \rangle)$. If we imagine that $\langle \vec{V} : t \rangle$ is an object in ω-dimension geometry, $\exists_k^{cyl}$ has the behaviour of fixing other dimensions and only allowing the kth dimension to change. This can also be viewed as a cylinder along the kth axis in the ω-dimension geometry, which is probably the origin of the name Cylindric Algebras. (k,j)-diagonal element $d_{k,j}$ can be expressed by a qBE, i.e.

$$\mathrm{o}^{cyl}(\langle \vec{V} : x_k \rangle, ft) \simeq \mathrm{o}^{cyl}(\langle \vec{V} : x_j \rangle, ft) \hookrightarrow ft \simeq d_{k,j},$$

---

[5]In fact, a complete axiomatization of Calculus of Constructions in FBO is done in [153] after completing this thesis.

where o^{cyl} is a BO for the composition functional associated with Cylindric Algebra, $ft = \langle \vec{V} : t \rangle$, $\langle \vec{V} : x_k \rangle$ and $\langle \vec{V} : x_j \rangle$ are kth and jth projection functions respectively.

1.3.6 Monadic Algebras and Polyadic Algebras

Monadic Algebras and Polyadic Algebras (see [58,59,67]) are alternatives to Cylindric Algebras. Monadic Algebras are special cases of Polyadic Algebras. Each of them is a special kind of Binding Algebras as well. To illustrate our point, let us look at Monadic Algebras first. A Monadic Algebra is a Boolean Algebra extended to include one "existential" quantifier $\exists^{mon}$, where $\exists^{mon}(\langle x : t \rangle)$ can be viewed as short for $\langle x : \exists_1(\langle x : t \rangle)\rangle$ and $\exists_1$ is the usual unary existential quantifier and the index 1 is to emphasize the arity. (1.3.6.b) For Polyadic Algebras, $\exists^{pol}_{\vec{x}}(\langle \vec{x} : t \rangle)$ $(\{\vec{x}\} \subseteq V)$ can be thought as $\langle \vec{x} : \exists_{|\vec{x}|}(\langle \vec{x} : t \rangle)\rangle$, where $\exists_k$ is the usual k-ary existential quantifier with $k = |\vec{x}\,|$.

The relationship between Cylindric Algebras and Polyadic Algebras can be found in [67]. There are other ways of algebraizing First Order Logic, say Projective Algebras (a special case of Cylindric Algebras) and Relation Algebras (another special case of Cylindric Algebras). see [67].

The above work can be viewed as extending first order algebras so that the resulting algebras can have the expressive power of First Order Logic, just like Boolean Algebra shares the same expressive power with Propositional Logic. In the present thesis, we seem to follows the same line by introducing binding primitives but with further extensions from pure equational calculus to dependent equational (equational implications), quasi-dependent equational (conditional equations or quasi-equations), universal equational calculi. The final output is very significant, i.e. we provide a complete equational reasoning for First Order Logic. Therefore, we show that there is no significant difference between equational reasoning and logical reasoning. This result seems not to be easily derivable from the other approaches demonstrated so far.

1.3.7 Logicals Relation vs Admissibility

Logical Relations have been used to characterize λ-definability in one way or another, see [154,76,46,121,142,143,144] for examples of using ordinary Logical Relations. [110] and [112] are references to second order Logical Relations and partial Logical Relations, respectively. Another way to look at Logical Relations is to view them as morphisms (correspondences) between two structures (models). However, full type hierarchy and their corresponding Logical Relations are not necessary in the present thesis. Therefore, our discussion on Logical Relations is of up to rank 2, which should not be difficult to extend to full type hierarchy.

In [121], Plotkin uses Logical Relations (or LR) to show that a functional satisfying every LR is λ-*definable* provided that the base domain is infinite. Since he let $A^{(\sigma,\tau)}$ be the full function space from $A^{(\sigma)}$ to $A^{(\tau)}$, his result can be classified into extensional approach.

In a contrast, Friedman does not give any relationship between $A^{(\sigma,\tau)}$ and $A^{(\sigma)}$, $A^{(\tau)}$ in [46]. Statman puts this in another way and says that Friedman only treats closed terms [143]. Therefore, Friedman's work [46] can be classified into intensional approach. Accordingly, his completeness corresponds to completeness of $\vdash_{iBA}$ but the latter is a generalized result of the former. His surjective partial homomorphisms are his terminology of Logical Relations (by omitting partiality).

With regard to Logical Relations in eBAs, we say a $\zeta : \mathbf{A} \rightarrow \mathbf{A}'$ is *logical* iff for each $m \in Nat$ and every pair of $g \in \mathcal{F}_m^{\mathbf{A}}$ and $h \in \mathcal{F}_m^{\mathbf{A}'}$, $\forall \vec{a} \in A^m.\zeta(g(\vec{a})) = h(\zeta(\vec{a}))$ implies $\zeta(g) = h$. Informally, this means that the image of a logical ζ has *enough points* in its base not only

(i) to identify the functions within its image function space (i.e. $\zeta(\mathcal{F}^{\mathbf{A}})$) but also

(ii) to identify functions inside the full function spaces of $\mathbf{A}'$ (i.e. $\mathcal{F}^{\mathbf{A}'}$).

However, an admissible ζ can only do the former (i) not necessarily the latter (ii). Therefore, a logical eBH must be admissible (see Section 3.12).

1.3.8 Natural Languages

The Framework for Binding Operators (FBO) is very powerful as demonstrated above, although the language is of restricted second order. People might wonder whether it is over powerful since none of the work related to binding has treated binding in so general a form. Therefore, it is worthwhile to point out a potential usage of the extra power.

Since a framework of Variable Binding Term Operators or the like can not treat first order objects and ordinary objects at the same time, the framework for Variable Binding Term Operators suffer a set-back on dealing with Natural Languages. That is, it may not be a proper framework for Natural Languages. As we know, dealing with Natural Languages is not an easy task. To show the difficulty, we just need to remind you of many semantic paradoxes, say the *Liar paradox* (see [100]). Compared with framework for Variable Binding Term Operators, The present Framework for BOs does not have this disadvantage. The aspect of treating first order objects and ordinary objects as the same is being considered as a very crucial point in providing a framework to deal with Natural Languages, see Barwise's argument in Situation Theory [11] for instance. Therefore, I am quite optimistic in predicting that the present Framework for BOs would have a bright future related to Natural Languages.

The work of Plotkin's [124] on the illative theory of relations can be regarded as applying the work presented in this thesis to Situation Theory. Also, the work of Fenstad and Benthem with others [45] on equational presentations of situation in natural languages, and Westerstahl's work on quantifiers of formal and natural languages [165] can be regarded steps toward this direction.

1.4 Future developments

I have to admit that the work presented in this thesis is just an opening for a wide area of research. For example, the issues related to Equational Logic discussed by Tarski, Taylor and Huet in [156,158,78] can be reconsidered in the present Framework for Binding Operators. However, we want to justify the claim by tentatively foreseeing some other developments briefly as follows.

1.4.1 Higher types vs higher orders

Pure equational theories are rather weak since their languages only contain assertions of the form $p \simeq q$. For instance, three well-known natural number systems (Presburger Arithmetic, Peano Arithmetic, Robinson Arithmetic) are not totally presented in equational forms. Naturally, one might wish to unite First Order Logic with equational theories. For such a "union", it would make more sense if we can provide a framework to contain First Order Logic into an algebraic system. Since quantifiers, say existential quantifier, are of non-first-order objects, the algebraic system can not be of first order. In the literature, it is a common practice to introduce higher types instead of raising the order of languages to resolve the problem of non-first-order objects. The Framework for Binding Operators can be regarded as trying (to lead) the other alternative.

The issue of extending languages to higher orders is quite delicate. Recently, Shapiro argued strongly in favour of extensions of languages from first order to second order [138]. One of his reason is *categoricity* (intended interpretations of mathematical theories). He proposes a "lower bound" for such an extension, and named it as *diminished* second order language. He also point out that such a lower bound language was implicitly proposed and called "applied" first order language by Church [27]. It turns out that Shapiro's diminished second order language coincides with the present Framework for Binding Operators.

Although there may not be an essential difference between higher types and higher orders, it is still a good idea to separate them for the conceptual reason. However, further introduction of the two kinds of higher types and higher orders into the Framework for Binding Operators is a worthy subject to pursue, and to see how the two kinds interact with each other (see Chapter 7 for many-sorted FBO).

1.4.2 2-category, eBAs and iBAs

Besides introducing higher types and higher orders into the present Framework for Binding Operators, there is another area (which might be fruitful), i.e. to connect the work here with Category Theory[6] [97]. This seems to be not trivial, since the usual category is first order in nature. Also, it is nice to see how eBAs and iBAs are presented in categorical terms, and the advantage of doing so seems to be that both eBAs and iBAs can be treated in one semantic framework. In turn, it would provide one semantic base to discuss relationship between eBAs and iBAs instead of many. We have made a few effort in connecting eBAs and iBAs. Corollary 5.0.4 confirms that the difference between eBAs and iBAs is indeed Extensionality, and Theorem 5.0.5 further shows that interpretations of eBAs (Definition 1.2.1.4) and interpretations of iBAs (Definition 1.2.2.2) are consistent with each others (see Section 11.1). We also made an attempt on connecting eBAs with iBAs through β-algebras (Binding Algebras satisfying β-conversion) by following Koymans' approach in [93] (see Subsection 9.5.4). However, this attempt seems unsuccessful, and it further strengthens the desire of linking Binding Algebras with Category Theory directly. Since Cartesian Closed Categories are closely related to higher types, it seems more appropriate to consider 2-categories as the semantic framework for the Framework for Binding Operators, along the introduction of higher types (see [97, 84,137] for some references of 2-categories). The reason for this is as follows. Recall

[6]The up and down of the movement in (classical) abstract algebras with Category Theory is excellently surveyed by Maclane in [98].

that a 2-category $\mathcal{C}$ can be viewed as a CAT-enriched category (see [97]); i.e. it contains

(a) a class $Ob(\mathcal{C})$ of objects,

(b) a category $\mathcal{C}(a,b)$ for a pair of $a,b \in Ob(\mathcal{C})$, and

(c) compositionals with the obvious laws.

An eBA $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$ is a special kind of 2-categories, i.e. a special $\mathcal{C}$, such that

(a') $Ob(\mathcal{C})$ contains A and its products (closed under finite limits),

(b') $C(A^m, A)$ is $\mathcal{F}_m$ with $m \in Nat$ (Discretive Category) where it contains:

1. constant functions (equalizers of arbitrary pair of homos),

2. projection functions (associated with limits)

(c') compositionals correspond to $\{\mathcal{F}_m | m \in Nat\}$ is closed under compositions (natural inheritance).

1.4.3 Lambda Calculus vs Linear Lambda Calculus

Referring to (i) β-conversions of Subsection 1.1.4, i.e.

$$(\beta) \quad \mathbf{app}(\lambda x.f(x), y) \simeq f(y),$$

and to (ii) the substitution rule of $\vdash_{iBA}$ in Subsection 1.2.2, we understand that f is only allowed to be substituted by a semi-closed function term[7]. If we consider that β-conversion is the same as the above then we are only considering linear Lambda Calculus, i.e. the terms involved must have at most one free ordinary variable. This

---

[7]This does not happen to admissible $\vdash_{eBA}$. In other words, the proof power of $\vdash_{iBA}$ is weaker than the one of $\vdash_{eBA}$ (see Section 11.2 for precise characterization between these two calculi).

tends to be over-restrictive. However, whether this is so is yet to been seen. It is simply because of the following. If we accept the equivalence between λ-definable functions and recursive functions, then we understand that linear Lambda Calculus has the full power of Lambda Calculus as the result of J. Robinson in [126,127]. Since this result is based on standard model of Natural Numbers, we wonder whether linear Lambda Calculus preserves the full power of Lambda Calculus in general.

1.4.4 Generalized Full Abstraction and non-Simple Algebras

From a theoretical computer science point of view, we would like to know what is the relationship between transition systems (say Plotkin's structural operational semantics [122]) and equational systems (say denotational semantics [147]). The core of this relationship is the Full Abstraction, see [120,106,15,114,145] and many others. Also, Knobel recently shows that one can obtain intuitionistic fully abstract λ-model from an classical fully abstract one [91]. However, the above mentioned fully abstract models are corresponding to *simple* algebras by algebraic terminology, which are algebras only having trivial congruences. Whether there exists a fully abstract model which is not a simple algebra deserves our attention as well. Nevertheless, the mentioned work on this subject are not general enough from the point of view of the present Framework for Binding Operators. Hence, a generalized Full Abstraction in the Framework for Binding Operators deserves our attention. However, how to generalize operational semantics to include the cases for function terms in the Framework for Binding Operators might be an essential step toward this direction.

1.4.5 Generalized Institutions

There are many collections of "logical systems" in literature. Within each collection, logical systems are more or less equivalent to each others. Therefore, they can be viewed as notational variants of each other. Goguen and Burstall take this phenomenon into account and proposed the theory of Institutions to treat the phenomenon in [49,50]. Following them, Sannella and Tarlecki have presented a way to

develop ML programs from specifications in [130,131]. The significant point of their work is institution-independency. Also, inside one institution, there are many subtle issues involving implementation aspects of Sannella and Tarlecki's work. Farres-Casals takes some of them into his thesis work, see [41] for a reference. However, all above mentioned work are presumed on first order languages since their signatures are not of higher orders. An extension of their work to present Framework, i.e. to increase the order of languages being used, is certainly desirable.

1.5 Summary of terminology and abbreviations

In this chapter, we begin by explaining the reason why FBO is introduced, and give some reasons for an existence of such a FBO. Then, we graduately introduce binding primitives, binding signatures, binding terms, binding equations, ..., etc. Along the way, we encounter semantic problems in working out complete equational logic for FBO. Our proposal to their solutions is classified into two approaches. One is Birkhoff's (extensional), and the other is Friedman's (intentional). Relationship between these two and the work in literature has been briefly discussed. Possible improvements of our results will be discussed in Section 3.13 (Birkhoff approach) and Section 11.3 (Friedman approach). Applications of FBO will be in Chapter 8, Chapter 9 and Chapter 10.

In what follows, we summarize some of our terminology for convenience, which have their correspondences in literature in the following tables (Table 1-1, Table 1-2 and Table 1-3).

in literature	equation	equational implication conditional equation	conditional equation quasi-equation (universal) Horn clause
Chapter 2 calculi axioms	equation $\vdash_{EQ}$ $t \simeq_X u$ or Δ_X	dependent equation $\vdash_{dEQ}$ $\gamma_X \mapsto \Delta_X$	quasi-dependent equation $\vdash_{qEQ}$ $\gamma_X \hookrightarrow \Delta_X$
Chapters 3 and 4 calculi axioms	admissible BE $\dot{\vdash}_{eBA}$ $p \simeq q$ or Δ	admissible dBE $\dot{\vdash}^{d}_{eBA}$ $\gamma \mapsto \Delta$	admissible qBE $\dot{\vdash}^{q}_{eBA}$ $\gamma \hookrightarrow \Delta$
Chapters 5 and 6 calculi axioms	BE $\vdash_{iBA}$ $p \simeq q$ or Δ	dependent BE $\vdash^{d}_{iBA}$ $\gamma \mapsto \Delta$	quasi-dependent BE $\vdash^{q}_{iBA}$ $\gamma \hookrightarrow \Delta$

Table 1–1: summary of terminology

Chapter 2	terms t, u	equation $t \simeq_X u$	algebra $\mathbf{D}$	homomorphism $\iota : \mathbf{D} \to \mathbf{D}'$ $\vec{\rho}^{\#}$ and $\iota^{\#}$
Chapter 3	binding terms p, q ordinary terms t, u	Binding Equations $p \simeq q$ ordinary equation $t \simeq u$	eBA $\mathbf{A}$	eBH $\zeta : \mathbf{A} \to \mathbf{A}'$
Chapter 5	function terms ft, fu $\langle \vec{x} : t \rangle, \langle \vec{y} : u \rangle$	function equation $ft \simeq fu$ $\langle \vec{x} : t \rangle \simeq \langle \vec{y} : u \rangle)$	iBA $\mathbf{B}$	translations $tr_{\vec{x}}, \underline{tr}$

Table 1–2: summary of notations and abbreviations

Chapter 2	algebra $\mathbf{D}$	subalgebra $\mathbf{D}' \preceq \mathbf{D}$	congruence θ	kernel $ker(\iota)$	environment $\vec{\rho} = \{\rho_i \mid i \in I\}$
Chapter 3	eBA $\mathbf{A}$	sub-eBA $\mathbf{A}' \prec \mathbf{A}$ perfect sub-eBA $\mathbf{A}' \preceq \mathbf{A}$	eBC ϑ	core ∇_ζ kernel $Ker(\zeta)$	$\vec{\rho} =< \rho, \varphi >$ $\vec{\varphi} = \{\varphi_k \mid k \in Nat\}$
Chapter 5	iBA $\mathbf{B}$				$\psi = \{\psi_k \mid k \in Nat\}$

Table 1–3: comparison among first order algebras, eBAs and iBAs

Part I

Many-Sorted Algebras

We mentioned previously in Subsection 1.3.5 and in Subsection 1.2.2 that

(a) Universal Algebra is a special case of eBAs and

(b) the equational completeness of iBAs is closely related to that of many-sorted first order algebras (or simply many-sorted algebra), respectively. Therefore, the equational completeness of many-sorted algebras is a prerequisite and an introduction to the one of eBAs. Thus, Part I is intended to serve such purpose.

We proceed along Birkhoff's approach. Firstly, completeness with variable indices is established (Section 2.1). Many ideas and concepts developed in this section will be used and exploited to obtain Admissible Completeness in Chapter 3. Then, a necessary and sufficient condition for index free completeness is given (Section 2.2). Thirdly, the comleteness will be extended to include (i) dependent equations (Section 2.3) and (ii) quasi-dependent equations (Section 2.4). Fourthly, results will be extended to include the united form of equations, dependent equations and quasi-dependent equations, i.e. universal equations (Section 2.5). An overview of the results in this part is provided in Section 2.6. It may also be a good idea to read Section 2.6 before any other sections.

Chapter 2

Many-Sorted Universal Algebra

We start at many-sorted first order algebras. For the single-sorted first order algebras, you can consult [23,28,56].

Let I be a set of sorts and Σ be a signature of I-sorted algebras (to be defined). Actually Σ, indexed by $I^* \times I$, is a family of sets $\Sigma_{\kappa,i}$ of operations with the finite *rank* $< \kappa, i >$ (or the *arity*) associated with $< \kappa, i > \in I^* \times I$; where $I^0 = \{\varepsilon\}$, ε stands for the empty list, $I^{n+1} = \{i\kappa | i \in I \, and \, \kappa \in I^n\}$ or $I \times I^n$ for short and $I^* =_{df} \bigcup_{n \in Nat} I^n$ and κ range over I^*.

Let $\vec{\chi}$ be a family of collections χ_i of variables associated with $i \in I$. We assume that $|\chi_i|$ (the cardinal of χ_i) is countably infinite for each $i \in I$ and pairwisely disjoint. They disjoint with Σ.

Now, let us look at the language, so-called *terms* (or *words*) built from signature Σ and $\vec{X} \subseteq \vec{\chi}$, i.e. $X_i \subseteq \chi_i$ for $i \in I$. Throughout this chapter, we assume an arbitrarily fixed signature Σ unless we explicitly say otherwise. Also, we will abbreviate $\vec{X}$ as X when no confusion involves.

Definition 2.0.1 (terms $T(X)$): *$T(X)$ ($X \subseteq \vec{\chi}$) is the family of sets $T(X)_i$, for $i \in I$, of (first order) terms of sort i. These sets are generated by using the formation rules:*

1.

$$\frac{x \in X_i}{x \in T(X)_i}$$

2.

$$\frac{t_1 \in T(X)_{i_1}, t_2 \in T(X)_{i_2}, ..., t_m \in T(X)_{i_m}}{\sigma(t_1, t_2, ..., t_m) \in T(X)_i} \ (\sigma \in \Sigma_{i_1 i_2 ... i_m, i}).$$

Once having terms (or words), we would wonder what are their meanings. So, we need to have a certain collection of objects which may be denoted by terms. Such a collection is called an *algebra*. Formally

Definition 2.0.2 (Σ-algebra D): *A* Σ-algebra $\mathbf{D}$ *is a pair of* $\mathbf{D} = < \vec{D}, \mathcal{D} >$ *such that*

(a) $\vec{D}$ *is a family of carriers* $\{D_i | i \in I\}$;

(b) $\mathcal{D}$ *is a family of functionals from* $\Sigma_{i_1 i_2 ... i_m, i}$ *to function space* $D_{i_1} \times D_{i_2} \times ... \times D_{i_m} \rightarrow D_i$ *for every* $< i_1 i_2 ... i_m, i > \in I^* \times I$. *In other words, it assigns a function to each operation in* Σ, *i.e. for each* $\sigma \in \Sigma_{\kappa,i}$ *(or* $\Sigma_{\kappa \rightarrow i}$ *and* $\Sigma_{i_1 i_2 ... i_m, i}$ *when* $\kappa = i_1 i_2 ... i_m$*),* $\mathcal{D}(\sigma)$ *is a function from* $D_{i_1} \times D_{i_2} \times ... \times D_{i_m}$ *to* D_i, *sometimes we write it as* $\mathbf{D}_\sigma$ *or* $\sigma^{\mathbf{D}}$.

A simple example of first order algebras is first order term algebras. Let $\mathbf{T}(X)$ be $< T(X), \mathcal{T} >$ such that for each $\sigma \in \Sigma_{\kappa,i}$ and for all $t_j \in T(X)_{\kappa_j}$, $\mathcal{T}_\sigma(\vec{t}) =_{df} \sigma(\vec{t})$, where $|\vec{t}| = |\kappa|$ and $t_j = \vec{t}(j)$. Then, $\mathbf{T}(X)$ is a Σ-algebra.

With terms and algebras, we can define the meaning of a term, so-called *interpretation*, such that every term has a meaning in an algebra.

Definition 2.0.3 (interpretation $\mathcal{D}$ and environment $\vec{\rho}$): *Let* $\mathbf{D}$ *be a* Σ*-algebra. For* $X \subseteq \vec{\chi}$ *and each* environment $\vec{\rho} : X \rightarrow \mathbf{D}$ *(i.e.* $\rho_i : X_i \rightarrow D_i$ *for* $i \in I$*),* an interpretation *under environment* $\vec{\rho}$, $\mathcal{D}[\![\bullet]\!](\vec{\rho})$, *is defined inductively as below; where interpretation* $\mathcal{D}$ *can be regarded as a functional* $\mathcal{D} : T(X) \rightarrow (Env \rightarrow \mathbf{D})$ *and* Env *is the indexed function space from* X *to* $\mathbf{D}$ *by sorts:*

(a) $\mathcal{D}[\![x]\!](\vec{\rho}) =_{df} \rho_i(x)$ *for* $x \in X_i$; *and*

(b) $\mathcal{D}[\![\sigma(\vec{t})]\!](\vec{\rho}) =_{df} \sigma^{\mathbf{D}}(\mathcal{D}[\![\vec{t}]\!](\vec{\rho}))$.

Note that we adopt the convention of confusing $\mathcal{D}$ in algebra $\mathbf{D}$ with the interpretation $\mathcal{D}$ of terms in algebra $\mathbf{D}$.

With the above set-up, we can discuss the *Indistinguishabilities*, which is commonly referring to as equality, in many-sorted first-order algebras. They correspond to equations (EQs), dependent equations (dEQs) and quasi-dependent equations (qEQs) respectively. Formally

Definition 2.0.4 (satisfactions of $\models_{EQ}$, $\models_{dEQ}$ and $\models_{qEQ}$): *Let t, u, t' (with or without subscripts) be terms in $T(X)$ where $X \subseteq \vec{\chi}$. Thus,*

(a) For equation $t \simeq_X u$, $\mathbf{D} \models_{EQ} t \simeq_X u$ *(or $\mathbf{D} \models t \simeq_X u$ for simplicity) iff $(\mathbf{D}, \vec{\rho}) \models t \simeq_X u$ for every environment $\vec{\rho} : X \to \mathbf{D}$, where $(\mathbf{D}, \vec{\rho}) \models t \simeq_X u$ iff $\mathcal{D}[\![t]\!](\vec{\rho}) = \mathcal{D}[\![u]\!](\vec{\rho})$.*

(b) For dependent equation $\{t_m \simeq_X t'_m | m \in M\} \mapsto t \simeq_X t'$ *(or $\gamma_X \mapsto \Delta_X$ where γ_X is $\{t_m \simeq_X t'_m | m \in M\}$ and Δ_X is $t \simeq_X t'$), $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_X$ (or $\mathbf{D} \models \gamma_X \mapsto \Delta_X$ for short) iff $\mathbf{D} \models t_m \simeq_X t'_m$ for each $m \in M$ implies $\mathbf{D} \models t \simeq_X t'$.*

(c) For quasi-dependent equation $\gamma_X \hookrightarrow \Delta_X$, $\mathbf{D} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$ (or $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$ as an abbreviation) iff for every environment $\vec{\rho}$, if for each $t_m \simeq_X t'_m \in \gamma_X$ $(\mathbf{D}, \vec{\rho}) \models t_m \simeq_X t'_m$ then $(\mathbf{D}, \vec{\rho}) \models t \simeq_X t'$.

Moreover, we can unify the above three forms into one, i.e.

$$\{\gamma_X^{(m)} \hookrightarrow \Delta_X^{(m)} | m \in M\} \mapsto (\gamma_X \hookrightarrow \Delta_X)$$

such that when $M = \emptyset$, it becomes a quasi-dependent equation; and when all γ's are $\emptyset$, it becomes a dependent equation; when $M = \emptyset$ and $\gamma = \emptyset$, it becomes an equation. Therefore, we call it a *universal* equation and simply written as Ω_X or even Ω. Formally,

Definition 2.0.5 (satisfaction $\models_{uEQ}$): *For* a universal equation Ω, $\mathbf{D} \models_{uEQ} \Omega$ *(or $\mathbf{D} \models \Omega$ when no confusion involves) iff $\mathbf{D} \models_{qEQ} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\}$ for every $m \in M$ implies $\mathbf{D} \models_{qEQ} \gamma \hookrightarrow \Delta$.*

Note that for simplicity we have omitted variable indices in the above definition. We should mention that *equational implications* in [158] and *conditional equations*

in [52] (or *quasi-equations* in [99] or *universal Horn clauses* in [23,116]) correspond to dependent equations and quasi-dependent equations in the present thesis respectively. Our idea is to unify unconnected terminology together.

In what follows in this chapter, Section 2.1 deals with equations with variable index. The elimination of variable index from deduction system $\vdash_{EQ}$ is attacked in Section 2.2. The dependent equations and quasi-dependent equations are the subjects of Section 2.3 and Section 2.4 respectively. Section 2.5 is devoted to the deduction system $\vdash_{uEQ}$ (of the universal equations), which is the first appearance of such kind, I believe. The last section (Section 2.6) will briefly provide some relations between the work done in this chapter and the work presented in literature.

2.1 Equational Logic $\vdash_{EQ}$ with variable indices

Let us briefly summarize Birkhoff's method of proving soundness and completeness of an equational calculus in single-sorted situation.

Basically, Birkhoff is interested in theories which can be expressed by axioms or equations of first-order languages. In such languages, different terms may express the same things. Naturally, we want to find an effective way of deducing whether two terms express the same thing. We observe that

(i) natural connections between algebras — *homomorphisms*, which are arbitrary functions preserving compositionality;

(ii) *kernels* of homomorphisms — capturing the terms denoting the same object;

(iii) *congruences* — capturing the kernels without depending on homomorphisms;

(iv) *quotient algebras* — capturing congruences, and in turn capturing kernels.

This is formalized in the three equivalent statements in Birkhoff's Theorem (see Theorem 2.1.18 and Theorem 2.1.20 below). With the fact of that homomorphisms on term algebras are substitutions, we complete our search for deduction rule besides the obvious ones (reflectivity, symmetricity, transitivity and compositionality) indicated by congruences.

The essential difference between single-sorted algebras and many-sorted algebras is the potentiality of *empty carriers* (or empty sorts). If Birkhoff's method is going to work in the many-sorted case, we must find a way to express empty carriers in the approach. This is accomplished by the *existence* of homomorphisms from the corresponding term algebra to an algebra (for denotation). This fact is demonstrated in Fact 2.1.3.

A Σ-equation (or equation for short) $t \simeq_X t'$ is a triple $< X, t, t' >$ with variable index X (an I-sorted set of variables), where the index X means $t, t' \in T(X)_i$ for some $i \in I$. The variable index X in equations is usually referred to as an universal quantification, say $\forall X.t \simeq t'$ in [51,53,38]. However, we prefer variable-indexed equations to universally-quantified ones, simply because variable indices make more semantic sense than quantifiers on the unification of the three equational logics. Besides universal quantifier is not a valid syntactic object in first order languages under our consideration. We will come back to this point later. Without variable indices (or universal quantifiers), $\mathbf{True} \simeq \mathbf{False}$ is derivable, see Example 2.1.1 below. But variable indices have no role in single-sorted situation, see Example 2.2.7.

Example 2.1.1 (necessity of variable indices [51,53]): *Let signature* Σ *have sorts* $\{ob, bool\}$ *and* $\Sigma_{\epsilon,bool} = \{\mathbf{True}, \mathbf{False}\}$, $\Sigma_{bool,bool} = \{\neg\}$, $\Sigma_{bool \times bool,bool} = \{\mathbf{and}, \mathbf{or}\}$, $\Sigma_{ob,bool} = \{\mathbf{fool}\}$, *and other* $\Sigma_{k,l}$ *be* $\emptyset$. *The axiom are the collection of*

$$\neg\,\mathbf{True} \simeq \mathbf{False}, \quad \neg\,\mathbf{False} \simeq \mathbf{True}, \quad x\,\mathbf{or}\,\neg x \simeq \mathbf{True}$$
$$x\,\mathbf{and}\,\neg x \simeq \mathbf{False}, \quad x\,\mathbf{and}\,x \simeq x, \quad x\,\mathbf{or}\,x \simeq x$$
$$\mathbf{fool}(y) \simeq \neg\,\mathbf{fool}(y)$$

where x *is a variable sorted by bool and* y *is a variable sorted by ob. We can derive* $\mathbf{True} \simeq \mathbf{False}$ *as follows.*

$$\begin{aligned}&\mathbf{True} \simeq \\ &\mathbf{fool}(x)\,\mathbf{or}\,\neg\,\mathbf{fool}(x) \simeq \mathbf{fool}(x)\,\mathbf{or}\,\mathbf{fool}(x) \simeq \mathbf{fool}(x) \simeq \\ &\mathbf{fool}(x)\,\mathbf{and}\,\mathbf{fool}(x) \simeq \mathbf{fool}(x)\,\mathbf{and}\,\neg\,\mathbf{fool}(x) \simeq \\ &\mathbf{False}.\end{aligned}$$

We start with some more definitions on *satisfaction of equations*. Formally,

Definition 2.1.2 (satisfaction $\models_{EQ}$): *Let Γ be a set of equations.*

(a) $\mathbf{D} \models_{EQ} \Gamma$ (or $\mathbf{D} \models \Gamma$) iff $\mathbf{D} \models t \simeq_X t'$ for every $t \simeq_X t' \in \Gamma$, where Γ is a collection of Σ-equations.

We also say that such a $\mathbf{D}$ is a (Σ, Γ)-algebra and let $Alg_{\Sigma,\Gamma}$ be the class of all possible (Σ, Γ)-algebras. Sometimes, $\mathbf{D}$ is referred to as a model *of Γ.*

(b) $\mathcal{K} \models_{EQ} t \simeq_X t'$ (or $\mathbf{D} \models t \simeq_X t'$) iff $\mathbf{D} \models t \simeq_X t'$ for each $\mathbf{D} \in \mathcal{K}$, where $\mathcal{K}$ is a collection of Σ-algebras.

(c) $\Gamma \models_{EQ} t \simeq_X t'$ (or $\Gamma \models t \simeq_X t'$) iff $\mathbf{D} \models \Gamma$ implies $\mathbf{D} \models t \simeq_X t'$ for every $\mathbf{D}$.

First of all, there are two simple facts which show the role of variable indices in Σ-equations. The proof is left as a simple exercise (hint: see Example 2.1.22)

Fact 2.1.3 (role of variable index): *Let $\mathbf{D}$ be a Σ-algebra, $X \cap Y = \emptyset$ where $X, Y \subseteq \vec{\chi}$, and $t, t' \in T(X)$. Then,*

1. *$\mathbf{D} \models t \simeq_{X \cup Y} t'$ implies $\mathbf{D} \models t \simeq_X t'$, only if there is a (total and non-empty) map ι from $X \cup Y$ to $\mathbf{D}$.*

2. *$\mathbf{D} \models t \simeq_X t'$ implies $\mathbf{D} \models t \simeq_{X \cup Y} t'$.*

Note that the side-condition of 1 in Fact 2.1.3 is because of possible empty carriers. It also point out that empty carriers can be captured by non-existence of interpretations.

In general, we are more interested in the property $\Gamma \models t \simeq_X t'$. To study this, we need to consider a quite arbitrary class of algebras, say $Alg_{\Sigma,\Gamma}$; since Γ can be arbitrary. In turn, we are interested in a generalization of it, i.e. $\mathcal{K} \models t \simeq_X t'$ for any class of $\mathcal{K}$. This is because of that $\Gamma \models t \simeq_X t'$ is equivalent to $\mathcal{K} \models t \simeq_X t'$ when $\mathcal{K} = Alg_{\Sigma,\Gamma}$. However, instead of considering every possible algebra, we wonder if there is a way to connect algebras with each other. Obviously, the most natural one has to be a homomorphism. Formally,

Definition 2.1.4 (Σ-homomorphism): *Let $\mathbf{D}, \mathbf{D}'$ be two Σ-algebras and ι be an indexed family of functions from D_i to D'_i according to sort $i \in I$. ι is said to be*

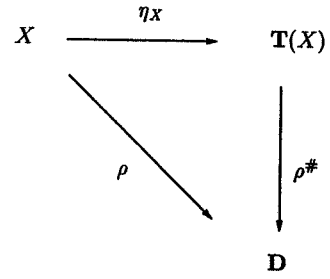

Figure 2–1: interpretation or unique extension of environments

a Σ-homomorphism from $\mathbf{D}$ *to* $\mathbf{D}'$, *written as* $\iota : \mathbf{D} \to \mathbf{D}'$, *iff for each* $\sigma \in \Sigma_{i_1 i_2 \ldots i_m, i}$ *and for* $d_k \in D_{i_k}$ $(1 \leq k \leq m)$, $\iota(\sigma^{\mathbf{D}}(d_1, d_2, ..., d_m)) = \sigma^{\mathbf{D}'}(\iota(d_1, d_2, ..., d_m))$.

For every environment $\vec{\rho} : X \to \mathbf{D}$, the associated interpretation $\mathcal{D}[\![\bullet]\!](\vec{\rho})$ sometimes denoted as $\vec{\rho}^{\#}$, is a Σ-homomorphism from term algebra $\mathbf{T}(X)$ to $\mathbf{D}$. It can be easily verified that every environment $\vec{\rho}$ determines a unique Σ-homomorphism $\vec{\rho}^{\#}$, and vice versa, see Figure 2-1 (where η_X is the indexed family of the usual inclusion functions from X_i to $T(X)_i$ according to sort i).

Therefore, we can have the following :

Property 2.1.5 (satisfaction and homomorphism): $\mathbf{D} \models t \simeq_X t'$ *iff* $\iota(t) = \iota(t')$ *for every* $\iota : \mathbf{T}(X) \to \mathbf{D}$.

Returning to our previous question about connections among algebras, this time it is under the situation of homomorphisms. We wonder whether $\mathbf{D}$ can be captured by $\mathbf{D}'$ through a homomorphism $\iota : \mathbf{D} \to \mathbf{D}'$. We may notice that if ι is injective, $\mathbf{D}$ can be viewed as a part of $\mathbf{D}'$. Such an observation leads to subalgebras. Formally

Definition 2.1.6 (subalgebra $\mathbf{D} \preceq \mathbf{D}'$**):** *Let* $\mathbf{D}'$ *be a Σ-algebra and* $\vec{D}$ *be a subfamily of* $\vec{D}'$. *Then* $\mathbf{D} = <\vec{D}, \mathcal{D}>$ *is said to be a* sub-Σ-algebra *of* $\mathbf{D}'$, *written as* $\mathbf{D} \preceq$

$\mathbf{D}'$, *iff for each* $\sigma \in \Sigma_{i_1 i_2 \ldots i_m, i}$ *and for* $d_k \in D_{i_k}$ $(1 \leq k \leq m)$, $\iota(\sigma^{\mathbf{D}}(d_1, d_2, ..., d_m)) = \sigma^{\mathbf{D}'}(\iota(d_1, d_2, ..., d_m))$.

Sometimes, $\mathcal{D}$ is denoted by $\mathcal{D}'\lceil_{\vec{D}}$ or $\mathcal{D}'\lceil_{\mathbf{D}}$, where $\lceil$ means projecting (or restriction).

The collection of all subalgebras has some nice properties. For example, it is closed under intersections, and inherits satisfaction. Formally,

Property 2.1.7 (intersection of subalgebras):

(a) Let $\mathbf{D}$ *be a* Σ*-algebra and* $\mathcal{K}$ *be* $\{\mathbf{D}' | \mathbf{D}' \preceq \mathbf{D}\}$. *Then,* $\bigcap \mathcal{K}$ *is an algebra and indeed a subalgebra of* $\mathbf{D}$, *i.e.* $\bigcap \mathcal{K} \preceq \mathbf{D}$.

(b) Let $\mathbf{D}' \preceq \mathbf{D}$. *Then,* $\mathbf{D} \models t \simeq_X t'$ *implies* $\mathbf{D}' \models t \simeq_X t'$.

On the other hand, let us return to our previous question on connections among algebras again. This time, we wonder whether $\mathbf{D}'$ can be captured by $\mathbf{D}$ through a homomorphism $\iota : \mathbf{D} \to \mathbf{D}'$. We observe that if ι is surjective and if we collect all equal element in $\vec{D}$ under ι (i.e. let $[d]$ be $\{d' \in D | \iota(d) = \iota(d')\}$) and unify those equal elements into one, then we have reasons to believe that $\mathbf{D}'$ should be equivalent to $\mathbf{D}$ after such collecting and unifying. This leads to congruences and kernels of homomorphisms. We introduce them gradually as follows.

Definition 2.1.8 (congruence $\vec{\theta}$): *Let* $\vec{\theta}$ *be a family of equivalence relations in* $\vec{D}$. $\vec{\theta}$ *is said to be a* Σ*-congruence on* $\mathbf{D}$ *iff* $\vec{\theta}$ *preserves compositionality; in other words, for each* $\sigma \in \Sigma_{i_1 i_2 \ldots i_m, i}$ *and for* $\vec{d}(j), \vec{d}'(j) \in D_{\kappa(j)}$, $\vec{d}\,\vec{\theta}\,\vec{d}'$ *implies* $\sigma^{\mathbf{D}}(\vec{d})\,\vec{\theta}\,\sigma^{\mathbf{D}}(\vec{d}')$ *(i.e.* $\vec{d}(j)\vec{\theta}\vec{d}(j)$ *for* $1 \leq j \leq m$*).*

Note that in general $\vec{e}(j)$ means the jth element of list $\vec{e}$, and $\vec{d}\ \vec{\theta}\ \vec{d}'$ means $\bigwedge_{1 \leq j \leq m} \vec{d}(j)\vec{\theta}\vec{d}'(j)$.

Like subalgebras, congruences have the nice property of being closed under intersections; i.e. let Φ be a collection of Σ-congruences of $\mathbf{D}$, then $\bigcap \Phi$ is also a Σ-congruence of $\mathbf{D}$.

Also, it is natural to introduce quotient algebras corresponding to congruences. Let $\vec{\theta}$ be a Σ-congruence on $\mathbf{D}$, we can define the following:

suppose $\vec{D}/\vec{\theta}$ is a family of $D_i/\vec{\theta}$ for each $i \in I$, where $D_i/\vec{\theta} =_{df} \{d/\vec{\theta} \mid d \in D_i\}$, $d/\vec{\theta} =_{df} \{d' \in D_i \mid d\,\vec{\theta}_i\,d'\}$ and $\sigma^{(\mathbf{D}/\vec{\theta})}(\vec{d}/\vec{\theta}) =_{df} \sigma^{\mathbf{D}}(\vec{d})/\vec{\theta}$. Sometimes, $d/\vec{\theta}$ is written as $[d]_{\vec{\theta}}$, or simply $[d]$ when $\vec{\theta}$ can be understood from the context. $\mathbf{D}/\vec{\theta}$ is said to be the *quotient* Σ-*algebra of* $\mathbf{D}$ *over* $\vec{\theta}$ and there is a natural homomorphism $\nu_{\vec{\theta}}$ from $\mathbf{D}$ to $\mathbf{D}/\vec{\theta}$ associated with $\vec{\theta}$, i.e. $\nu_{\vec{\theta}}(d) = d/\vec{\theta}$. On the other hand, let $\vec{\theta}'$ be another Σ-congruence of $\mathbf{D}$ and $\vec{\theta} \subseteq \vec{\theta}'$. Then, a relation $\vec{\theta}'/\vec{\theta}$, as that $<d/\vec{\theta}, d'/\vec{\theta}> \in \vec{\theta}'/\vec{\theta}$ iff $<d, d'> \in \vec{\theta}'$, is a Σ-congruence on quotient algebra $\mathbf{D}/\vec{\theta}$. So, we can obtain another quotient Σ-algebra $(\mathbf{D}/\vec{\theta})/(\vec{\theta}/\vec{\theta}')$ by $\sigma^{(\mathbf{D}/\vec{\theta})/(\vec{\theta}/\vec{\theta}')}((\vec{d}/\vec{\theta})/(\vec{\theta}/\vec{\theta}')) =_{df} \sigma^{(\mathbf{D}/\vec{\theta})}((\vec{d}/\vec{\theta})/(\vec{\theta}/\vec{\theta}'))$. Such double quotient algebra has a close relation with quotient algebra $\mathbf{D}/\vec{\theta}'$. Formally

Theorem 2.1.9 (quotient algebras and double quotient algebras): *Let* $\vec{\theta}$ *and* $\vec{\theta}'$ *be* Σ-*congruences of* $\mathbf{D}$ *and* $\vec{\theta} \subseteq \vec{\theta}'$. *Then, there is a* Σ-*isomorphism* ι *from* Σ-*quotient algebras* $(\mathbf{D}/\vec{\theta})/(\vec{\theta}/\vec{\theta}')$ *to* $\mathbf{D}/\vec{\theta}'$ *such that* $\nu_{\vec{\theta}'} = \iota \circ \nu_{(\vec{\theta}/\vec{\theta}')} \circ \nu_{\vec{\theta}}$ *where a* Σ-*homomorphism is a* Σ-*isomorphism iff it is bijective.*

Now, we come to the point of introducing kernels of homomorphisms. Formally,

Definition 2.1.10 (kernel of a homomorphism $ker(\iota)$**):** *Let* $\iota : \mathbf{D} \to \mathbf{D}'$. *Then the kernel of* ι, *written as* $ker(\iota)$, *is the family of relations* $ker_i(\iota)$ *such that* $<d, d'> \in ker(\iota)$ *iff* $\iota(d) = \iota(d')$ *and* $d, d' \in D_i$.

It is easy to verify that $ker(\iota)$ is a Σ-congruence of $\mathbf{D}$. Therefore, with the natural $\nu_{ker(\iota)}$ (or ν_ι natural homomorphism associated with $ker(\iota)$) in mind, we wonder whether there is a Σ-homomorphism $\hat{\iota} : \mathbf{D}/ker(\iota) \to \mathbf{D}'$ such that $\iota = \hat{\iota} \circ \nu_\iota$. The answer is positive, since $\hat{\iota}$ can be obtained through ι. That is, $\hat{\iota}(d/ker(\iota)) =_{df} \iota(d)$. Such a $\hat{\iota}$ is a Σ-homomorphism. Moreover, it is a Σ-isomorphism if ι is surjective. We sum up these as a theorem below (Theorem 2.1.11).

Theorem 2.1.11 (homomorphism theorem): *Let* $\iota : \mathbf{D} \to \mathbf{D}'$ *be a* Σ-*homomorphism (or surjective). Then, there exists an injective (or bijective)* $\hat{\iota} : \mathbf{D}/ker(\iota) \to \mathbf{D}'$ *such that* $\iota = \hat{\iota} \circ \nu_\iota$.

So, $\mathbf{D}/ker(\iota)$ can totally capture $\mathbf{D}'$. Formally

Property 2.1.12 (invariance of satisfaction under isomorphism): *Let* $\mathbf{D}'$ *and* $\mathbf{D}$ *be* Σ-*isomorphic to each other. Then,* $\mathbf{D} \models t \simeq_X t'$ *iff* $\mathbf{D}' \models t \simeq_X t'$.

So far, it seems that we have a quite satisfactory answer to our previous question on connections among algebras. But such a question would make more sense if we can find a *"universal"* algebra so that each algebra of a class $\mathcal{K}$, say the class $Alg_{\Sigma,\Gamma}$, can be expressed by a universal one in one way or the other. Further, we hope that such a "universal" algebra can capture the satisfaction $\models_{EQ}$ of class $\mathcal{K}$ completely.

Checking what we have in hand at the moment, we will see that the term algebra $\mathbf{T}(X)$ has certain "universal" property. For example, it has homomorphisms to every algebra. Also, $\mathbf{T}(X)$ can be generated from X. Combining both together, we would come to *"free"* algebras. Firstly, we introduce a concept of generated algebras. Formally

Definition 2.1.13 (generated subalgebra $G(\vec{D})$): *Let* $\mathbf{D}'$ *be a* Σ*-algebra and* $\vec{D} \subseteq \vec{D}'$. $G(\vec{D}) =_{df} \bigcap\{\vec{C} | \mathbf{C} \preceq \mathbf{D}'$ *and* $\vec{C} \supseteq \vec{D}\}$.

(a) $\mathbf{G}(\vec{D}) = <G(\vec{D}), \mathcal{D}'\lceil_{G(\vec{D})}>$ *is called the* generated sub-Σ-algebra *of* $\mathbf{D}'$ *from* $\vec{D}$. $\vec{D}$ *is also called* generator *of* $\mathbf{G}(\vec{D})$.

(b) further $\mathbf{D}'$ *is said to be the* generated Σ-algebra *from* $\vec{D}$, *if* $\vec{D} \subseteq \vec{D}'$ *and* $\mathbf{G}(\vec{D}) = \mathbf{D}'$.

To justify the word "generated", we can define a function *gen* and *gen** as $gen(\vec{D}) = \vec{C}$ such that $C_i =_{df} D_i \cup \{\sigma^{\mathbf{D}'}(\vec{d}) | \vec{d} \in D_\kappa$ *and* $\sigma \in \Sigma_{\kappa,i}\}$ and $gen^*(\vec{D}) =_{df} \bigcup_{\ell \in Nat} gen^\ell(\vec{D})$, where $gen^{\ell+1}(\vec{D}) =_{df} gen^\ell(gen(\vec{D}))$. It is an easy exercise to verify that $G(\vec{D}) = <gen^*(\vec{D}), \mathcal{D}'\lceil_{gen^*(\vec{D})}>$ and $\vec{d} \in D_\kappa$ means $d_k \in D_{i_k}$ for $1 \leq k \leq m$ and $\kappa = i_1 i_2 ... i_m$. Also, it can be easily checked that the term algebra $\mathbf{T}(X)$ is the generated Σ-algebra by X.

The concept of the generated algebra does not quite capture the term algebra totally. For instance, the property of every environment can be uniquely extended to a homomorphism does not hold for generated algebras in general. In other words, not every map from the generator $\vec{D}$ to $\mathbf{G}(\vec{D})$ can be extended to a homomorphism. The reason for this kind of a failure comes from that the elements in $\vec{D}$ may not be independent. This observation leads to free algebras. Formally

Definition 2.1.14 (free algebra): *Let* $\mathcal{K}$ *be a class of* Σ*-algebra and* $\mathbf{D}$ *be a generated algebra from* $\vec{D}$. $\mathbf{D}$ *is said to be a* free Σ-algebra *of* $\mathcal{K}$ *iff for every algebra*

$\mathbf{D}' \in \mathcal{K}$, *any map from* $\vec{D}$ *to* $\mathbf{D}'$ *can be extended to a* Σ*-homomorphism from* $\mathbf{D}$ *to* $\mathbf{D}'$. *We also say that* $\mathbf{D}$ *is* free *in* $\mathcal{K}$ *and* $\vec{D}$ *is a* free generator *of* $\mathbf{D}$ *over* $\mathcal{K}$.

Note that if $\mathbf{D}$ is free over $\mathcal{K}$ with generator $\vec{D}$, and if ι_1, ι_2 are Σ-homomorphism from $\mathbf{D}$ to $\mathbf{A}$ which agree on generator $\vec{D}$, then $\iota_1 = \iota_2$ since ι_1 and ι_2 agree on $gen^{\ell+1}(\vec{D})$ if they agree on $gen^{\ell}(\vec{D})$ for $\ell \geq 0$.

An obvious example of free algebras is the term algebra $\mathbf{T}(X)$, and it is a free Σ-algebra of any class $\mathcal{K}$, say $Alg_{\Sigma,\emptyset}$ (or Alg). Now, the uniqueness of homomorphism's extension on term algebras can be generalized to free algebras. That is, let $\mathbf{D}$ be a free Σ-algebra of $\mathcal{K}$ where $\vec{D}$ is its free generator, then, every $\iota : \mathbf{D} \to \mathbf{D}'$ is uniquely and completely decided by the map $\iota\lceil_{\vec{D}}$ from the generator $\vec{D}$ to $\mathbf{D}'$.

For a relation among free algebras, we have

Lemma 2.1.15 (uniqueness of free algebras): *Let* $\mathbf{D}$ *and* $\mathbf{D}'$ *be two free* Σ*-algebras over* $\mathcal{K}$ *generated by* $\vec{D}$ *and* $\vec{D}'$ *respectively. If* $|\vec{D}| = |\vec{D}'|$ *then* $\mathbf{D}$ *is* Σ*-isomophic to* $\mathbf{D}'$ *where* $\mathbf{D}, \mathbf{D}' \in \mathcal{K}$.

For free algebras, we can generalize the result of Theorem 2.1.11. Formally

Lemma 2.1.16 (free algebras and congruences): *Let* $\mathbf{D}$ *be a free algebra over* $\mathcal{K}$ *and* $\vec{D}$ *be its free generator,* $\mathbf{D}' \in \mathcal{K}$. *Then,*

(i) $\mathbf{D}/\vec{\theta} = \mathbf{G}(\vec{D}/\vec{\theta})$, *where* $\vec{\theta}$ *is a congruence of* $\mathbf{D}$, *and*

(ii) for all $\iota : \mathbf{D} \to \mathbf{D}'$, *there is an* Σ*-homomorphism* $\hat{\iota} : \mathbf{D}/\vec{\theta} \to \mathbf{D}'$ *such that* $\iota = \hat{\iota} \circ \nu_{\vec{\theta}}$, *where* $\vec{\theta}$ *is* $\bigcap_{\iota' : \mathbf{D} \to \mathbf{D}'} ker(\iota')$.

(iii) if let $\vec{\theta}'$ *be a* Σ*-congruence of* $\mathbf{D}'$ *and* $\mathbf{D}'/\vec{\theta}' \in \mathcal{K}$, *then for every* $\iota : \mathbf{D} \to \mathbf{D}'/\vec{\theta}'$, *there is an* Σ*-homomorphism* $\check{\iota} : \mathbf{D} \to \mathbf{D}'$ *such that* $\iota = \nu_{\vec{\theta}'} \circ \check{\iota}$.

Note that the condition $\mathbf{D}$ is a free algebra over $\mathcal{K}$ is important in the proof of (iii) in Lemma 2.1.16. Other items (i) and (ii) are easy exercises.

From Property 2.1.5, we understand that $\mathbf{D} \models t \simeq_X t'$ iff $< t, t' > \in \bigcap_{\iota : \mathbf{T} \to \mathbf{D}} ker_X(\iota)$ (we use $ker_X(\iota)$ instead of $ker(\iota)$ to emphasize the role of X). Also the index X means that $ker(\iota)$ is a Σ-congruence of $\mathbf{T}(X)$). We restate it as a lemma (Lemma 2.1.17).

Lemma 2.1.17 (satisfaction and index congruence): *Let* $Ker_X(\dot{\mathbf{D}})$ *be* $\bigcap_{\iota:\mathbf{T}\to\mathbf{D}} ker_X(\iota)$. *Then,* $\mathbf{D} \models t \simeq_X t'$ *iff* $< t, t' > \in Ker_X(\mathbf{D})$.

Applying the above two lemmas (i.e. (ii) in Lemma 2.1.16 and Lemma 2.1.17) to term algebra $\mathbf{T}(X)$, we would have a Birkhoff's Theorem in many-sorted first order algebras.

Theorem 2.1.18 (Many-sorted Birkhoff's Theorem 1): *The following three statements are equivalent:*

(a) $\mathbf{T}(X)/Ker_X(\mathbf{D}) \models t \simeq_X t'$,

(b) $\mathbf{D} \models t \simeq_X t'$,

(c) $< t, t' > \in Ker_X(\mathbf{D})$.

Proof

By Lemma 2.1.17, we have that (b) iff (c); and by (ii) in Lemma 2.1.16 with $Ker_X(\mathbf{D}) = ker_X(\nu_{Ker_X(\mathbf{D})}) \subseteq \bigcap_{\iota:\mathbf{T}(X)\to(\mathbf{T}(X)/Ker_X(\mathbf{D}))} ker_X(\iota)$, we have that (a) iff (c). □

The many-sorted Birkhoff's Theorem seems to suggest that a term algebra $\mathbf{T}(X)$ may be a "universal" algebra representing all algebras in any (non-empty) $\mathcal{K}$. The links between $\mathbf{T}(X)$ and other algebras are homomorphisms. This is confirmed by the theorem below (Theorem 2.1.19).

Theorem 2.1.19 (free quotient term algebras): *Let* $\mathcal{K}$ *be a (non-empty) collection of* Σ*-algebras, and* Ker_X *be* $\bigcap_{\mathbf{D}\in\mathcal{K}} Ker_X(\mathbf{D})$. *Then,* $\mathbf{T}(X)/Ker_X$ *is a free* Σ*-algebra over it (when* $\mathcal{K} = \emptyset$, *we assume* $\bigcap_{\mathbf{D}\in\mathcal{K}} Ker_X(\mathbf{D}) = \mathbf{T}(X) \times \mathbf{T}(X)$*).*

Proof

- Let $\mathbf{D} \in \mathcal{K}$. Then, for every map $\iota : X/Ker_X \to \mathbf{D}$, we know that $\iota \circ (\nu_X \lceil_X)$ is a map from X to $\mathbf{D}$, where ν_X is the natural homomorphism associated with Ker_X. Thus, there is an unique extension ι' of $\iota \circ (\nu_X \lceil_X)$ such that ι' is a Σ-homomorphism from $\mathbf{T}(X)$ to $\mathbf{D}$.

- since $\iota' : \mathbf{T}(X) \rightarrow \mathbf{D}$, there is an unique $\hat{\iota'}$ such that $\iota' = \hat{\iota'} \circ \nu_{\iota'}$ by Lemma 2.1.16.

- for $\nu_{\iota'}$, applying the same lemma (Lemma 2.1.16) again, we get that there is an unique $\hat{\nu_{\iota'}}$ such that $\nu_{\iota'} = \hat{\nu_{\iota'}} \circ \nu$.

- We conclude that $\mathbf{T}(X)/Ker_X$ is a free Σ-algebra over $\mathcal{K}$. □

Now, Theorem 2.1.18 (many-sorted Birkhoff's Theorem on an individual term algebra) can be extended. Namely,

Theorem 2.1.20 (many-sorted Birkhoff's Theorem 2): *The following three statements are equivalent:*

1. $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$,

2. $\mathcal{K} \models t \simeq_X t'$,

3. $< t, t' > \in Ker_X$.

From the above theorem (Theorem 2.1.20), we understand that for every ι : $\mathbf{T}(X) \rightarrow \mathbf{T}(X)/Ker_X$, if $< t, t' > \in Ker_X$, then there are u, u' such that $\iota(t) = u/Ker_X$, $\iota(t') = u'/Ker_X$ and $< u, u' > \in Ker_X$. This observing leads to Lemma 2.1.21.

Lemma 2.1.21 : *Let $\mathcal{K}$ be a (non-empty) collection of Σ-algebras and for every $\iota : \mathbf{T}(X) \rightarrow \mathbf{T}(X)/Ker_X$, there exists $\check{\iota} : \mathbf{T}(X) \rightarrow \mathbf{T}(X)$ such that $\iota = \nu_X \circ \check{\iota}$.*

Proof

We only need to verify the fact that semantic substitutions (environments) can be replaced by syntactic substitutions and vice versa. Then, with two facts of every term algebra is a free algebra and of each environment can be uniquely extended to a homomorphism, the result follows. □

Note that we can not apply Lemma 2.1.16 to the proof of above lemma (Lemma 2.1.21). Why? (hint : the quotient algebra might not be in $\mathcal{K}$).

Since we have not only that for every $\iota : \mathbf{T}(X) \to \mathbf{T}(X)$ there is an $\iota : \mathbf{T}(X) \to (\mathbf{T}(X)/Ker_X)$ such that $\iota = \nu_X \circ \iota$, but also Lemma 2.1.21, we know that $\{Ker_X | X \subseteq \vec{\chi}\}$ is an invariance under Σ-homomorphisms from $\mathbf{T}(X)$ to itself. This invariance brings in the concept of *fully invariant congruences*. Formally

Definition 2.1.22 (fully invariant congruences):

(a) Let $\vec{\theta}$ be a Σ-congruence of Σ-algebra $\mathbf{D}$. Then, $\vec{\theta}$ is said to be fully invariant *iff $d\,\vec{\theta}\,d'$ implies $\iota(d)\vec{\theta}\iota(d')$, for every Σ-homomorphism (endomorphism) ι from $\mathbf{D}$ to itself.*

(b) Let $\tilde{\theta}$ be a family of collections $\vec{\theta}^{\mathbf{D}}$ which is a Σ-congruence of $\mathbf{D} \in \mathcal{K}$. $\tilde{\theta}$ is said fully invariant over $\mathcal{K}$ iff for each member $\mathbf{D}$ of $\mathcal{K}$, the associated congruence $\vec{\theta}^{\mathbf{D}}$ is fully invariant.

For example, Ker_X is fully invariant Σ-congruence of $\mathbf{T}(X)$, where $\mathcal{K}$ is not empty; and $\{Ker_X | X \subseteq \vec{\chi}\}$ is fully invariant over the family $\{\mathbf{T}(X) | X \subseteq \vec{\chi}\}$ of term algebras.

Referring to Lemma 2.1.21, we notice that $\bar{\iota}$ is actually a commonly used syntactic technique, so-called a *substitution* (map). This understanding leads to an extension of Lemma 2.1.21. Formally

Theorem 2.1.23 (substitution $\bar{\iota}$): *Let $\mathcal{K}$ be a (non-empty) collection of Σ-algebras and $X, Y \subseteq \vec{\chi}$. Then, for every $\iota : \mathbf{T}(X) \to \mathbf{T}(Y)/Ker_Y$, there is a Σ-homomorphism $\bar{\iota} : \mathbf{T}(X) \to \mathbf{T}(Y)$ such that $\iota = \nu_Y \circ \bar{\iota}$ and vice versa.*

This theorem (Theorem 2.1.23) remind us to seek relations between satisfactions of term algebras with different variable indices, since we only discuss the satisfaction under a fixed X (referring to 2.1.18 and 2.1.20). For example, whether the satisfactions are preserved by different indexed term algebras.

With this purpose in mind, let

(a) $\tilde{\mathbf{T}}(\vec{\chi})$ be $\{\mathbf{T}(X) | X \subseteq \vec{\chi}\}$,

(b) $\tilde{K}(\mathbf{D})$ be $\{Ker_X(\mathbf{D}) | X \subseteq \vec{\chi}\}$,

(c) $\tilde{K}(\mathcal{K})$ (or $\tilde{K}$) be $\bigcap_{\mathbf{D} \in \mathcal{K}} \tilde{K}(\mathbf{D})$ where it is the same as $\{Ker_X | X \subseteq \vec{\chi}\}$, and

(d) $\tilde{\mathbf{T}}/\tilde{K}$ be $\{\mathbf{T}(X)/Ker_X | X \subseteq \vec{\chi}\}$.

Note that sign $\vec{\bullet}$ means that $\bullet$ is a family of things indexed by different variable sets and sign $\vec{\bullet}$ means that $\bullet$ is a family of things indexed by different sorts.

Thus, we define that $\tilde{\mathbf{T}}/\tilde{K} \models t \simeq_X t'$ iff $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$ and its natural extension to $\tilde{\mathbf{T}}/\tilde{K} \models \Gamma$. Then, we return to our starting point, $\Gamma \models t \simeq_X t'$ (i.e. $\mathcal{K} \models t \simeq_X t'$ if $\mathcal{K}$ be $Alg_{\Sigma,\Gamma}$). On the other hand, what we have is: for each member $\mathbf{T}(X)/Ker_X$ of $\tilde{\mathbf{T}}/\tilde{K}$, it satisfies a certain sub-collection Γ'_X of Γ, which share the same variable index X with the term algebra $\mathbf{T}(X)$. Therefore, we naturally wonder what is the inter-relationship among the satisfactions of different members of $\tilde{\mathbf{T}}$?

The followings are some simple facts related to the above question.

Fact 2.1.24 (relations among satisfaction and variable indices): *Let* $t, t' \in \mathbf{T}(X)$. *Then,*

1. $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$ *implies* $\mathbf{T}(X)/Ker_X \models t \simeq_{X \cup Y} t'$;

2. $\mathbf{T}(X)/Ker_X \models t \simeq_{X \cup Y} t'$ *implies* $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$, *if there is a* Σ-*homomorphism* ι *from* $\mathbf{T}(X \cup Y) \to \mathbf{T}(X)$;

3. $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_{X \cup Y} t'$ *implies* $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_X t'$;

4. $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_X t'$ *implies* $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_{X \cup Y} t'$.

Informally, Fact 2.1.24 say that increasing and decreasing the sizes of variables (as indices in Σ-equations) have no impact on satisfactions (note that there is a side-condition in 2 of Fact 2.1.24). However, such facts do not answer the question of inter-relationship of satisfactions directly. Nevertheless, from the second part of Fact 2.1.3, we have that $< t, t' > \in Ker_X(\mathbf{D})$ implies $< t, t' > \in Ker_{X \cup Y}(\mathbf{D})$, where $X \cap Y = \emptyset$. So, $< t, t' > \in Ker_X$ implies $< t, t' > \in Ker_{X \cup Y}$. We restate this result in another way (see Lemma 2.1.25) because of Theorem 2.1.19.

Lemma 2.1.25 (index increasing):

$\mathbf{T}(X)/Ker_X \models t \simeq_X t'$ *implies* $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_{X \cup Y} t'$.

With Lemma 2.1.25 and the first part of Fact 2.1.3, we can obtained the following.

Lemma 2.1.26 (index decreasing):

$\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_{X \cup Y} t'$ *implies* $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$, *provided that there exists* $\iota : \mathbf{T}(X \cup Y) \to \mathbf{T}(X)$ $(X \cap Y = \emptyset)$.

Proof

For each $\mathbf{D} \in \mathcal{K}$, and for every $\iota_1 : \mathbf{T}(X) \to \mathbf{D}$, we have $\iota_1 \circ (id_{\mathbf{T}(X \cup Y)} \lceil_X \uplus \iota)^\#$ is a Σ-homomorphism from $\mathbf{T}(X \cup Y) \to \mathbf{D}$ and $(id \lceil_X \uplus \iota)^\# \lceil_{\mathbf{T}(X)} = id_{\mathbf{T}(X)}$, where $dom(\iota_2)$ and $dom(\iota_1)$ mean their domains and

$$\iota_2 \uplus \iota_1(b) = \begin{cases} \iota_2(b) & if\ b \in dom(\iota_2) \\ \iota_1(b) & if\ b \notin dom(\iota_2)\ and\ b \in dom(\iota_1). \end{cases}$$

Therefore,

$$\iota_1(t) = \iota_1 \circ (id_{\mathbf{T}(X \cup Y)} \lceil_X \uplus \iota)^\#(t) = \iota_1 \circ (id_{\mathbf{T}(X \cup Y)} \lceil_X \uplus \iota)^\#(t') = \iota_1(t'). \ \square$$

This result (Lemma 2.1.26) can be generalized to the following theorem (Theorem 2.1.27).

Theorem 2.1.27 (independent increase and decrease of variable indices): *Let* $X \cap Y = \emptyset$ *(i.e.* $X_i \cap Y_i = \emptyset$ *for* $i \in I$*). Then,*

1. $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$ *implies* $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_X t'$;

2. $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models t \simeq_{X \cup Y} t'$ *implies* $\mathbf{T}(X)/Ker_X \models t \simeq_{X \cup Y} t'$.

Proof

(i) For the first part of theorem, we would have it by Lemma 2.1.25 and Fact 2.1.24.

(ii) For the second part of the theorem, we proceed as follows.

(ii.a) If there is no map from $X \cup Y$ to $\mathbf{T}(X)$, then it is trivial; otherwise,

(ii.b) for each such map ι, we have $\mathbf{T}(X \cup Y)/Ker_{X \cup Y} \models \iota^\#(t) \simeq_{X \cup Y} \iota^\#(t')$.

By Lemma 2.1.26, we have $\mathbf{T}(X)/Ker_X \models \iota^{\#}(t) \simeq_X \iota^{\#}(t')$. Then, by Lemma 2.1.25, we get $\mathbf{T}(X)/Ker_X \models \iota^{\#}(t) \simeq_{X \cup Y} \iota^{\#}(t')$.

(ii.c) On the other hand, for every $\iota_1 : \mathbf{T}(X \cup Y) \rightarrow \mathbf{T}(X)/Ker_X$, there is a Σ-homomorphism $\check{\iota}_1$ from $\mathbf{T}(X \cup Y)$ to $\mathbf{T}(X)$ such that $\iota_1 = \nu_X \circ \check{\iota}_1$ and vice versa (see Theorem 2.1.23).

By (ii.b) above, i.e. let ι be $\check{\iota}_1$, we have $\mathbf{T}(X)/Ker_X \models \check{\iota}_1(t) \simeq_X \check{\iota}_1(t')$.

In other words, $< \check{\iota}_1(t), \check{\iota}_1(t') > \in Ker_X$. Hence, $\iota^{\#}(t) = \iota^{\#}(t')$.

(ii.d) So, $\mathbf{T}(X)/Ker_X \models t \simeq_X t'$. □

Therefore, every member of $\tilde{\mathbf{T}}$ is a model of Γ if $\mathcal{K}$ is $Alg_{\Sigma,\Gamma}$. Also, Lemma 2.1.25 and Theorem 2.1.27 together give us a new concept *"cross-invariance"*. It is unique to the *many-sorted* Universal Algebra and has no role in the *single-sorted* Universal Algebra.

Definition 2.1.28 (cross-fully invariance): *Let $\tilde{\theta}$ be a (fully invariant) family of collections $\vec{\theta}^{\mathbf{D}}$ which are Σ-congruences of $\mathbf{D} \in \mathcal{K}^T$. $\tilde{\theta}$ is said to be* cross-fully invariant *over $\mathcal{K}^T$ iff for all $\mathbf{D}, \mathbf{D}' \in \mathcal{K}^T$ and for every $< d, d' > \in \vec{\theta}^{\mathbf{D}}$, we have that for each $\iota : \mathbf{D} \rightarrow \mathbf{D}'$, $< \iota(d), \iota(d') > \in \vec{\theta}^{\mathbf{D}'}$.*

An example of cross-fully invariant family is to let $\tilde{\theta} = \{Ker_X | X \subseteq \vec{X}\}$ and $\mathcal{K}^T = \tilde{\mathbf{T}}$, which $\tilde{\theta}$ and $\mathcal{K}^T$ are associated with each other between members of them by sharing a same X.

We expect that many previous results on $\{Ker_X \mid X \subseteq \vec{X}\}$ or on its member can be extended to (a) a fully invariant congruence, to (b) a fully invariant family of congruences, and to (c) a cross-fully invariant family of congruences. For example, Theorem 2.1.20 can be extended, and the extended result is as follows.

Lemma 2.1.29 (fully invariance and satisfaction): *Let $\vec{\theta}_X$ be a fully invariant congruence on $\mathbf{T}(X)$. Then, $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_X t'$ iff $< t, t' > \in \vec{\theta}_X$.*

Proof

Let ι denote a Σ-homomorphism from $\mathbf{T}(X)$ to $\mathbf{T}(X)/\vec{\theta}_X$. Then, $\bigcap_\iota ker_X(\iota) \subseteq ker_X(\nu_{\vec{\theta}_X}) = \vec{\theta}_X$.

For the reversed containment, by Lemma 2.1.21, we have that $< t, t' \dot{>}\in \vec{\theta}_X$ implies $< \iota(t), \iota(t') >\in \vec{\theta}_X$, i.e. $\iota(t) = \iota(t')$. □

Similar to Fact 2.1.24, we have Fact 2.1.30.

Fact 2.1.30 (satisfaction and different variable indices): *Let $\tilde{\theta}$ be a family of Σ-congruence over $\tilde{\mathbf{T}}$.*

1. $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_X t'$ *implies* $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_{X\cup Y} t'$.
2. $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_{X\cup Y} t'$ *implies* $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_X t'$, *if there exists a* Σ-*homomorphism* $\iota : \mathbf{T}(X \cup Y) \to \mathbf{T}(X)$.
3. $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_{X\cup Y} t'$ *implies* $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_X t'$.
4. $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_X t'$ *implies* $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_{X\cup Y} t'$.

Theorem 2.1.27 has its extension as follows (see Theorem 2.1.31).

Theorem 2.1.31 (cross-fully invariance and satisfaction): *Let $\tilde{\theta}$ be a cross-fully invariant family of Σ-congruences over $\tilde{\mathbf{T}}$. Then,*

1. $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_X t'$ *implies* $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_X t'$;
2. $\mathbf{T}(X \cup Y)/\vec{\theta}_{X\cup Y} \models t \simeq_{X\cup Y} t'$ *implies* $\mathbf{T}(X)/\vec{\theta}_X \models t \simeq_{X\cup Y} t'$.

Proof

Both can be proved by applying (iii) of Lemma 2.1.16 with the cross-invariant property. □

A cross-fully invariant family of Σ-congruences over $\tilde{\mathbf{T}}$ seems to totally capture equality. That is, so far, we have gotten a quite satisfactory answer to the question of inter-relationship among satisfactions. Since we believe of total capture of Indistinguishability, we are naturally seeking a syntactic counterpart of the cross-fully

invariance. Theorem 2.1.23 provides us a clue, i.e. substitutions. Therefore, we come to a definition as below (see Definition 2.1.32).

Definition 2.1.32 (calculus $\vdash_{EQ}$): *Let Γ be a set of Σ-equations (possibly empty). Then, $\vdash_{EQ}$ is defined, in the judgement form of*

$$\Gamma \vdash_{EQ} t \simeq_X u \text{ or } \Gamma \vdash t \simeq_X u,$$

as follows.

1. *(eq-id)*

$$\{t \simeq_X u\} \vdash_{EQ} t \simeq_X u$$

2. *(eq-rfl)*

$$\vdash_{EQ} t \simeq_X t \quad (t \in \mathbf{T}(X))$$

3. *(eq-sym)*

$$\frac{\Gamma \vdash_{EQ} t \simeq_X u}{\Gamma \vdash_{EQ} u \simeq_X t}$$

4. *(eq-trs)*

$$\frac{\Gamma \vdash_{EQ} t \simeq_X u;\ \Gamma \vdash_{EQ} u \simeq_X v}{\Gamma \vdash_{EQ} t \simeq_X v}$$

   Later, $\Gamma \vdash_{EQ} t_1 \simeq_{X(1)} u_1, t_2 \simeq_{X(2)} u_2, ..., t_m \simeq_{X(m)}$ is abbreviated as list $\Gamma \vdash_{EQ} t_1 \simeq_X u_1, \Gamma \vdash_{EQ} t_2 \simeq_X u_2, ..., \Gamma \vdash_{EQ} t_m \simeq_X u_m$.

5. *(eq-cmp)*

$$\frac{\Gamma \vdash_{EQ} t_1 \simeq_X u_1, t_2 \simeq_X u_2, ..., t_n \simeq_X u_n}{\Gamma \vdash_{EQ} \sigma(t_1, t_2, ..., t_n) \simeq_X \sigma(u_1, u_2, ..., u_n)} \ (\sigma \in \Sigma_{i_1 i_2 ... i_n, i})$$

6. *(crs-sub)*

$$\frac{\Gamma \vdash_{EQ} t \simeq_X u}{\Gamma \vdash_{EQ} \iota(t) \simeq_Y \iota(t')} \ (\iota : \mathbf{T}(X) \to \mathbf{T}(Y))$$

7. *(eq-ctr) (or (eq-cut), (eq-MP))*

$$\frac{\Gamma' \vdash_{EQ} t \simeq_X u;\ \{\Gamma \vdash_{EQ} t' \simeq_{X'} u' | t' \simeq_{X'} u' \in \Gamma'\}}{\Gamma \vdash_{EQ} t \simeq_X u}$$

8. (eq-wkn)

$$\frac{\Gamma \vdash_{EQ} t \simeq_X u}{\Gamma \cup \Gamma' \vdash_{EQ} t \simeq_X u}$$

Note that the non-existence of a substitution map indicates empty sorts (or carriers). So, *Abstraction* rule and *Concretion* rule in [51,53] are derivable from the (crs-sub) rule.

Now, instead of using Γ, we use $\tilde{\Gamma}$ $(= \{\Gamma \bigcap (\{X\} \times \mathbf{T}(X) \times \mathbf{T}(X)) \mid X \subseteq \vec{\chi}\})$ to show its uniformity with other notation. Actually, we can define $\tilde{\mathcal{M}}_{EQ}$ (or simply $\tilde{\mathcal{M}}$) as a function which produces a set $\tilde{\mathcal{M}}(\Gamma)$ of equations if given Γ, i.e. $t \simeq_X u \in \tilde{\mathcal{M}}(\Gamma)$ iff $\Gamma \vdash_{EQ} t \simeq_X u$ without using rule (eq-ctr). Then, $\tilde{\mathcal{M}}$ is *monotonic* and $\tilde{\Gamma} \subseteq \tilde{\mathcal{M}}(\tilde{\Gamma})$ $(= \{\mathcal{M}(\Gamma) \bigcap (\{X\} \times \mathbf{T}(X) \times \mathbf{T}(X)) \mid X \subseteq \vec{\chi}\})$ where the partial order is the usual set inclusion. Therefore, there are (least) fixed points of $\tilde{\mathcal{M}}$, where $\tilde{\Gamma}'$ is a fixed point of $\tilde{\mathcal{M}}$ iff $\tilde{\mathcal{M}}(\tilde{\Gamma}') \subseteq \tilde{\Gamma}'$.

Since our aim is to have deduction systems, We are only interested in minimal fixed points (or the least fixed point relative to a given $\tilde{\Gamma}$), written as $\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma})$, which is $\bigcap \mathcal{R}$ and $R \in \mathcal{R}$ iff R is a fixed point of $\tilde{\mathcal{M}}$ and it contains $\tilde{\Gamma}$ (N.B. The fixed points of $\tilde{\mathcal{M}}$ are closed under intersections, i.e. $\bigcap \mathcal{R}$ is a fixed point of $\tilde{\mathcal{M}}$ if $\mathcal{R}$ is a collection of fixed point of $\tilde{\mathcal{M}}$). When without a confusion, we will sometimes use $\tilde{\mathcal{M}}(\tilde{\Gamma})$ to denote $\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma})$.

Let $\tilde{\phi}$ be a family of functions from $\tilde{\Gamma}$ to $\tilde{\mathbf{T}} \times \tilde{\mathbf{T}}$ such that $\phi_X(t \simeq_X t') =< t, t' > \in T(X) \times T(X)$ for $t \simeq_X t' \in \Gamma_X$ (i.e. somehow $\tilde{\phi}$ is a family of identity functions) and we denote the family of the reversed functions as $\tilde{\phi}^{-1}$. Then, we have the following

Theorem 2.1.33 (deduction and fully invariance and cross-fully invariance):

1. *For each* $X \subseteq \vec{\chi}$, $\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$ *is a* Σ*-congruence on* $\mathbf{T}(X)$ *and* $\tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$ *contains* $\tilde{\phi}(\tilde{\Gamma})$.

2. $\tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$ *is fully invariant and further, it is cross-fully invariant over* $\mathbf{T}(\vec{\chi})$.

Proof

- From 1 of Definition of 2.1.32, we have that $\tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$ contains $\tilde{\phi}(\tilde{\Gamma})$. From 2, 3 and 4 of the same definition (Definition 2.1.32), we have that $\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$ is an equivalence relation for each $X \subseteq \vec{\chi}$. From 5 of Definition 2.1.32, we come to that $\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$ is a Σ-congruence on $\mathbf{T}(X)$. So, we got a proof of the first part of theorem.

- From 6 of Definition 2.1.32 and for each X, if let Y be the same as X, we have that $\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$ is fully invariant.

  If we let Y be arbitrary, then we get that $\tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$ is cross-fully invariant. □

Now, the remaining thing is to see the relationship between $\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma})$ and $\tilde{K}(\mathcal{K})$.

Let us look at $\tilde{\phi}^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))$:

(2.1.a) It contains $\tilde{\Gamma}$.

(2.1.b) Since $\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}})$ is a family of Σ-congruences, the following four statements are true:

(2.1.b.i) for every $t \in \mathbf{T}(X)$, $t \simeq_X t \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$;

(2.1.b.ii) for every $X \subseteq \vec{\chi}$, if $t \simeq_X t' \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$ then $t' \simeq_X t \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$;

(2.1.b.iii) for every $X \subseteq \vec{\chi}$, if $t \simeq_X t', t' \simeq_X t'' \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$ then $t \simeq_X t'' \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$;

(2.1.b.iv) for each $\sigma \in \Sigma_{\kappa,i}$, if $\vec{t} \simeq_X \vec{t'} \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$ then $\sigma(\vec{t}) \simeq_X \sigma(\vec{t'}) \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))_X$.

(2.1.c) Since $\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}})$ is cross-fully invariant, we have that for every $X \subseteq \vec{\chi}$, if $t \simeq_X t' \in \phi^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))$, then for all $Y \subseteq \vec{\chi}$ and for all $\iota : \mathbf{T}(X) \to \mathbf{T}(Y)$ $\iota(t) \simeq_Y \iota(t') \in \phi^{-1}(\tilde{K}(Alg_{\tilde{\Gamma}}))$.

To sum (2.1.a), (2.1.b) and (2.1.c) up, we have that $\tilde{\phi}^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))$ is a (least) fixed point of $\tilde{\mathcal{M}}$ relative to $\tilde{\Gamma}$. Therefore, $\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}) \subseteq \tilde{\phi}^{-1}(\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}))$. Formally

Theorem 2.1.34 (soundness of $\vdash_{EQ}$): $\vdash_{EQ}$ *is sound, i.e.* $\tilde{\Gamma} \vdash_{EQ} t \simeq_X t'$ *implies* $\tilde{\Gamma} \models_{EQ} t \simeq_X t'$.

Note that $\tilde{\Gamma} \vdash t \simeq_X t'$ iff $t \simeq_X t' \in \sqcup\tilde{\mathcal{M}}(\tilde{\Gamma})$.

From Lemma 2.1.29, we know that $\mathbf{T}(X)/\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X \models t \simeq_X t'$ iff $< t, t' > \in \phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$. Then, as a result of Fact 2.1.30 and Theorem 2.1.31, we have that every member $\mathbf{T}(X)/\phi(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))_X$ of $\tilde{\mathbf{T}}(X)/\tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$ is a model of $\tilde{\Gamma}$, i.e. it belongs to $Alg_{\Sigma,\tilde{\Gamma}}$. Therefore, $\tilde{K}(Alg_{\Sigma,\tilde{\Gamma}}) = \{\bigcap_{\mathbf{D} \in Alg_{\Sigma,\tilde{\Gamma}}} Ker_X(\mathbf{D}) | X \subseteq \vec{\chi}\} \subseteq \tilde{\phi}(\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma}))$. In another word, we get the completeness of $\vdash_{EQ}$. Formally

Theorem 2.1.35 (completeness of $\vdash_{EQ}$): $\vdash_{EQ}$ *is complete (and sound), i.e.* $\tilde{\Gamma} \models_{EQ} t \simeq_X t'$ *implies* $\tilde{\Gamma} \vdash_{EQ} t \simeq_X t'$.

A complete syntactic characterization of Indistinguishability leads us to wonder whether there is a syntactic characterization of the subtlety of empty carriers. Next section (Section 2.2) is aiming at this.

2.2 Elimination of variable indices

In the end of last section (Section 2.1), we raise an interesting question, namely

Question 2.2.1: *Can we somehow syntactically capture the subtlety of empty carriers, instead of semantic one so that* $\vdash_{EQ}$ *is variable index free?*

The success on Completeness of Equational Logic leads us to expect an optimistic answer to Question 2.2.1. With such a question in mind, we firstly introduce a definition for index free $\vdash_{EQ}$ (see Definition 2.2.2). Formally

Definition 2.2.2 (consequence family and variable index eliminable): *Let* $\mathcal{K}$ *be a collection of* Σ*-algebras. Then,*

(a) $< t, t' > \in \tilde{C}_X(\mathcal{K})_Y$ *(or* $\tilde{C}_{X,Y}$*) iff* $\mathbf{T}_\Sigma(X)/Ker_X \models t \simeq_Y t'$.

(b) $\vdash_{EQ}$ *is* variable index eliminable *(or variable index free) iff for all* $X, Y \subseteq \vec{\chi}$ *such that* $\tilde{C}_{X,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)} = \tilde{C}_{X,X}$ *and* $\tilde{C}_{Y,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)} = \tilde{C}_{X,X}$.

The consequence family in Definition 2.2.2 is to capture the role of the indexed quotient algebras (index X) at one hand, and the role of variable index in equations (index Y) on the other's. We would see which one is more essential and characterizes the subtlety of empty carriers. For this purpose, we turn to Fact 2.1.23 and Theorem 2.1.26. We would get the following fact (Fact 2.2.3). Namely

Fact 2.2.3 (relationship among members in a consequence family): *Let $X, Y \subseteq \vec{\chi}$ and $X \subseteq Y$. Then,*

$$\begin{array}{ll} & \tilde{C}_{Y,X} \\ & \quad \subseteq \\ \cup|(?) & \quad \tilde{C}_{Y,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)} \subseteq \hat{C}_{X,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)} \overset{\exists\iota}{\subseteq} \tilde{C}_{X,X} \\ & \quad = \\ & \tilde{C}_{X,X} \end{array}$$

where $\overset{\exists\iota}{\subseteq}$ means that such an inclusion holds if there exists $\iota : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X)$.

Note that the inclusion beside the sign (?) can not directly obtain from Fact 2.1.23 and Theorem 2.1.26. However, it is an easy exercise to verify that $\tilde{C}_{X,X} \subseteq \tilde{C}_{Y,X}$ (hint: consider two separated cases, i.e. whether there is a Σ-homomorphism $\iota : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X)$). Hence, (i) since $\mathbf{T}_\Sigma(X) \preceq \mathbf{T}_\Sigma(Y)$ $(X \subseteq Y)$, the first part of Fact 2.2.3 roughly says that if an equation is satisfied by a bigger algebra, then it is certainly satisfied by a smaller one. So, the sizes of algebras have no impact on satisfactions.

(ii) The second part of the same fact (Fact 2.2.3) indicates that if an equation is satisfied by an algebra with a larger variable index, then it is satisfied by the same algebra with a smaller index provided that the larger index can be expressed by the smaller one under the signature Σ. Therefore, the variable indices play roles for empty carriers rather than the size of algebras. This result justifies our formalism for equations.

Now, Question 2.2.1 becomes Question 2.2.4, which is more interesting.

Question 2.2.4 (variable index eliminable in $\vdash_{EQ}$): *Under what condition is the variable index eliminable in $\vdash_{EQ}$?*

Since given an $X \subseteq \vec{\chi}$ where $|X_i| \neq \emptyset$ for each $i \in I$, we have that every $Y \subseteq \vec{\chi}$ can certainly be expressed by X, i.e. there is a Σ-homomorphism $\iota : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X)$. For example, let $|X_i| = 1$ for each $i \in I$ and there is a unique $\iota : Y \to X$ such that $\iota^\# : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X)$.

On the other hand, since for anther $Y \subseteq \vec{\chi}$ provided $|Y_i| = 1$ for each $i \in I$, there is a Σ-isomorphism $\iota : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X)$. So, for X and Y, we have that if every $\vec{Z} \subseteq \vec{\chi}$ can be expressed by such a Y, then $\vec{Z}$ can be expressed by X; and vice versa. Thus, we can fixed an X which has the property $|X_i| = 1$ for each $i \in I$.

For the fixed X, if there is an $i \in I$ such that $\mathbf{T}_\Sigma(X - \chi_i)_i \neq \emptyset$, then for all $Y \subseteq \vec{\chi}$, there is a Σ-homomorphism $\iota : \mathbf{T}_\Sigma(Y) \to \mathbf{T}_\Sigma(X - X_i)$.

Further, if $|I| > 1$ and there is an $i \in I$ such that $\mathbf{T}_\Sigma(X - \chi_i)_i = \emptyset$ (i.e. the variable in X_i can not be expressed by other variable in X), then there is a $\tilde{\Gamma}$ such that the variable indices in equations are not eliminable in the deduction system $\vdash_{EQ}$.

Example 2.2.5 (non-eliminable index): *Let $\mathbf{T}_\Sigma(X - \chi_i)_i = \emptyset$ for a signature Σ. A simple example for a variable index non-eliminable $\tilde{\Gamma}$ is $\{y \simeq_{\{x,y,z\}} z\}$ where $x \in X_i$ and $y, z \in X_j$ $(j \in I)$ provided $y \neq z$ and $i \neq j$ (hint: considering to eliminate x from the index $\{x, y, z\}$).*

Consequently, we have what we want, Theorem 2.2.6.

Theorem 2.2.6 (variable index eliminable condition): *Given a signature Σ, the following three statements are equivalent:*

1. *for all $i, j \in I$, $\mathbf{T}_\Sigma(X_j)_i \neq \emptyset$;*
2. *for all $Y \subseteq \vec{\chi}$ and $Y \neq \emptyset$, $\mathbf{T}_\Sigma(Y)_i \neq \emptyset$ for each $i \in I$;*
3. *the variable indices in $\vdash_{EQ}$ are eliminable.*

Proof

The implication from 2 to 1 is trivial, the implication from 1 to 3 is obtained by examining Definition 2.2.2, and the implication from 3 to 2 is by Example of 2.2.5. □

Theorem 2.2.6 says that as long as the given signature Σ has the property of 1 in Theorem 2.2.6, calculus $\vdash_{EQ}$ is always variable index free, no matter what kind of $\tilde{\Gamma}$ would be possibly provided. Such a result is very strong and it is very easy to check as well, e.g. see Example 2.2.7. It is very interesting to point out that such a condition is totally independent of possible algebras and only relates to the signatures.

Example 2.2.7 (variable index free $\vdash_{EQ}$):

1. *If it is single-orted case, i.e.* $|I| = 1$, *then* $\vdash_{EQ}$ *is variable index free.*

2. *for* $|I| > 1$, *if for each* $i \in I$ $\Sigma_{\epsilon,i} \neq \emptyset$, *then* $\vdash_{EQ}$ *is variable index free. This example justifies a commonly used technique of getting variable free* $\vdash_{EQ}$ *by introducing extra constants.*

As a summary of last section (Section 2.1) and this section, we have reached a sound and complete $\vdash_{EQ}$ for Indistinguishability, and also a nice necessary and sufficient condition is found for checking whether $\vdash_{EQ}$ is variable index free, i.e. the usual deduction works in the many-sorted situation under such a condition.

2.3 Dependent equations

In this section, we are studying the dependent Indistinguishability, or dependent implications. We start with definitions on satisfactions of dependent equations below (see Definition 2.3.1).

Definition 2.3.1 (satisfaction $\models_{dEQ}$): *Let* $\mathbf{D}$ *be a* Σ*-algebra,* γ_X *be a set of equations,* Δ_X *be an equation,* $\mathcal{K}$ *be a class of* Σ*-algebras, and* Γ *be a set of dependent equations. Thus,*

1. $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *(or* $\mathbf{D} \models \gamma_X \mapsto \Delta_Y$ *for short) iff* $\mathbf{D} \models \gamma_X$ *implies* $\mathbf{D} \models \Delta_X$*;*

2. $\mathcal{K} \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *(or* $\mathcal{K} \models \gamma_X \mapsto \Delta_Y$ *for simplicity) iff* $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *for each* $\mathbf{D} \in \mathcal{K}$*;*

3. $\mathbf{D} \models_{dEQ} \Gamma$ *(or* $\mathbf{D} \models \Gamma$ *as an abbreviation) iff* $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *for each* $\gamma_X \mapsto \Delta_Y \in \Gamma$;

4. $\Gamma \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *(or simply* $\Gamma \models \gamma_X \mapsto \Delta_Y$*) iff* $\mathbf{D} \models_{dEQ} \Gamma$ *implies* $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_Y$ *for every* $\mathbf{D}$.

In $\gamma_X \mapsto \Delta_Y$, we say γ_X as its *pre-condition* (or *premise*) and Δ_Y as its *post-condition* (or *conclusion*). For its satisfaction, there is a simple but important equivalence for it (see Fact 2.6.2.1), i.e.

> $\mathbf{D} \models \gamma_X \mapsto t \simeq_Y t'$ *iff* $\mathbf{D} \models u \simeq_X u'$ *for each* $u \simeq_X u' \in \gamma_X$ *implies* $\mathbf{D} \models t \simeq_Y t'$.

Also, we would like to mention that $\Gamma \models \gamma_X \mapsto \Delta_Y$ in Definition 2.3.1 is equivalent to $Alg_{\Sigma,\Gamma} \models \gamma_X \mapsto \Delta_Y$ (where $\mathbf{D} \in Alg_{\Sigma,\Gamma}$ iff $\mathbf{D} \models \Gamma$) in the sense that they share the same consequences.

For simplicity, we always assume $X = Y$ in $\{t_m \simeq_X t'_m | m \in M\} \mapsto (t \simeq_Y t')$; otherwise, we can replace it by $\{t_m \simeq_{X \cup Y} t'_m | m \in M\} \mapsto t \simeq_{X \cup Y} t'$. This assumption about indices will be applied in later part of this thesis. Also, we notice that a dependent Σ-equation $\gamma_X \mapsto \Delta_X$ can be viewed as a Σ-equation Δ_X, when $\gamma_X = \emptyset$ (i.e. $|M| = 0$), since $\mathbf{D} \models_{dEQ} \emptyset \mapsto \Delta_X$ iff $\mathbf{D} \models_{EQ} \Delta_X$. So, we assume Δ_X as an abbreviation of $\emptyset \mapsto \Delta_X$. This also provides us a clue to understand that dependent Indistinguishability corresponds to Indistinguishability. This formally has the property below, see Property 2.3.2.

Property 2.3.2 (equations and dependent equations): *Let* Γ *be a set of dependent* Σ*-equations,* $Eq(\Gamma) =_{df} \{\Delta_X | \Delta_X \in \Gamma\}$, *and* $\mathbf{D} \in Alg_{\Sigma,Eq(\Gamma)}$.

(a) If $\mathbf{D} \not\models \gamma_X$ *then* $\mathbf{D} \in Alg_{\Sigma,Eq(\Gamma) \cup \{\gamma_X \mapsto \Delta_X\}}$.

(b) If $\mathbf{D} \models \gamma_X$ *then* $\mathbf{D} \in Alg_{\Sigma,Eq(\Gamma) \cup \{\gamma_X \mapsto \Delta_X\}}$ *iff* $\mathbf{D} \models \Delta_X$.

(c) If replaced $Eq(\Gamma)$ *by* Γ, *the above two properties (a) and (b) hold.*

From Theorem 2.1.19, we can have the following corollary (Corollary 2.3.3). Formally,

Corollary 2.3.3 (a similar Birkhoff's Theorem): *Let $\mathcal{K}$ be a class of Σ-algebras, Ker_X be $\bigcap_{\mathbf{D}\in\mathcal{K}} Ker_X(\mathbf{D})$. Thus, for the following four statements, (a) 1 and 2 are equivalent with each other, (b) 3 and 4 are equivalent with each other, and (c) 2 implies 3:*

1. $\mathcal{K} \models \{t_m \simeq_X t'_m | m \in M\} \mapsto t \simeq_X t'$;
2. *for each* $\mathbf{D} \in \mathcal{K}$, *if* $<t_m, t'_m> \in Ker_X(\mathbf{D})$ *for every* $m \in M$ *(or* $\gamma_X \subseteq Ker_X(\mathbf{D})$*), then* $<t, t'> \in Ker_X(\mathbf{D})$ *(or* $\Delta_X \in Ker_X(\mathbf{D})$*);*
3. *if* $<t_m, t'_m> \in Ker_X$ *for all* $m \in M$ *(or* $\gamma_X \subseteq Ker_X$*), then* $<t, t'> \in Ker_X$ *(or* $\Delta_X \in Ker_X$*);*
4. $\mathbf{T}_\Sigma(X)/Ker_X \models \{t_m \simeq_X t'_m | m \in M\} \mapsto t \simeq_X t'$.

Note that the above 2 and 3 together correspond 2 in Theorem 2.1.20. Also, observe that in Corollary 2.3.3, 2 is not in general equivalent to 3. This makes a sharp difference between Theorem 2.1.20 and Corollary 2.3.3. However, when $M = \emptyset$, the four statements in Corollary 2.3.3 are equivalent with each other. In other words, analogous to Section 2.1, we immediately get a sound and complete calculus $\vdash^-_{dEQ}$ with respect to Σ-equations. The reasoning runs the same as the one in Section 2.1. Based on this, we give its definition below (Definition 2.3.4).

Definition 2.3.4 (calculus $\vdash^-_{dEQ}$): *The follwing calculus $\vdash^-_{dEQ}$ of dependent equations is defined in the judgement form of*

$$\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X \text{ (or simply } \Gamma \vdash \gamma_X \mapsto \Delta_X),$$

where Γ is a set of dependent Σ-equations, γ_X is a set of equations and Δ_X is an equation. Then,

1. *(d-id)*
$$\{\gamma_X \mapsto \Delta_X\} \vdash \gamma_X \mapsto \Delta_X$$
2. *(d-rfl)*
$$\vdash^-_{dEQ} t \simeq_X t \ (t \in \mathbf{T}(X))$$

3. (d-sym')

$$\{t \simeq_X u\} \vdash^-_{dEQ} u \simeq_X t$$

4. (d-trs')

$$\{t \simeq_X u, u \simeq_X v\} \vdash^-_{dEQ} t \simeq_X v$$

5. (d-cmp')

$$\{t_n \simeq_X u_n | 1 \leq n \leq m\} \vdash^-_{dEQ} \sigma(\vec{t}) \simeq_X \sigma(\vec{u})$$

where $\sigma \in \Sigma_{i_1 i_2 \ldots i_m, i}$ *and* $t_n, u_n \in \mathbf{T}(X)_{i_n}$ *for* $n = 1, 2, \ldots, m$.

6. (d-sub)

$$\frac{\Gamma \vdash^-_{dEQ} \emptyset \mapsto \Delta_X}{\Gamma \vdash^-_{dEQ} \emptyset \mapsto \iota(\Delta_X)}$$

where $\iota : \mathbf{T}(X) \to \mathbf{T}(Y)$, $\iota(t \simeq_X u) =_{df} \iota(t) \simeq_Y \iota(u)$ *and* $\iota(\gamma_X)$ *is its natural extension to a set of* Σ*-equations (note :* $\iota(\emptyset) = \emptyset$*).*

7. (d-wkn)

$$\frac{\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X}{\Gamma \cup \Gamma' \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X}$$

8. (d-cut) (or (d-MP))

$$\frac{\{\Gamma \vdash^-_{dEQ} \gamma'_X \mapsto \Delta'_X | \gamma'_X \mapsto \Delta'_X \in \Gamma'\}; \Gamma' \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X}{\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X}$$

9. (d-ctr)

$$\frac{\{\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta'_X | \Delta'_X \in \gamma'_X\}; \Gamma \vdash^-_{dEQ} \gamma'_X \mapsto \Delta_X}{\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X}$$

Also, we can define $\tilde{\mathcal{M}}^-_{dEQ}$ as a function which produces a set $\tilde{\mathcal{M}}^-_{dEQ}(\Gamma)$ of dependent equations when given a set Γ of dependent equations; i.e. $\gamma_X \mapsto \Delta_X \in \tilde{\mathcal{M}}^-_{dEQ}(\Gamma)$ iff $\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X$ without using (d-cut) rules. Then, $\tilde{\mathcal{M}}^-_{dEQ}$ is monotonic, and the least fixed point of $\tilde{\mathcal{M}}^-_{dEQ}$ containing Γ, written as $\sqcup\tilde{\mathcal{M}}^-_{dEQ}(\Gamma)$ is equivalent to the derivability of $\vdash^-_{dEQ}$. In other words, $\Gamma \vdash^-_{dEQ} \gamma_X \mapsto \Delta_X$ iff $\gamma_X \mapsto \Delta_X \in \sqcup\tilde{\mathcal{M}}^-_{dEQ}(\Gamma)$.

Actually, the soundness and completeness of $\vdash^{-}_{dEQ}$ with respect to Σ-equations are $\phi(\sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma))\lceil_{\simeq} = \bigcap_{\mathbf{D}\in Alg_{\Sigma,\Gamma}} \check{\mathcal{K}}(\mathbf{D})$, where $\phi(\sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)_X)\lceil_{\simeq} = \{<t,t'> \mid \emptyset \mapsto t \simeq_X t' \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\check{\Gamma})\}$. This can be achieved by the same reason as $\tilde{\mathcal{M}}$ with the fact that the rules from (d-id) to (d-sub) and (d-MP) of $\vdash^{-}_{dEQ}$ include all rules of $\vdash_{EQ}$.

Therefore, the soundness and completeness of $\vdash^{-}_{dEQ}$ for all possible dependent Σ-equations are whether we have that $\gamma_X \mapsto \Delta_X \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$ iff $\gamma_X \subseteq \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$ implies $\Delta_X \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$ (i.e. $\emptyset \mapsto \Delta_X \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$ when $\emptyset \mapsto \Delta'_X \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$ for every $\Delta'_X \in \gamma_X$). The proof for the soundness direction is easy. But for the completeness direction, we have a counter-example, see Example 2.3.5.

Example 2.3.5 (counter-example to completeness of $\vdash^{-}_{dEQ}$): *$\vdash^{-}_{dEQ}$ is incomplete for the dependent Σ-equations. A counter-example against completeness of $\vdash^{-}_{dEQ}$ is provided as follows:*

let bool be a sort in I, $\Sigma_{i,bool} = \{\mathbf{true}\}$ (assuming $i = bool$ for simplicity) and other $\Sigma_{\kappa,j} = \emptyset$, Γ is $\{\mathbf{true}(x) \simeq_{\{x,y\}} \mathbf{true}(y)\}$.

Also, let $\mathbf{D}$ be an Σ-algebra such that $A_{bool} = \{true, false\}$ and **true** *is the constant function which maps everything to true. Apparently, no algebra can satisfy $x \simeq_{\{x,y\}} y$ (except trivial ones), say $\mathbf{D} \not\models x \simeq_{\{x,y\}} y$. Obviously, $\mathbf{D} \models \mathbf{true}(x) \simeq_{\{x,y\}} y$ provided that $\mathbf{D} \models x \simeq_{\{x,y\}} y$. So, $x \simeq_{\{x,y\}} y \mapsto \mathbf{true}(x) \simeq_{\{x,y\}} y$ is always satisfied by any algebra, i.e. $\mathbf{D} \models x \simeq_{\{x,y\}} y \mapsto \mathbf{true}(x) \simeq_{\{x,y\}} y$ for every algebra $\mathbf{D}$. However, it is not derivable by $\vdash^{-}_{dEQ}$. The full proof is omitted but a hint is given as follows: to show this, you can define $n_{\mathbf{true}}(t)$ as the number of* **true** *occurring in t and show that every derivable $\gamma_X \mapsto t \simeq_X u$ has the property of either $t = u$ or $t, u \in \gamma_X$ or $n_{\mathbf{true}}(t) + n_{\mathbf{true}}(u) > 1$.*

So, we have to extend $\vdash^{-}_{dEQ}$ in order to get a complete calculus for dependent equations.

Definition 2.3.6 (calculus $\vdash_{dEQ}$): *Calculus $\vdash_{dEQ}$ is defined as follows:*

1. *$\vdash_{dEQ}$ contains all rules in $\vdash^{-}_{dEQ}$, and an extra rule below*

2. *(d-intr)*

$$\frac{\Gamma \cup \{\emptyset \mapsto t \simeq_X t' | t \simeq_X t' \in \gamma_X\} \vdash_{dEQ} \Delta_X}{\Gamma \vdash_{dEQ} \gamma_X \mapsto \Delta_X}$$

Since $\vdash^{-}_{dEQ}$ is complete for Σ-equations and $\vdash_{dEQ}$ does not increase the derivable Σ-equations of $\vdash^{-}_{dEQ}$, we have that $\vdash_{dEQ}$ is sound and complete with respect to Σ-equations. For soundness and completeness of $\vdash_{dEQ}$, we have a theorem below (Theorem 2.3.7).

Theorem 2.3.7 (soundness and completeness of $\vdash_{dEQ}$): *$\vdash_{dEQ}$ is a sound and complete deduction system of dependent Σ-equations. i.e. Given Γ,*

$$\Gamma \vdash_{dEQ} \gamma_X \mapsto \Delta_X \text{ iff } \Gamma \models_{dEQ} \gamma_X \mapsto \Delta_X.$$

Proof

The proof for the soundness direction is easy and is left out.

For the completeness direction, we know that for every $\mathbf{D} \models \Gamma$, $\mathbf{D} \models \gamma_X \mapsto \Delta_X$ iff $\mathbf{D} \models \gamma_X$ implies $\mathbf{D} \models \Delta_X$. Let us suppose $\Gamma \models \gamma_X \mapsto \Delta_X$. That is, for every $\mathbf{D} \models \Gamma$, if $\mathbf{D} \models \gamma_X$ then $\mathbf{D} \models \Delta_X$. This is equivalent to that for every $\mathbf{D}' \models \Gamma \cup \{\emptyset \mapsto t \simeq_X t' | t \simeq_X t' \in \gamma_X\}$, $\mathbf{D}' \models \Delta_X$. By the completeness of $\vdash^{-}_{dEQ}$ with respect to equations, we have that $\Gamma \cup \{\emptyset \mapsto t \simeq_X t' | t \simeq_X t' \in \gamma_X\} \vdash^{-}_{dEQ} \Delta_X$. So, we have $\Gamma \vdash_{dEQ} \gamma_X \mapsto \Delta_X$ by (d-intr) rule (in Definition 2.3.6). □

Before leaving for next section, we make a few comments on the substitution rule in $\vdash_{dEQ}$. There are other two forms for (d-sub) in $\vdash_{dEQ}$ They are

1. (d-post-sub)

$$\Gamma \vdash \Delta_X \mapsto \iota(\Delta_X) \quad (\iota : \mathbf{T}(X) \to \mathbf{T}(Y));$$

and

2. (d-both-sub)

$$\frac{\Gamma \vdash \gamma_X \mapsto \Delta_X; \{\Gamma \vdash \Delta'_X | \Delta'_X \in \gamma_X\}}{\Gamma \vdash \iota(\gamma_X) \mapsto \iota(\Delta_X)} \ (\iota : \mathbf{T}(X) \to \mathbf{T}(Y)).$$

These three substitution rules are equivalent to each other in the sense of that we can obtain three deduction systems by using each of them in $\vdash_{dEQ}$, and the resulting deduction systems for share the same set of derivable dependent equations. However, these three rules can not be generalized to the following

$$(d\text{-}wrong\text{-}sub) \quad \frac{\Gamma \vdash \gamma_X \mapsto \Delta_X}{\Gamma \vdash \iota(\gamma_X) \mapsto \iota(\Delta_X)} \, (\iota : \mathbf{T}(X) \to \mathbf{T}(Y)).$$

We further remark that when (d-intr) is available, we can slightly modified some rules in $\vdash_{dEQ}^{-}$ and obtain an equivalent variant of $\vdash_{dEQ}$ which presents the calculus in a uniform way; i.e. to replace (d-sym'), (d-trs') and (d-cmp') by the following (d-sym), (d-trs) and (d-cmp) respectively.

1. (d-sym)

$$\vdash \{t \simeq_X u\} \mapsto u \simeq_X t$$

2. (d-trs)

$$\vdash \{t \simeq_X u, u \simeq_X v\} \mapsto t \simeq_X v$$

3. (d-cmp)

$$\vdash \{t_n \simeq_X u_n | 1 \leq n \leq m\} \mapsto \sigma(\vec{t}) \simeq_X \sigma(\vec{u})$$

where $\sigma \in \Sigma_{i_1 i_2 \ldots i_m, i}$ and $t_n, u_n \in \mathbf{T}(X)_{i_n}$ for $n = 1, 2, \ldots, m$.

The condition for variable index free $\vdash_{dEQ}$ is still the same as it expressed for $\vdash_{EQ}$ in Section 2.2.

2.4 Quasi-dependent equations

According to their definitions (see Definition 2.0.6), quasi-dependent equations are close to dependent equations. For example, we have the following simple fact but very important one (see Fact 2.6.3.2):

> *Two kinds of implications "$\mapsto$" and "$\hookrightarrow$" are the same one if we only consider* ground cases *(or closed terms).*

Formally, when γ_X and Δ_X only contain ground terms, $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_X$ iff $\mathbf{D} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$, where $\mathbf{D}$ is a Σ-algebra. However, we will soon discover that quasi-dependent equations are more complicated than dependent equations.

Definition 2.4.1 (satisfaction $\models$): *Let $\gamma_X \hookrightarrow \Delta_X$ be a quasi-dependent Σ-equation (we assume that $\emptyset \hookrightarrow \Delta_X$ is Δ_X) and $\mathbf{D}$ be a Σ-algebra. Thus,*

(i) $\mathbf{D} \models_{qEQ} \Gamma$ (or $\mathbf{D} \models \Gamma$ for short) iff $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$ for each $\gamma_X \hookrightarrow \Delta_X \in \Gamma$,

where Γ be an indexed family of $\{\Gamma_X \mid X \subseteq \vec{\chi}\}$ of quasi-dependent Σ-equations, and $\emptyset \hookrightarrow \Delta_X \in \Gamma$ means a Σ-equation;

(ii) $\mathcal{K} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$ (or $\mathcal{K} \models \gamma_X \hookrightarrow \Delta_X$ for simplicity) iff $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$ for each $\mathbf{D} \in \mathcal{K}$,

where $\mathcal{K}$ is a collection of Σ-algebras.

(iii) $\Gamma \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$ (or $\Gamma \models \gamma_X \hookrightarrow \Delta_X$ as an abbreviation) iff $\mathbf{D} \models \Gamma$ implies $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$ for every $\mathbf{D}$.

We point out that $\Gamma \models \gamma_X \hookrightarrow \Delta_X$ is equivalent to $Alg_{\Sigma,\Gamma} \models \gamma_X \hookrightarrow \Delta_X$ (where $\mathbf{D} \in Alg_{\Sigma,\Gamma}$ iff $\mathbf{D} \models_{qEQ} \Gamma$) in the sense that they share the same consequences.

For the relationship between dependent equations and quasi-dependent equations, we define an obvious translation between them and we have the following.

Definition 2.4.2 (q-d): *A natural translation q-d from quasi-dependent Σ-equations to dependent Σ-equations is defined as $q\text{-}d[\![\gamma_X \hookrightarrow \Delta_X]\!] =_{df} \gamma_X \mapsto \Delta_X$.*

We can extend *q-d* naturally from defining on an individual quasi-dependent equation to defining on a collection of quasi-dependent Σ-equations. Therefore, we would know that there is certain link between dependent equations and quasi-dependent equations through *q-d*. For example, given a Γ which only contains quasi-dependent equations, and let Γ^d be $q\text{-}d[\![\Gamma]\!]$, from $\mathbf{D} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$ implies $\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_X$, we have that

(2.4.a) $Alg_{\Sigma,\Gamma} \subseteq Alg_{\Sigma,\Gamma^d}$, hence, $Alg_{\Sigma,\Gamma^d} \models_{dEQ} t \simeq_X t'$ implies $Alg_{\Sigma,\Gamma} \models_{qEQ} t \simeq_X t'$;

(2.4.b) $\mathcal{K} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$ implies $\mathcal{K} \models_{dEQ} \gamma_X \mapsto \Delta_X$, for (non-empty) $\mathcal{K}$.

Note that when we only consider pure equations, the three satisfaction relations $\models_{EQ}, \models_{dEQ}$ and $\models_{qEQ}$ coincide with each other.

From (2.4.b), we understand that $\vdash_{qEQ} \subseteq \vdash_{dEQ}$ if we consider derivabilities as relations. From (2.4.a), we have whenever a Σ-equation can be deduced by $\vdash_{dEQ}$, it can also be deduced by $\vdash_{qEQ}$ (if there is a sound and complete $\vdash_{qEQ}$). Based on this, we know that $\vdash_{qEQ}$ and $\vdash_{dEQ}$ are equivalent if we only consider Σ-equations.

Analogous to Section 2.1, we try to establish a similar result of Theorem 2.3.3 for quasi-dependent Σ-equations, which composes of two lemmas (Lemma 2.4.3 and Lemma 2.4.4) and one example (Example 2.4.5) below. Formally

Lemma 2.4.3 (index congruence, **its quotient algebra and substitutions**): *Let $\mathcal{K}$ be a class of Σ-algebras. Then, the following two statements are equivalent:*

1. $\mathbf{T}(X)/Ker_X(\mathbf{D}) \models \gamma_X \hookrightarrow \Delta_X$.

2. *if* $\imath(\gamma_X) \subseteq Ker_X(\mathbf{D})$, *then* $\imath(\Delta_X) \in Ker_X(\mathbf{D})$, *for every* $\imath : \mathbf{T}(X) \to \mathbf{T}(X)$.

Proof

(a) In general, we have that for each $\imath' : \mathbf{T}(X) \to \mathbf{T}(X)/Ker_X(\mathbf{D})$ there is a $\imath' : \mathbf{T}(X) \to \mathbf{T}(X)$ such that $\imath' = \nu_{Ker_X(\mathbf{D})} \circ \imath'$ and vice versa. Hence, we have that for every $u \simeq_X u'$, $\imath'(u) = \imath'(u')$ iff $<\imath'(u), \imath'(u')> \in Ker_X(\mathbf{D})$.

(b) For "2 $\Rightarrow$ 1", i.e. in the case of $\gamma_X \hookrightarrow \Delta_X$, we have that $\iota'(\gamma_X)$ implies that for each m, $<\iota'(t_m), \iota'(t'_m)> \in Ker_X(\mathbf{D})$. By 2 (of Lemma 2.4.3), we have $<\iota'(t), \iota'(t')> \in Ker_X(\mathbf{D})$, i.e. $\iota'(t) = \iota'(t')$.

(c) For "1 $\Rightarrow$ 2", we have that for every ι', suppose that for each m, $<\iota'(t_m), \iota'(t'_m)> \in Ker_X(\mathbf{D})$, i.e. $\iota'(\gamma_X)$.

Since $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$, $\iota'(\gamma_X)$ implies $\iota'(\Delta_X)$, i.e. $<\iota'(t), \iota'(t')> \in Ker_X(\mathbf{D})$. □

Before proceeding further, we look at the opposite of 2 of Lemma 2.4.3, i.e. there exists a ι such that $\iota(\gamma_X) \subseteq Ker_X(\mathbf{D})$ and $\iota(\Delta_X) \notin Ker_X(\mathbf{D})$. In other words, γ_X becomes a valid Σ-equation by a substitution ι but not Δ_X. An understanding of this will lead to a proof of the following lemma (Lemma 2.4.4).

Lemma 2.4.4 (implication between satisfaction and substitutions): $\mathbf{D} \models \gamma_X \hookrightarrow \Delta_X$ *implies that if* $\iota(\gamma_X) \subseteq Ker_X(\mathbf{D})$ *then* $\iota(\Delta_X) \in Ker_X(\mathbf{D})$, *for every* $\iota : \mathbf{T}(X) \to \mathbf{T}(X)$.

Proof

Suppose the opposite of that for every $\iota' : \mathbf{T}(X) \to \mathbf{T}(X)$, if $\iota'(\gamma_X) \subseteq Ker_X(\mathbf{D})$, then $\iota'(\Delta_X) \in Ker_X(\mathbf{D})$. Then we have a ι'' such that $\iota''(\gamma_X) \subseteq Ker_X(\mathbf{D})$ and $\iota''(\Delta_X) \notin Ker_X(\mathbf{D})$. In other words, $(\mathbf{D}, \iota'') \not\models \gamma_X \hookrightarrow \Delta_X$. □

For the reversed implication between satisfactions and substitutions, we give a counter-example.

Example 2.4.5 (counter-example to the reversed implication in Lemma 2.4.4): *Let* $\mathbf{D}$ *and* Γ *be the same as in Example 2.3.5. Thus, (a)* Γ *is sound in* $\mathbf{D}$.

(b) $x \simeq_{\{x,y\}} y \hookrightarrow \mathrm{true}(x) \simeq_{\{x,y\}} y$ *is not sound in* $\mathbf{D}$ *(consider the assignment of both* x *and* y *to value* $false$*), i.e.*

$$\mathbf{D} \not\models x \simeq_{\{x,y\}} y \hookrightarrow \mathrm{true}(x) \simeq_{\{x,y\}} y.$$

(c) However, for every substitution $\iota : \mathbf{T}(X) \to \mathbf{T}(X)$,

$$\iota(x \stackrel{\cdot}{\simeq}_{\{x,y\}} y) \subseteq Ker_X(\mathbf{D}) (or\ <\iota(x), \iota(y)> \in Ker_X(\mathbf{D})>)$$

implies

$$\hat{\imath}(\mathrm{true}(x) \simeq_{\{x,y\}} y) \in Ker_X(\mathbf{D}),$$

(Note that by Birkhoff's Theorem we know that $\mathbf{D} \models t \simeq_X u$ *iff* $< t, u > \in Ker_X(\mathbf{D})$ *and see Example 2.6.3.1 for more).* □

From Lemma 2.4.3 and Lemma 2.4.4, we understand that quasi-dependent deduction systems have a very close relation with dependent deduction systems. This close relation has been explicitly expressed, say in 2 of Lemma 2.4.3. So, we introduce a deduction $\vdash^d_{qEQ}$ containing $\vdash_{dEQ}$ as follows.

Definition 2.4.6 (calculus $\vdash^d_{qEQ}$:) *The following calculus* $\vdash^d_{qEQ}$ *of dependent and quasi-dependent equations is defined in the judgement forms of*

$$\text{either } \Gamma \vdash^d_{qEQ} \gamma_X \mapsto \Delta_X \text{ or } \Gamma \vdash^d_{qEQ} \gamma_X \hookrightarrow \Delta_X,$$

i.e. it contains rules in $\vdash_{dEQ}$ *and the following extra rules (where* Γ *is a set of dependent and quasi-dependent equations):*

1. *(q-id)*

$$\{\gamma_X \hookrightarrow \Delta_X\} \vdash^d_{qEQ} \gamma_X \hookrightarrow \Delta_X$$

2. *(q-rfl)*

$$\vdash^d_{qEQ} t \simeq_X t$$

where $t \simeq_X u$ *is short for either* $\emptyset \hookrightarrow t \simeq_X u$ *or* $\emptyset \mapsto t \simeq_X u$ *and the possible confusion may be resolved by context.*

3. *(q-sym)*

$$\vdash^d_{qEQ} \{t \simeq_X u\} \hookrightarrow u \simeq_X t$$

4. *(q-trs)*

$$\vdash^d_{qEQ} \{t \simeq_X u, u \simeq_X v\} \hookrightarrow t \simeq_X v$$

5. *(q-cmp)*

$$\vdash^d_{qEQ} \{t_m \simeq_X u_m | 1 \leq m \leq n\} \hookrightarrow \sigma(\vec{t}\,) \simeq_X \sigma(\vec{u}\,)$$

where $\sigma \in \Sigma_{i_1 i_2 \ldots i_n, i}$ *and* $t_m, u_m \in \mathrm{T}(X)_{i_m}$ *for* $m = 1, 2, \ldots, n$.

6. (q-sub)

$$\frac{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}{\Gamma \vdash^{d}_{qEQ} \iota(\gamma_X) \hookrightarrow \iota(\Delta_X)} \, (\iota : \mathbf{T}(X) \to \mathbf{T}(Y))$$

7. (q-wkn)

$$\frac{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}{\Gamma \cup \Gamma' \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}$$

8. (q-cut)

$$\frac{\{\Gamma \vdash^{d}_{qEQ} \gamma'_X \hookrightarrow \Delta'_X | \gamma'_X \hookrightarrow \Delta'_X \in \Gamma'\}; \Gamma' \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}$$

9. (q-d)

$$\frac{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}{\Gamma \vdash^{d}_{qEQ} \gamma_X \mapsto \Delta_X}$$

10. (q-ctr)

$$\frac{\{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta'_X | \Delta'_X \in \gamma'_X\}; \Gamma \vdash^{d}_{qEQ} \gamma'_X \hookrightarrow \Delta_X}{\Gamma \vdash^{d}_{qEQ} \gamma_X \hookrightarrow \Delta_X}$$

Since the rules from (q-id) to (q-ctr) of $\vdash^{d}_{qEQ}$ imply all rules of $\vdash_{EQ}$, the soundness and completeness of $\vdash^{d}_{qEQ}$ with respect to Σ-equations can be obtained by a same reasoning as in Section 2.1. On the other hand, analogous to the reasoning in Section 2.3 for $\vdash_{dEQ}$ we come to that $\vdash^{d}_{qEQ}$ is sound and complete with respect to dependent equations. In order to get a complete deduction for quasi-dependent equations, we need to further extend $\vdash^{d}_{qEQ}$. One possible choice is $\vdash^{+}_{qEQ}$ which contains all rules in $\vdash^{d}_{qEQ}$ along with an extra rule below

$$(\omega\text{-}d\text{-}q) \quad \frac{\{\Gamma \vdash^{+}_{qEQ} \iota(\gamma_X) \mapsto \iota(\Delta_X) | \iota : \mathbf{T}(X) \to \mathbf{T}(Y)\}}{\Gamma \vdash^{+}_{qEQ} \gamma_X \hookrightarrow \Delta_X}.$$

This rule is to reverse the implication in Lemma 2.4.4 as a mean to extend $\vdash^{d}_{qEQ}$. Informally, (ω-d-q) is to collect all those dependent Σ-equations which are closed under substitutions and make them become quasi-dependent Σ-equations. However, such a collecting get more quasi-dependent Σ-equations than necessary, see Example 2.4.5. In other words, $\vdash^{+}_{qEQ}$ is complete but unsound for quasi-dependent Σ-equations.

To introduce the correct extension of $\vdash^{d}_{qEQ}$, we should remind you of the fact stated at the beginning of this section. Further, we have the following.

Theorem 2.4.7 (skolemization): *Given* Γ, *let* $x_1, x_2, ..., x_n$ *be all possible free variables in* γ_X *and* Δ_X, *and* $\Sigma' = \Sigma \cup \{c_1, c_2, ..., c_n\}$ *(where* $c_1, c_2, ..., c_n$ *are fresh constants not occurring in signature* Σ *and* x_j *and* c_j *share a same sort for* $1 \leq j \leq n$*), then* $Alg_{\Sigma,\Gamma} \models \gamma_X \hookrightarrow \Delta_X$ *iff* $Alg_{\Sigma',\Gamma} \models \delta(\gamma_X) \hookrightarrow \delta(\Delta_X)$, *where* $\delta : X \to \mathbf{T}_{\Sigma'}(X - \{x_1, x_2, ..., x_n\})$ *and* $\delta(x_i) = c_i$.

Proof

Since $\mathbf{D} \in Alg_{\Sigma',\Gamma}$ can be regarded as a Σ-algebra, we have $\mathbf{D} \in Alg_{\Sigma,\Gamma}$, i.e. $Alg_{\Sigma',\Gamma} \subseteq Alg_{\Sigma,\Gamma}$. Hence, $Alg_{\Sigma,\Gamma} \models \gamma_X \hookrightarrow \Delta_X$ implies $Alg_{\Sigma',\Gamma} \models \gamma_X \hookrightarrow \Delta_X$; in turn, it implies $Alg_{\Sigma',\Gamma} \models \iota'(\gamma_X) \hookrightarrow \iota'(\Delta_X)$, where $\iota' : X \to \mathbf{T}_{\Sigma'}(X)$. In particular, $\iota' : X \to \mathbf{T}_{\Sigma'}(X - \{x_1, x_2, ..., x_n\})$ and $\iota'(x_i) = c_i$.

For the reversed implication, suppose $\mathbf{D}' \in Alg_{\Sigma,\tilde{\mathcal{M}}^q}$, and $\iota : \mathbf{T}(X) \to \mathbf{D}'$, and $(\mathbf{D}', \iota) \models \gamma_X$, we know that $\mathbf{D}'$ can extended to be a Σ'-algebra $\mathbf{D}''$ such that $\mathcal{D}''_\sigma = \mathcal{D}'_\sigma$ for every $\sigma \in \Sigma$ and $\mathcal{D}''_{c_i} = \iota(x_i)$. Apparently, we have $\mathbf{D}'' \models \Gamma$ and $\mathbf{D}'' \models \delta(\gamma_X)$ where $\delta : X \to \mathbf{T}_{\Sigma'}(X - \{x_1, x_2, ..., x_n\})$, $\delta(x_i) = c_i$ and $\delta(y) = y$ for $y \in X - \{x_1, x_2, ..., x_n\}$. By hypothesis, we have $\mathbf{D}'' \models \delta(\Delta_X)$, i.e. $(\mathbf{D}', \iota) \models \Delta_X$. Therefore, $\mathbf{D}' \models \gamma_X \hookrightarrow \Delta_X$. □

From the above proof we notice that replacing rule (ω-*d*-*q*) by

$$(q\text{-}skm) \quad \frac{\Gamma \vdash_{qEQ} \delta(\gamma_X) \mapsto \delta(\Delta_X)}{\Gamma \vdash_{qEQ} \gamma_X \hookrightarrow \Delta_X}$$

where δ is defined in the proof of Theorem 2.4.7, we can obtain a sound and complete calculus $\vdash_{qEQ}$. And rule (*q-skm*) is referred to as *skolemization* techniques (see Subsection 2.6.3 for more).

Definition 2.4.8 (calculus $\vdash_{qEQ}$): *Calculus* $\vdash_{qEQ}$ *contains all rules in* $\vdash^d_{qEQ}$ *with one extra rule (q-skm).*

From the proof of Theorem 2.4.7, we know that the completeness of $\vdash_{qEQ}$ seems to rely on the assumption of non-existence of empty sorts in the signature in general to guarantee the introduction of constants. But if we notice variable assignments are void when there exist empty sorts, then the above assumption is not required in the proof of completeness. So, we have the completeness of $\vdash_{qEQ}$ below.

Theorem 2.4.9 (completeness of $\vdash_{qEQ}$): *$\vdash_{qEQ}$ is sound and complete (with respect to quasi-dependent equations).*

Proof It is based on Theorem 2.4.7 with the fact of that $\vdash^{d}_{qEQ}$ is sound and complete with respect to dependent equations. □

2.5 Universal equations

In the previous sections, we have obtained deduction systems for equations, dependent equations and quasi-dependent equations, i.e. one for each. On the other hand, these three equational forms can be unified into Ω_X (or simply Ω), i.e. the form of

$$\{\gamma^{(m)}_X \hookrightarrow \Delta^{(m)}_X | m \in M\} \mapsto (\gamma_X \hookrightarrow \Delta_X),$$

so-called *universal equations* as pointed out in the introduction part of the present chapter. So, in this section we discuss a deduction system of universal equations.

$\mathbf{D}$ is said to be a model of Ω, written as $\mathbf{D} \models_{uEQ} \Omega$ (or $\mathbf{D} \models \Omega$ for short) iff $\mathbf{D} \models_{qEQ} \{\gamma^{(m)}_X \hookrightarrow \Delta^{(m)}_X | m \in M\}$ implies $\mathbf{D} \models_{qEQ} \gamma \hookrightarrow \Delta$. Therefore, for an universal equation Ω we understand that (1) when $M = \emptyset$ it is a pure quasi-dependent equation; (2) further if $\gamma = \emptyset$, it is a pure equation; (3) when all γ's are empty, it is a pure dependent equation; (4) of course, further if $M = \emptyset$ it is a pure equation again. So, the results in previous sections are very useful for universal equations.

Analogous to previous sections, we can naturally define the followings:

(a) $\mathcal{K} \models_{uEQ} \Omega$ (or $\mathcal{K} \models \Omega$ for short) is the natural extension of $\mathbf{D} \models \Omega$ to a collection $\mathcal{K}$ of algebras;

(b) $\mathbf{D} \models_{uEQ} \Gamma$ (or $\mathbf{D} \models \Gamma$ for simplicity) is $\mathbf{D} \models \Omega$ for each $\Omega \in \Gamma$, which is a collection of Ω's;

(c) $\mathcal{K} \models_{uEQ} \Gamma$ (or $\mathcal{K} \models \Gamma$ as an abbreviation) is the extension of $\mathbf{D} \models \Gamma$ to a collection of algebras.

Hence, some similar results to Lemma 2.4.3 and Lemma 2.4.4 can be obtained and state in the following two of a theorem and of a lemma.

Theorem 2.5.1 (a similar Birkhoff's Theorem): *The following two statements are equivalent:*

1. $\mathbf{T}(X)/Ker_X(\mathbf{D}) \models \Omega$;

2. *whenever for every* $\iota : \mathbf{T}(X) \to \mathbf{T}(X)$, $\iota(\gamma_X^{(m)}) \subseteq Ker_X(\mathbf{D})$ *implies* $\iota(\Delta_X^{(m)}) \in Ker_X(\mathbf{D})$ *for each* $m \in M$, *we would have that for every* $\iota' : \mathbf{T}(X) \to \mathbf{T}(X)$, $\iota'(\gamma) \subseteq Ker_X(\mathbf{D})$ *implies* $\iota'(\Delta) \in Ker_X(\mathbf{D})$.

Comparing with Birkhoff's Theorem, there is one equivalence missing from the above theorem. However, we have an implication for the missing equivalence.

Lemma 2.5.2 (implication of universal satisfaction): $\mathbf{D} \models \Omega$ *implies that whenever for every* $\iota : \mathbf{T}(X) \to \mathbf{T}(X)$, $\iota(\gamma_X^{(m)}) \subseteq Ker_X(\mathbf{D})$ *implies* $\iota(\Delta_X^{(m)}) \in Ker_X(\mathbf{D})$ *for each* $m \in M$, *we would have that for every* $\iota' : \mathbf{T}(X) \to \mathbf{T}(X)$, $\iota'(\gamma) \subseteq Ker_X(\mathbf{D})$ *implies* $\iota'(\Delta) \in Ker_X(\mathbf{D})$.

Nevertheless, we understand from the previous result, see Example 2.4.5, that the reversed implication of Lemma 2.5.2 can not hold in general. However, we can go through a same reasoning as in the previous sections, say Section 2.3 and Section 2.4, and obtain a sound and complete deduction system with respect to equations, dependent equations and quasi-dependent equations. For the completeness up to universal equations, we need the following theorem.

Theorem 2.5.3 (universal equations and quasi-dependent equations): $\Gamma \models \Omega$ *iff* $\mathbf{D} \models \emptyset \mapsto (\gamma_X^{(m)} \hookrightarrow \Delta_X^{(m)})$ *for each* $m \in M$ *implies* $\mathbf{D} \models \emptyset \mapsto (\gamma_X \hookrightarrow \Delta_X)$, *for every* $\mathbf{D} \models \Gamma$.

We should know that the satisfaction relation $\models_{uEQ}$ coincides (1) with $\models_{EQ}$ for equations, (2) with $\models_{dEQ}$ for dependent equations and (3) with $\models_{qEQ}$ for quasi-dependent equations. For convenience, we introduce the following abbreviations.

(i) $t \simeq u$ is short for $\emptyset \mapsto (\emptyset \hookrightarrow t \simeq u)$,

(ii) $\gamma \mapsto t \simeq u$ is for $\{\emptyset \hookrightarrow t' \simeq u' | t' \simeq u' \in \gamma\} \mapsto (\emptyset \hookrightarrow t \simeq u)$,

(iii) $\gamma \hookrightarrow t \simeq u$ is for $\emptyset \mapsto (\gamma \hookrightarrow t \simeq u)$.

From Theorem 2.5.3, we are able to get a sound and complete calculus for universal equations. Now, we give the definition for it below.

Definition 2.5.4 (calculus $\vdash_{uEQ}$): *The following calculus $\vdash_{uEQ}$ for universal equations is defined in the judgement form of*

$$\Gamma \vdash_{uEQ} \Omega_X \text{ or simply } \Gamma \vdash \Omega$$

where Γ is a set of Ω's.

1. *(u-id)*

$$\{\Omega\} \vdash_{uEQ} \Omega$$

2. *(u-rfl)*

$$\Gamma \vdash_{uEQ} t \simeq_X t$$

3. *(u-sym)*

$$\Gamma \vdash_{uEQ} \{t \simeq_X u\} \hookrightarrow u \simeq_X t$$

4. *(u-trs)*

$$\Gamma \vdash_{uEQ} \{t \simeq_X u, u \simeq_X v\} \hookrightarrow t \simeq_X v$$

5. *(u-cmp)*

$$\Gamma \vdash_{uEQ} \{t_m \simeq_X u_m | 1 \leq m \leq n\} \hookrightarrow \sigma(\vec{t}) \simeq_X \sigma(\vec{u})$$

where $\sigma \in \Sigma_{i;i_2 \ldots i_n,i}$ and $t_m, u_m \in \mathbf{T}(X)_{i_m}$ for $m = 1, 2, \ldots, n$.

6. *(u-sub)*

$$\frac{\Gamma \vdash_{uEQ} \gamma_X \hookrightarrow \Delta_X}{\Gamma \vdash_{uEQ} \iota(\gamma_X) \hookrightarrow \iota(\Delta_X)} \; (\iota : \mathbf{T}(X) \to \mathbf{T}(Y))$$

7. *(u-d-ctr)*

$$\frac{\Gamma \vdash_{uEQ} \Omega'; \{\Gamma \vdash_{uEQ} \Omega^{(n)} \mid n \in N\}}{\Gamma \vdash_{uEQ} \Omega}$$

where

$\Omega' = \{\gamma^{(n)} \hookrightarrow \Delta^{(n)} | n \in N\} \mapsto (\gamma \hookrightarrow \Delta)$,

$\Omega^{(n)} = \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma^{(n)} \hookrightarrow \Delta^{(n)})$ *for* $n \in N$, *and*

$\Omega = \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$, *respectively.*

8. (u-q-ctr)

$$\frac{\Gamma \vdash_{uEQ} \Omega'; \{\Gamma \vdash_{uEQ} \Omega^{(t' \simeq u')} \mid t' \simeq u' \in \gamma'\}}{\Gamma \vdash_{uEQ} \Omega}$$

where

$\Omega' = \{\gamma^{(n)} \hookrightarrow \Delta^{(n)} | n \in N\} \mapsto (\gamma' \hookrightarrow \Delta)$,

$\Omega^{(t' \simeq u')} = \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow t' \simeq u')$ *for* $t' \simeq u' \in \gamma'$, *and*

$\Omega = \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$, *respectively.*

9. (u-d-intr)

$$\frac{\Gamma \cup \gamma \vdash_{uEQ} \Delta}{\Gamma \vdash_{uEQ} \gamma \mapsto \Delta}$$

10. (u-q-d)

$$\frac{\Gamma \vdash_{uEQ} \gamma \hookrightarrow \Delta}{\Gamma \vdash_{uEQ} \gamma \mapsto \Delta}$$

11. (u-skm)

$$\frac{\Gamma \vdash_{uEQ} \delta(\gamma) \mapsto \delta(\Delta)}{\Gamma \vdash_{uEQ} \gamma \hookrightarrow \Delta}$$

where δ *is a (skolemized) family of substitution functions such that (i)* $\delta(y) = y$ *if* y *is neither in* γ *nor in* Δ *and (ii)* $\delta(x_i) = c_i$ *if* x_i *is either in* γ *or in* Δ *and (iii)* c_i *is a fresh constant (i.e. not available in* Σ*).*

12. (u-q-u)

$$\frac{\Gamma \cup \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \vdash_{uEQ} \gamma \hookrightarrow \Delta}{\Gamma \vdash_{uEQ} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)}$$

13. (u-wkn)

$$\frac{\Gamma \vdash_{uEQ} \Omega}{\Gamma \cup \Gamma \vdash_{uEQ} \Omega}$$

14. (u-cut)

$$\frac{\Gamma' \vdash_{uEQ} \Omega; \{\Gamma \vdash_{uEQ} \Omega' | \Omega' \in \Gamma'\}}{\Gamma \vdash_{uEQ} \Omega}$$

Theorem 2.5.5 (completeness of $\vdash_{uEQ}$): *Calculus* $\vdash_{uEQ}$ *is sound and complete, i.e.* $\Gamma \models_{uEQ} \Omega$ *iff* $\Gamma \models_{uEQ} \Omega$.

Proof

The completeness with respect to pure equations is obtained by (u-id), (u-rfl), (u-sym), (u-trs), (u-cmp), (u-sub), (u-wkn), and (u-cut). Besides this, with extra (u-q-d) and (u-d-intr), we can obtain the completeness with respect to dependent equations. After that, we get the completeness with respect to quasi-dependent equations through another (u-skm). From the above and the last one (u-q-u), we have the full completeness. □.

2.6 Discussion on related works

In the present chapter, we are interested in many-sorted theories which can be expressed by equations or axioms of first order language. This kind of equational theories, including their corresponding software system environments (say systems of term rewriting [36,90], logic programming and/or functional programming), has been developed dramatically in recent years, and is still rapidly growing. Apparently, *equational logics* play fundamental roles in these theories. However, there are confusions in the literature about the logics. Some of them are terminological or notational, and the others of them are theoretical. This section is an attempt at a brief survey and subsequent clarification in the area.

Terminologically, there is no confusion in equational logics for pure equations. But the situations are different for the other two kinds of equational logics. One is for *equational implications* [158,56] and the other is for *conditional equations* [52]. For equational implications, some people refer them as "conditional equations", and even confuse them as conditional equations. For conditional equations, they are referred to either as *quasi-equations* [99] or as *universal Horn Clauses* [116] by different schools. Actually, the three kinds of equational logics have a close relationship between them. In order to avoid a future confusion we propose to use

dependent equations and *quasi-dependent equations* for equational implications and conditional equations respectively in this thesis (see Table 1.5.1 in Section 1.5).

Henkin has recognized the difference between the dependent implication and the quasi-dependent implication in [65]. He obtains deduction systems for natural numbers $< Nat, +_{Nat}, 0 >$ corresponding to dependent and quasi-dependent implications (see his $\models$ for $\mapsto$ in [65, Section 4] and his $\models^*$ for $\hookrightarrow$ in [65, (7.1) Definition]). The examples to show the difference between $\mapsto$ and $\hookrightarrow$ by Henkin are given in [65, (7.2)], i.e. (with notational modifications)

$$\{x +_{Nat} y \simeq y +_{Nat} x\} \mapsto x +_{Nat} z \simeq z +_{Nat} x, \textit{ but}$$
$$\textit{not } \{x +_{Nat} y \simeq y +_{Nat} x\} \hookrightarrow x +_{Nat} z \simeq z +_{Nat} x$$

In the following three subsections, we discuss the relationship between the work in this chapter the work in the literature.

2.6.1 Equational Logic $\vdash_{EQ}$

The soundness and completeness of many-sorted $\vdash_{EQ}$, was believed to be a trivial extension of the single-sorted one, and was first claimed by Goguen and Meseguer in [51] which demonstrated that the naive belief did not hold. The simple example provided by them is Example 2.1.1. Two special rules, *Abstraction* and *Concretion*

$$(abst)\ \frac{\Gamma \vdash_{EQ} t \simeq_X u}{\Gamma \vdash_{EQ} t \simeq_{X \cup \{y\}} u} \qquad (concr)\ \frac{\Gamma \vdash_{EQ} t \simeq_X u}{\Gamma \vdash_{EQ} t \simeq_{X-\{x\}} u}\ (\mathbf{T}(\emptyset)_i \neq \emptyset \text{ and } x \in X_i)$$

where y is not in X and x is neither in t nor in u, are given in the deduction system (see [51,53]) besides the ordinary rules to emphasize the potentiality of empty sorts (or empty carriers). The full version of the soundness and completeness of their system appears in [53]. This proof involves building function spaces from term algebras and the verification that these spaces form clones (see [28] for basic properties of clones). Since they only allow arbitrary finite quantification over equations, they use the co-limit result from Category Theory [97] in eliminating quantification over equations, i.e. in obtaining a quantifier-free (or variable index free) calculus. Ehrig and Mahr seem to follow the outline of the proof in [51] and provide another proof

in [38] but they exclude the case of empty sorts. There has been some confusion between these two proofs, see [37] and [54].

We take a different approach from that used by the authors above in our proof of soundness and completeness of $\vdash_{EQ}$. Following Birkhoff (see [23,28,56]), we extend the proof of the single-sorted case by introducing the concept of *cross-fully invariant congruences*, which differs from fully invariant congruences as in [23]. This approach removes the need to build function spaces from term algebras to form clones, and use clone properties to obtain the completeness result as in [53]; nor do we need to exclude algebras with empty carriers as in [38]. We should point out that there is no place for cross-fully invariant congruences in the single-sorted case. The necessary and sufficient condition for variable index free $\vdash_{EQ}$ is derived within this approach (Section 2.2).

Birkhoff's approach is conceptually simpler, and it is more coherent when we consider the extensions to include dependent equations and quasi-dependent equations. Also, information about models, e.g. what are equationally definable, dependent equationally definable and/or quasi-dependent equationally definable classes of algebras (see [25,99] for examples about models of quasi-dependent equations), can easily be carried out. This kind of benefit is not easily available in the other approaches such as [53]. To understand this point more clearly, let us consider universally-quantified equations, like $\forall X.t \simeq u$. It is easy to extend universally-quantified equations to dependent equations, i.e.

$$(\forall X.\gamma) \mapsto (\forall X.t \simeq u)$$

where $\forall X.\gamma$ is $\{\forall X.t' \simeq u' | t' \simeq u' \in \gamma\}$. However, since every equations are universally-quantified, it is hard to represent a quasi-dependent equation, like

$$\forall X.(\gamma \hookrightarrow t \simeq u),$$

in this universally-quantified framework for equations. There is no such a problem in generalizing variable-indexed equations to other kinds of equational forms, see the following

$$t \simeq_X u, \quad \gamma_X \mapsto t \simeq_X u, \quad \gamma_X \hookrightarrow t \simeq_X u.$$

2.6.2 Calculus $\vdash_{dEQ}$ for dependent equations

Birkhoff's approach provides further clarification on the soundness and completeness of $\vdash_{dEQ}$ and of $\vdash_{qEQ}$ Essentially, a sound and complete calculus for dependent equations can be obtained by the following fact with completeness of $\vdash_{EQ}$.

Fact 2.6.2.1: $\Gamma \models \gamma \mapsto t \simeq u$ *iff* $\Gamma \models t \simeq u$ *when* $\Gamma \models \gamma$.

Therefore, let $\vdash_{dEQ}$ have all the obvious rules[1] with an extra rule (d-intr).

Thus, it is sound and complete.

2.6.3 Calculus $\vdash_{qEQ}$ for quasi-dependent equations

For quasi-dependent equations, there is one obvious temptation as follows.

(i) $Congr_{S,\Sigma}$ denotes the complete lattice of congruences on term algebra $\mathbf{T}(X)$.

(ii) θ_Γ is the functional : $Congr_{S,\Sigma} \to Congr_{S,\Sigma}$ such that given a congruence $\equiv_1$, $\theta_\Gamma(\equiv_1)$ is the smallest congruence family $\equiv_2$ satisfying: for each $\gamma_X \hookrightarrow t \simeq_X u \in \Gamma$ and every $\iota : X \to \mathrm{T}(X)$,

(ii.a) if $\iota(t') \equiv_1 \iota(u')$ for each $t' \simeq_X u' \in \gamma_X$ then $\iota(t) \equiv_2 \iota(u)$,

(ii.b) $\equiv_1 \subseteq \equiv_2$.

(iii) $\equiv_\Gamma$ is the least fixpoint of θ_Γ, i.e. $\equiv_\Gamma = \bigcup_{i \in Nat} \theta_\Gamma^{[i]}(\equiv)$ where $\equiv$ is the plain equality on $\mathbf{T}(X)$ (i.e. pure identity of terms). Note that $\equiv_\emptyset = \equiv$.

(iv) Replacing the $\equiv_i$ by $\equiv_\Gamma$ in the two conditions of (2.6.3.ii) above, we let the closure of Γ, written as $[\Gamma]$, be the least collection of quasi-dependent equations (like $\gamma_X \hookrightarrow t \simeq_X u$), satisfying:

(*) if $\iota(t') \equiv_\Gamma \iota(u')$ for each $t' \simeq_X u' \in \gamma_X$ then $\iota(t) \equiv_\Gamma \iota(u)$.

---

[1] There is only one requirement on these rules, i.e. they must form a sound and complete calculus with respect to pure equations.

(v) Obviously, $\equiv_\Gamma = \equiv_{[\Gamma]}$.

Therefore, the new context can be achieved by adding a new rule

$$(\omega\text{-}d\text{-}q) \quad \frac{\{\Gamma \vdash \iota(\gamma_X) \mapsto \iota(t) \simeq_Y \iota(u) | \iota \in \mathbf{T}(X) \to \mathbf{T}(Y)\}}{\Gamma \vdash \gamma_X \hookrightarrow t \simeq_X u}$$

Actually, rule (ω-d-q) is (2.6.3.v) above. However, this rule is not sound in general, but the calculus containing this rule is sound and complete with respect to pure equations and dependent equations. So, we name this calculus as $\vdash^+_{qEQ}$. A counter-example to soundness of $\vdash^+_{qEQ}$ (or more properly to the rule (ω-d-q)) is as follows (the continued Example 2.4.5).

Example 2.6.3.1 (counter-example to soundness of (ω-d-q)): *Let* $\mathbf{D}$ *and* Γ *be the same as in Example 2.3.5. Thus,*

(a) Γ *is sound in* D.

(b) $x \simeq_{\{x,y\}} y \hookrightarrow \mathrm{true}(x) \simeq_{\{x,y\}} y$ *is not sound in* $\mathbf{D}$ *(consider the assignment of both* x *and* y *to truth-value* $false$*), i.e.*

$$\mathrm{D} \not\models x \simeq_{\{x,y\}} y \hookrightarrow \mathbf{true}(x) \simeq_{\{x,y\}} y.$$

(c) However, for every substitution $\check{\iota} : \mathbf{T}(X) \to \mathbf{T}(X)$, *dependent equation*

$$\check{\iota}(x \simeq_{\{x,y\}} y) \mapsto \check{\iota}(\mathbf{true}(x) \simeq_{\{x,y\}} y)$$

is derivable. So, $x \simeq_{\{x,y\}} y \hookrightarrow \mathrm{true}(x) \simeq_{\{x,y\}} y$ *is derivable by* (ω-d-q).

For (c), we only need to show four simple cases. The general case is not hard to prove from the observation of these four, and is left out.

(c.i) when $\check{\iota}$ *is a permutation of variables, say the identity substitution,*

$$x \simeq_{\{x,y\}} y \mapsto \mathbf{true}(x) \simeq_{\{x,y\}} y$$

is derivable because of the rule (d-intr);

c.ii when $\check{\iota}$ *assigns* x *to* $\mathrm{true}(x)$ *and* y *to* y,

$$\mathrm{true}(x) \simeq_{\{x,y\}} y \mapsto \mathbf{true}(\mathbf{true}(x)) \simeq_{\{x,y\}} y$$

is derivable by the same reason as (c.i);

(c.iii) when $\check{\iota}$ assigns x to x and y to $\mathbf{true}(y)$,

$$x \simeq_{\{x,y\}} \mathbf{true}(y) \mapsto \mathbf{true}(x) \simeq_{\{x,y\}} \mathbf{true}(y)$$

is derivable by (d-intr);

(c.iv) when $\check{\iota}$ assigns x to $\mathbf{true}(x)$ *and y to* $\mathbf{true}(y)$,

$$\mathbf{true}(x) \simeq_{\{x,y\}} \mathbf{true}(y) \mapsto \mathbf{true}(\mathbf{true}(x)) \simeq_{\{x,y\}} \mathbf{true}(y)$$

is derivable by (d-intr).

Actually, rule *(ω-d-q)* is not necessary. The key point to have a sound and complete calculus for quasi-dependent equations is the following two facts.

Fact 2.6.3.2 (dEQ and qEQ): $\Gamma \models \gamma \mapsto t \simeq u$ *iff* $\Gamma \models \gamma \hookrightarrow t \simeq u$, *provided that there is no free variable in γ (and $t \simeq u$).*

and

Fact 2.6.3.3 (skolemization and Theorem **2.4.7**): $\Gamma \models \gamma \hookrightarrow t \simeq u$ *iff* $\Gamma \models \gamma' \mapsto t' \simeq u'$, *where γ' and $t' \simeq u'$ are the results of the substitution of fresh constants for free variables*[2] *in γ and $t \simeq u$.*

From the above two facts, we can borrow "skolemization" technique from logic and apply it to here. This is exemplified by *(q-skm)* rule. This rule coupled with other obvious rules[3] forms a sound and complete calculus $\vdash_{qEQ}$. To be more explicit, we apply the above rule to Example 2.6.3.1 and demonstrate that $x \simeq y \hookrightarrow \mathbf{true}(x) \simeq y$ is not derivable.

Example 2.6.3.4: *After introducing two extra constants* **True** *and* **False** *into Example 2.6.3.1 to instantiate x and y respectively, we have that*

$$x \simeq_{\{x,y\}} y \hookrightarrow \mathbf{true}(x) \simeq_{\{x,y\}} y$$

---

[2]This technique is commonly called "skolemization" in logic.

[3]As long as they form a sound and complete calculus with respect to dependent equations.

is derivable iff

$$\frac{\mathbf{True} \simeq \mathbf{False}}{\mathbf{true(True)} \simeq \mathbf{False}}.$$

However, $\mathbf{true(True)} \simeq \mathbf{False}$ *is not derivable by adding an extra axiom* $\mathbf{True} \simeq \mathbf{False}$*, since every derivable equation has the property that either constructor* **true** *does not occur in both sides of* $\simeq$ *or it appears in both sides of* $\simeq$ *at the same time. Therefore,* $x \simeq_{\{x,y\}} y \hookrightarrow \mathbf{true}(x) \simeq_{\{x,y\}} y$ *is not derivable.*

On the other hand, $\mathbf{true}(x) \simeq_{\{x,y\}} y \hookrightarrow \mathbf{true}(\mathbf{true}(x)) \simeq_{\{x,y\}} y$ in Example 2.6.3.1 is obviously derivable (by (q-sub) rule).

Because of the application of "skolemization", the *non-emptyness* of sorts seems to be the crucial condition for our solution to work. However, this condition can be removed if we know the fact that the existence of empty sorts implies non-existence of variable assignment (or homomorphisms).

2.6.4 Comments

So far, we have briefly clarified the soundness and completeness of the three equational logics. In First Order Logic, we know that the validation (or satisfaction) relation is actually the *quasi-dependent* implications. So, as a by-product of this chapter, we know that First Order Logic without quantifiers (i.e. quantifier-free First Order Logic) is sound and complete. Also, this by-product is not easily foreseen.

For a complete treatment of First Order Logic, we need to introduce *binding* into algebraic framework. This direction of research is taken in this thesis. It will be presented in Chapter 8 (and in [153]).

Part II

Birkhoff's Approach

In this part, which consists of Chapter 3 and Chapter 4, we are going to establish the necessity and the sufficiency of *admissible* condition for the commutative property shown in Figure 1-1, and to prove the Admissible Completeness of $\dot{\vdash}_{eBA}$.

Recall that a Birkhoff-like theorem is the equivalences among (1.2.1) $\mathbf{A} \models p \simeq q$; (1.2.2) $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$ for every $\zeta : \mathbf{T} \to \mathbf{A}$, where $\mathbf{T}$ is the term eBA; (1.2.3) $< \bullet_{[p]}, \bullet_{[q]} > \in \bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$; and (1.2.4) $\mathbf{T}/\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta) \models p \simeq q$ (see Subsection 1.2.1). Under the binding context, the equivalence between (1.2.1) and (1.2.2) is obvious, and the equivalence between (1.2.3) and (1.2.4), especially the implication from (1.2.3) to (1.2.4), is a result of Lemma 3.10.1. However, the equivalence between (1.2.2) and (1.2.3) is not necessarily valid, although the implication from (1.2.2) to (1.2.3) is evident. To be more explicit, let us fix a $\zeta : \mathbf{T} \to \mathbf{A}$ and look at (1.2.2') and (1.2.3'):

(1.2.2') $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$, and

(1.2.3') $< \bullet_{[p]}, \bullet_{[q]} > \in Ker(\zeta)$.

The necessary and sufficient condition for the implication from (1.2.3') to (1.2.2') is Admissiblity (a result from Theorem 3.8.5, Corollary 3.8.7 and Lemma 3.10.1). Therefore, by weakening satisfaction $\models_{eBA}$ to admissible one $\dot{\models}_{eBA}$, we obtain a Birkhoff-like theorem. That is, the following four statements are equivalent:

(1.2.1") $\mathbf{A} \dot{\models}_{eBA} p \simeq q$,

(1.2.2") $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$ for every $\zeta : \mathbf{T} \dot{\to} \mathbf{A}$ (where a dot on top of an arrow $\dot{\to}$ means that *admissible* eBHs),

(1.2.3") $< \bullet_{[p]}, \bullet_{[q]} > \in \bigcap_{\zeta:\mathbf{T}\dot{\to}\mathbf{A}} Ker(\zeta)$, and

(1.2.4") $\mathbf{T}/\bigcap_{\zeta:\mathbf{T}\dot{\to}\mathbf{A}} Ker(\zeta) \dot{\models}_{eBA} p \simeq q$.

The relationship between Completeness and Admissible Completeness will be discussed in Section 3.13.

We further extend this result to include admissible dBEs (Section 4.1), admissible qBEs (Section 4.2) admissible uBEs (Section 4.3).

Chapter 3

Remedy to Birkhoff's Approach

In this chapter, we use extensional Binding algebras (or eBAs) instead of Σ^{BO}-algebras. Section 3.1 provides some basic definitions, say sub-extensional Binding Algebras (sub-eBAs), *perfect* sub-eBAs and extenional Binding Homomorphisms (eBHs). Some simple properties about them will be given in Section 3.2 and Section 3.3. An important example of eBAs, besides the ones mentioned after Definition 1.2.1.3, is the term eBA **T**. It is given and verified in Section 3.4, but it is not trivial at all, unlike the case in Chapter 2. Section 3.5 will provide a way to *generate* the term eBA from variables. This generation provides the clue to define Universal Property (Freeness), which means that term eBA possess of *freeness*. Then, extensional Binding Congruences (eBCs) and quotient Binding Algebras (quotient eBAs) are introduced in Section 3.7. However, in this section we will run across the major problem mentioned in Section 1.2, i.e. the natural commutativity among eBHs of two eBAs and the quotient eBA does not hold in general. This failure demonstrates that Birkhoff's approach does not work in the Framework for Binding Operators (FBO). Therefore, the work to be presented in later chapters (Chapter 5 and Chapter 6) becomes important, complementary and necessary.

Nevertheless, we manage to find a way out, i.e. to reduce the equality to admissible equality. Our "reduced ambition" is carried out successfully in Section 3.8, Section 3.9 and Section 3.10. The Admissible Freeness is introduced in Section 3.8. The weaken equality will be defined in Section 3.9. Section 3.10 establishes the Admissible Completeness through a modified standard technique — admissibly

invariant eBCs (congruences). We will also show that "admissibility" is a weaker property than Plotkin's "logical Relations", in the sense that an eBH is logical implies that it is admissible. This is the subject of Section 3.12. But there are still some open problems related to Admissibility to be solved. One of them is the variety problem listed in Section 3.11. Lastly, the relationship between Completeness and Admissible Completeness (or between satisfaction $\models_{eBA}$ and admissible satisfaction $\dot{\models}_{eBA}$) is not completely clear and it is discussed in Section 3.13.

3.0 Well-definedness of interpretations in eBAs

Before proceeding further, we check the well-definedness of interpretations in eBAs in this section.

Lemma 3.0.1 (semantic substitution): *Let* $< \rho, \varphi >$ *be a valuation of* V *and* FV *on* $\mathbf{A}$. *Then,*

(i) if $x \notin \{\vec{y}\}$, *then* $\rho[a/x][\vec{a}/\vec{y}] = \rho[\vec{a}/\vec{y}][a/x]$;

(ii) if $x = y_j$ *for some* $y_j \in \{\vec{y}\}$ *(i.e.* $\vec{y}(j) = y_j$*), then* $\rho[a/x][\vec{a}/\vec{y}] = \rho[\vec{a}/\vec{y}]$.

Lemma 3.0.2 (non-free variable and semantic substitution): *For any* $a \in A$,

(i) if $x \notin Free(t)$, *then* $\mathcal{A}[\![t]\!](\rho[a/x], \varphi) = \mathcal{A}[\![t]\!](\rho, \varphi)$,

(ii) if $x \notin Free(ft)$, *then* $\mathcal{A}[\![ft]\!](\rho[a/x], \varphi) = \mathcal{A}[\![ft]\!](\rho, \varphi)$.

Proof. By structural induction on binding terms and *Lemma 3.0.1.* □

The above lemmas said that the valuation on non-free (or bound) variables has no effect.

Lemma 3.0.3 (interpretation and semantic substitution): *Let* $|\vec{x}| = |\vec{z}|$ *and they satisfy the following,*

$$z_j = \begin{cases} x_j & if\ x_j \notin \{\vec{y}\} \\ z_j^* & if\ x_j \in \{\vec{y}\} \end{cases}$$

where $z_j^* \notin Free(t) \cup \{\vec{y}\} \cup \{\vec{z}\lceil_{j-1}\}$, *in which* $\vec{z}\lceil_j$ *means the prefix list of list* $\vec{z}$ *with lenth* j. *Then,* $\mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{x}], \varphi) = \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{z}], \varphi)$ *for all valuation* $< \rho, \varphi >$.

Proof.

If all $z_j = x_j$ $(1 \leq j \leq |\vec{z}|)$, then it is trivial. So, we restrict our interest to that there is at least one $z_j \neq x_j$ for some $1 \leq j \leq |\vec{z}|$.

Let n be the smallest j which does not satisfy $z_j = x_j$. i.e. $z_j = x_j$ $(1 \leq j < n)$ and $z_n \neq x_n$. Since $x_n \in \{\vec{y}\}$, we know $x_n \notin Free(\langle \vec{y} : t\rangle)$. On the other hand, $z_n \notin Free(t) \cup \{\vec{y}\}$. Then, $z_n \notin Free(\langle \vec{y} : t\rangle)$. Therefore, by using twice Lemma 3.0.2, we get

$$
\begin{aligned}
&\mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{x}], \varphi) \\
&= \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}\lceil_{n-1}, a_n, \vec{a}\lfloor_n / \vec{x}\lceil_{n-1}, x_n, \vec{x}\lfloor_n], \varphi) \\
&= \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}\lceil_{n-1}, a_n, \vec{a}\lfloor_n / \vec{z}\lceil_{n-1}, x_n, \vec{x}\lfloor_n], \varphi) \\
&= \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}\lceil_{n-1}, \vec{a}\lfloor_n / \vec{z}\lceil_{n-1}, \vec{x}\lfloor_n], \varphi) \\
&= \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}\lceil_{n-1}, a_n, \vec{a}\lfloor_n / \vec{z}\lceil_{n-1}, z_n, \vec{x}\lfloor_n], \varphi) \\
&= \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}\lceil_n, \vec{a}\lfloor_n / \vec{z}\lceil_n, \vec{x}\lfloor_n], \varphi)
\end{aligned}
$$

where $\vec{\bullet}\lfloor_k$ means the tail list obtained by dropping off the first k elements from list $\vec{\bullet}$.

The above process can be carried on until all x_j replaced by proper z_j $(1 \leq j \leq |\vec{z}|)$. □

Lemma 3.0.4 (definability):

(i) For all $t \in T$, *we have that for any* $k \geq 0$, *and for any* $\{\vec{x}\} \subseteq V$, *if* $g(\vec{a}) = \mathcal{A}[\![t]\!](\rho[\vec{a}/\vec{x}], \varphi)$ *for all* $\vec{a} \in A$, *then* $g \in \mathcal{F}_k$.

(ii) For any $m \geq 0$, *and all* $ft \in FT_m$, *we have that for any* $\{\vec{x}\} \subseteq V$, *if* $g(\vec{a}, \vec{b}) = \mathcal{A}[\![ft]\!](\rho[\vec{a}/\vec{x}], \varphi)(\vec{b})$ *for all* $\vec{b} \in A$, *then* $g \in \mathcal{F}_{|\vec{x}|+m}$.

Proof.

By mutual structural induction on t and ft.

case x: for any $\{\vec{x}\} \subseteq V$, if $x \notin \{\vec{x}\}$, then $g = C_{|\vec{x}|,\rho(x)} \in \mathcal{F}_{|\vec{x}|}$; if $x = x_i \in \{\vec{x}\}$ such that $x_i = \vec{x}(i)$ for some $1 \leq i \leq |\vec{x}|$ then $g = \pi_{|\vec{x}|,i} \in \mathcal{F}_{|\vec{x}|}$.

case $f(\vec{t})$:

$$\mathcal{A}[\![f(\vec{t})]\!](\rho[\vec{a}/\vec{x}],\varphi) = \varphi(f)(\mathcal{A}[\![\vec{t}]\!](\rho[\vec{a}/\vec{x}],\varphi))$$
$$= \varphi(f)(g_1(\vec{a}), g_2(\vec{a}), ..., g_m(\vec{a})) = \varphi(f)\odot < \vec{g} > (\vec{a})$$

So, $\varphi(f)\odot < \vec{g} >= g \in \mathcal{F}_{|\vec{x}|}$.

case $\sigma(\vec{ft}, \vec{t})$: for any $\{\vec{x}\} \subseteq V$, by structural hypothesis, we have $h_i \in \mathcal{F}_{|\vec{x}|+m_i}$ $(1 \leq i \leq |\vec{m}|)$ and $h'_j \in \mathcal{F}_{|\vec{x}|}$ $(1 \leq j \leq n)$ where given $\vec{a} \in A$

$$h_i(\vec{a}, \vec{b^i}) = \mathcal{A}[\![ft_i]\!](\rho[\vec{a}/\vec{x}],\varphi)(\vec{b^i}) \ (1 \leq i \leq |\vec{m}|)$$

for all $b^i_j \in A$ $(1 \leq j \leq |\vec{b^i}|)$ and $h'_j(\vec{a}) = \mathcal{A}[\![t_j]\!](\rho[\vec{a}/\vec{x}],\varphi)$ $(1 \leq j \leq n)$.

By Extensionality, we have

$$(h''_j =)curry_{|\vec{x}|,m_i}(h_i)(\vec{a}) = \mathcal{A}[\![ft_i]\!](\rho[\vec{a}/\vec{x}],\varphi) \ (1 \leq i \leq |\vec{m}|)$$

So,

$$\mathcal{A}[\![t]\!](\rho[\vec{a}/\vec{x}],\varphi) = \sigma^{\mathbf{A}}(\vec{h''}(\vec{a}), \vec{h'}(\vec{a})) = \sigma\odot < \vec{h''}, \vec{h'} > (\vec{a}).$$

By Extensionality, we come to $g = \sigma^{\mathbf{A}}\odot < \vec{h''}, \vec{h'} > \in \mathcal{F}_{|\vec{x}|}$.

case $\langle \vec{y} : t\rangle$: for any $\{\vec{x}\} \subseteq V$, let $\vec{z}$ satisfy the condition of Lemma 3.0.3, by Definition 1.5.1.2 we get

$$h(\vec{a}, \vec{b}) = \mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{x}],\varphi)(\vec{b}) = \mathcal{A}[\![t]\!](\rho[\vec{a}/\vec{z}][\vec{b}/\vec{y}],\varphi).$$

By structural hypothesis, we have $h \in \mathcal{F}_{|\vec{x}|+m}$. □

Essentially, what Lemma 3.0.4 said is that 4 of Definition 1.5.1.2 is well-defined.

Theorem 3.0.5 (well-definedness of interpretations in an eBA): *$\mathcal{A}$ is well-defined in $\mathcal{F}$ (or on $\mathbf{A}$).*

Proof. by the structural induction and *Lemma 3.0.4.* □

So, we have examined the well-definedness of interpretations in an eBA. In the next section, we are going to introduce sub-eBAs and perfect sub-eBAs, and eBHs.

3.1 Sub-eBAs, perfect Sub-eBAs and eBHs

To motivate certain intuitions, we recall the comparison between eBAs and first order algebras in Section 1.3.3. Let us suppose that $\Sigma^{BO}_{<\vec{m},n>} = \emptyset$ if $\vec{m} \neq \varepsilon$. So, this Σ^{BO} can be regarded as a first order signature, and a Σ^{BO}-algebra $\mathbf{A}$ (i.e. an eBA) consists of

(a) $\mathcal{F}^{\mathbf{A}} =< A, \mathcal{F} >$ and $\mathcal{F} = \{\mathcal{F}_m(A) | m \in Nat\}$, and

(b) for each $\sigma \in \Sigma^{BO}_{<\varepsilon,n>}$ with $n \in Nat$, its interpretation $\mathcal{A}_\sigma$ is uniform over $\mathcal{F}^{\mathbf{A}}$.

For each $n \in Nat$ if $\mathcal{F}_n(A)$ is the all function space from A^n to A, then any arbitrary interpretation of the $\sigma \in \Sigma^{BO}_{<\varepsilon,n>}$ is uniform over $\mathcal{F}$. Thus, eBAs are more general than first order algebras. In other words, the latter is a special case of the former.

From the above analogy, we can generalize sub-algebras and homomorphisms from Chapter 2 to the present chapter, and to accommodating FBOs. That is, sub-eBAs and perfect sub-eBAs are introduced in this section. We give a definition for sub-eBAs first.

Let $\mathbf{A} =< \mathcal{F}^{\mathbf{A}}, \mathcal{A} >$ such that $\mathcal{F}^{\mathbf{A}} =< A, \mathcal{F} >$ and $\mathcal{F} = \{\mathcal{F}_m | m \in Nat\}$ be an eBA, and $\mathcal{F}' = \{\mathcal{F}'_m | m \in Nat\}$ such that $\mathcal{F}'_m \subseteq \mathcal{F}_m$ for $m \in Nat$. When $\mathcal{F}^{\mathbf{A}'} =< A', \mathcal{F}' >$ is explicitly closed, given $k \geq 0$ and $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$, we obtain a unique function h in $\mathcal{F}_k$ for every $g_i \in \mathcal{F}'_{k+m_i}$ and each $g_{\ell+j} \in \mathcal{F}'_k$ by the uniformity of $\mathcal{A}_\sigma$ (see Definition 1.2.1.2). We use $\mathcal{F}^{\mathbf{A}}_k(\sigma, \mathcal{F}^{\mathbf{A}'})$ to refer to the collection of such functions as h inside $\mathcal{F}^{\mathbf{A}}_k$ which are associated with σ, k and $\mathcal{F}^{\mathbf{A}'}$.

Definition 3.1.1 (sub-eBA): *Let* $\mathbf{A}$ *be an eBA.* $\mathbf{A}' =< \mathcal{F}^{\mathbf{A}'}, \mathcal{A} >$ *is said to be an sub-extensional Binding Algebra, sub-eBA for short, of* $\mathbf{A}$ *iff*

(i) $\mathcal{F}^{\mathbf{A}'} =< A', \mathcal{F}' >$ *such that* $A' \subseteq A$ *and for all* $m \geq 0$, $\mathcal{F}'_m \subseteq \mathcal{F}_m$;

(ii) $\mathcal{F}^{\mathbf{A}'}$ *is explicitly closed; and*

(iii) for each σ, $\mathcal{F}^{\mathbf{A}}_k(\sigma, \mathcal{F}^{\mathbf{A}'}) \subseteq \mathcal{F}^{\mathbf{A}'}_k$ *(or* $\mathcal{F}'_k$*), i.e.* $\mathcal{A}_\sigma$ *is uniform over* $\mathbf{A}'$.

Sometimes, we use $\mathbf{A}' \prec \mathbf{A}$ to mean $\mathbf{A}'$ is a sub-eBA of $\mathbf{A}$.

Note that a sub-eBA $\mathbf{A}' \prec \mathbf{A}$ is well-defined since we have the following two conditions. Firstly, for $m \geq 0$ and $g \in \mathcal{F}_m^{\mathbf{A}'}$, $g(\vec{a})$ is in A' for $a_1, a_2, ..., a_m \in A'$ where $|\vec{a}| = m$. Secondly, for $k \geq 0$ and $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$, given $g_i \in \mathcal{F}^{\mathbf{A}'}_{k+m_i}$ and given $g_j \in \mathcal{F}_k^{\mathbf{A}'}$, h is in $\mathcal{F}_k^{\mathbf{A}'}$ where $h(\vec{a}) = \mathbf{A}_\sigma(\vec{h}, \vec{b}) \in A'$ for $a_1, a_2, ..., a_k \in A'$ $(k = |\vec{a}|)$ as defined in the uniformity condition (see Definition 1.2.1.2 and the comment after it).

As we know, first order subalgebras have two properties from Chapter 2. They are that

(1) each subalgebra is closed under functionality (or compositionality), and

(2) each subalgebra is an algebra if it is restricted to its basis.

However, sub-eBAs only possess the first property (1) not necessarily the second (2). The loss possession of the second property comes from the fact that function spaces are under consideration in eBAs but not under consideration in first order algebras. On the other hand, the previous analogy between eBAs and first order algebras can be carried over between sub-eBAs and sub-algebras. Therefore, sub-eBAs are regarded as a generalization of the concept of first order sub-algebras. Since this generalization loses the second property (2) of first order subalgebras, it is our intention to introduce a concept of "*perfect*" sub-eBAs which are sub-eBAs with property (2). This "perfectness" turns out to be very important to Admissibility. Formally,

Definition 3.1.2 (perfect sub-eBA): *Let* $\mathbf{A}'$ *be a sub-eBA of* $\mathbf{A}$, *i.e.* $\mathbf{A}' \prec \mathbf{A}$. $\mathbf{A}'$ *is said to be a perfect sub-eBA of* $\mathbf{A}$, *written as* $\mathbf{A}' \preceq \mathbf{A}$, *iff* $\mathbf{A}'$ *has Extensionality on its basis* A', *i.e. for* $m \geq 0$ *and given* $g, h \in \mathcal{F}_m^{\mathbf{A}'}$, $g(\vec{a'}) = h(\vec{a'})$ *for all* $a'_j \in A'$ *implies* $g = h$.

So, it is easy to verify that a perfect sub-eBA is an eBA if we restrict it to its basis, since a sub-eBA is well-defined (see the comment after Definition 3.1.1). Next, we are going to present a definition for extensional Binding Homomorphisms. Similar to the above analogy, we understand from Chapter 2 that a homomorphism is basically a function which preserves functionality (or compositionality). However,

since function spaces are not under consideration of first order algebras, homomorphisms among them do not need to take function spaces into account in Chapter 2. Obviously, we are in a different situation for the case of extensional Binding Homomorphisms. Some minimal requirements are needed besides functionality. Informally, these requirements can be expressed by certain preservations, i.e. preservations of certain primitive functions like constant functions, projections, compositions of functions. However, it is not obvious that the minimal requirements imply the preservation of the explicitly-closedness and of the uniformity of interpretations of BOs. So, we will return to this issue later and will verify them in Lemma 3.3.1 and Lemma 3.3.2.

Definition 3.1.3 (eBH): *An extensional Binding Homomorphism (eBH)* ζ *from eBA* $\mathbf{A}$ *to eBA* $\mathbf{A}'$, *written as* $\zeta : \mathbf{A} \to \mathbf{A}'$, *is a family of functions from* A *to* A' *and functionals from* $\mathcal{F}_m$ *to* $\mathcal{F}'_m$ *for each* $m \in Nat$ *such that*

(eBH-func) $\zeta(g(\vec{a})) = \zeta(g)(\vec{a'})$,

where $g \in \mathcal{F}_k$ $(k \geq 0)$, *and* $a'_j = \zeta(a_j)$;

(eBH-cons) $\zeta C^{\mathbf{A}}_{k,a} = C^{\mathbf{A}'}_{k,\zeta a}$, *for* $k \geq 0$, *and* $a \in A$;

(eBH-proj) $\zeta \pi^{\mathbf{A}}_{k,i} = \pi^{\mathbf{A}'}_{k,i}$, *for* $1 \leq i \leq k$;

(eBH-cmp) $\zeta(g \odot < \vec{h} >) = \zeta g \odot < \zeta \vec{h} >$ *for any* $m, k > 0$, *each* $g \in \mathcal{F}_m$ *and any* $h_j \in \mathcal{F}_k$ $(1 \leq j \leq m)$;

(eBH-unif) $\zeta(\sigma^{\mathbf{A}} \odot <\vec{g}, \vec{h}>) = \sigma^{\mathbf{A}'} \odot <\vec{g''}, \vec{h'}>$ *for* $\sigma \in \Sigma_{<\vec{m},n>}$, $k \geq 0$ *and each* $g_i \in \mathcal{F}^{\mathbf{A}}_{k+m_i}$ *and* $h_j \in \mathcal{F}^{\mathbf{A}}_k$ $(1 \leq j \leq n)$,

where $g'_i = curry_{k,m_i}(g_i)$, $g''_i = curry_{k,m_i}(\zeta g_i)$ *and* $h'_j = \zeta h_j$.

For the well-definedness of eBHs, all items are obvious, perhaps except the last one. However, let h be $curry_{k,m}(g)$ for $g \in \mathcal{F}_{k+m}$ (where $curry_{k,m}$ is conventionally written as $\lambda\vec{a}(\lambda\vec{x}g(\vec{a},\vec{x}))$ in a λ-notataion), then for every $\vec{a} \in A^k$ $h(\vec{a}) = curry_{k,m}(g)(\vec{a}) = g \odot < C_{m,a_1}, C_{m,a_2}, ..., C_{m,a_k}, \Pi^m_{1,m} >$ where $\Pi^m_{i,j}$ is the

list of $\Pi_{m,i}, \Pi_{m,i+1}, ..., \Pi_{m,j} (i \leq j)$. Hence, suppose $\zeta : \mathbf{A} \to \mathbf{A}'$, we would have the following:

$$\begin{aligned}
&\zeta(h(\vec{a})) \\
&= \zeta(curry_{k,m}(g)(\vec{a})) \\
&= \zeta(g \odot < C_{m,a_1}, C_{m,a_2}, ..., C_{m,a_k}, \Pi^m_{1,m} >) \\
&= \zeta(g) \odot < C_{m,a_1}, C_{m,a_2}, ..., C_{m,a_k}, \Pi^m_{1,m} >) \\
&= \zeta(g) \odot < \zeta(C_{m,a_1}), \zeta(C_{m,a_2}), ..., \zeta(C_{m,a_k}), \zeta(\pi_{1,m}), \zeta(\pi_{2,m}), ..., \zeta(\pi_{m,m}) > \\
&= \zeta(g) \odot < C_{m,\zeta(a_1)}), C_{m,\zeta(a_2)}), ..., \zeta(C_{m,\zeta(a_k)}), \pi_{1,m}), (\pi_{2,m}), ..., \pi_{m,m} > \\
&= curry_{k,m}(\zeta(g))(\vec{b}),
\end{aligned}$$

where $b_j = \zeta(a_j)$. In other words, (eBH-unif) is well-defined.

After introducing those basic definitions, we will try to follow Birkhoff's approach for FBO. The next section (Section 3.2) is to discuss some properties of sub-eBAs and generated sub-eBAs. Properties of eBHs will be provided in Section 3.4.

3.2 Sub-eBAs and generated sub-eBAs

We are examining that the intersection of two sub-eBAs is a sub-eBAs. Formally,

Lemma 3.2.1 (intersection of two sub-eBAs): *Let* $\mathbf{A}'_1, \mathbf{A}'_2 \prec \mathbf{A}$. *Then,* $\mathbf{A}'_1 \bigcap \mathbf{A}'_2 \prec \mathbf{A}$, *where* $\mathbf{A}'_1 \bigcap \mathbf{A}'_2$ *is* $<B_1 \cap B_2, \mathcal{F}^{\mathbf{A}'_1} \bigcap \mathcal{F}^{\mathbf{A}'_2}>$ *such that* $\mathcal{F}^{\mathbf{A}'_1} \bigcap \mathcal{F}^{\mathbf{A}'_2} >= \mathcal{F}^{\mathbf{A}'_1} \bigcap \mathcal{F}^{\mathbf{A}'_2} = \{\mathcal{F}^{\mathbf{A}'_1}_m \cap \mathcal{F}^{\mathbf{A}'_2}_m | m \in Nat\epsilon\}$.

Proof.

- $A_1 \cap A_2 \subseteq A$.
- For any $k \geq 0$, we have $\mathcal{F}^{\mathbf{A}'_1}_k \cap \mathcal{F}^{\mathbf{A}'_2}_k \subseteq \mathcal{F}^{\mathbf{A}}_k$, since $\mathcal{F}^{\mathbf{A}'_l}_k \subseteq \mathcal{F}^{\mathbf{A}}_k \quad (l = 1, 2)$.
- – for each $a \in A_1 \cap A_2$ and any $k > 0$, we get $b \in A_j$ for $j = 1, 2$. So, $C_{k,a} \in \mathcal{F}^{\mathbf{A}'_l}_k \quad (l = 1, 2)$, i.e. $C_{k,a} \in \mathcal{F}^{\mathbf{A}'_1}_k \cap \mathcal{F}^{\mathbf{A}'_2}_k$.
  – for any $k > 0$, and $1 \leq j \leq k$, we know that $\pi_{k,j} \in \mathcal{F}^{\mathbf{A}'_l}_k \quad (l = 1, 2)$. So, $\pi_{k,j} \in \mathcal{F}^{\mathbf{A}'_1}_k \cap \mathcal{F}^{\mathbf{A}'_2}_k$.

– for any $m > 0$, $k \geq 0$, $g \in \mathcal{F}_m^{\mathbf{A}'_1} \cap \mathcal{F}_m^{\mathbf{A}'_2}$ and $h_j \in \mathcal{F}_k^{\mathbf{A}'_1} \cap \mathcal{F}_k^{\mathbf{A}'_2}$ $(1 \leq j \leq m)$, we get $g \in \mathcal{F}_m^{\mathbf{A}'_l}$ $(l = 1, 2)$ and $h_j \in \mathcal{F}_k^{\mathbf{A}'_l}$ $(l = 1, 2;\ 1 \leq j \leq m)$.

So, $g \odot < \vec{h} > \in \mathcal{F}_k^{\mathbf{A}'_l}$ $(l = 1, 2)$, i.e. $g \odot < \vec{h} > \in \mathcal{F}_k^{\mathbf{A}'_1} \cap \mathcal{F}_k^{\mathbf{A}'_2}$.

- for each $\sigma \in \Sigma_{<\vec{m},n>}$, for any $k \geq 0$, each $g_i \in \mathcal{F}_{m_i+k}^{\mathbf{A}'_1} \cap \mathcal{F}_{m_i+k}^{\mathbf{A}'_2}$ $(1 \leq i \leq |\vec{m}|)$, and $h_j \in \mathcal{F}_k^{\mathbf{A}'_1} \cap \mathcal{F}_k^{\mathbf{A}'_2}$ $(1 \leq j \leq n)$, we have $g_i \in \mathcal{F}_{m_i+k}^{\mathbf{A}'_l}$ $(1 \leq i \leq |\vec{m}|)$ and $h_j \in \mathcal{F}_k^{\mathbf{A}'_l}$ $(1 \leq j \leq n; l = 1, 2)$.

  So, $\sigma \odot < curry_{k,\vec{m}}(\vec{g}), \vec{h} > \in \mathcal{F}_k^{\mathbf{A}'_l}$ $(l = 1, 2)$.

  Hence, $\sigma^{\mathbf{A}} \odot < curry_{k,\vec{m}}(\vec{g}), \vec{h} > \in \mathcal{F}_k^{\mathbf{A}'_1} \cap \mathcal{F}_k^{\mathbf{A}'_2}$. □

From the proof of the above lemma, we know that Lemma 3.2.1 can be extended to an intersection of a class of sub-eBAs instead of the intersection of just two sub-eBAs. Formally

Lemma 3.2.2 (intersection of sub-eBAs): *Let $\mathcal{K}$ be a class of sub-eBAs of an eBA $\mathbf{A}$. Then $\bigcap \mathcal{K} \prec \mathbf{A}$.*

Proof. omitted, since it is similar to the above one. □

Now, we introduce a concept of sub-eBAs generated by a sub-structure, which is an analogy to generated subalgebras in Definition 2.1.13.

Definition 3.2.3 (generated sub-eBA): *Let $\mathbf{X}$ be a family of X and X_k which are subsets of A and $\mathcal{F}_k$ of an eBA $\mathbf{A}$ $(k \geq 0)$. The smallest sub-eBA containing $\mathbf{X}$, written as $[\mathbf{X}]$ or $Sub(\mathbf{X})$, is $\bigcap \mathcal{K}$ where $\mathcal{K} = \{\mathbf{A}' | \mathbf{A}' \prec \mathbf{A}$ and $\mathbf{X} \subseteq \mathbf{A}'\}$.*

To justify the word "*generated*", we show how to construct the least sub-eBA $[\mathbf{X}]$, which contains $\mathbf{X}$ as follows.

Definition 3.2.4 (extension *ext* from a base): *The extension of $\mathbf{X}$, $\mathbf{ext}(\mathbf{X})$, is a family of extensions $ext(X)$ and $ext_k(X)$ of $\mathbf{X}$ $(k \in Nat)$, where*

1.

$$ext(\mathbf{X}) =_{df} X \cup \{g(\vec{a}) | g \in X_m \text{ for some } m \wedge a_j \in X\ (1 \leq j \leq m)\}$$
$$\cup \{\sigma(\vec{g}, \vec{a}) | \sigma \in \Sigma_{<\vec{m},n>} \wedge g_i \in \mathbf{X}_{m_i}\ (1 \leq i \leq |\vec{m}|) \wedge a_j \in X\ (1 \leq j \leq n)\}$$

2.

$$ext_k(\mathbf{X}) =_{df} X_k \cup \{C_{k,a} | a \in X\} \cup \{\pi_{k,i} | 1 \leq i \leq k\}$$
$$\cup\{g \odot < \vec{h} > | g \in \mathbf{X}_m \wedge h_j \in X_k \ (1 \leq j \leq m) \ for \ m > 0 \}$$
$$\cup\{\sigma^{\mathbf{A}} \odot < curry_{k,\vec{m}}(\vec{g}), \vec{h} > | \sigma \in \Sigma_{<\vec{m},n>} \wedge g_i \in X_{m_i+k} \wedge h_j \in X_k\}$$

Let $\mathbf{ext}^{j+1}(\mathbf{X}) = \mathbf{ext}(\mathbf{ext}^j(\mathbf{X}))$ $(j \geq 0)$ where $\mathbf{ext}^0(\mathbf{X})$ is $\mathbf{X}$. In what follows, we firstly show that the extension grows larger and larger by every extending if we recursively use **ext**. Then, we verify (3.2.a) that the least upper bound of such extensions forms a sub-eBA and (3.2.b) that it coincides with the generated sub-eBA.

Lemma 3.2.5 (monotonicity of *ext*):

(a) $\mathbf{ext}^j(X) \subseteq \mathbf{ext}^{j+1}(X)$, *for any* $j \geq 0$;

(b) $\mathbf{ext}^j(X) \subseteq \mathbf{ext}^{j'}(X)$ *for any* $0 \leq j \leq j'$.

Next, we show that such an extension from a sub-collection of an eBA is inside the eBA. Formally,

Lemma 3.2.6 (boundness of *ext*): $\mathbf{ext}^j(\mathbf{X}) \subseteq \mathcal{F}^{\mathbf{A}}$ *for any* $j \geq 0$, *where* $\mathbf{X} \subseteq \mathbf{A}$.

Then, we confirm that the least upper bound of such extensions (extended recursively by **ext** from $\mathbf{X}$ which is inside of an eBA) is an sub-eBA of the eBA.

Lemma 3.2.7 (countably-limit of *ext*):

(a) $\bigcup_{j\in\omega} \mathbf{ext}^j(\mathbf{X})$ *is explicitly closed;*

(b) for any $\sigma \in \Sigma_{<\vec{m},n>}$, $\sigma^{\mathbf{A}}$ *is uniform over* $\bigcup_{j\in\omega} \mathbf{ext}^j(\mathbf{X})$.

Therefore,

Theorem 3.2.8 (*ext* and sub-eBA): $\bigcup_{j\in\omega} \mathbf{ext}^j(\mathbf{X}) \prec \mathbf{A}$ *provided* $\mathbf{X} \subseteq \mathbf{A}$.

From Theorem 3.2.8, we know that a sub-eBA can be constructed from $\mathbf{X}$. In what follows, we show that it is indeed the generated sub-eBA containing $\mathbf{X}$.

From Lemma 3.2.6, we understand that $\mathbf{ext}(\mathbf{X}) \subseteq \mathcal{F}^{[\mathbf{X}]}$ and $\mathbf{ext}^j(\mathbf{X}) \subseteq \mathcal{F}^{[\mathbf{X}]}$, for any $j \geq 0$. So, $\bigcup_{j\in\omega} \mathbf{ext}^j(\mathbf{X}) \subseteq [\mathbf{X}]$. Subsequently, by Theorem 3.2.8 and by Definition 3.2.3, we have the following:

Theorem 3.2.9 (*ext* **and the generated sub-eBA**): $\bigcup_{j\in\omega} \mathbf{ext}^j(\mathbf{X}) = [\mathbf{X}]$.

This theorem justifies the terminology of "generated" sub-eBAs, and says that $[\mathbf{X}]$ is the sub-eBA generated by $\mathbf{X}$. Further, let $\mathbf{A}$ be an eBA and $\mathbf{X}$ be a subcollections of $\mathbf{A}$, then $\mathbf{A}$ is said to be the eBA *generated by* $\mathbf{X}$ if $[\mathbf{X}] = \mathbf{A}$. From this, we are led to the following: the restriction of a perfect sub-algebra to its basis is an eBA. Formally,

Lemma 3.2.10 (perfect sub-eBA and eBA): *Let* $\mathbf{A}$ *be an eBA, and* $\mathbf{A}' \preceq \mathbf{A}$. *Then* $\mathbf{A}'\lceil_{A'}$ *is an eBA, where* $\mathcal{F}(A)^{(\mathbf{A}'\lceil_{A'})} = (\mathcal{F}(A)^{\mathbf{A}'})\lceil_{A'} = \{\mathcal{F}_m^{\mathbf{A}'}(A)\lceil_{A'} | m \in Nat\}$.

Note that the domain of function g in $\mathcal{F}(A)^{\mathbf{A}'}$ is in a product of multiple copies of A and $\mathcal{F}(A)^{\mathbf{A}'}\lceil_{A'}$ is to limit the domains of its functions to products of multiple copies of A'.

The great difference between an eBA and a sub-eBA (or between a perfect sub-eBA and a sub-eBA) is Extensionality. Because of Extensionality and the presence of function carriers, we lose some important properties of first order algebras in the present FBO. For example, see Theorem 3.7.22, whether can we drop off the condition "onto" to an ordinary eBH? This "dropping-off" plays a crucial role in showing that an eBA is a *free* eBA over a class $\mathcal{K}$. To get a positive answer, we consequently have to introduce "*admissible*" concept, see section 3.8.

3.3 eBHs and their uniqueness over generated eBAs

This section is mainly to prove Theorem 3.3.9, which is important in Section 3.6 related to free eBAs. Firstly, we show that eBHs preserve "explicit closedness". The preservation of uniformity under eBHs follows. Therefore, an image of an eBH is a sub-eBA (Lemma 3.3.3).

Lemma 3.3.1 (explicit-closedness preserved by eBHs): *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs, where* $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$; ζ *be an eBH from* $\mathbf{A}$ *to* $\mathbf{A}'$. *Thus, if a*

sub-family $< A', \{\mathcal{G}_k | k \in Nat\} >$ *of the family* $< A, \{\mathcal{F}_k | k \in Nat\} >$ *is explicitly closed, so is* $\zeta(\mathcal{G})$.

Secondly, eBHs preserve "uniformity" over explicitly-closed families. Formally,

Lemma 3.3.2 (uniformity preserved by eBHs): *Let* $\mathbf{A}$ *and* $\mathbf{A}'$ *be eBAs, where* $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$; ζ *be an eBH from* $\mathbf{A}$ *to* $\mathbf{A}'$; $\mathcal{G} \subseteq \mathcal{F}$ *be explicitly closed. Then, if* $\sigma^{\mathbf{A}}$ *is uniform over* $\mathcal{G}$, *then so is* $\sigma^{\mathbf{A}'}$ *over* $\zeta(\mathcal{G})$.

From the previous two lemmas, we can conclude that the image of an eBH is a sub-eBA. That is,

Lemma 3.3.3 (sub-eBA and image of an eBH): *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs,* ζ *be an eBH from* $\mathbf{A}$ *to* $\mathbf{A}'$. *Then,* $\zeta(\mathbf{A}) \prec \mathbf{A}'$.

Proof Obvious from Lemma 3.3.1 and Lemma 3.3.2. □

Consequently, we have that eBHs preserve sub-eBAs. Formally,

Lemma 3.3.4 (sub-eBAs preserved by eBHs): *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs,* ζ *be an eBH from* $\mathbf{A}$ *to* $\mathbf{A}'$, *and* $\mathbf{A}'' \prec \mathbf{A}$. *Then,* $\zeta(\mathbf{A}'') \prec \mathbf{A}'$.

Next, we demonstrate that a composition of two eBHs is a eBH.

Lemma 3.3.5 (composition of eBHs): *Let* $\mathbf{A}$, $\mathbf{A}'$, $\mathbf{A}''$ *be eBAs,* ζ' *be an eBH from* $\mathbf{A}$ *to* $\mathbf{A}'$ *and* ζ *be an eBH from* $\mathbf{A}'$ *to* $\mathbf{A}''$. *Then* $\zeta \odot \zeta'$ *is an eBH from* $\mathbf{A}$ *to* $\mathbf{A}''$.

Now, we would say that two eBHs agree on the extension of $\mathbf{X}$ if they agree on $\mathbf{X}$.

Lemma 3.3.6 (*ext* **and eBHs):** *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs, where* $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$; *and* $\mathbf{X}$ *be a sub-family* $< X, \{X_k | k \in Nat\} >$ *of the family* $< A, \{\mathcal{F}_m | m \in Nat\} >$. *Thus, if* $\zeta, \zeta' : \mathbf{A} \to \mathbf{A}'$ *agree on* $\mathbf{X}$, *so do they on* $\mathbf{ext}(\mathbf{X})$.

Proof.

we have the followings.

(i) for $a \in ext(\mathbf{X})$, there are three cases:

(i.a) if $a \in X$, then $\zeta' a = \zeta a$, by the agreement on $\mathbf{X}$.

(i.b) if $a = g(\vec{a})$ for some $m > 0$, $g \in X_m$ and $a_j \in X$ $(1 \leq j \leq m)$, then

$$\zeta' a = (\zeta' g)(\zeta' \vec{a}) = (\zeta g)(\zeta \vec{a}) = \zeta a.$$

(i.c) if $a = \sigma^{\mathbf{A}}(\vec{g}, \vec{a})$ for some $\sigma \in \Sigma_{<\vec{m},n>}$, $g_i \in X_{m_i}$ $(1 \leq i \leq n)$ and $a_j \in X$ $(1 \leq j \leq n)$, then $\zeta' a = \sigma^{\mathbf{A}'}(\zeta' \vec{g}, \zeta' \vec{a}) = \sigma^{\mathbf{A}'}(\zeta \vec{g}, \zeta \vec{a}) = \zeta a$.

(ii) for $k \in Nat$, any $h \in ext_k(\mathbf{X})$, we have five cases:

(ii.a) if $h \in \mathbf{X}_k$, then $\zeta' h = \zeta h$, by the agreement on $\mathbf{X}$.

(ii.b) if $h = C^{\mathbf{A}}_{k,a}$ for some $a \in X$, then $\zeta' h = C^{\mathbf{A}'}_{k,\zeta' a} = C^{\mathbf{A}'}_{k,\zeta a} = \zeta h$.

(ii.c) if $h = \pi^{\mathbf{A}}_{k,i}$ for some $1 \leq i \leq k$, then $\zeta' h = \pi^{\mathbf{A}'}_{k,i} = \zeta h$.

(ii.d) if $h = g \odot < \vec{h} >$ for some $g \in X_m$ and $h_j \in X_k$ $(1 \leq j \leq m)$, then

$$\zeta' h = (\zeta' g) \odot < \zeta' \vec{h} >= (\zeta g) \odot < \zeta \vec{h} >= \zeta h$$

(ii.e) if $h = \sigma^{\mathbf{A}} \odot < curry_{k,\vec{m}}(\vec{g}), \vec{g'} >$ for $\sigma \in \Sigma_{<\vec{m},n>}$ and $g_i \in X_{m_i+k}$ $(1 \leq i \leq \ell)$ and $g'_j \in X_k$ $(1 \leq j \leq n)$, then

$$\zeta' h = \sigma^{\mathbf{A}'} \odot < curry_{k,\vec{m}}(\zeta' \vec{g}), \zeta' \vec{g'} >= \sigma^{\mathbf{A}'} \odot < curry_{k,\vec{m}}(\zeta \vec{g}), \zeta \vec{g'} >= \zeta h. \square$$

From this result, we expect to generalize it to arbitrary finite times of applying **ext** to $\mathbf{X}$, instead of only once. Formally,

Lemma 3.3.7 (monotonicity with *ext* and eBHs): *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs, where* $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$; $\mathbf{X}$ *be a sub-family* $< X, \{X_k | k \in Nat\} >$ *of the family* $< A, \{\mathcal{F}_m | m \in Nat\} >$. *Thus, if* $\zeta', \zeta : \mathbf{A} \to \mathbf{A}'$ *agree on* $\mathbf{X}$, *so do they* $\mathbf{ext}^j(\mathbf{X})$ *for* $j \geq 0$.

Proof

We omit the proof and only remind you of that the proof needs to incorporate an induction on j. $\square$

Therefore, we would have that two eBHs agree on a generated sub-eBA if they agree on the generators. Formally,

Lemma 3.3.8 (eBHs and generated sub-eBAs): *Let* $\mathbf{A}$, $\mathbf{A}'$ *be eBAs, where* $\mathbf{A} =< A, \{\mathcal{F}_m | m \in Nat\} >$; $\mathbf{X}$ *be a sub-faimly* $< X, \{X_k | k \in Nat\} >$ *of the family* $< A, \{\mathcal{F}_m | m \in Nat\} >$. *Thus, if* $\zeta' : \mathbf{A} \to \mathbf{A}'$ *and* $\zeta : \mathbf{A} \to \mathbf{A}'$ *agree on* $\mathbf{X}$, *then so do they on the generated sub-eBA* $[\mathbf{X}]$.

Proof by Lemma 3.3.7 and Theorem 3.2.9.□

Naturally, we come to the following.

Theorem 3.3.9 (uniqueness of eBHs on generated eBAs): *Let* $\mathbf{A}$ *be a generated eBA and* $\mathbf{X}$ *be its generator. Thus, if* $\zeta', \zeta : \mathbf{A} \to \mathbf{A}'$ *agree on* $\mathbf{X}$, *then* $\zeta' = \zeta$.

Proof. by Lemma 3.3.8. □

3.4 Constructing term eBA

In this section, we are to introduce the term eBA $\mathbf{T}$. Apparently, the introduction is dominated by an actual construction. The key point of the construction is to build carriers $< T, \{\mathcal{F}^{\mathbf{T}}_m | m \in Nat\} >$ from binding terms BT, and the carriers form an explicitly-closed family with uniform interpretations over the family. Firstly, let us introduce an equivalence relation $\dot{\approx}$ on binding terms BT, which is essentially a version of α-conversions in the Framework for Binding Operators.

Definition 3.4.1 ($\dot{\approx}$ **on** BT): $\dot{\approx}$ *is the least equivalence relation family (on binding terms* BT*) which is closed under* α*-conversions,* ξ*-conversions, anti-*ξ*-conversions* (ξ^{-1}) *and compositions of function variables and of Binding Operators; i.e.* $\dot{\approx}$ *is the least fixed point of* $\mathcal{M}_{\dot{\approx}}$ *where* $\mathcal{M}_{\dot{\approx}}$ *is defined as follows: for every equivalence relation* R,

($\dot{\approx}$-α) $< \langle \vec{x} : t \rangle, \langle \vec{y} : t[\vec{x} := \vec{y}] \rangle > \in \mathcal{M}_{\dot{\approx}}(R)$,

where $t \in T$ *and* $y_j \notin Free(t)$ $(1 \leq j \leq |\vec{x}| = |\vec{y}|)$;

($\dot{\approx}$-ξ^{-1}) *if* $< \langle \vec{z} : u \rangle, \langle \vec{z'} : u' \rangle > \in R$ *then* $< u[\vec{z} := \vec{y}], u'[\vec{z'} := \vec{y}] > \in \mathcal{M}_{\dot{\approx}}(R)$,

where $\{\vec{y}\} \subseteq V$ *and* $\{\vec{y}\} \cap ((Free(u) - \{\vec{z}\}) \cup (Free(u') - \{\vec{z'}\})) = \emptyset$;

($\dot{\approx}$-ξ) if $< t, u > \in R$, *then* $< \langle \vec{x} : t \rangle, \langle \vec{x} : u \rangle > \in \mathcal{M}_{\dot{\approx}}(R)$;

($\dot{\approx}$-cmp-1) if $< t_j, u_j > \in R$ $(1 \leq j \leq |\vec{t}| = |\vec{u}|)$, *then* $< f(\vec{t}), f(\vec{u}) > \in \mathcal{M}_{\dot{\approx}}(R)$;

($\dot{\approx}$-cmp-2) if $< ft_i, fu_i > \in R$ $(1 \leq i \leq \ell)$ *and* $< t_j, u_j > \in R$ $(1 \leq j \leq n)$, *then* $< \sigma(\vec{ft}, \vec{t}), \sigma(\vec{fu}, \vec{u}) > \in \mathcal{M}_{\dot{\approx}}(R)$; *where* $\sigma \in \Sigma_{<\vec{m},n>}$ *and* $|\vec{m}| = \ell$.

Note that $\dot{\approx}$ is the least fixed point of $\mathcal{M}_{\dot{\approx}}$ on binding terms BT, i.e. $\dot{\approx} =_{df} \sqcup \mathcal{M}_{\dot{\approx}}(\mathcal{I}) = \bigcup_{j \in Nat} \mathcal{M}^j_{\dot{\approx}}(\mathcal{I})$ where $\mathcal{I}$ is the pure identity relation on binding terms BT.

In what follows, we try to establish the validity of $\dot{\approx}$ (Theorem 3.4.3). For this purpose, we show that syntactic substitutions are exchangeable with semantic substitutions as a first step toward the validity.

Lemma 3.4.2 (syntactic substitution vs semantic substitution): *For* $t \in T$ *and* $ft \in FT_m$ $(m \in Nat)$, *we have that for every environment* $< \rho, \varphi >$ *on an eBA* **A**, *the following holds:*

(a) $\mathcal{A}[\![t]\!](\rho[\mathcal{A}[\![\vec{t}]\!](\rho,\varphi)/\vec{x}], \varphi[\mathcal{A}[\![\vec{fu}]\!](\rho,\varphi)/\vec{f}]) = \mathcal{A}[\![t\,[\vec{x},\vec{f} := \vec{t},\vec{fu}]\,]\!](\rho,\varphi)$,

(b) $\mathcal{A}[\![ft]\!](\rho[\mathcal{A}[\![\vec{t}]\!](\rho,\varphi)/\vec{x}], \varphi[\mathcal{A}[\![\vec{fu}]\!](\rho,\varphi)/\vec{f}]) = \mathcal{A}[\![ft\,[\vec{x},\vec{f} := \vec{t},\vec{fu}]\,]\!](\rho,\varphi)$,

where $\vec{f}$, $\vec{fu}$ *are a list of distinct function variables and a list of function terms with compatible arities to* $\vec{f}$, *respectively.*

We refer you to Section 5.1 for substitutions involved.

Proof

By combining (a) and (b) together with structural induction on binding terms BT. □

Theorem 3.4.3 (validation of relation $\dot{\approx}$): *$\dot{\approx}$-equivalence is sound, i.e.* $p \dot{\approx} q$ *implies* $\mathcal{A}[\![p]\!](\rho,\varphi) = \mathcal{A}[\![q]\!](\rho,\varphi)$ *for every* $< \rho, \varphi >$.

Proof We know that $\mathcal{M}_{\dot{\approx}}$ is monotonic and its least fixed point is $\dot{\approx}$. So, by induction on j of $\mathcal{M}^j_{\dot{\approx}}(\mathcal{I})$, we would have what we want. □

Let $[t]$ be $\{t' \in T | t' \dot{\approx} t\}$ for $t \in T$, and $[ft]$ be $\{ft' \in FT_m | ft' \dot{\approx} ft\}$ for $ft \in FT_m$. We are going to give the term eBA below, but we avoid the terminology of "term eBA" before its verification.

Definition 3.4.4 (carriers and interpretations from BT): *Let*

(i) $[T]$ *be* $\{[t] | t \in T\}$;

(ii) $g_{[ft]}([\vec{u}]) = [t\,[\vec{x} := \vec{u}]\,]$ *if* $ft = \langle \vec{x} : t \rangle$;

(iii) for $\sigma \in \Sigma_{<\vec{m},n>}$, *its interpretation* $\sigma^{\mathbf{T}}$ *be a functional such that* $\sigma^{\mathbf{T}}(\vec{g}_{[\vec{ft}]}, [\vec{t}]) = [\sigma(\vec{ft}, \vec{t})]$,

where $\ell = |\vec{m}| = |\vec{ft}|$, $n = |\vec{t}|$ *and* $\vec{g}_{[\vec{ft}]}$ *is the list* $g_{[ft_1]}, g_{[ft_2]}, \cdots, g_{[ft_{|\vec{ft}|}]}$.

For simplicity, we sometimes refer $ft(\vec{t})$ as $ft||_{|\vec{x}|}\,[\vec{x} := \vec{t}]$ in future, where $\vec{x} = ft|_{|\vec{x}|}$ (see definitions for $|_m$ and $||_m$ after Definition 5.1.2 in Section 5.1).

Since $\dot{\approx}$ is an equivalence relation, this guarantees the well-definedness of Definition 3.4.4, i.e. the values do not depend on their representative. Thus, we can also simplify $[t]$ and $g_{[ft]}$ as $\bullet_{[t]}$ and $\bullet_{[ft]}$ respectively. This is to emphasize the fact that they can be generated from binding terms t and ft respectively through the indices.

In what follows, we are to verify that such a construction yields an eBA (Theorem 3.4.9). Firstly, we show that the construction has Extensionality (Lemma 3.4.5). It is followed by a verification of that compositions are closed in the family constructed in Definition 3.4.4 (Lemma 3.4.6). Thirdly, that the family is explicitly-closed is examined (Theorem 3.4.7). Lastly, the uniformity of interpretations of BOs is established (Lemma 3.4.8). To be explicit, let $\mathbf{T}$ be $< [T], \mathcal{F} >$, and $\mathcal{F}$ be the family $\{\mathcal{F}_k | k \in Nat\}$, where $\mathcal{F}_k =_{df} \{g_{[ft]} | ft \in FT_k\}$ for each $k \in Nat$. Thus, we have the following.

Lemma 3.4.5 ($\dot{\approx}$*-ext*): *For* $ft, fu \in FT_m$ $(m \in Nat)$, *if for every* $[t_j]$ $(1 \leq j \leq m)$ $g_{[ft]}([\vec{t}\,]) = h_{[fu]}([\vec{t}\,])$ *then* $ft \dot{\approx} fu$.

Proof

To verify this, just let t_j be $z_j \in V$ such that $z_j \notin Free(ft) \cup Free(fu) \cup \{\vec{z}\lceil_{j-1}\}$; and by ($\dot{\approx}$-$\xi$), ($\dot{\approx}$-$\alpha$) and transitivity, you can get it (i.e. $\dot{\approx}$-ext). □

Next, we verify that the family $\mathbf{T}$ is closed under function compositions. Formally,

Lemma 3.4.6 (closedness of compositions in $\mathcal{F}^{\mathbf{T}}$): *Let $g_{[ft]}$ be the function generated by $ft \in FT_m$ ($m \in Nat$), $\vec{g}$ be a list of functions generated by a list $\vec{ft}$ of function terms where $ft_i \in FT_k$ ($k \in Nat$). Then, $\bullet_{[ft]} \odot < \vec{g} >= \bullet_{[(\vec{x}:fu)]}$, where $fu = ft||_m [\vec{y} := \vec{t}]$, $t_i = ft_i||_k [\vec{y^i} := \vec{x}]$, $ft_i|_k = \vec{y^i}$, and $x_j \notin (Free(ft) \cup \bigcup_{1 \leq i \leq k} Free(ft_i)) \cap V$ ($1 \leq j \leq |\vec{x}|$).*

Proof. By $\dot{\approx}$-ext (i.e. Lemma 3.4.5). □

Theorem 3.4.7 (explicit closedness of $\mathcal{F}^{\mathbf{T}}$): *$\mathcal{F}([T])$ is explicitly closed.*

Proof.

- For any $[t] \in [T]$, and any $k > 0$, let $\vec{x} \notin Free(t)$. Then

  $C_{k,[t]} = g_{[(\vec{x}:t)]} \in \mathcal{F}_k([T])$

  by *Lemma 5.1.9* (see Section 5.1 in Chapter 5).

- For any $k > 0$, let $\vec{x} \in V$. Then, for any $1 \leq i \leq k$, by *Lemma 5.1.9* we have

  $\pi_{k,i} = g_{[(\vec{x}:x_i)]} \in \mathcal{F}_k([T])$.

- For any $m > 0$, $k \geq 0$, and given $g_{[ft]} \in \mathcal{F}_m([T])$ and $h_{j[ft_j]} \in \mathcal{F}_k([T])$ ($1 \leq j \leq m$), we get $g_{[ft]} \dot{\odot} < h_{[\vec{ft}]} > \in \mathcal{F}_k([T])$ by *Lemma 3.4.6.* □

The next lemma is about "uniformity" over the family $\mathcal{F}^{\mathbf{T}}$.

Lemma 3.4.8 (uniformity over $\mathcal{F}^{\mathbf{T}}$): *For $\sigma \in \Sigma_{<\vec{m},n>}$ with $|\vec{m}| = \ell$, let $\vec{ft}$ be a list of function terms such that the ith element ft_i of $\vec{ft}$ has arity $k + m_i$, and all elements in list $\vec{fu}$ share a same arity k. Then,*

$$\sigma^{\mathbf{T}} \odot < \vec{g}, \vec{g'} >= h,$$

where $\vec{x}$ are not free in $\vec{ft}$ and $|\vec{x}| = k$; $g_i = curry_{k,m_i}(\bullet_{[ft_i]})$ ($1 \leq i \leq \ell$); $g'_j = \bullet_{[fu_j]}$ ($1 \leq j \leq n$); $h = \bullet_{[(\vec{x}:\sigma(\vec{fv},\vec{v}))]}$ such that $fv_i = \langle \vec{y^i} : ft_i||_{k+m_i} [\vec{z^i} := \vec{x}, \vec{y^i}] \rangle$,

$ft_i|_{k+m_i} = \vec{z^i}$, *and* $y^i_j \notin Free(ft_i) \cup \{\vec{x}\} \cup \{\vec{y^i}\lceil_{j-1}\}$; *and* $v_j = fu_j||_k [\vec{w^j} := \vec{x^i}]$ *such that* $fu_j|_k = \vec{w^j}$.

Proof.

By *Lemma 3.4.6*, ($\tilde{\approx}$-α) and ($\tilde{\approx}$-ext), see Lemma 3.4.5.

Also, there is another technical point we should mention. That is,

$curry_{k,m}(g_{[ft]})([t_1],[t_2],...,[t_k])$ can be generated by $[\langle \vec{y} : ft||_{k+m} [\vec{z} := \vec{t}, \vec{y}]\rangle]$, where $\vec{z} = ft|_{k+m}$ and y_j is a $y \in V$ such that $y \notin Free(ft) \cup \bigcup_{j=1}^{k} Free(t_j) \cup \{\vec{y}\lceil_{j-1}\}$. □

In other words, Lemma 3.4.8 actually said is that $\sigma^{\mathbf{T}}$ is uniform over $\mathcal{F}([T])$ for each $\sigma \in \Sigma_{<\vec{m},n>}$. Therefore, it is appropriate to say that $\mathbf{T}$ is an eBA, and term eBA will be used to refer it.

Theorem 3.4.9 (term eBA): $\mathbf{T} = <[T], \mathcal{F}([T])>$ *is an eBA.*

Proof. By *Theorem 3.4.7* and *Lemma 3.4.8.* □

3.5 Generatability of term eBA

In this section, we are going to show that the term eBA $\mathbf{T}$, defined in last section (Section 3.4), can actually be generated through its ordinary variables and function variables (Theorem 3.5.8). Firstly we seek a syntactic counterpart $\vec{\Im}$ of *ext* as in Section 3.2. The one provided below is not a direct counterpart but close enough to serve for our purpose.

Definition 3.5.1 (syntactic extension $\vec{\Im}$): *Let* $\vec{X}$ *be a pair* $< X, \{X_k | k \in Nat\} >$ *where* X *is a subset of ordinary binding terms* T *and* X_k *is a subset of function terms* FT_k *with arity* $k \geq 0$. *We define* $\vec{\Im}$ *inductively as follows:*

1.

$$\Im(\vec{X}) =_{df} X \cup \{ft||_m [\vec{x} := \vec{t}] \mid ft \in (X_m - FV_m) \wedge t_j \in X (1 \leq j \leq m)\}$$
$$\cup \{f(\vec{t}) | f \in (X_m \cap FV_m) \wedge t_j \in X (1 \leq j \leq m)\}$$
$$\cup \{\sigma(\vec{ft}, \vec{t}) | \sigma \in \Sigma_{<\vec{m},n>} \wedge ft_i \in X_{m_i} (1 \leq i \leq \ell) \wedge t_j \in X (1 \leq j \leq n)\}$$

where $\ell = |\vec{m}|$;

2. $\Im_k(\vec{X}) =_{df} X_k \cup \{ \langle \vec{x} : t \rangle | x_j \in V(1 \leq j \leq k) \wedge t \in \Im(X) \}$, *for* $k \geq 0$ *and* $\vec{x}(i) \neq \vec{x}(j)$ $(i \neq j)$.

For convenience, let $\Im^0(\vec{X}) = \vec{X}$ and $\Im^{j+1}(\vec{X}) = \Im(\Im^j(\vec{X}))$. Also we let $\mathbf{X}$ be a pair $< \{\bullet_{[t]} | t \in X\}, \{\mathbf{X}_k | k \in Nat\} >$ such that $\mathbf{X}_k = \{\bullet_{[ft]} | ft \in X_k\}$ for $k \geq 0$. Firstly, we show that $\vec{\Im}$ is a proper extension for binding terms. Formally,

Lemma 3.5.2 (boundness of $\vec{\Im}$): *For* $X \subseteq T$ *and* $X_k \subseteq FT_k$ $(k \geq 0)$, *we have the following:*

(a) $\vec{\Im}(\vec{X}) \subseteq T$ *and* $\vec{\Im}_k(\vec{X}) \subseteq FT_k$ *for* $k \geq 0$;

(b) $\vec{\Im}^j(\vec{X}) \subseteq T$ *and* $\vec{\Im}^j_k(\vec{X}) \subseteq FT_k$ $(k \geq 0)$, *for every* $j \geq 0$.

Next, we show that $\vec{\Im}$ is monotonic under the usual order of set inclusions. Formally,

Lemma 3.5.3 (monotonicity of $\vec{\Im}$):

(i) $\vec{\Im}^j(\vec{X}) \subseteq \vec{\Im}^{j+1}(\vec{X})$, *for any* $j \geq 0$;

(ii) $\vec{\Im}^j(\vec{X}) \subseteq \vec{\Im}^{j'}(\vec{X})$, *for* $0 \leq j \leq j'$.

From Lemma 3.5.2 and Lemma 3.5.3, we understand that the sequence $\{\vec{\Im}^j(\vec{X}) | j \geq 0\}$ has an upper bound and it is non-decreasing. Apparently, it has a least upper bound. We give one of them as follows.

Lemma 3.5.4 (binding terms BT and $\vec{\Im}$): *In* $\vec{X}$, *let* $X = V$ *and* $X_k = \{\langle \vec{x} : f(\vec{x}) \rangle | f \in FV_k$ *for* $k \geq 0$ *and* $x_j \in V\}$. *then* $\bigcup_{j \in \omega} \vec{\Im}^j(\vec{X}) = T$ *and* $\bigcup_{j \in \omega} \vec{\Im}^j_k(\vec{X}) = FT_k$ *for* $k \geq 0$.

Proof

By Lemma 3.5.2, we know that $\bigcup_{j \in \omega} \vec{\Im}^j(\vec{X}) \subseteq T$ and $\bigcup_{j \in \omega} \vec{\Im}^j_k(\vec{X}) \subseteq FT_k$ for $k \geq 0$.

For the reversed containments, just to check $\bigcup_{j \in \omega} \vec{\Im}^j(\vec{X})$ is closed under the conditions of Definition 1.3.1. □

Lemma 3.5.4 says that the collection of all possible arbitrary finite times of syntactic extensions is just the collections of binding terms, and no term is missing from the union of all these extensions. Lemma 3.5.5 and Lemma 3.5.6 exhibit a relation from the syntactic extension $\Im$ (Definition 3.5.1) to the semantic extension *ext* (Definition 3.2.4).

Lemma 3.5.5 (from syntactic $\vec{\Im}$ to semantic *ext*): *Let $\vec{X}$ be the same as in Lemma 3.5.4. Then,*

(a) $[t] \in ext^2(\mathbf{X})$ *for* $t \in \Im(\vec{X})$;

(b) $g_{[ft]} \in ext^2_k(\mathbf{X})$ *for* $ft \in \Im_k(\vec{X})$.

Proof.

The proof for the first part (a) is left out.

We are considering the second part (b), i.e. $ft = \langle \vec{x} : t \rangle$ for some $t \in \Im(\vec{X})$ and $\vec{x} \in V$. There are two possibilities as follows :

(i) $t = y$ for some $y \in X$;

(ii) $t = f(\vec{y}\,)$ for some $f \in X_m$ and $y_j \in X$ $(1 \leq j \leq m)$.

For (i), if $y = x_i \in \{\vec{x}\}$ for some $1 \leq i \leq k = |\vec{x}|$, then $g_{[\,\langle \vec{x}:y \rangle\,]} = \pi_{k,i} \in ext_k(\mathbf{X})$; if $y \notin \{\vec{x}\}$ then $g_{[\,\langle \vec{x}:y \rangle\,]} = C_{k,[y]} \in ext_k(\mathbf{X})$.

For (ii), we have $g_{[\langle \vec{x}:f(\vec{y}) \rangle]} = \bullet_{[\langle \vec{x}:f(\vec{x}) \rangle]} \odot < \bullet_{[\langle \vec{x}:y_1 \rangle]}, \bullet_{[\langle \vec{x}:y_2 \rangle]}, \ldots, \bullet_{[\langle \vec{x}:y_m \rangle]} >$ From (i), we have that $\bullet_{[\langle \vec{x}:y_j \rangle]} \in ext_k(\mathbf{X})$, for $1 \leq j \leq m$ and $\bullet_{[\langle \vec{x}:f(\vec{x}) \rangle]} \in \mathbf{X}_m$. So, we have $g_{[\,\langle \vec{x}:f(\vec{y}) \rangle\,]} \in ext^2_k(\mathbf{X})$. □

The above result can be extended to the following.

Lemma 3.5.6 (syntactic $\vec{\Im}$ and semantic *ext* with term eBA): *Let $\vec{X}$ be the same as in Lemma 3.5.4. Then, for every $j \geq 0$, we have the followings:*

(i) if $t \in \vec{\Im}^j(\vec{X})$, $[t] \in ext^{3^j}(\mathbf{X})$;

(ii) if $ft \in \vec{\Im}^j_k(\vec{X})$, $g_{[ft]} \in ext^{3^j}_k(\mathbf{X})$ *for* $k > 0$.

Proof.

Similar to the proof of *Lemma 3.5.5*, we leave the first part (i) out simply because that it is quite easy.

For the second part (ii), i.e. $ft \in \vec{\Im}_k^{j+1}(\vec{X})$, there are two possibilities:

(a) $ft \in \vec{\Im}_k^j(\vec{X})$,

(b) $ft = \langle x_1, x_2, ..., x_k : t \rangle$ for some $t \in \vec{\Im}^{j+1}(\vec{X})$.

For (a), by inductive hypothesis, $g_{[ft]} \in ext^{3j}(\mathbf{X})$.

For (b), there are four cases:

(b.1) $t \in \vec{\Im}^j(\vec{X})$,

(b.2) $t = f(\vec{t})$ for some $f \in FV_m$ and $t_j \in \vec{\Im}^j(\vec{X})$ $(1 \leq j \leq m)$,

(b.3) $t = fu||_m [\vec{y} := \vec{u}]$ for $fu \in \vec{\Im}_m^j(\vec{X})$ and $fu|_m = \vec{y}$ and $u_j \in \vec{\Im}^j(\vec{X})$.

(b.4) $t = \sigma(\vec{ft}, \vec{t})$ for some $\sigma \in \Sigma_{<\vec{m},n>}$, $ft_i \in \vec{\Im}_{m_i}^j(\vec{X})$ and $t_j \in \vec{\Im}^j(\vec{X})$.

For (b.1), we get $\langle \vec{x} : t \rangle \in \vec{\Im}_k^j(X)$. Then by inductive hypothesis, we have $g_{[\langle \vec{x}:t \rangle]} \in ext_k^{3j}(\mathbf{X})$.

For (b.2), we have $\bullet_{[\langle \vec{x}:t_j \rangle]} \in ext_k^{3j}(\mathbf{X})$ by inductive hypothesis. On the other hand, since $f \in FV_m$, we have

$g_{[\langle \vec{x}:f(\vec{t}) \rangle]} = h_{[\langle \vec{x}:f(\vec{z}) \rangle]} \odot < \bullet_{[\langle \vec{x}:t_1 \rangle]}, \bullet_{[\langle \vec{x}:t_2 \rangle]}, ..., \bullet_{[\langle \vec{x}:t_m \rangle]} >$
$\in ext_k^{3j+1}(\mathbf{X})$ $(k = |\vec{x}| = |\vec{z}|)$.

For (b.3), we have $g = h \odot < \vec{h} > \in ext_k^{3j+2}(\mathbf{X})$, where h is generated by $[\langle \vec{x}\,\vec{z} : fu||_m [\vec{y} := \vec{z}] \rangle]$, $\{\vec{x}\} \cap \{\vec{z}\} = \emptyset$, $|\vec{x}| = k$ and $|\vec{z}| = m$, h_i is generated by $[\langle \vec{x}, \vec{z} : x_i \rangle]$ $(1 \leq i \leq k)$ and h_{k+j} is generated by $[\langle \vec{x}, \vec{z} : u_j \rangle]$ $(1 \leq j \leq m)$.

For (b.4), let h be the function generated by $[\langle \vec{x'} : \sigma(\vec{fu}, \vec{u}) \rangle]$ where $fu_i = \langle \vec{y^i} : (ft_i||_{m_i} [\vec{z^i} := \vec{y^i}]) [\vec{x}, \vec{y^i} := \vec{x'}, \vec{y^i}] \rangle$ and $u_j = t_j [\vec{x} := \vec{x'}]$ such that $\vec{z^i} = ft_i|_{m_i}$, $\{\vec{y^i}\}$ are not in $Free(ft_i)$, $\{\vec{y^i}\}$ disjoint with $\{\vec{x'}\}$ and are not in $Free(ft_i)$ $(1 \leq i \leq \ell)$. Then, by *Lemma 3.4.8* h is equal to $\sigma^{\mathbf{T}} \odot < \vec{h'}, \vec{h''} >$ where $h'_i = curry_{k,m_i}(\bullet_{[\langle \vec{x}, \vec{y^i} : ft_i||_{m_i} [\vec{z^i} := \vec{y^i}] \rangle]})$, $h''_j = \bullet_{[\langle \vec{x}:\vec{t} \rangle]}$.

So, $h \in ext_k^{3j+3}(\mathbf{X})$. This is

because (b.4.i) we have $\bullet_{[\langle \vec{x},\vec{y^i}:ft_i(\vec{y^i})\rangle]} \in ext^{3j+2}_{m_i+k}(\mathbf{X})$ by (b.2) since $\langle \vec{x},\vec{y^i} : ft_i(\vec{y^i})\rangle \in \Im^{n+1}_{m_i+k}(X)$; and

because (b.4.ii) we have $h''_{j[\langle \vec{x}:t_j\rangle]} \in ext^{3j}_k(\mathbf{X})$ by inductive hypothesis, since $\langle \vec{x} : t_j\rangle \in \vec{\Im}^j_k(X)$.

Therefore, the remaining thing for (b.4) is to check whether $h = g$, where g is generated by $[\langle \vec{x} : \sigma(\vec{ft},\vec{t})\ \rangle]$. For this purpose, we have the following: let $fv_i = \langle \vec{x}\,\vec{z^{ji}} : ft_i(\vec{z^{ji}})\rangle$, $fv'_i = \langle \vec{y^i} : fv_i(\vec{x'}\vec{y^i})\rangle$, $fv''_i = \langle \vec{y^i} : fv_i(\vec{u}\,\vec{y^i})\rangle$, $fw_j = \langle \vec{x} : t_j\rangle$, $w'_j = fw_j(\vec{x'})$ and $w''_j = fw_j(\vec{u})$; thus, $\langle \vec{x'} : \sigma(\vec{fv'},\vec{w'})\rangle(\vec{u}) \dot{\approx} \sigma(\vec{fv''},\vec{w''})$.

By ($\dot{\approx}$-α) and the definition of substitutions,

$$(\textit{the above}) \dot{\approx} \sigma(\vec{ft}\ [\vec{x} := \vec{u}]\vec{t}\ [\vec{x} := \vec{u}]) = \sigma(\vec{ft},\vec{t})\,[\vec{x} := \vec{u}] \dot{\approx} \langle \vec{x} : \sigma(\vec{ft},\vec{t})\rangle(\vec{u})$$

Hence, $h = g$ because of Extensionality. □

Therefore,

Lemma 3.5.7 (term eBA and the closure of *ext*):

(i) $[t] \in \bigcup_{j\in\omega} ext^j(\mathbf{X})$, *for each* $t \in T$;

(ii) $g_{[ft]} \in \bigcup_{j\in\omega} ext^j_k(\mathbf{X})$ $(k > 0)$ *for every* $ft \in FT_k$.

As a summary, we have the following.

Theorem 3.5.8 (generated term eBA): *The term eBA is generatable from its ordinary variables and function variables, i.e.* $\mathbf{T} = [\mathbf{X}]$.

Proof by Lemma 3.5.7 and the fact $[\mathbf{X}] \subseteq \mathbf{T}$.□

As a side interest, we give an explicit relation from semantic extension *ext* to syntactic extension $\vec{\Im}$ below. Their proofs are easy and left out as exercises.

Lemma 3.5.9 (from semantic *ext* to syntactic $\vec{\Im}$):

(a) For any $[t] \in ext(\mathbf{X})$, *we have that there is a* $t' \in \vec{\Im}^2(X)$ *such that* $[t'] = [t]$.

(b) For any $g \in ext_k(\mathbf{X})$, *we have that there is a* $ft \in \vec{\Im}^2_k(X)$ *such that* $g = h_{[ft]}$.

This lemma can be generalized to the following.

Lemma 3.5.10 (monotonicity from semantic *ext* to syntactic $\vec{\Im}$): *For any $j > 0$, we have*

(i) if $[t] \in ext^j(\mathbf{X})$, then there is a $t' \in \vec{\Im}^{2j}(X)$ such that $[t] = [t']$;

(ii) if $g \in ext^j_k(\mathbf{X})$, then there is a $ft \in \vec{\Im}^{2j}_k(X)$ such that $g = h_{[ft]}$.

Unlike first order algebras, the verification of term eBA $\mathbf{T}$ is generatable is not trivial as demonstrated in this section. From this generatability, we are led to consider of extension of the universal property (Freeness) from first order algebras to "second order" eBAs. This is the subject of next section.

3.6 Universal property – free eBA

This section is to show the term eBA $\mathbf{T}$ is a *free* eBA (Theorem 3.6.5). If this is the case, then when $FV_m = \emptyset$ for all $m \geq 0$, then the corresponding term eBA is an initial object in the category of eBAs (provided that there are always countably-infinite ordinary variables). Firstly, we give a definition for the "freeness" below.

Definition 3.6.1 (Free eBA): *Let $\mathcal{K}$ be a class of eBAs, and let $\mathbf{A}$ be an eBA which is generated by $\mathbf{X}$, i.e. $\mathbf{A} = [\mathbf{X}]$. If for every $\mathbf{A}' \in \mathcal{K}$ and for every family $\zeta' : \mathbf{X} \to \mathbf{A}'$ of functions from X to A' and of functionals from X_m to $\mathcal{F}^{\mathbf{A}'}_m$ ($m \in Nat$), there is an eBH $\zeta : \mathbf{A} \to \mathbf{A}'$ which extends ζ' (i.e. $\zeta = \zeta'$ for $a \in X$ and $\zeta(g) = \zeta'(g)$ for $g \in X_k$, $k \in Nat$ where $\mathbf{X}$ is a pair of $< X, \{X_k | k \in Nat\} >$), then we say $\mathbf{A}$ has the* universal mapping property *for $\mathcal{K}$ over $\mathbf{X}$. $\mathbf{X}$ is called a set of* free generators *of $\mathbf{A}$, and $\mathbf{A}$ is said to be* freely-generated *by $\mathbf{X}$.*

Theorem 3.6.2 (Free eBA and unique family of extensions): *Suppose $\mathbf{A}$ has the universal mapping property for $\mathcal{K}$ over $\mathbf{X}$. Thus, if we are given $\mathbf{A}' \in \mathcal{K}$ and a map $\zeta' : \mathbf{X} \to \mathbf{A}'$, then the extension ζ of ζ' such that ζ is an eBH from $\mathbf{A}$ to $\mathbf{A}'$ is unique.*

Proof. by *Theorem 3.3.9.* □

Next, we provide a relation between eBHs and generated sub-eBAs.

Lemma 3.6.3 (eBH and generated sub-eBA): *Let* $\zeta' : \mathbf{A} \to \mathbf{A}'$ *be an eBH and* $\mathbf{X} \subseteq \mathbf{A}$, *then* $\zeta'(\mathbf{X}) = [\zeta'(\mathbf{X})]$.

Proof

Just check the following:

(a) $\zeta'(\mathbf{ext}(\mathbf{X})) = \mathbf{ext}(\zeta'\mathbf{X})$,

(b) for all $j \geq 0$, $\zeta'(\mathbf{ext}^j(\mathbf{X})) = \mathbf{ext}^j(\zeta'\mathbf{X})$,

(c) $\zeta'(\bigcup_{j\in Nat} \mathbf{ext}^j(\mathbf{X})) = \bigcup_{j\in Nat} \mathbf{ext}^j(\zeta'\mathbf{X})$. □

The following is to provide the uniqueness between free eBAs.

Theorem 3.6.4 (Isomorphism between free eBAs): *Let* $\mathbf{A}$, $\mathbf{A}'$ *have the universal mapping property for* $\mathcal{K}$ *over* $\mathbf{X}$ *and* $\mathbf{X}'$ *respectively, and* $|\mathbf{X}| = |\mathbf{X}'|$. *Then* $\mathbf{A} \cong \mathbf{A}'$, *where* $\mathbf{A}, \mathbf{A}' \in \mathcal{K}$ *and* $\cong$ *means isomorphic and* $|\mathbf{X}| = |\mathbf{X}'|$ *is short for both* $|X| = |X'|$ *and* $|X_k| = |X'_k|$ *for* $k \in Nat$.

The proof is simple and left out.

Informally, Theorem 3.6.4 says that "freeness" is unique, in the sense that if two eBAs have the universal mapping property over a same class of eBAs and if their bases share a same cardinality, then these two eBAs are isomorphic.

Theorem 3.6.5 (freeness of term eBA): *Let* V *be a set of ordinary variables with* $|V| = \aleph_0$, *and* FV *be a family of* FV_k *function variables such that* $|FV_k| = \aleph_0$ *with arity* $k \in Nat$. *Then, term eBA* $\mathbf{T}$ *has the universal property for the class of all eBAs over* $< V, FV >$.

Proof. by *Theorem 3.6.2* and *Theorem 3.5.9.* □

So, term eBA $\mathbf{T}$ is a free eBA.

3.7 Binding Congruences and quotient eBAs

For first order algebras, a congruence is an equivalence preserving functionality (or compositionality). However, an extensional Binding Congruence (eBC) can not be so simple. We have to consider some primitive functions such as constants, projections, and compositions of functions, i.e. certain cares are need to deal with them. Also, since function spaces are parts of carriers, Extensionality has to be considered as well. Formally,

Definition 3.7.1 (eBCs): *Let ϑ be a family of equivalence relations on an eBA* **A**. *ϑ is said to be an extensional Binding Congruence (eBC) on* **A** *if ϑ satisfies the followings:*

(eBC-ext) $g(\vec{a})\vartheta h(\vec{a})$ *for every* $\vec{a} \in A^m$ *iff* $g\vartheta h$, *where* $g, h \in \mathcal{F}_m$ $(m \in Nat)$;

(eBC-comp-1) $a_j \vartheta b_j$ *for each* j *implies* $g(\vec{a})\vartheta g(\vec{b})$, *where* $g \in \mathcal{F}_m$;

(eBC-comp-2) $g_i \vartheta h_i$ *(for every* i*) and* $a_j \vartheta b_j$ *(for each* j*) implies* $\sigma^{\mathbf{A}}(\vec{g}, \vec{a})\vartheta\sigma^{\mathbf{A}}(\vec{h}, \vec{b})$,

where $\sigma \in \Sigma_{<\vec{m},n>}$ *with* $|\vec{m}| = \ell$, $g_i, h_i \in \mathcal{F}_{m_i}$ $(1 \leq i \leq \ell)$ *and* $a_j, b_j \in A$ $(1 \leq j \leq n)$.

Considering the analogy between eBAs and first order algebras, a similar analogy between eBCs and first order congruences is as follows. An eBC is a generalization of a congruence, since it take function spaces into account. However, an eBC is not as general as a congruence since it is not as arbitrary as a first order congruence seems to be, and it commits itself on some primitive functions.

The well-definedness of eBCs is not obvious, and we examine it below. In other words, we are to show that an eBC preserves function compositions and uniformity.

Lemma 3.7.2 (eBCs, compositions and uniformity):

(a) $g\vartheta h$ *and* $g_i\vartheta h_i$ *(for each* i*) implies* $g(\vec{g})\vartheta h(\vec{h})$,

where $g, h \in \mathcal{F}_m$ *and* $g_i, h_i \in \mathcal{F}_k$ *for* $m \in Nat$, $1 \leq i \leq m$ *and* $k \in Nat$;

(b) $g_i^1 \vartheta g_i^2$ *(for each* i*) and* $h_j^1 \vartheta h_j^2$ *(for every* j*) implies* $\sigma^{\mathbf{A}} \odot < \vec{g^1}, \vec{h^1} > \vartheta^j \sigma^{\mathbf{A}} \odot < \vec{g^2}, \vec{h^2} >$ *for* $\sigma \in \Sigma_{<\vec{m},n>}$ *with* $|\vec{m}| = \ell$*, and* $k \in Nat$*,*

where $g_i^{'j} = curry_{k,m_i}(g_i^j)$ *and* $g_i^j \in \mathcal{F}_{k+m_i}$ *(*$j = 1,2$*) for* $1 \leq i \leq \ell$*, and* $h_j^i \in \mathcal{F}_k$ *(*$i = 1,2$*) for* $1 \leq j \leq n$*.*

Proof

For (a), it is by (eBC-ext).

For (b), we know that

(b.1) $C_{m,a} \vartheta C_{m,b}$ if $a \vartheta b$,

(b.2) $\pi_{m,i} \vartheta \pi_{m,i}$ and

(b.3) $curry_{k,m}(g)(\vec{a}) = g \odot < C_{m,a_1}, C_{m,a_2}, ..., C_{m,a_k}, \Pi_{1,m}^m >$ for $\vec{a} \in A^k$.

So, by (eBC-ext), we have (b). □

Similar to quotient algebras in first order case, we turn to quotient eBAs after introducing eBCs (Definition 3.7.7). We proceed as follows.

Definition 3.7.3 (quotient family): *Let* ϑ *be an eBC on an eBA* **A***. Then,* $a/\vartheta =_{df} \{b \in A | a \vartheta b\}$ *for* $a \in A$*, and* $g/\vartheta =_{df} \{h \in \mathcal{F}_m | g \vartheta h\}$ *for* $g \in \mathcal{F}_m$*.*

It can be easily checked that the $\bullet/\vartheta$ in Definition 3.7.3 is well-defined, i.e. the values do not depend on their representative. Further, let ϑ be an eBC on an eBA **A**. Then, we define the followings:

(i) for $m \in Nat$, $g \in \mathcal{F}_m$ and $a_j \in A$, $\bullet_{(g/\vartheta)}(\vec{a}/\vartheta) =_{df} g(\vec{a})/\vartheta$ (later, we will use g/ϑ to refer $\bullet_{(g/\vartheta)}$ for simplicity);

(ii) for $m \in Nat$, $1 \leq i \leq m$, and $k \in Nat$, given $g \in \mathcal{F}_m$ and $h_i \in \mathcal{F}_k$, $(g/\vartheta) \odot < \vec{h}/\vartheta > =_{df} (g \odot < \vec{h} >)/\vartheta$;

(iii) for $\sigma \in \Sigma_{<\vec{m},n>}$ with $|\vec{m}| = \ell$, $g_i \in \mathcal{F}_{k+m_i}$ $(1 \leq i \leq \ell)$ and $h_j \in \mathcal{F}_k$ $(1 \leq j \leq n)$, $(\sigma^{\mathbf{A}}/\vartheta) \odot < curry_{k,\vec{m}}(\vec{g}/\vartheta), \vec{h}/\vartheta > =_{df} (\sigma^{\mathbf{A}} \odot < curry_{k,\vec{m}}(\vec{g}), \vec{h} >)/\vartheta$. It is also easily verified that the above definitions of $\bullet/\vartheta$ are well-defined, i.e. their values do not depend on their representatives. Therefore, we can define $\mathcal{F}/\vartheta =_{df}$

$< A/\vartheta, \mathcal{F}_m/\vartheta_{m\in Nat} >$ where $\mathcal{F}_m/\vartheta = \{g/\vartheta | g \in \mathcal{F}_m\}$ for $m \in Nat$. Below, we will show that $\mathcal{F}/\vartheta$ has the "explicitly closedness".

Lemma 3.7.4 (explicit closedness of $\mathcal{F}/\vartheta$): *$\mathcal{F}/\vartheta$ is explicitly-closed.*

Proof

(a) For each $a/\vartheta \in A/\vartheta$ and $k \in Nat$, we have: $(C_{k,a}/\vartheta)(\vec{a}/\vartheta) = C_{k,a}(\vec{a})/\vartheta = a/\vartheta$ for every $a_j/\vartheta \in A/\vartheta$ $(1 \leq j \leq k)$. So, $C_{k,(a/\vartheta)} = C_{k,a}/\vartheta \in \mathcal{F}_k/\vartheta$.

(b) For each $k \in Nat$ and $1 \leq i \leq k$, we have: $(\pi_{k,i}/\vartheta)(\vec{a}/\vartheta) = \pi_{k,i}(\vec{a})/\vartheta = a_i/\vartheta$ for every $a_j/\vartheta \in A/\vartheta$ $(1 \leq j \leq k)$. So, $\pi_{k,i}/\vartheta \in \mathcal{F}_k/\vartheta$.

(c) The rest proof is obvious by the definitions of $\bullet/\vartheta$. □

Next, we concern of uniformity of interpretations of BOs.

Lemma 3.7.5 (uniformity of $\mathcal{F}/\vartheta$): *For each $\sigma \in \Sigma_{<\vec{m},n>}$, $\sigma^{\mathbf{A}}/\vartheta$ is uniform over $\mathcal{F}/\vartheta$.*

Proof Obvious by the definitions of $\bullet/\vartheta$. □

Since $\bullet/\vartheta$ preserves the explicit-closedness and the uniformity, we arrive at that $\mathbf{A}/\vartheta$ is an eBA. That is,

Theorem 3.7.6 ($\mathcal{F}/\vartheta$ and eBA): *Let $\mathbf{A}$ be an eBA and ϑ be an eBC on it. Then, $< A/\vartheta, \mathcal{F}/\vartheta >$ is an eBA.*

Finally, we can define a quotient eBA from an eBA and its eBC. Comparing with the first order quotient algebras, the existence of "second order" quotient eBAs are not apparent as we expected.

Definition 3.7.7 (quotient eBA): *$\mathbf{A}/\vartheta$ is said to be the quotient eBA of an eBA $\mathbf{A}$ by an eBC ϑ iff $\mathbf{A}/\vartheta =< A/\vartheta, \mathcal{F}/\vartheta >$ and $\mathcal{A}_\sigma^{(\mathbf{A}/\vartheta)} = \mathcal{A}_\sigma/\vartheta$.*

Recall the comment after Definition 3.7.3, the well-definedness of $\bullet/\vartheta$ can also be viewed as that for each environment $< \rho, \varphi >$ if $< \mathcal{A}[\![p]\!]_{<\rho,\varphi>}, \mathcal{A}[\![q]\!]_{<\rho,\varphi>} > \in \vartheta$ then $(\mathcal{A}/\vartheta)[\![p]\!]_{<\rho/\vartheta,\varphi/\vartheta>} = (\mathcal{A}/\vartheta)[\![q]\!]_{<\rho/\vartheta,\varphi/\vartheta>}$. In other words, this means that quotient algebras preserve satisfaction.

Let $\mathbf{A}$ be an eBA and ϑ be an eBC on $\mathbf{A}$. The natural map $\nu_\vartheta : \mathbf{A} \to \mathbf{A}/\vartheta$ is defined by (a) $\nu_\vartheta(a) =_{df} a/\vartheta$, (b) $\nu_\vartheta(g) =_{df} g/\vartheta$ and (c) $\nu_\vartheta(\sigma^{\mathbf{A}}) =_{df} \sigma^{\mathbf{A}/\vartheta}$. We are to demonstrate that the natural map is an eBH. Formally

Lemma 3.7.8 (preservations of ν_ϑ):

(ν-cons) $\nu_\vartheta(C_{k,a}) = C_{k,\nu_\vartheta(a)}$;

(ν-proj) $\nu_\vartheta(\pi^{\mathbf{A}}_{k,i}) = \pi^{(\mathbf{A}/\vartheta)}_{k,i}$ *for* $1 \leq i \leq k$;

(ν-cmp) $\nu_\vartheta(g\odot < \vec{h} >) = \nu_\vartheta(g)\odot < \nu_\vartheta(\vec{h}) >$;

(ν-unif) $\nu_\vartheta(\sigma^{\mathbf{A}}\odot < \vec{g'}, \vec{h} >) = \sigma^{(\mathbf{A}/\vartheta)}\odot < \vec{g''}, \nu_\vartheta(\vec{h}) >$

where $\sigma \in \Sigma_{<\vec{m},n>}$ *with* $|\vec{m}| = \ell$, $g_i \in \mathcal{F}_{k+m_i}$, $g'_i = curry_{k,m_i}(g_i)$ *and* $g''_i = curry_{k,m_i}(\nu_\vartheta(g_i))$ $(1 \leq i \leq \ell)$, *and* $h_j \in \mathcal{F}_k$ $(1 \leq j \leq n)$.

Proof By Extensionality. □

Therefore,

Theorem 3.7.9 (ν_ϑ and eBH): *Let ϑ be an eBC on an eBA $\mathbf{A}$. Then, the natural map ν_ϑ from $\mathbf{A}$ to $\mathbf{A}/\vartheta$ is an onto (surjective) eBH.*

The proof of this theorem is obvious from Lemma 3.7.8 and is left out.

Later, the natural map ν_ϑ is referred to as the *natural* eBH associated with the eBC ϑ. Analogous to first order algebras, we define an eBC over a quotient eBA as follows. Suppose ϑ, ϑ' are eBCs on $\mathbf{A}$ and $\vec{\vartheta} \subseteq \vec{\vartheta'}$, i.e. $\vartheta \subseteq \vartheta'$ and $\vartheta_m \subseteq \vartheta'_m$ for $m \in Nat$. Later, we will simply refer to them as $\vartheta \subseteq \vartheta'$. Then, we define a double eBC as the following. Let $\vartheta'/\vartheta =_{df} \{< g/\vartheta, h/\vartheta > \mid < g, h > \in \vartheta'\}$. We have two results, Lemma 3.7.10 and Theorem 3.7.11.

Lemma 3.7.10 (double eBC): *If ϑ, ϑ' are eBCs on $\mathbf{A}$ and $\vartheta \subseteq \vartheta'$, then ϑ'/ϑ is an eBC on $\mathbf{A}/\vartheta$.*

The proof is an straightforward verification, and is left out. Informally, this lemma says that a double eBC ϑ'/ϑ is an eBC on quotient eBA $\mathbf{A}/\vartheta$. Further, from

a double eBC we can obtain a double quotient eBA. The relationship among these eBA, quotient eBA and double quotient eBA is as follows.

Theorem 3.7.11 (double quotient eBA): *If ϑ', ϑ are eBCs on $\mathbf{A}$ and $\vartheta \subseteq \vartheta'$, then the map $\zeta' : (\mathbf{A}/\vartheta)/(\vartheta'/\vartheta) \to \mathbf{A}/\vartheta'$, defined by $\zeta'((a/\vartheta)/(\vartheta'/\vartheta)) =_{df} a/\vartheta'$ and $\zeta'((g/\vartheta)/(\vartheta'/\vartheta)) =_{df} g/\vartheta'$, is an extensional binding isomorphism (i.e. a bijective eBH).*

The proof is also an easy verification and is left out.

Considering quotient eBAs, we present a result related to generated eBAs and their quotient eBAs below. That is,

Theorem 3.7.12 (generated eBA and its quotient eBA): *Let $\mathbf{A}$ be an eBA generated by $\mathbf{X}$. Thus, if ϑ is an eBC on $\mathbf{A}$, then its quotient eBA $\mathbf{A}/\vartheta$ can be generated by $[\mathbf{X}/\vartheta]$.*

Proof

By Theorem 3.7.9, we know that ν_ϑ is an eBH. Hence, by Lemma 3.6.3 we have that $\mathbf{A}/\vartheta$ is generated by $[\mathbf{X}/\vartheta]$. □

From first order algebras, we understand that the purpose of introducing congruences and their quotient algebras is to provide a general framework of working on kernels of homomorphisms. This is a crucial step toward the success of Birkhoff's approach. We intend to follow this in what follows.

A first order congruence plays two roles. They are

(a) an equivalence relation preserving compositionality, and

(b) yielding an quotient algebra.

However, these two roles are tied to each other in first order case. The situation is different for eBAs, i.e. these two roles are separated. Corresponding to (a), there is a concept of the *binding core* of an eBH; and the *binding kernel* of an eBH corresponds to (b). In other words, the binding core of an eBH is a technical extension of kernels of first order homomorphisms. Informally, a pair of elements are in the binding core of an eBH iff they share a same image under the eBH. Formally,

Definition 3.7.13 (binding core $\vec{\nabla}_\zeta$): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$. Then, the binding core of ζ (the core of ζ for short), written as $\vec{\nabla}_\zeta$, is a pair $< \nabla, \{\nabla_k | k \in Nat\} >$ where ∇ is a relation on ordinary objects of $\mathbf{A}$ and ∇_k is a relation on functions of $\mathbf{A}$ with arity $k \in Nat$ such that $\nabla =_{df} \{< a, b >\in A^2 | \zeta(a) = \zeta(b)\}$ and $\nabla_k =_{df} \{< g, h >\in \mathcal{F}_k^2 | \zeta(g) = \zeta(h)\}$.*

However, the above basic requirement on a relation obtained from an eBH is not enough to be the binding kernel of the eBH but the core of the eBH. The reason for this is Extensionality of eBCs. We introduce kernels of eBHs as follows.

Definition 3.7.14 (binding kernel of an eBH): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$. The binding kernel of ζ (or simply kernel of ζ), written as $Ker(\zeta)$, is defined as the least pair $< R, \{R_k | k \in Nat\} >$ on $\mathbf{A}$ such that the followings hold:*

(a) R is an equivalence relation on ordinary object;

(b) R_k is an equivalence relation on functions with arity k;

(c) $\vec{\nabla}_\zeta \subseteq \vec{R}$, i.e. $\nabla \subseteq R$ and $\nabla_k \subseteq R_k$ with $k \in Nat$; and

(c) $\mathbf{ker}(\vec{R}) \subseteq \vec{R}$, where $\mathbf{ker}$ is defined as follows,

$$ker(\vec{R}') = R' \cup \{< g(\vec{a}), h(\vec{b}) > | < g, h >\in R'_k \text{ and } < a_j, b_j >\in R'\}$$
$$\cup\{< \sigma^{\mathbf{A}}(\vec{g}, \vec{a}), \sigma^{\mathbf{A}}(\vec{h}, \vec{b}) > | < g_i, h_i >\in R'_{m_i} \text{ and } < a_j, b_j >\in R'\}$$

and

$$ker_k(\vec{R}') = R'_k \cup \{< g, h >\in \mathcal{F}_k^2 | < g(\vec{a}), h(\vec{b}) >\in R' \text{ for every } < a_j, b_j >\in R'\}$$

for any relation $\vec{R}'$ on $\mathbf{A}$.

To see the difference between $\vec{\nabla}_\zeta$ and $Ker(\zeta)$, we have following two facts:

(1) $g\vec{\nabla}_\zeta g'$ implies $g(\vec{a})\vec{\nabla}_\zeta g'(\vec{a})$ for every $\vec{a} \in A^m$, and

(2) $gKer(\zeta)g'$ iff $g(\vec{a})Ker(\zeta)g('(\vec{a})$ for every $\vec{a} \in A^m$;

where $\mathbf{A}$ and $\mathbf{A}'$ are eBAs, $\zeta : \mathbf{A} \to \mathbf{A}'$ and $g, g' \in \mathcal{F}_m$.

For concrete examples, let $\Sigma^{BO}_{<\vec{m},n>} = \emptyset$ when $\vec{m} \neq \varepsilon$ (i.e. essentially a first order signature), you can take $\mathbf{A} =< \mathcal{F}, \mathcal{A} >$ be an eBA with the signature such that $\mathcal{F} =< A, \{\mathcal{F}_m | m \in Nat\} >$ where $A = \{a, b, c\}$ and $\mathcal{F}_m$ is some part of full function

space from A^m to A closed under constant functions, projections and compositions with arity $m \geq 0$; and let $\mathbf{A}'$ be another identical eBA with $\mathbf{A}$ and eBH ζ is a family of function from A to A' and functionals from $\mathcal{F}_m$ to $\mathcal{F}'_m$ for $m \in Nat$. More specifically,

(i) the "object" function of ζ maps both a and c to a, and it maps b to b:

$$\zeta(\bullet) = \begin{cases} a & \text{if } \bullet = a \\ b & \text{if } \bullet = b \\ a & \text{if } \bullet = c. \end{cases}$$

and

(ii) the "function" functionals of ζ from $\mathcal{F}_m$ to $\mathcal{F}'_m$ are almost identity functionals between these two functions spaces except that functions like constant function $C_{m,c}$ whose image is constant function $C_{m,a}$ not $C_{m,c}$ for $m \in Nat$.

Since the number of functions in the full function space of $A^m \to A$ is 3^{3^m} with $m \in Nat$, i.e. 27 if $m = 1$ (see Table 3-1), we only give the details of ∇_ζ and Ker_ζ for the case of $m = 1$. Note that both $\mathcal{F}_1(A)$ and $\mathcal{F}'_1(A')$ have, at most, 15 functions in total, i.e. they are

$$\{id, f_1, f_2, f_6, f_7, C_{1,c}, C_{1,b}, g_4, g_5, h_0, C_{1,a}, h_2, h_6, h_7, h_8\},$$

in order to preserving compositionality of ζ; e.g. otherwise there is a contradition of ζ being an eBH:

$$\zeta(g_0)(a) = \zeta(g_0)(\zeta(a)) = \zeta(g_0(a)) = \zeta(a) = a$$
$$\zeta(g_0)(a) = \zeta(g_0)(\zeta(c)) = \zeta(g_0(c)) = \zeta(b) = b.$$

On the other hand, this can also be regarded as a justification of working on explicitly closed family of functions in eBAs instead of full function spaces. Also, this situation will not appear if eBA $\mathbf{A}$ is the term eBA $\mathbf{T}$ regardless what eBA $\mathbf{A}'$ is (why?).

Since ζ is the "identity" identity on $\mathcal{F}_1 \times \mathcal{F}'_m$ (where $\mathcal{F}_m = \mathcal{F}'_m$), ∇_1 is the "identity" relation on $\mathcal{F}_1$ but Ker_1 properly contains ∇_1 with extra

$$\left\{ \begin{array}{llll} < C_{1,a}, f_1 >, & < C_{1,a}, f_2 >, & < C_{1,a}, f_7 >, & < id, f_6 >, \quad < id, h_0 > \\ < C_{1,a}, h_2 >, & < id, h_6 >, \quad < C_{1,a}, h_7 >, & < C_{1,a}, h_8 >, & < g_4, g_5 > \end{array} \right\}$$

$$id(x) = \begin{cases} a & \text{if } x = a \\ b & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_0(y) = \begin{cases} a & \text{if } y = a \\ b & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_0(z) = \begin{cases} a & \text{if } z = a \\ b & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_1(x) = \begin{cases} a & \text{if } x = a \\ a & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_1(y) = \begin{cases} a & \text{if } y = a \\ a & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad C_{1,a}(z) = \begin{cases} a & \text{if } z = a \\ a & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_2(x) = \begin{cases} a & \text{if } x = a \\ c & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_2(y) = \begin{cases} a & \text{if } y = a \\ c & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_2(z) = \begin{cases} a & \text{if } z = a \\ c & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_3(x) = \begin{cases} b & \text{if } x = a \\ b & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad C_{1,b}(y) = \begin{cases} b & \text{if } y = a \\ b & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_3(z) = \begin{cases} b & \text{if } z = a \\ b & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_4(x) = \begin{cases} b & \text{if } x = a \\ a & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_4(y) = \begin{cases} b & \text{if } y = a \\ a & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_4(z) = \begin{cases} b & \text{if } z = a \\ a & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_5(x) = \begin{cases} b & \text{if } x = a \\ c & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_5(y) = \begin{cases} b & \text{if } y = a \\ c & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_5(z) = \begin{cases} b & \text{if } z = a \\ c & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_6(x) = \begin{cases} c & \text{if } x = a \\ b & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_6(y) = \begin{cases} c & \text{if } y = a \\ b & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_6(z) = \begin{cases} c & \text{if } z = a \\ b & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$f_7(x) = \begin{cases} c & \text{if } x = a \\ a & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_7(y) = \begin{cases} c & \text{if } y = a \\ a & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_7(z) = \begin{cases} c & \text{if } z = a \\ a & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

$$C_{1,c}(x) = \begin{cases} c & \text{if } x = a \\ c & \text{if } x = b \\ c & \text{if } x = c \end{cases} \quad g_8(y) = \begin{cases} c & \text{if } y = a \\ c & \text{if } y = b \\ b & \text{if } y = c \end{cases} \quad h_8(z) = \begin{cases} c & \text{if } z = a \\ c & \text{if } z = b \\ a & \text{if } z = c \end{cases}$$

Table 3–1: full function space of $\{a, b, c\} \to \{a, b, c\}$

and their reflexive and tranistive closure. Interested readers are encouraged to work the rest and compare the difference between binding core $\vec{\nabla}_\zeta$ and binding kernel $Ker(\zeta)$.

Furthermore, keeping our previous analogy between eBAs and first order algebras, we know that the binding core and the binding kernel of an eBH becomes almost the same if the signature Σ^{BO} has the property of $\Sigma^{BO}_{<\vec{m},n>} = \emptyset$ for every $\vec{m} \neq \varepsilon$. The difference is on the carriers of functions spaces. If we disregard of this, then the binding core and the binding kernel are the same.

Anyway, a first order kernel is a first order congruence and yields a first order quotient algebra. In contrast, a (binding) core of an eBH is not an eBC, but a (binding) kernal of the eBH is and it yields a quotient eBA (Theorem 3.7.16) The next lemma is to justify this point of view. However, let us verify that Definition 3.7.14 yields an eBC (Theorem 3.7.15).

Theorem 3.7.15 (kernel and eBC): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$, Then, $Ker(\zeta)$ is an eBC on* $\mathbf{A}$.

Proof Obvious, just check their definitions. □

Theorem 3.7.16 (kernel and its quotient eBA): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$ be an onto eBH. Then, there is an isomorphic eBH $\hat{\zeta} : \mathbf{A}/Ker(\zeta) \to \mathbf{A}'$ such that $\zeta = \hat{\zeta} \odot \nu_\zeta$ where $\nu_\zeta =_{df} \nu_{Ker(\zeta)}$. I.e. the following diagram commutes (see Figure 3-1):*

Proof The key point of the proof is the fact of coincidence between the core ∇_ζ and the kernel $Ker(\zeta)$. This coincidence is because of that ζ is an onto eBH. Others are just routine checks. □

Now, we come to the key issue of Birkhoff's approach, i.e. whether the *"onto"* condition on ζ can be withdrawn from Theorem 3.7.16. To seek a general answer to this question, we are led to Admissibility, which is the subject of next section.

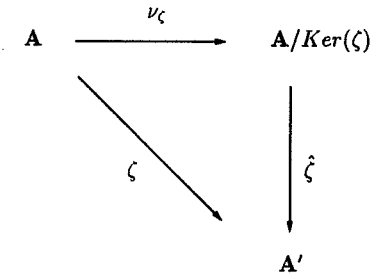

Figure 3–1: commutativity

3.8 Admissible freeness

Referring to Theorem 3.7.16, can we weaken the condition "*onto*" and still has the diagram commutes? To answer this question, we introduce a concept of "*Admissibility*". The essence of Admissibility is Extensionality.

Definition 3.8.1 (admissible eBH): *Let* $\zeta : \mathbf{A} \to \mathbf{A}'$. *Thus,* ζ *is admissible iff* $\zeta(\mathbf{A}) \preceq \mathbf{A}'$.

In other words, $\zeta : \mathbf{A} \to \mathbf{A}'$ is admissible, sometimes written as $\zeta : \mathbf{A} \dot{\to} \mathbf{A}'$ (the dot $\cdot$ on top of $\to$ shows the Admissibility) iff ζ preserves Extensionality in its image. The following is the introduction of "*admissible freeness*".

Definition 3.8.2 (admissibly free eBA): *Let* $\mathcal{K}$ *be a class of eBAs, and* $\mathbf{U}$ *be an eBA generated by* $\mathbf{X}$. *Thus,*

(a) a family of function from X *to* A' *and functionals from* X_k *to* $\mathcal{F}'_k$ *for* $k \in Nat$ *is said to be* admissible *if* $[\zeta(\mathbf{X})] \preceq \mathbf{A}'$, *where* $\mathbf{A}'$ *is an eBA;*

(b) for every $\mathbf{A}' \in \mathcal{K}$ *and if for every admissible eBH* $\zeta : \mathbf{X} \to \mathbf{A}'$, *there is an* $\zeta' : \mathbf{U} \to \mathbf{A}'$ *which extends of* ζ, *then we say that* $\mathbf{U}$ *has the* admissible universal mapping property *for* $\mathcal{K}$ *over* $\mathbf{X}$. $\mathbf{X}$ *is said to be* admissible free generators *of* $\mathbf{U}$ *and* $\mathbf{U}$ *is said to be an* admissible-freely generated *by* $\mathbf{X}$.

By Theorem 3.3.9, for an admissibly-free eBA, we naturally have the following: suppose $\mathbf{U}$ has the admissible universal mapping property for $\mathcal{K}$ over $\mathbf{X}$, thus, if given $\mathbf{A}' \in \mathcal{K}$ and an admissible map $\zeta : \mathbf{X} \to \mathbf{A}'$, then there exists an unique extension ζ' of ζ such that ζ' is an eBH and $\zeta' : \mathbf{U} \to \mathbf{A}'$.

Considering different admissibly-free eBAs, we have the following.

Theorem 3.8.3 (isomorphism between admissibly-free eBAs): *Let* $\mathbf{U}$ *and* $\mathbf{U}'$ *have the admissible universal mapping property for* $\mathcal{K}$ *over* $\mathbf{X}$ *and* $\mathbf{X}'$ *respectively, and* $|\mathbf{X}| = |\mathbf{X}'|$ *(i.e.* $|X| = |X'|$ *and* $|X_k| = |X'_k|$ *for* $k \in Nat$*), then* $\mathbf{U} \cong \mathbf{U}'$ *where* $\mathbf{U}, \mathbf{U}' \in \mathcal{K}$.

The proof is a simple verification and is left out.

Admissibility makes the difference between the binding core and the binding kernel of an eBH disappear. In other words, it means that "Admissibility" eliminate the difference between binding kernels of eBHs and kernels of first order homomorphism, see next Lemma 3.8.4.

Lemma 3.8.4 (coincidence of kernels and cores of admissible eBHs): *Let* $\zeta : \mathbf{A} \to \mathbf{A}'$ *be an admissible eBH, then* $Ker(\zeta) = \nabla_\zeta$.

The proof of this lemma is easy (hint: $\vec{\nabla}_\zeta$ is an fixpoint of **ker** in Definition 3.7.14 under the admissible condition) and is left out.

Examples of eBHs which have coincident kernels and cores are either

(i) eBHs with surjective images on base level, or

(ii) eBHs with 1-1 mapping on base level.

Apparently, they are admissible.

Considering the kernel of an admissible eBH and its quotient eBA, we have the following.

Theorem 3.8.5 (kernel of an admissible eBH and its quotient eBA): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$ be an admissible eBH. Then there is an 1-1 eBH $\hat{\zeta} : \mathbf{A}/Ker(\zeta) \to \mathbf{A}'$ such that $\zeta = \hat{\zeta} \odot \nu_\zeta$ (see Figure 3-1).*

Proof We define $\zeta(g/Ker(\zeta)) =_{df} \zeta(g)$. By Lemma 3.8.4, it is well-defined and is an 1-1 eBH. □

The importance of Admissibility is established by a few subsequent results, which are summarized as: the result of Theorem 3.8.6 is the best we can have in Birkhoff's approach. Recall that the essence of Birkhoff's method is the equivalences among (1) satisfaction of algebras, (2) kernels of the algebras and (3) satisfaction of the corresponding quotient term algebras. We proceed this as follows.

Lemma 3.8.6 (injectiveness of eBH $\hat{\zeta}$): *Let ζ be an eBH and $\zeta : \mathbf{A} \to \mathbf{A}'$, and $\hat{\zeta}$ be another eBH and $\hat{\zeta} : \mathbf{A}/Ker(\zeta) \to \mathbf{A}'$ such that $\zeta = \hat{\zeta} \odot \nu_\zeta$, then $\hat{\zeta}$ must be 1-1 (see Figure 3-1).*

Proof Let $g'/Ker(\zeta), h'/Ker(\zeta) \in \mathbf{A}/Ker(\zeta)$ and $\hat{\zeta}(g'/Ker(\zeta)) = \hat{\zeta}(h'/Ker(\zeta))$. Since ν_ζ is onto, there exists $g, h \in \mathbf{A}$ such that $\nu_\zeta(g) = g'/Ker(\zeta)$ and $\nu_\zeta(h) = h'/Ker(\zeta)$. Therefore, for every such a pair g, h, we have $\hat{\zeta} \odot \nu_\zeta(g) = \hat{\zeta} \odot \nu_\zeta(h)$, i.e. $\zeta(g) = \zeta(h)$. So, $< g, h > \in \nabla_\zeta \subseteq Ker(\zeta)$, i.e. $g'/Ker(\zeta) = h'/Ker(\zeta)$. □

See Figure 3-1, this lemma said that if the diagram commutes then the eBH starting from the quotient eBA must be an injective. Consequently, the original eBH which we start with, must be admissible, see Corollary 3.8.7.

Corollary 3.8.7 (Admissibility and commutativity): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$ and $\hat{\zeta} : \mathbf{A}/Ker(\zeta) \to \mathbf{A}'$ such that $\zeta = \hat{\zeta} \odot \nu_\zeta$, then ζ is admissible (see Figure 3-1).*

Proof By Lemma 3.8.6, we know the image of $\hat{\zeta}$ is a perfect sub-eBA, so is the image of ζ. □

From previous results (Theorem 3.8.5 and Corollary 3.8.7), we understand that Admissibility is a necessary and sufficient condition for the diagram (Figure 3-1) to commute. Because of this, we claim that Admissible Completeness (Corollary 3.10.11) is the best result for Birkhoff's approach.

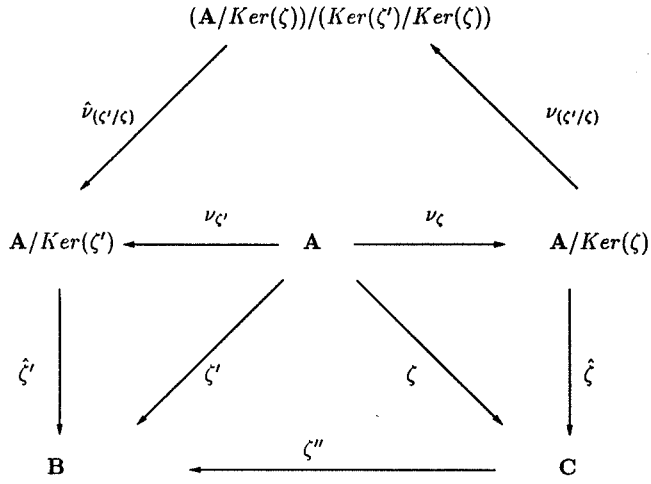

Figure 3–2: quotient eBA and double quotient eBA

Theorem 3.8.8 (Admissibility and double quotient eBA): *Let ζ and ζ' be eBHs and $\zeta' : \mathbf{A} \to \mathbf{A}'$, $\zeta : \mathbf{A} \to \mathbf{A}''$ and $Ker(\zeta) \subseteq Ker(\zeta')$ such that ζ' is admissible and ζ is onto, then there exists $\zeta'' : \mathbf{A}'' \to \mathbf{A}'$ such that $\zeta' = \zeta'' \odot \zeta$.*

Proof See Figure 3-2, and we define $\zeta'' =_{df} \hat{\zeta}' \odot \hat{\nu}_{(\zeta'/\zeta)} \odot \nu_{(\zeta'/\zeta)} \odot \hat{\zeta}^{-1}$, where $\hat{\zeta}'$ is an 1-1 eBH by Theorem 3.8.5. $\hat{\zeta}^{-1}$ is an isomorphic eBH by Theorem 3.7.16. $\hat{\nu}_{(\zeta'/\zeta)}$ is an isomorphic eBH by Lemma 3.7.11. □

The following two lemmas provide the fact that eBCs are closed under intersections.

Lemma 3.8.9 (intersection of two eBCs): *Let ϑ', ϑ be eBCs on an eBA $\mathbf{A}$. Then $\vartheta' \cap \vartheta$ is an eBC on $\mathbf{A}$.*

Consequently,

Lemma 3.8.10 (intersection of eBCs): *Let Ψ be a class of eBCs on $\mathbf{A}$. Then $\cap\Psi$ is an eBC on $\mathbf{A}$.*

As a result of Theorem 3.8.5 and Corollary 3.8.7, we define admissible eBCs with respect to an eBA and a collection of eBAs as follows.

Definition 3.8.11 (admissible eBC $\dot{\vartheta}$): *Let* $\mathbf{A}$ *be an eBA and* $\mathcal{K}$ *be a class of eBAs. Thus,*

(a) the admissible eBC $\dot{\vartheta}_{\mathbf{A}}$ *on* $\mathbf{T}$ *with respect to* $\mathbf{A}$ *is defined to be* $\bigcap_{\zeta:\mathbf{T}\dashrightarrow\mathbf{A}} Ker(\zeta)$;

(b) the admissible BC $\dot{\vartheta}_{\mathcal{K}}$ *on* $\mathbf{T}$ *with respect to* $\mathcal{K}$ *is defined to be* $\bigcap_{\mathbf{A}\in\mathcal{K}} \dot{\vartheta}_{\mathbf{A}}$.

Next, we give two conditions under which the composition of two admissible eBHs is admissible (Lemma 3.8.12); or more precisely under which Admissibility is preserved by a composition of two admissible eBHs.

Lemma 3.8.12 (Admissibility under compositions): *Let* $\zeta' : \mathbf{A} \to \mathbf{A}'$ *be admissible (or onto) and* $\zeta : \mathbf{A}' \to \mathbf{A}''$ *be 1-1 (or admissible), then* $\zeta \odot \zeta' : \mathbf{A} \to \mathbf{A}''$ *is admissible.*

The proof is obvious and is left out.

Theorem 3.8.13 (admissibly free quotient term eBA): *Let* $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}})$ *be* $\mathbf{T}(\mathbf{X})/\dot{\vartheta}_{\mathcal{K}}$ *where* $\dot{\mathbf{X}} = \mathbf{X}/\dot{\vartheta}_{\mathcal{K}}$, *then it has the admissible universal mapping property for* $\mathcal{K}$ *over* $\dot{\mathbf{X}}$.

Proof

(1) By Theorem 3.7.12 and Lemma 3.6.3, $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}})$ is generated by $\dot{\mathbf{X}}$.

(2) Let ζ be admissible as in Figure 3-3.

(2.a) By Theorem 3.6.3, $\zeta \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}\lceil_{\mathbf{X}}$ can be uniquely extended to an eBH $\zeta' : \mathbf{T} \to \mathbf{A}$.

(2.b) The Admissibility of ζ is preserved by such an extension ζ' because ζ' is admissible by a similar proof of Lemma 3.8.12 where ζ is admissible and $\nu_{\dot{\vartheta}_{\mathcal{K}}}$ is onto.

(2.c) Hence, by Theorem 3.8.5 and by Theorem 3.8.8, there exists $\hat{\zeta}' : \dot{\mathbf{T}}_{\mathcal{K}}(\dot{X}) \to \mathbf{A}$ such that $\zeta' = \hat{\zeta}' \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}$ where $\hat{\zeta}'\lceil_{\dot{\mathbf{X}}} = \zeta$, since $Ker(\nu_{\dot{\vartheta}_{\mathcal{K}}}) = \dot{\vartheta}_{\mathcal{K}} \subseteq Ker(\zeta')$ and $\nu_{\dot{\vartheta}_{\mathcal{K}}}$ is onto.

(3) $\hat{\zeta}'(\dot{\mathbf{X}}) = \hat{\zeta}' \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}(\mathbf{X}) = \zeta'(\mathbf{X}) = \zeta \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}\lceil_{\mathbf{X}}(\mathbf{X}) = \zeta(\dot{\mathbf{X}})$.□

Although quotient term eBA $\dot{\mathbf{T}}$ is not a free eBA, it is *admissibly-free*. Nevertheless, Theorem 3.8.5 and Corollary 3.8.7 told us that this result can not be improved in Birkhoff's approach.

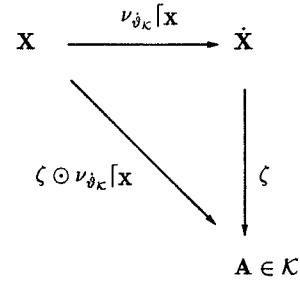

Figure 3–3: admissible evironment and admissibly freeness

The central idea of Birkhoff's approach is to capture equality through kernels of homomorphisms. This can be done in first order algebras simply because the cores and kernels of their homomorphisms coincide with each other. This coincidence does not hold in general for eBAs. Therefore, we turn to admissible BEs instead of BEs. This is the subject of next section.

3.9 Admissible Binding Equations

Analogous to Chapter 2, we can define the followings:

(a) $\mathbf{A} \models_{eBA} p \simeq q$ (or $\mathbf{A} \models p \simeq q$ for short) as for all environment $< \rho, \varphi >$, $\mathcal{A}[\![p]\!](\vec{\rho}) = \mathcal{A}[\![q]\!](\vec{\rho})$;

(b) a class $\mathcal{K}$ of eBAs satisfies $p \simeq q$, written as $\mathcal{K} \models_{eBA} p \simeq q$ or simply $\mathcal{K} \models p \simeq q$, iff each member $\mathbf{A}$ of $\mathcal{K}$ satisfies $p \simeq q$;

(c) let Γ_b (or Γ as an abbreviation) be a set of BEs, we say $\mathcal{K}$ satisfies Γ_b, written as $\mathcal{K} \models_{eBA} \Gamma_b$ (or $\mathcal{K} \models \Gamma$ for simplicity), iff $\mathcal{K} \models p \simeq q$ for each $p \simeq q \in \Gamma$; we also

use $Mod_{\Sigma^{BO}}(\Gamma)$ (or $Mod(\Gamma)$ for simplicity) to denote the class of eBAs, where each member $\mathbf{A}$ satisfies Γ;

(d) $\Gamma \models_{eBA} p \simeq q$ (or shortened as $\Gamma \models p \simeq q$) iff $\mathbf{A} \models \Gamma$ implies $\mathbf{A} \models p \simeq q$ for each $\mathbf{A}$.

Further, an environment $\vec{\rho}$ is said to be *admissible* iff its image can generate a perfect sub-eBA, i.e. $[\vec{\rho}(V \cup FV)]$ is a perfect sub-eBA. Then, we can define the above concepts accordingly in the context of Admissibility. Formally,

Definition 3.9.1 (admissible satisfaction $\dot{\models}_{eBA}$): *For binding term $p, q \in T$ or $p, q \in FT_m$ for some $m \in Nat$,*

(i) $\mathbf{A} \dot{\models}_{eBA} p \simeq q$ (or $\mathbf{A} \dot{\models} p \simeq q$), iff $\mathcal{A}[\![p]\!](\vec{\rho}) = \mathcal{A}[\![q]\!](\vec{\rho})$ for every admissible $\vec{\rho}$ on $\mathbf{A}$;

(ii) $\mathcal{K} \dot{\models}_{eBA} p \simeq q$ (or $\mathcal{K} \dot{\models} p \simeq q$), iff $\mathbf{A} \dot{\models} p \simeq q$ for each $\mathbf{A} \in \mathcal{K}$;

(iii) $\mathcal{K} \dot{\models}_{eBA} \Gamma$ (or $\mathcal{K} \dot{\models} \Gamma$) iff $\mathcal{K} \dot{\models} p \simeq q$ for each $p \simeq q \in \Gamma$;

(iv) $Adm_{\Sigma^{BO}}(\Gamma)$, or $Adm(\Gamma)$ when Σ^{BO} can be decided by context (anyway it is alway convenient to assume that Σ^{BO} is arbitrarily fixed), is defined to be $\{\mathbf{A} | \mathbf{A} \dot{\models} \Gamma\}$;

(v) $\Gamma \dot{\models}_{eBA} p \simeq q$ iff for each $\mathbf{A}$, $\mathbf{A} \dot{\models} \Gamma$ implies $\mathbf{A} \dot{\models} p \simeq q$;

(vi) $\dot{I}_{\mathcal{K}}(\mathbf{X}) =_{df} \{p \simeq q | \mathcal{K} \dot{\models}_{eBA} p \simeq q\}$, where $\mathcal{K}$ is a class of eBAs and

$\mathbf{X} =< V, \{FV_k | k \in Nat\} >$.

For Definition 3.9.1, we make the following remark.

(1) if $\mathbf{A} \models p \simeq q$, then $\mathbf{A} \dot{\models} p \simeq q$ where $\mathbf{A}$ is an eBA.

(2) $Mod(\Gamma) \subseteq Adm(\Gamma)$, where Γ is a set of BEs.

On the other hand, let $p \simeq q \in \Gamma^1$ iff $Adm(\Gamma) \dot{\models} p \simeq q$; and let $p' \simeq q' \in \Gamma^2$ iff $Mod(\Gamma) \models p' \simeq q'$. Then, we know in general neither that

(3) $p \simeq q \in \Gamma^1$ implies $p \simeq q \in \Gamma^2$ nor that

(4) $p' \simeq q' \in \Gamma^2$ implies $p' \simeq q' \in \Gamma^1$.

Essentially, these are the relations between satisfaction $\models_{eBA}$ and admissible satisfaction $\dot{\models}_{eBA}$. A further discussion of this will be resumed in Section 3.13.

Lemma 3.9.3 (effect of an eBH over an interpretation): *If* $\zeta : \mathbf{A} \stackrel{\cdot}{\to} \mathbf{A}'$ *is an eBH, then* $\zeta(\mathcal{A}[\![p]\!](\vec{\rho})) = \mathcal{A}'[\![p]\!](\zeta \odot \vec{\rho})$.

The proof is obvious and is left out.

The above lemma says that the effect of an eBH over an interpretation is completely determined by the effect of the eBH over the environment of the interpretation. Further exploiting this result, we have the following.

Lemma 3.9.4 (admissible BEs and admissible eBHs): *Let* $\mathcal{K}$ *be a collection of eBAs, then* $\mathcal{K} \dot{\models} p \simeq q$ *iff* $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$ *for each admissible eBH* $\zeta : \mathbf{T} \to \mathbf{A}$, *and every* $\mathbf{A} \in \mathcal{K}$.

Proof By Lemma 3.9.3 and Theorem 3.6.3. □

Lemma 3.9.4 says that the admissible equality can be completely captured by admissible eBHs. Consequently, we have that

Theorem 3.9.5 (admissible BE and admissible eBC): $\mathcal{K} \dot{\models} p \simeq q$ *iff* $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$, *where* $\mathcal{K}$ *is a class of eBAs.*

Proof

For "$\Leftarrow$", we suppose $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$ and given a $\mathbf{A} \in \mathcal{K}$. Let $\zeta : \mathbf{T} \to \mathbf{A}$ be an admissible eBH. Then, $\dot{\vartheta}_{\mathcal{K}} \subseteq Ker(\zeta)$. So, there exists $\check{\zeta} : \dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \to \mathbf{A}$ such that $\zeta = \check{\zeta} \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}$. Hence,

$$\begin{aligned} \zeta(\bullet_{[p]}) &= \check{\zeta} \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}(\bullet_{[p]}) = \check{\zeta} \odot \nu_{\dot{\vartheta}_{\mathcal{K}}}(\bullet_{[q]}) \quad (since\ Ker(\nu_{\mathcal{K}}) = \dot{\vartheta}_{\mathcal{K}}) \\ &= \zeta(\bullet_{[q]}) \end{aligned}$$

Consequently, $\mathcal{K} \dot{\models} p \simeq q$ (since for every environment $< \rho, \varphi >$, there exists $\zeta : \mathbf{T} \to \mathbf{A}$ such that the environment and ζ agree on $\mathbf{X}$ or $V \cup FV$).

For "$\Rightarrow$", assume $\mathcal{K} \dot{\models} p \simeq q$, i.e. for every admissible eBH $\zeta : \mathbf{T}(\mathbf{X}) \to \mathbf{A}$ where $\mathbf{A} \in \mathcal{K}$, $\zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$. In other words, $< \bullet_{[p]}, \bullet_{[q]} > \in Ker(\zeta)$. Therefore, since this does not depend on each individual ζ, $< \bullet_{[p]}, \bullet_{[q]} > \in \bigcap_{\zeta : \mathbf{T} \dot{\to} \mathbf{A}} Ker(\zeta)$. Again this dose not depend on each individual $\mathbf{A} \in \mathcal{K}$. So, $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$. □

Informally, Theorem 3.9.5 expresses that admissible equality is totally decided by certain eBC. Similar to Birkhoff's Theorem in Chapter 2, we have a theorem below. Formally

Theorem 3.9.6 (admissible Birkhoff's Theorem): *The following three statements are equivalent:*

1. $\mathcal{K} \dot{\models} p \simeq q$;

2. $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$;

3. $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \dot{\models} p \simeq q$.

Proof

The equivalence between 1 and 2 was established in Theorem 3.9.5. So, we only need to consider the equivalence between 2 and 3.

For "2 $\Rightarrow$ 3", we have that for every admissible eBH $\zeta : \mathbf{T} \to \dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}})$ there exists eBH $\hat{\zeta} : \dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \to \dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}})$ such that $\zeta = \nu_{\dot{\vartheta}_{\mathcal{K}}} \odot \hat{\zeta}$, since Theorem 3.8.5, Corollary 3.8.8, and the equivalence between 1 and 2. So, $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$ implies $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \dot{\models} p \simeq q$.

For "3 $\Rightarrow$ 2", we have that $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \dot{\models} p \simeq q$ implies $\nu_{\dot{\vartheta}_{\mathcal{K}}}(\bullet_{[p]}) = \nu_{\dot{\vartheta}_{\mathcal{K}}}(\bullet_{[q]})$, since $\nu_{\dot{\vartheta}_{\mathcal{K}}}$ is an admissible eBH. Hence, $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$. □

The above result shows that Admissibility is indeed a remedy to Birkhoff's approach. Further, it shows that admissible BEs are captured by admissible eBC, and this, in turn, can be captured by term eBA $\mathbf{T}$ through certain way. The remaining problem is to catch the certain way syntactically. This is the subject of next section.

3.10 Admissible Completeness of $\dot{\vdash}_{eBA}$

Analogous to first order case, we introduce admissibly-invariant eBC to characterize eBC $\dot{\vartheta}_{\mathcal{K}}$ over a class $\mathcal{K}$. To motivate the definition for admissibly-invariant eBC we need the following lemma. Namely

Lemma 3.10.1 (binding substitutions on T): *Let ϑ be an eBC on term eBA $\mathbf{T}$. For every $\zeta : \mathbf{T} \to \mathbf{T}/\vartheta$, there exists a $\check{\zeta} : \mathbf{T} \to \mathbf{T}$ such that $\zeta = \check{\zeta} \odot \nu_{\vartheta}$, i.e. the following diagram (see Figure 3-4) commutes.*

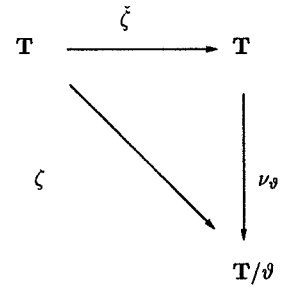

Figure 3–4: binding substitution

Proof

For each $x \in X$ and each $f \in X_m$, let us fix a value in $\nu_\vartheta^{-1}(\zeta(x))$ and in $\nu_\vartheta^{-1}(\zeta(f))$ for $\check{\zeta}$ on $\mathbf{X}$ by Axiom of Choice (note that if $\zeta(x) = \zeta(y)$ then $\nu_\vartheta^{-1}(\zeta(x)) = \nu_\vartheta^{-1}(\zeta(y))$). For readability, we keep the notation as $\check{\zeta}(x) =_{df} \nu_\vartheta^{-1}(\zeta(x))$ and $\check{\zeta}(f) =_{df} \nu_\vartheta^{-1}(\zeta(f))$.

Since $\mathbf{T}$ is free term eBA, then $\check{\zeta}$ can be extended as an eBH from $\mathbf{T}$ to $\mathbf{T}$. Obviously, $\zeta\lceil_{\mathbf{X}} = (\nu_\vartheta \odot \check{\zeta})\lceil_{\mathbf{X}}$. By the uniqueness of free eBA about eBHs, we come to $\zeta = \nu_\vartheta \odot \check{\zeta}$. □

As you may notice, $\check{\zeta}$ is a substitution as it is commonly called. This observation leads us to the definition of admissibly-invariant eBC. Formally,

Definition 3.10.2 (admissibly-invariant eBC): *An eBC ϑ on term eBA* $\mathbf{T}$ *is* admissibly invariant *iff for every admissible eBH* $\zeta : \mathbf{T} \to \mathbf{T}/\vartheta$, $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta$ *implies* $< \check{\zeta}(\bullet_{[p]}), \check{\zeta}(\bullet_{[q]}) > \in \vartheta$

Tricky examples of admissibly-invariant eBCs are

(a) $\mathbf{T} \times \mathbf{T}$ is an admissibly invariant eBC on $\mathbf{T}$;

(b) $\dot{\vartheta}_K$ is an admissibly invariant eBC on $\mathbf{T}$.

To justify the introduction of admissibly-invariant concept, we provide a lemma below. Formally

Lemma 3.10.3 (admissibly invariant eBC and admissible BE): *Let ϑ be an admissibly invariant eBC on term eBA* $\mathbf{T}$. *Then* $\mathbf{T}/\vartheta \dot{\models} p \simeq q$ *iff* $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta$.

Proof

Let $\{\vec{f}, \vec{x}\} = Free(p) \cup Free(q)$. So, we only need to consider the environments over $\{\vec{f}, \vec{x}\}$.

(i) For "$\Leftarrow$", $\mathbf{T}/\vartheta \dot{\models} p \simeq q \Leftrightarrow$ for all admissible $< \rho, \varphi >$, $\mathcal{T}[\![p]\!](\rho, \varphi) = \mathcal{T}[\![q]\!](\rho, \varphi)$.

From these $\mathcal{T}$ and $< \rho, \varphi >$, we can get an admissible eBH $\zeta : \mathbf{T} \to \mathbf{T}/\vartheta$ such that $\mathcal{T}[\![p]\!](\rho, \varphi) = \mathcal{T}[\![q]\!](\rho, \varphi) \Leftrightarrow \zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$.

Conversely, given an admissible $\zeta : \mathbf{T} \to \mathbf{T}/\vartheta$, we have an admissible $< \rho, \phi >$ such that $\mathcal{T}[\![p]\!](\rho, \varphi) = \mathcal{T}[\![q]\!](\rho, \varphi) \Leftrightarrow \zeta(\bullet_{[p]}) = \zeta(\bullet_{[q]})$.

Since $\nu_\vartheta : \mathbf{T} \to \mathbf{T}/\vartheta$ is onto, we have that $\nu_\vartheta(\check{\zeta}(\bullet_{[p]})) = \nu_\vartheta(\check{\zeta}(\bullet_{[q]}))$, i.e. $< \check{\zeta}(\bullet_{[p]}), \check{\zeta}(\bullet_{[q]}) > \in \vartheta$. Because of fully-invariant property of ϑ, we have that $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta$ implies for every $\check{\zeta} : \mathbf{T} \to \mathbf{T}$, $< \check{\zeta}(\bullet_{[p]}), \check{\zeta}(\bullet_{[q]}) > \in \vartheta$.

(ii) For "$\Rightarrow$", since ν_ϑ is admissible, $\mathbf{T}/\vartheta \dot{\models} p \simeq q$ implies $\nu_\vartheta(\bullet_{[p]}) = \nu_\vartheta(\bullet_{[q]})$. In other words, $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta$. □

The above lemma says that an admissible invariant eBC decides the admissible equality of its quotient eBA. This convinces us that we are on the right track.

Definition 3.10.4 (least admissible invariance containing $\vec{E}$): *Let $\vec{E}$ be a family of $E \subseteq T \times T$ and $E_k \subseteq FT_k \times FT_k$ for $k \in Nat$. Then, $\vartheta_A(\vec{E})$ denote the* least admissibly-invariant *eBC on* $\mathbf{T}$ *containing* $\vec{E}$.

Sometimes, eBC $\vartheta_A(\vec{E})$ is called the *admissibly invariant eBC generated by* $\vec{E}$. It comes from the fact that $\vartheta_A(\vec{E}) =_{df} \bigcap \Psi(\vec{E})$ where $\Psi(\vec{E})$ is the collection of all possible admissibly invariant eBC congruence on $\mathbf{T}$ containing $\vec{E}$. Such a definition is well-defined, since $\mathbf{T} \times \mathbf{T}$ is an admissibly invariant eBC (i.e. $\Psi(\vec{E}) \neq \emptyset$) and the property of admissibly invariant is preserved by an arbitrary intersection (e.g. $\vartheta_1 \cap \vartheta_2$ is admissibly invariant if both of them are).

Definition 3.10.5 (function ϕ): *Given a pair $< X, \{X_k | k \in Nat\} >$ where X is set of ordinary variables and X_k is a set of function variables with arity $k \in Nat$. Let $\vec{\mathbf{X}}$ be such a pair, and $\phi :\ll \dot{I}(\vec{X}) \gg\!\!\rightarrow \mathbf{T} \times \mathbf{T}$ be the bijection defined by $\phi([p] \simeq [q]) =_{df} < \bullet_{[p]}, \bullet_{[q]} >$, where $\ll \dot{I}(\vec{X}) \gg =_{df} \{[p] \simeq [q] | p \simeq q \in \dot{I}(\vec{X})\}$ (and $[p] =_{df} \{q | p \dot{\approx} q\}$).*

Obviously, we would have $\vartheta_A(\phi(\Gamma)) = \dot{\vartheta}_{Adm(\Gamma)}$ for every $\Gamma \subseteq (T \times \{\simeq\} \times T) \cup (FT \times \{\simeq\} \times FT)$ (hint : see the tricky examples of previously given admissibly-invariant eBCs and the least condition of ϑ_A for one direction of inclusions, and see Lemma 3.10.3 to get the fact $\mathbf{T}/\vartheta_A(\phi(\Gamma)) \in Adm(\Gamma)$ for the other direction of inclusions). We state it as theorem. Formally,

Theorem 3.10.6 (least admissible invariance and admissible BEs): $\Gamma \dot{\models} p \simeq q$ *iff* $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta_A(\phi(\Gamma))$.

Such an observation leads to a concept of admissible substitutions. Formally

Definition 3.10.7 (admissible substitution with respect to Γ): *Given a set Γ of (admissible) BEs, a substitution $\vec{\varrho}$ is said to be* admissible *with respect to Γ iff the corresponding eBH $\zeta : \mathbf{T} \rightarrow \mathbf{T}$ (i.e. $\zeta(\bullet_{[p]}) = \bullet_{[p\vec{\varrho}]}$) has the property that $\nu_{\vartheta_A(\phi(\Gamma))} \odot \zeta$ is an admissible eBH.*

We denote an admissible substitution map $\vec{\varrho}$ as $\vec{\varrho} \in Sub_A(\Gamma)$. An example of such substitution map is the identity substitution map $\vec{\imath} \in Sub_A(\Gamma)$.

The well-definedness of this definition (Definition 3.10.7) is guaranteed by the fact that $\mathbf{T}$ is a free eBA over all possible eBAs

From the above definition, we can easily get a lemma. Namely

Lemma 3.10.8 (admissible BEs and admissible substitutions):

If $Adm(\Gamma) \dot{\models} p \simeq q$ then $Adm(\Gamma) \dot{\models} p\vec{\varrho} \simeq q\vec{\varrho}$, where $\vec{\varrho} \in Sub_A(\Gamma)$.

Proof

We simply have a corresponding eBH ζ for $\vec{\varrho}$. Then, since $\nu_{\vartheta_A(\phi(\Gamma))} \odot \zeta$ is admissible and $\vartheta_A(\phi(\Gamma)) = \dot{\vartheta}_{Adm(\Gamma)}$, we have that $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{Adm(\Gamma)}$ implies $< \zeta(\bullet_{[p]}), \zeta(\bullet_{[q]}) > \in \dot{\vartheta}_{Adm(\Gamma)}$. Putting it in another way, we would have that $Adm(\Gamma) \dot{\models} p \simeq q$ implies $Adm(\Gamma) \dot{\models} p\vec{\varrho} \simeq q\vec{\varrho}$. □

From Lemma 3.10.8, we understand that $\dot{\vartheta}_{Adm(\Gamma)}$ is closed under admissible substitutions. However, we have no feasible criteria to judge whether a given substitution is admissible. Nevertheless, the effect of applying substitutions to binding terms is completely decided by the images of the substitutions on the free variables of the binding terms. Luckily, there is a way to get around it, that is, the number of free variables in binding terms must be finite, and the identity substitution $\vec{\iota}$ is obviously admissible. So, by shifting variables in the identity substitution $\vec{\iota}$ to preserve the Admissibility and to share the images of an arbitrary substitution $\vec{\varrho}$ on the free variables of a binding term p, we would get an admissible substitution $\vec{\varrho}$ such that $p\vec{\varrho} = p\vec{\varrho}$. Hence, we can use arbitrary substitutions in practice. This completes our effort in search of syntactic characterization of admissible equality. To confirm this claim, we define the admissible calculus $\dot{\vdash}_{eBA}$ (or $\dot{\vdash}$) as follows and prove the claim in Theorem 3.10.10.

Calculus $\dot{\vdash}$ is almost the same as $\vdash_{EQ}$ in Chapter 2 except that (a) it has three extra rules (α), (ξ) and (ξ^{-1}); (b) composition rule (eq-cmp) in $\vdash_{EQ}$ is replaced by two rules (cmp-1) and (cmp-2); (c) its substitution is $\vec{\varrho}$ not ι; (d) meta variables are mainly p, q, r instead of t, u, v respectively.

Definition 3.10.9 (admissible calculus $\dot{\vdash}_{eBA}$): *The admissible calculus $\dot{\vdash}_{eBA}$ (or $\dot{\vdash}$) is almost the same as calculus $\vdash_{iBA}$ (defined in Theorem 1.2.3.1) except that substitution rule (func-sub) should be replaced by (b-sub) below:*

$$(b\text{-}sub) \quad \frac{\Gamma\dot{\vdash}_{eBA} p \simeq q}{\Gamma\dot{\vdash}_{eBA} p\vec{\varrho} \simeq q\vec{\varrho}}$$

where $\vec{\varrho}$ is an arbitrary family of substitution functions ϱ from ordinary variables V to ordinary terms T and ϱ_m from function variables FV_m to function terms FT_m with arity $m \in Nat$.

Let $\dot{\mathcal{M}}_b$ (or $\dot{\mathcal{M}}$) be a function which produces a set $\dot{\mathcal{M}}_b(\Gamma)$ when given a set Γ, $p \simeq q \in \dot{\mathcal{M}}(\Gamma)$ iff $\Gamma\dot{\vdash}_{eBA} p \simeq q$ without using cut rules. Obviously, $\dot{\mathcal{M}}_b$ is monotonic, and we can easily have that $p \simeq q \in \sqcup\dot{\mathcal{M}}(\Gamma)$ iff $\Gamma\dot{\vdash}_{eBA} p \simeq q$.

Theorem 3.10.10: $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg) = \vartheta_A(\phi(\ll \Gamma \gg))$.

Proof

(i) First of all, we define a new rule (adm-ξ), called admissible ξ, as

$$(\text{adm-}\xi)\quad \frac{\Gamma\dot{\vdash} t[\vec{x}:=\vec{u}] \simeq t'[\vec{x}:=\vec{u}]}{\Gamma\dot{\vdash}\langle \vec{x}:t\rangle \simeq \langle \vec{x}:t'\rangle}$$

where $\vec{\imath}[\vec{x}:=\vec{u}]$ is a family of substitution functions and $\{\vec{x}\} \subseteq V$.

Then, we understand that if we define a new $\dot{\mathcal{M}}_b$ which is alomst the same as the previous one except without (ξ) rule but with (adm-ξ) rule, both of them are actually equivalent to each other (hint: comparing the two premises of (ξ) rule and (adm-ξ) rule with (b-sub) rule, and the fact that the identity substitution $\vec{\imath}$ is admissible for arbitrary Γ).

(ii.a) $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg)$ is obviously an equivalence relation and containing $\phi(\ll \Gamma \gg)$.

(ii.b) By (α) rule, (ξ^{-1}) rule, (b-sub) rule and (adm-ξ) rule with the above (ii.a), we have $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg)$ satisfies (eBH-ext) (in Definition 3.7.1).

(ii.c) By (cmp-1) and (cmp-2) rules with the above (ii.a) and (ii.b), we have that $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg)$ is an eBC.

(ii.d) By the (b-sub) rule in Definition 3.10.9, we get that $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg)$ is admissibly invariant.

So, combine with all items in (i) and (ii)'s, we know $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg) \supseteq \vartheta_A(\phi(\ll \Gamma \gg))$.

(iii) On the other hand, it is quite easy to see that $\phi^{-1}(\vartheta_A(\phi(\ll \Gamma \gg)))$ is closed under the following rules, where $\phi^{-1}(\vartheta) =_{df} \{[p] \simeq [q] \mid\mid < \bullet_{[p]}, \bullet_{[q]} > \in \vartheta\}$ (and $[p] = \{q \mid p \dot{\approx} q\}$):

(iii.a) for (α) rule, trivial;

(iii.b) for (ξ^{-1}) rule, a simple exercise (hint : $\bullet_{[(\vec{x}:u)]}(\bullet_{[\vec{y}]}) = \bullet_{[u[\vec{x}:=\vec{y}]]}$);

(iii.c) for (cmp-1) and (cmp-2) rules, trivial;

(iii.d) for (b-sub) rule, an easy exercise (hint: if $\check{\zeta} \in Sub_A(\Gamma)$ then $\nu_{\vartheta_A(\phi(\ll\Gamma\gg))} \odot \check{\zeta}$ is an admissible eBH);

(iii.e) for (adm-ξ) rule, we show the closeness below: suppose $< \bullet_{[t]}, \bullet_{[t']} > \in \vartheta_A(\phi(\ll \Gamma \gg))$.

For every $\zeta : \mathbf{T} \to \mathbf{T}$ such that $\check{\zeta} \odot \nu_{\vartheta_A(\phi(\ll \Gamma \gg))}$ is an admissible eBH. Let $\zeta(x_j)$ be u_j $(1 \leq j \leq |\vec{x}|)$, we would have that $\bullet_{[(\vec{x}:t)]}(\bullet_{[u_1]}, \bullet_{[u_2]}, ..., \bullet_{[u_{|\vec{x}|}]}) = \bullet_{[t[\vec{x}:=\vec{u}]]} = \zeta(t)$ and $\bullet_{[(\vec{x}:t')]}(\bullet_{[u_1]}, \bullet_{[u_2]}, ..., \bullet_{[u_{|\vec{x}|}]}) = \bullet_{[t'[\vec{x}:=\vec{u}]]} = \zeta(t')$. Hence, we would have $< \zeta(t), \zeta(t') > \in \vartheta_A(\phi(\ll \Gamma \gg))$, since $\check{\zeta} \odot \nu_{\vartheta_A(\phi(\ll \Gamma \gg))}$ is an admissible eBH,

Then, by (eBH-ext), we get that $< \bullet_{[(\vec{x}:t)]}, \bullet_{[(\vec{x}:t')]} > \in \vartheta_A(\phi(\ll \Gamma \gg))$. In other words, $\phi^{-1}(\vartheta_A(\phi(\ll \Gamma \gg)))$ is closed under the admissible (adm-ξ) rule.

Therefore, by (i) and by the least condition of $\sqcup\dot{\mathcal{M}}_b(\Gamma)$, we have $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg) \supseteq \vartheta_A(\phi(\ll \Gamma \gg))$.

(iv) By the above (ii) and (iii), we come to $\phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg) = \vartheta_A(\phi(\ll \Gamma \gg))$. □

From Theorem 3.10.10, we can infer that $\Gamma \dot{\models} p \simeq q$ iff $p \simeq q \in \sqcup\dot{\mathcal{M}}_b(\Gamma)$, since $\Gamma \dot{\models} p \simeq q$ iff $\dot{\vartheta}_{Mod_{\Sigma BO}(\Gamma)}$ iff $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta_A(\phi(\ll \Gamma \gg))$ iff $< \bullet_{[p]}, \bullet_{[q]} > \in \phi(\ll \sqcup\dot{\mathcal{M}}_b(\Gamma) \gg)$ iff $p \simeq q \in \sqcup\dot{\mathcal{M}}_b(\Gamma)$. So, I state it as a corollary.

Corollary 3.10.11 (soundness and completeness of $\dot{\vdash}_{eBA}$): *$\Gamma \dot{\models} p \simeq q$ iff $\Gamma \dot{\vdash} p \simeq q$.*

Although we have no general syntactic characterization of admissible substitutions, this is hardly a limitation, (See comments before Definition 3.10.9). Of course, a general characterization of admissible substitution (and/or non-admissible substitutions) deserves our future attention.

3.11 Admissible variety problem

Since the admissible models seems not closed under products, there is an open problem whether there is a variety concept such that it is the least closure of certain operations. On the other hand, we know that the quotient eBA $\dot{\mathbf{T}}(\dot{\mathbf{X}})$ admissibly satisfies $\dot{I}_{\mathcal{K}}$ for the class $\mathcal{K}$ of eBAs. This naturally leads to a question whether there is an operation to replace products, which enables us to get a Birkhoff's variety concept.

Definition 3.11.1 (direct product of eBA): *Let $\mathbf{A}_k$ ($k \in Ind$) be an indexed family of eBAs, the direct product $\mathbf{A}$, written as $\prod_{k\in Ind}\mathbf{A}_k$, is defined as*

(a) $g^{\mathbf{A}} =< g^{\mathbf{A}_k} >_{k\in Ind}$ *and* $g^{\mathbf{A}}(\vec{a})(k) = g^{\mathbf{A}_k}(\vec{a}(k))$, *where* $k \in Ind$ *and* $a_k \in \prod_{k\in Ind} A_k$.

(b) $\sigma^{\mathbf{A}}(\vec{g}^{\mathbf{A}}, \vec{a})(k) = \sigma^{\mathbf{A}_k}(g_1^{\mathbf{A}_k}, g_2^{\mathbf{A}_k}, ..., g_{|\vec{m}|}^{\mathbf{A}_k}, \vec{a}(k))$ *for* $k \in Ind$ *and* $g_i^{\mathbf{A}} \in \prod_{k\in Ind} \mathcal{F}_{m_i}^{\mathbf{A}_k}(A_k)$ *and* $a_j \in \prod_{k\in Ind} A_k$.

Apparently, eBAs are closed under direct products. For sub-direct products, we introduce them as follows.

Definition 3.11.2 (sub-direct product): *An sub-eBA $\mathbf{A}$ ($= \sum_{k\in Ind} A_k$) of $\prod_{k\in Ind}\mathbf{A}_k$, where $\mathbf{A}_k$ is an eBA indexed by $k \in Ind$, is said to be a sub-direct product of the indexed family $\mathbf{A}_k$ ($k \in Ind$) of eBAs iff*

(i) $\mathbf{A}$ is a perfect sub-eBA of $\prod_{k\in Ind}\mathbf{A}_k$;

(ii) $\pi_k(\mathbf{A}) = \mathbf{A}_k$ for each $k \in Ind$.

Similar to first order algebras, we introduce sub-direct embedding as follows.

Definition 3.11.3 (sub-direct embedding): *An embedding (1-1 and eBH) $\zeta : \mathbf{A} \to \prod_{k\in Ind}\mathbf{A}_k$ is a sub-direct iff $\zeta(\mathbf{A})$ is a sub-direct product of $\mathbf{A}_k$ ($k \in Ind$).*

An example of sub-direct embedding is given below.

Lemma 3.11.4 (natural eBH and sub-direct embedding): *If ϑ_k is eBC on an eBA $\mathbf{A}$, and $\bigcap_{k\in Ind}\vartheta_k = Diag$ (diagonal relation, it is an eBC), then the natural eBH $\nu : \mathbf{A} \to \prod_{k\in Ind}(\mathbf{A}/\vartheta_k)$ defined by $\nu(g)(k) = g/\vartheta_k$ for $g \in \mathcal{F}_m^{\mathbf{A}}(A)$ ($m \geq 0$) is a sub-direct embedding.*

The intersection of a collection of eBCs is an eBC, and it has an interesting property with the members of the collection. The property is formalized as follows.

Lemma 3.11.5 (intersection of eBCs): *Let $\vartheta = \bigcap_{k\in Ind}\vartheta_k$ and ϑ_k be a eBC on an eBA $\mathbf{A}$. Then $\bigcap_{k\in Ind}(\vartheta_k/\vartheta) = diag$ on $\mathbf{A}/\vartheta$, where diag is the diagonal eBC.*

Also, the isomorphic property among components of a product of eBAs is preserved by the product. Formally,

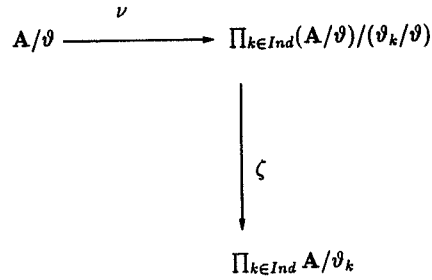

Figure 3–5: quotient eBA and sub-direct embedding

Lemma 3.11.6 (isomorphism and product): *If each eBA* $\mathbf{A}_k$ *is isomorphic to* $\mathbf{A'}_k$ *($k \in Ind$), then* $\prod_{k\in Ind}\mathbf{A}_k$ *is isomorphic to* $\prod_{k\in ind}\mathbf{A'}_k$.

There is an simple fact about preservation of perfectness under eBHs. That is,

Lemma 3.11.7 (perfectness and eBHs): *If* $\mathbf{A}$ *is a perfect sub-eBA of an eBA* $\mathbf{A'}$, *and* $\zeta : \mathbf{A'} \to \mathbf{A''}$ *is 1-1, then* $\zeta(\mathbf{A})$ *is a perfect sub-eBA of* $\mathbf{A''}$.

This lemma can be regarded as an improvement of Lemma 3.3.3.

For a collection of eBCs and their quotient eBAs, the following is an interesting fact.

Theorem 3.11.8 (quotient eBA and sub-direct embedding): *If* $\mathbf{A}$ *is an eBA and* ϑ_k *is an eBC ($k \in Ind$), let* $\vartheta = \bigcap_{k\in Ind}\vartheta_k$, *then* $\mathbf{A}/\vartheta$ *can be sub-directly embedded in* $\prod_{k\in Ind}\mathbf{A}/\vartheta_k$.

Proof See Figure 3-5.

By Lemma 3.11.4, Lemma 3.11.5 and Lemma 3.11.7, $\mathbf{A}/\vartheta$ can sub-directly embedded in $\prod_{k\in Ind}(\mathbf{A}/\vartheta)/(\vartheta_k/\vartheta)$. By Lemma 3.7.11 and Lemma 3.11.6, we have that $\prod_{k\in Ind}(\mathbf{A}/\vartheta)/(\vartheta_k/\vartheta)$ is isomorphic to $\prod_{k\in Ind}\mathbf{A}/\vartheta_k$. □

Theorem 3.11.9 (admissible quotient eBA): *For $\mathcal{K} \neq \emptyset$, $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \in \mathbf{IS}_p\mathbf{P}(\mathcal{K})$, in particular, if $\mathcal{K}$ is closed under $\mathbf{I}, \mathbf{S}_p, \mathbf{P}$, then $\dot{\mathbf{T}}_{\mathcal{K}}(\dot{\mathbf{X}}) \in \mathcal{K}$, where*

$\mathbf{I}(\mathcal{K}) = \{\mathbf{A}' \mid$ *for some* $\mathbf{A} \in \mathcal{K}$ *and* $\mathbf{A}' \cong \mathbf{A}\}$ *($\cong$ means "is ismorphic to")*

$\mathbf{S}_p(\mathcal{K}) = \{\mathbf{A}' \mid$ *for some* $\mathbf{A} \in \mathcal{K}$ *and* $\mathbf{A}' \preceq \mathbf{A}\}$ *and*

$\mathbf{P}(\mathcal{K}) = \{\mathbf{A}' \mid \mathbf{A}'$ *is a product of a subset of* $\mathcal{K}\}$.

Almost all proof of this section are left out, either they can be straightly verified or the proof of later lemmas (or theorems) depends directly on the proof of an earlier lemma (or theorem). There is an interesting question which comes from Theorem 3.11.10. Namely,

Question 3.11.10 (closure under products): *Under what condition, especially a equationally definable condition, $\mathcal{K}$ is closed under direct product?*

Also, we name the existence of the operator under which variety is closed as an open problem.

Open Problem 3.11.11 (admissible variety): *Whether there is an constructible Υ over a class $\mathcal{K}$ of algebras such that $Adm(\dot{I}_{\mathcal{K}}) = \Upsilon(\mathcal{K})$?*

3.12 Logical Relations vs Admissibility

In this subsection, we are going to discuss the relationship between Logical Relations and Admissibility.

Definition 3.12.1 (Logical Relation on eBAs): *A relation $\zeta \subseteq \mathbf{A} \times \mathbf{A}'$ is logical iff given $g \in \mathcal{F}_m^{\mathbf{A}}$ and $h \in \mathcal{F}_m^{\mathbf{A}'}$, $g\zeta_m h \Leftrightarrow (\forall a_j \in A, \forall b_j \in A'. \bigwedge_{j=1}^m a_j \zeta b_j \Rightarrow g(\vec{a})\zeta h(\vec{b}))$.*

Let $\zeta : \mathbf{A} \to \mathbf{A}'$ be an eBH, then we have that it is logical iff for all $m \geq 0$, $g \in \mathcal{F}_m^{\mathbf{A}}$ and $h \in \mathcal{F}_m^{\mathbf{A}'}$ such that $\forall a_j \in A. \zeta(g(\vec{a})) = h(\zeta\vec{a}) \Rightarrow \zeta g = h$.

From this, we understand that the image of a logical eBH ζ has enough points to identify not only

(a) the functions in its image function spaces (i.e. $\zeta(\mathcal{F}^{\mathbf{A}'})$) but also

(b) all functions in the function spaces of $\mathcal{F}^{\mathbf{A}'}$.

In a contrast, the image of an admissible ζ can only identify the functions in its mage function spaces (i.e. only (a) holds). Consequently,

Lemma 3.12.2 (logical and admissible): *Let $\zeta : \mathbf{A} \to \mathbf{A}'$, then, ζ is logical implies ζ is admissible*

This lemma says that an eBH is certainly admissible if it is logical.

3.13 Completeness and admissible completeness

In this section, we are going to discuss more on relationship between $\models_{eBA}$ and $\dot{\models}_{eBA}$ (or between $Mod(\Gamma)$ and $Adm(\Gamma)$ for given Γ) in order to clarify more on relationship between Completeness and Admissible Completeness of $\dot{\vdash}_{eBA}$.

Let us start with summarizing the main results of Section 2.1 and comparing these with the ones in this chapter.

From Section 2.1, we can get a congruence θ_Γ by given Γ through calculus $\vdash_{EQ}$; i.e.

(3.13.a) $< X, t, u > \in \theta_\Gamma$ iff $\Gamma \vdash_{EQ} t \simeq_X u$.

This θ_Γ is cross-invariant so that

(3.13.b) $\mathbf{T}/\theta_\Gamma \models_{EQ} t \simeq_X u$ iff $< X, t, u > \in \theta_\Gamma$.

Therefore, we have

(3.13.c) $\Gamma \vdash_{EQ} t \simeq_X u$ iff $\mathbf{T}/\theta_\Gamma \models_{EQ} t \simeq_X u$.

Furthermore, by Birkhoff Theorems (through Freeness), there is a congruence ϑ such that

(3.13.d) $\Gamma \models_{EQ} t \simeq_X u$ iff $\mathbf{T}/\vartheta \models_{EQ} t \simeq_X u$.

Then, by establishing the coincidence of ϑ and θ_Γ, we have Completeness of $\vdash_{EQ}$; i.e.

(3.13.e) $\Gamma \models_{EQ} t \simeq_X u$ iff $\Gamma \vdash_{EQ} t \simeq_X u$.

Similarily, we can carry the same argument about $\vdash_{EQ}$ to the present $\dot{\vdash}_{eBA}$; i.e.

(3.13.a') $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta_\Gamma$ iff $\Gamma \dot{\vdash}_{eBA} p \simeq q$.

We can still get

(3.13.b') $\mathbf{T}/\vartheta_\Gamma \models_{eBA} p \simeq q$ iff $< \bullet_{[p]}, \bullet_{[q]} > \in \vartheta_\Gamma$.

Therefore, we have

(3.13.c') $\Gamma \dot{\vdash}_{eBA} p \simeq q$ iff $\mathbf{T}/\vartheta_\Gamma \models_{eBA} p \simeq q$.

However, the crucial point of this chapter is that validation (3.13.d') can not be established by Birkhoff method (through Freeness);

(3.13.d') $\Gamma \dot{\vdash}_{eBA} p \simeq q$ iff $\mathbf{T}/\vartheta_\Gamma \models_{eBA} p \simeq q$.

That is, we do not know whether

(3.13.d") $Mod(\Gamma) \models_{eBA} p \simeq q$ iff $\mathbf{T}/\vartheta_\Gamma \models_{eBA} p \simeq q$.

The reason for this is that $\mathbf{T}/\vartheta_\Gamma$ can not be established as a free eBA over the class $Mod(\Gamma)$ of eBAs. We are only able to establish that $\mathbf{T}/\vartheta_\Gamma$ is admissibly free over an enlarged class $Adm(\Gamma)$, where $Mod(\Gamma) \subseteq Adm(\Gamma)$ since $\mathbf{A} \models_{eBA} p \simeq q$ implies $\mathbf{A} \dot{\models}_{eBA} p \simeq q$ for any eBA $\mathbf{A}$; i.e.

(3.13.d*) $Adm(\Gamma) \dot{\models}_{eBA} p \simeq q$ iff $\mathbf{T}/\vartheta_\Gamma \dot{\models}_{eBA} p \simeq q$.

Actually, it is easy to verify that

(3.13.f) $\mathbf{T}/\vartheta_\Gamma \dot{\models}_{eBA} p \simeq q$ iff $\mathbf{T}/\vartheta_\Gamma \models_{eBA} p \simeq q$.

So, we have

(3.13.e') $\Gamma \dot{\models}_{eBA} p \simeq q$ iff $\Gamma \dot{\vdash}_{eBA} p \simeq q$.

Secondly, we know that

(3.13.g) $Adm(\Gamma) \models_{eBA} p \simeq q$ implies both $Adm(\Gamma) \dot{\models}_{eBA} p \simeq q$ and $Mod(\Gamma) \models_{eBA} p \simeq q$; and

(3.13.h) eitehr $Adm(\Gamma) \dot{\models}_{eBA} p \simeq q$ or $Mod(\Gamma) \models_{eBA} p \simeq q$ implies $Mod(\Gamma) \dot{\models}_{eBA} p \simeq q$.

Question 3.13.1: *What is the relation between* $Mod(\Gamma) \models_{eBA} p \simeq q$ *and* $Adm(\Gamma) \dot{\models}_{eBA} p \simeq q$*?*

We already point out that $Mod(\Gamma) \subseteq Adm(\Gamma)$. So, if $Mod(\Gamma) \subseteq Adm(\Gamma)$, then we would have the coincidence between Completeness and Admissible Completeness. Unfortunately, we are not able to establish this. In what follows, we are trying to identify where it is possible to cause a difference between $Adm(\Gamma)$ and $Mod(\Gamma)$.

Recall (3.13.f), we know that the crucial point is that there exists a surjective eBH from $\mathbf{T}$ to $\mathbf{T}/\vartheta_\Gamma$. This can be further generalized to the following.

Lemma 3.13.2: *Let* $\mathbf{A} =<< A, \mathcal{F} >, \mathcal{A} >$ *be an eBA such that* $|A| \leq \aleph_0$ *(i.e. up to the countable infinite cardinal), then* $\mathbf{A} \models_{eBA} p \simeq q$ *iff* $\mathbf{A} \dot{\models}_{eBA} p \simeq q$.

From this lemma, we know that $\mathbf{A} \in Adm(\Gamma)$ and $|A| \leq \aleph_0$ implies $\mathbf{A} \in Mod(\Gamma)$. For the case of large cardinals, i.e. $|A| > \aleph_0$, we can think of applying the idea of the existence of surjective eBHs from a term eBA to $\mathbf{A}$ here. In other words, we can enlarge the set V to V' such that $|V'| = \aleph \geq \aleph_0 = |V|$ and obtain a new term eBA $\mathbf{T}(V', FV)$ instead of $\mathbf{T}(V, FV)$ (see [155,82,80] for references on large cardinals). We shall abbreviate them as $\mathbf{T}'$ and $\mathbf{T}$ respectively. Furthermore, we introduce the following:

(i) $\mathbf{A} \models^{\aleph}_{eBA} p \simeq q$ iff $\beta(\bullet_{[p]}) = \beta(\bullet_{[q]})$ for every eBH $\beta : \mathbf{T}' \to \mathbf{A}$;

(ii) $\mathbf{A} \dot{\models}^{\aleph}_{eBA} p \simeq q$ iff $\beta(\bullet_{[p]}) = \beta(\bullet_{[q]})$ for each admissible eBH $\beta : \mathbf{T}' \dot{\to} \mathbf{A}$;

(iii) $\mathbf{A} \in Mod^{\aleph}(\Gamma)$ iff $\mathbf{A} \models^{\aleph}_{eBA} p \simeq q$ for every $p \simeq q \in \Gamma$;

(iv) $\mathbf{A} \in Adm^{\aleph}(\Gamma)$ iff $\mathbf{A} \dot{\models}^{\aleph}_{eBA} p \simeq q$ for each $p \simeq q \in \Gamma$.

Note that $\dot{\models}^{\aleph_0}_{eBA}$ and $\dot{\models}_{eBA}$ coincide with each other so that $Mod(\Gamma) = Mod^{\aleph_0}(\Gamma)$ and $Adm(\Gamma) = Adm^{\aleph_0}(\Gamma)$.

Similarily, we have Calculus $\dot{\vdash}^{\aleph}_{eBA}$ corresponding to $\dot{\models}^{\aleph}_{eBA}$. Also, $\dot{\vdash}_{eBA}$ and $\dot{\vdash}^{\aleph_0}_{eBA}$ coincide with each other.

Recall the proof of Admissible Completeness of $\dot{\vdash}_{eBA}$, by carefully examining the proof we understand that the same proof can be carried through for $\dot{\vdash}^{\aleph}_{eBA}$ so that $\dot{\vdash}^{\aleph}_{eBA}$ is sound and admissibly complete. That is,

Theorem 3.13.3: *For any cardinal* $\aleph \geq \aleph_0$, $\Gamma \dot{\models}^{\aleph}_{eBA} p \simeq q$ *iff* $\Gamma \dot{\vdash}^{\aleph}_{eBA}$.

Actually, there is no essential difference between $\dot{\vdash}_{eBA}$ and $\dot{\vdash}^{\aleph}_{eBA}$ except that the number of the ordinary variables available in $\dot{\vdash}_{eBA}$ is countably infinite $\aleph_0$ and the number of the ones available in $\dot{\vdash}^{\aleph}_{eBA}$ is $\aleph$ ($> \aleph_0$). In this sense, we regard both $\dot{\vdash}_{eBA}$ and $\dot{\vdash}^{\aleph}_{eBA}$ as identical. Also, it can be verified that

(1) $Mod(\Gamma) = Mod^{\aleph_0}(\Gamma) = Mod^{\aleph}(\Gamma)$ for each $\aleph \geq \aleph_0$;

(2) $Adm(\Gamma) = Adm^{\aleph_0}(\Gamma) \supseteq Adm^{\aleph}(\Gamma)$ for every $\aleph \geq \aleph_0$.

So, the result of Lemma 3.13.2 can be generalized. That is,

Lemma 3.13.4: *Let* $\mathbf{A} =<< A, \mathcal{F} >, \mathcal{A} >$ *be an eBA such that* $\aleph_0 \leq |A| \leq \aleph$, *then* $\mathbf{A} \models^{\aleph}_{eBA} p \simeq q$ *iff* $\mathbf{A} \dot{\models}^{\aleph}_{eBA} p \simeq q$.

From this lemma, we have that $\mathbf{A} \in Adm^{\aleph}(\Gamma)$ and $\aleph_0 \leq |A| \leq \aleph$ implies $\mathbf{A} \in Mod(\Gamma)$. So, for $\aleph \geq \aleph_0$, if $\mathbf{A} \in Adm(\Gamma)$ and $|A| = \aleph$, then $\mathbf{A} \in Adm^{\aleph}(\Gamma)$ implies $\mathbf{A} \in Mod(\Gamma)$. Hence, a possible difference can only lie between $Adm(\Gamma)$ and $Adm^{\aleph}(\Gamma)$; i.e. whether there is an eBA $\mathbf{A} \in Adm(\Gamma)$ such that $\mathbf{A} \notin Adm^{\aleph}(\Gamma)$ for every $\aleph > \aleph_0$.

If there is no such an eBA, then $\dot{\vdash}_{eBA}$ is sound and complete; i.e. Completeness and Admissible Completeness of $\dot{\vdash}_{eBA}$ are one. Furthermore, the freeness requirement on term eBAs is too strong for Completeness and a weaker one like admissible freeness is enough. But we suspect that there exists such an eBA. However, we fail to construct an example of such. Basically, the reason for this is that if such an examples exists, it must have uncountable ordinary objects (see Lemma 3.13.2); e.g.

- in its function spaces, it has two different functions which agree on countable ordinary objects which forms a part of a perfect sub-eBA;

- on the other hand, it may not have a perfect sub-EBA which has countably ordinary objects.

We leave these as open problems.

Chapter 4

Extended Remedy

Analogous to Chapter 2, especially analogous to the relations among $\vdash_{EQ}$, $\vdash_{dEQ}$, $\vdash_{qEQ}$ and $\vdash_{uEQ}$, we are considering to extend the remedy of Birkhoff's approach in Chapter 3 to admissible dependent implication of eBAs and admissible quasi-dependent implication of eBAs in this chapter. That is, calculi $\dot{\vdash}^{d}_{eBA}$ and $\dot{\vdash}^{q}_{eBA}$ are the subject of this chapter. Their soundness and completeness are provided. An effort is also devoted to unify the above three forms of the calculi, i.e. admissible universal BEs (or admissible uBEs) $\dot{\vdash}^{u}_{eBA}$.

Section 4.1 will deal with calculus $\dot{\vdash}^{d}_{eBA}$, and calculus $\dot{\vdash}^{q}_{eBA}$ is the subject of Section 4.2, Section 4.3 will treat calculus $\dot{\vdash}^{u}_{eBA}$.

4.1 Admissible dBEs

Let Δ range over BEs (i.e. $p \simeq q$ for $p, q \in T$ or $p, q \in FT_m$ of some $m \in Nat$), γ range over sets of BEs, $\mathbf{A}$ be an eBA, $\mathcal{K}$ be a class of eBAs, and Γ be a set of dBEs. Thus,

Definition 4.1.1 (admissible dependent satisfaction $\dot{\vdash}^{d}_{eBA}$):

(a) $\mathbf{A}\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$ *(or* $\mathbf{A}\dot{\models}_{eBA}\gamma \mapsto \Delta$ *even* $\mathbf{A}\dot{\models}\gamma \mapsto \Delta$*) iff* $\mathcal{A}[\![\gamma]\!]_{<\rho_1,\varphi_1>}$ *for every admissible* $< \rho_1, \varphi_1 >$ *implies* $\mathcal{A}[\![\Delta]\!]_{<\rho_2,\varphi_2>}$ *for each admissible* $< \rho_2, \varphi_2 >$,

where $\mathcal{A}[\![p \simeq q]\!]_{<\rho,\varphi>}$ *means* $\mathcal{A}[\![p]\!]_{<\rho,\varphi>} = \mathcal{A}[\![q]\!]_{<\rho,\varphi>}$ *for admissible* $< \rho, \varphi >$ *and* $\mathcal{A}[\![\gamma]\!]_{<\rho,\varphi>}$ *is its natural extension to a collection of admissible BEs;*

(b) $\mathbf{A}\dot{\models}^{d}_{eBA}\Gamma$ *(or* $\mathbf{A}\dot{\models}_{eBA}\Gamma$ *even* $\mathbf{A}\dot{\models}\Gamma$*) iff* $\mathbf{A}\dot{\models}\gamma \mapsto \Delta$ *for each* $\gamma \mapsto \Delta \in \dot{\Gamma}$;

(c) $\mathcal{K}\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$ *(or* $\mathcal{K}\dot{\models}_{eBA}\gamma \mapsto \Delta$ *even* $\mathcal{K}\dot{\models}\gamma \mapsto \Delta$*) iff* $\mathbf{A}\dot{\models}\gamma \mapsto \Delta$ *for every* $\mathbf{A} \in \mathcal{K}$;

(d) $\mathcal{K}\dot{\models}^{d}_{eBA}\Gamma$ *(or* $\mathcal{K}\dot{\models}_{eBA}\Gamma$ *even* $\mathcal{K}\dot{\models}\Gamma$*) iff* $\mathbf{A}\dot{\models}\Gamma$ *for each* $\mathbf{A} \in \mathcal{K}$;

(e) $\Gamma\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$ *(or*$\Gamma\dot{\models}_{eBA}\gamma \mapsto \Delta$ *even* $\Gamma\dot{\models}\gamma \mapsto \Delta$*) iff for every eBA* $\mathbf{A}$, $\mathbf{A}\dot{\models}\Gamma$ *implies* $\mathbf{A}\dot{\models}\gamma \mapsto \Delta$.

The above items from (a) to (e) are obvious perhaps except item (e). However, item (e) is equivalent to $Adm_{\Sigma Bo}(\Gamma)\dot{\models}\gamma \mapsto \Delta$ where $\mathbf{A} \in Adm_{\Sigma Bo}(\Gamma)$ iff $\mathbf{A}\dot{\models}\Gamma$. Sometimes, we write $Mod(\Gamma)$ and $Adm(\Gamma)$ for $Mod_{\Sigma Bo}(\Gamma)$ and $Adm_{\Sigma Bo}(\Gamma)$ respectively.

From Theorem 3.9.6, we have that $\mathcal{K}\dot{\models}p \simeq q$ iff $< \bullet_{[p]}, \bullet_{[q]} > \in \dot{\vartheta}_{\mathcal{K}}$ iff $\mathbf{T}/\dot{\vartheta}_{\mathcal{K}}\dot{\models}p \simeq q$. Since admissible dBEs are close to admissible BEs, we can get the following theorem, which corresponds to Corollary 2.3.3.

Theorem 4.1.2 (admissible Birkhoff's Theorem for dBEs): *For the following four statments, we have that*

(a) 1 and 2 are equivalent to each other,

(b) 3 and 4 are equivalent to each other, and

(c) 2 implies 3:

1. $Adm(\Gamma)\dot{\models}\gamma \mapsto \Delta$;

2. *For each* $\mathbf{A} \in Adm(\Gamma)$, $\gamma \subseteq \dot{\vartheta}_{\mathbf{A}}$ *implies* $\Delta \in \dot{\vartheta}_{\mathbf{A}}$, *where* $\dot{\vartheta}_{\mathbf{A}}$ *is the admissible binding kernel (with respect to* Γ*), i.e.* $\dot{\vartheta}_{\mathbf{A}} = \bigcap_{\zeta:\mathbf{T}\dot{\to}\mathbf{A}} Ker(\zeta)$ *and* $\mathbf{A}\dot{\models}\Gamma$;

3. $\gamma \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *implies* $\Delta \in \dot{\vartheta}_{Adm(\Gamma)}$, *where* $\dot{\vartheta}_{\mathcal{K}} = \bigcap_{\mathbf{A}\in\mathcal{K}} \dot{\vartheta}_{\mathbf{A}}$;

4. $\mathbf{T}/\dot{\vartheta}_{Adm(\Gamma)} \models \gamma \mapsto \Delta$.

Now, we introduce calculus $\dot{\vdash}^{d}_{eBA}$ for dBEs below.

Definition 4.1.3 (calculus $\dot{\vdash}^{d}_{eBA}$): *Calculus $\dot{\vdash}^{d}_{eBA}$ (or $\dot{\vdash}_{eBA}$ even $\dot{\vdash}$) is defined in the judgement form of*

$$\Gamma\dot{\vdash}^{d}_{eBA}\gamma \mapsto \Delta \text{ or } \Gamma\dot{\vdash}\gamma \mapsto \Delta,$$

where Γ is a set of dBEs, γ is a set of BEs and Δ is a BE. It is almost same as either $\dot{\vdash}_{eBA}$ or $\vdash_{dEQ}$ except that

(a) BEs will be presented in form of $\emptyset \mapsto p \simeq q$ instead of in form of $p \simeq q$ as in $\dot{\vdash}_{eBA}$,

(b) $\dot{\vdash}^{d}_{eBA}$ has three extra rules (α), (ξ) and (ξ^{-1}) along with two obvious compositional rules (one is for function variables and the other is for Binding Operators) if we compare them with the ones in $\vdash_{dEQ}$, and

(c) it replaces t, u, v in $\vdash_{dEQ}$ by p, q, r respectively as well.

Note that rules (ξ) and (ξ^{-1}) can be presented in forms of dBEs as follows.

1. (d-ξ)

$$\dot{\vdash}\{t \simeq u\} \mapsto \langle \vec{x} : t\rangle \simeq \langle \vec{x} : u\rangle$$

2. (d-ξ^{-1})

$$\dot{\vdash}\{\langle \vec{z} : u\rangle \simeq \langle \vec{z'} : u'\rangle\} \mapsto u\,[\vec{z} := \vec{y}] \simeq u'\,[\vec{z'} := \vec{y}]$$

where $\{\vec{y}\} \subseteq V$, $\{\vec{y}\} \cap ((Free(u) - \{\vec{z}\}) \cup (Free(u) - \{\vec{z}\})) = \emptyset$.

Similar to the proof of Theorem 2.3.7, we have Admissible Completeness for dBEs. That is,

Theorem 4.1.4 (soundness and completeness of $\dot{\vdash}^{d}_{eBA}$): *$\Gamma\dot{\vdash}\gamma \mapsto \Delta$ iff $\Gamma\dot{\models}\gamma \mapsto \Delta$.*

The proof is similar to Theorem 2.3.7 and is left out. But we point out the key point of the proof. The key point is that $\Gamma\dot{\models}\gamma \mapsto p \simeq q$ iff for every $\mathbf{A} \in Adm(\Gamma)$, if $\mathbf{A}\dot{\models}\gamma$ then $\mathbf{A}\dot{\models}p \simeq q$.

4.2 Admissible qBEs

Let $\mathbf{A}$ be an eBA, $\mathcal{K}$ be a class of eBAs, Γ be a set of qBEs, γ be a set of BEs, and Δ be a BE. Thus, we have admissible quasi-dependent satisfactions in the following.

Definition 4.2.1 (admissible quasi-dependent satisfaction $\dot{\models}^{q}_{eBA}$):

(i) $(\mathbf{A}, < \rho, \varphi >) \dot{\models}^{q}_{eBA} \gamma \hookrightarrow \Delta$ *(or* $(\mathbf{A}, < \rho, \varphi >) \dot{\models}_{eBA} \gamma \hookrightarrow \Delta$ *even* $(\mathbf{A}, < \rho, \varphi >) \dot{\models} \gamma \hookrightarrow \Delta$*) iff* $\mathcal{A}[\![\gamma]\!]_{<\rho,\varphi>}$ *implies* $\mathcal{A}[\![\Delta]\!]_{<\rho,\varphi>}$ *for an admissible* $< \rho, \varphi >$*;*

(ii) $\mathbf{A} \dot{\models}^{q}_{eBA} \gamma \hookrightarrow \Delta$ *(or* $\mathbf{A} \dot{\models}_{eBA} \gamma \hookrightarrow \Delta$ *even* $\mathbf{A} \dot{\models} \gamma \hookrightarrow \Delta$*) iff* $(\mathbf{A}, < \rho, \varphi >) \dot{\models} \gamma \hookrightarrow \Delta$ *for every admissible* $< \rho, \varphi >$*.*

(iii) $\mathbf{A} \dot{\models}^{q}_{eBA} \Gamma$ *(or* $\mathbf{A} \dot{\models}_{eBA} \Gamma$ *even* $\mathbf{A} \dot{\models} \Gamma$*) iff* $\mathbf{A} \dot{\models} \gamma \hookrightarrow \Delta$ *for each* $\gamma \hookrightarrow \Delta \in \Gamma$*;*

(iv) $\mathcal{K} \dot{\models}^{q}_{eBA} \gamma \hookrightarrow \Delta$ *(or* $\mathcal{K} \dot{\models}_{eBA} \gamma \hookrightarrow \Delta$ *even* $\mathcal{K} \dot{\models} \gamma \hookrightarrow \Delta$*) iff* $\mathbf{A} \dot{\models} \gamma \hookrightarrow \Delta$ *for every* $\mathbf{A} \in \mathcal{K}$*;*

(v) $\mathcal{K} \dot{\models}^{q}_{eBA} \Gamma$ *(or* $\mathcal{K} \dot{\models}_{eBA} \Gamma$ *even* $\mathcal{K} \dot{\models} \Gamma$*) iff* $\mathbf{A} \dot{\models} \Gamma$ *for each* $\mathbf{A} \in \mathcal{K}$*;*

(vi) $\Gamma \dot{\models}^{q}_{eBA} \gamma \hookrightarrow \Delta$ *(or* $\Gamma \dot{\models}_{eBA} \gamma \hookrightarrow \Delta$ *even* $\Gamma \dot{\models} \gamma \hookrightarrow \Delta$*) iff for every eBA* $\mathbf{A}$*,* $\mathbf{A} \dot{\models} \Gamma$ *implies* $\mathbf{A} \dot{\models} \gamma \mapsto \Delta$*.*

Analogous to Lemma 2.4.3, we have a theorem for admissible qBEs as follows. Let Γ be a set of dBEs and qBEs, and $Adm(\Gamma)$ be the class of eBAs admissibly satisfying Γ.

Theorem 4.2.2 (admissible Birkhoff's Theorem for qBEs): *The two statements below are equivalent:*

1. $\mathbf{T}/\dot{\vartheta}_{Adm(\Gamma)} \dot{\models} \gamma \hookrightarrow \Delta$;

2. *for every* $\check{\zeta} : \mathbf{T} \to \mathbf{T}$*, if* $\check{\zeta}(\gamma) \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *then* $\check{\zeta}(\Delta) \in \dot{\vartheta}_{Adm(\Gamma)}$*.*

Similar to Lemma 2.4.4, there is an implication for admissible qBEs. Formally,

Lemma 4.2.3 (admissible substitutions and qBEs): $Adm(\Gamma) \dot{\models} \gamma \hookrightarrow \Delta$ *implies that for every admissible* $\check{\zeta} : \mathbf{T} \to \mathbf{T}$ *if* $\check{\zeta}(\gamma) \subseteq \vartheta_{Adm(\Gamma)}$ *then* $\check{\zeta}(\Delta) \in \vartheta_{Adm(\Gamma)}$*.*

Further along, by skolemization we obtain the reversed implication (see Theorem 2.4.7).

Definition 4.2.4 (calculus $\dot{\vdash}^{q}_{eBA}$): *Calculus $\dot{\vdash}^{q}_{eBA}$ is defined in the judgement form of*

$$\Gamma\dot{\vdash}^{q}_{eBA}\gamma \hookrightarrow \Delta \text{ and } \Gamma\dot{\vdash}^{q}_{eBA}\gamma \mapsto \Delta$$

or

$$\Gamma\dot{\vdash}\gamma \hookrightarrow \Delta \text{ and } \Gamma\dot{\vdash}\gamma \mapsto \Delta$$

which is almost the same as either $\vdash_{qEQ}$ or $\dot{\vdash}^{d}_{eBA}$ except that

(a) $\dot{\vdash}^{q}_{eBA}$ contains all rules in $\dot{\vdash}^{d}_{eBA}$,

(b) it has two compositional rules (one is for function variables and the other is for Binding Operators) instead of one as in $\vdash_{qEQ}$,

(c) t, u, v in $\vdash_{qEQ}$ are replaced by p, q, r respectively in $\dot{\vdash}^{q}_{eBA}$, and

(d) $\dot{\vdash}^{q}_{eBA}$ has one extra rule (q-ξ^{-1}) below:

$$(q\text{-}\xi^{-1}) \quad \dot{\vdash}^{q}_{eBA}\{\langle\vec{z} : u\rangle \simeq \langle\vec{z'} : u'\rangle\} \hookrightarrow u[\vec{z} := \vec{y}] \simeq u'[\vec{z'} := \vec{y}]$$

where $\{\vec{y}\} \subseteq V$ and $\{\vec{y}\} \cap ((Free(u) - \{\vec{z}\}) \cup (Free(u) - \{\vec{z}\})) = \emptyset$.

About calculi $\dot{\vdash}^{d}_{eBA}$ and $\dot{\vdash}^{q}_{eBA}$, we have a remark below.

Remark 4.2.5 ((q-ξ) and (d-ξ)): *We do not have a rule like (q-ξ) in $\dot{\vdash}^{q}_{eBA}$,*

$$(q\text{-}\xi) \quad \Gamma\dot{\vdash}\{t \simeq u\} \hookrightarrow \langle\vec{x} : t\rangle \simeq \langle\vec{x} : u\rangle$$

where $t, u \in T$ and $\{\vec{x}\} \subseteq V$. But we do have a rule like (d-ξ) in $\dot{\vdash}^{q}_{eBA}$. The reason for these is simple, i.e. rule (q-ξ) is unsound in general. A counter-example to its soundness is

$$x \simeq \mathbf{true}(y) \hookrightarrow \langle x, y : x\rangle \simeq \langle x, y : \mathbf{true}(y)\rangle$$

in Example 2.4.5.

Similar to Theorem 2.4.9, we can get that $\dot{\vdash}^{q}_{eBA}$ is sound and complete with respect to admissible BEs. With (d-intr) rule, we can extend the soundness and

completeness to include admissible dBEs. Further with (q-skm) rule, we would have the soundness and completeness for admissible qBEs. That is,

Theorem 4.2.6 (soundness and completeness of $\dot{\vdash}^{q}_{eBA}$): *Given* Γ, $\Gamma\dot{\vdash}^{q}_{eBA}\gamma \hookrightarrow \Delta$ *iff* $\Gamma\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$.

The proof is similar to the one for Theorem 2.4.9 and is left out.

4.3 Admissible uBEs

Similar to Section 2.5, a deduction system of the form

$$\{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$$

is investigated in this section, which is referred to as an admissible universal BE (admissible uBEs). Sometimes, we refer it simply as Ω.

An eBA $\mathbf{A}$ is said to be a *model* of Ω, written as $\mathbf{A}\dot{\models}^{u}_{eBA}\Omega$ (or $\mathbf{A}\dot{\models}_{eBA}\Omega$ even $\mathbf{A}\dot{\models}\Omega$) iff $\mathbf{A}\dot{\models}^{q}_{eBA}\{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\}$ implies $\mathbf{A}\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$. So, when $M = \emptyset$, Ω is an (pure) admissible qBEs, further if $\gamma = \emptyset$, it is an admissible BE; when all γ's are empty, it is an admissible dBE.

Analogous to Section 2.5, we can naturally define the followings:

1. $\mathcal{K}\dot{\models}^{u}_{eBA}\Omega$ (or $\mathcal{K}\dot{\models}_{eBA}\Omega$ even $\mathcal{K}\dot{\models}\Omega$) is the natural extension of $\mathbf{A}\dot{\models}\Omega$ to a class $\mathcal{K}$ of eBAs;

2. $\mathbf{A}\dot{\models}^{u}_{eBA}\Gamma$ (or $\mathbf{A}\dot{\models}_{eBA}\Gamma$ even $\mathbf{A}\dot{\models}\Gamma$) is $\mathbf{A}\dot{\models}\Omega$ for each $\Omega \in \Gamma$, which is a set of (admissible) uBEs.

3. $\mathcal{K}\dot{\models}^{u}_{eBA}\Gamma$ (or $\mathcal{K}\dot{\models}_{eBA}\Gamma$ even $\mathcal{K}\dot{\models}\Gamma$) is the natural extension of $\mathbf{A}\dot{\models}\Gamma$ to a collection Γ of admissible uBEs.

4. $\Gamma\dot{\models}^{u}_{eBA}\Omega$ (or $\Gamma\dot{\models}_{eBA}\Omega$ even $\Gamma\dot{\models}\Omega$) iff for every eBA $\mathbf{A}$, $\mathbf{A}\dot{\models}\Gamma$ implies $\mathbf{A}\dot{\models}\Omega$.

Let $Adm(\Gamma)$ be the class of eBAs admissibly satisfying Γ, which is a set of uBEs.

Note that the last one of the above definitions of admissible universal satisfaction is equivalent to $Adm(\Gamma) \models \Omega$.

We proceed analogous to Theorem 2.5.1 and have a theorem for admissible uBEs below.

Theorem 4.3.1 (admissible Birkhoff's theorem for uBEs): *The following two statement are equivalent:*

1. $\mathbf{T}/\dot{\vartheta}_{Adm(\Gamma)} \dot{\models} \Omega$;

2. *whenever for every* $\check{\zeta}_1 : \mathbf{T} \to \mathbf{T}$, $\check{\zeta}_1(\gamma^{(m)}) \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *implies* $\check{\zeta}_1(\Delta^{(m)}) \in \dot{\vartheta}_{Adm(\Gamma)}$ *for each* $m \in M$, *we would have that for each* $\check{\zeta}_2 : \mathbf{T} \to \mathbf{T}$, $\check{\zeta}_2(\gamma) \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *implies* $\check{\zeta}_2(\Delta) \in \dot{\vartheta}_{Adm(\Gamma)}$.

Similar to Lemma 2.5.2, we have the following.

Lemma 4.3.2 (admissible universal equality): *If* $Adm(\Gamma) \dot{\models} \Omega$, *then whenever for every* $\check{\zeta}_1 : \mathbf{T} \to \mathbf{T}$, $\check{\zeta}_1(\gamma^{(m)}) \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *implies* $\check{\zeta}_1(\Delta^{(m)}) \in \dot{\vartheta}_{Adm(\Gamma)}$ *for each* $m \in M$, *we would have that for each* $\check{\zeta}_2 : \mathbf{T} \to \mathbf{T}$, $\check{\zeta}_2(\gamma) \subseteq \dot{\vartheta}_{Adm(\Gamma)}$ *implies* $\check{\zeta}_2(\Delta) \in \dot{\vartheta}_{Adm(\Gamma)}$.

Similar to Theorem 2.5.3, we have a theorem below.

Theorem 4.3.3 (admissible uBEs and admissible qBEs):

$\Gamma \dot{\models} \Omega$ *iff* $\mathbf{A} \dot{\models} \gamma^{(m)} \hookrightarrow \Delta^{(m)}$ *for each* $m \in M$ *implies* $\mathbf{A} \dot{\models} \gamma \to \Delta$ *for every* $\mathbf{A} \models \Gamma$, *where the form of* $\gamma \hookrightarrow \Delta$ *is short for the form of* $\emptyset \mapsto (\gamma \hookrightarrow \Delta)$.

From the above, we can introduce calculus $\dot{\vdash}^{u}_{eBA}$ as follows.

Definition 4.3.4 (calculus $\dot{\vdash}^{u}_{eBA}$): *Calculus* $\dot{\vdash}^{u}_{eBA}$ *is defined in the judgement form of*

$$\Gamma \dot{\vdash}^{u}_{eBA} \Omega \text{ or } \Gamma \dot{\vdash} \Omega$$

which is almost the same as either $\vdash_{uEQ}$ *or* $\dot{\vdash}_{eBA}$ *except of the following:*

(a) t, u, v *in* $\vdash_{uEQ}$ *are replaced by* p, q, r *respectively in* $\dot{\vdash}^{u}_{eBA}$ *like in* $\dot{\vdash}_{eBA}$,

(b) $\dot{\vdash}^{u}_{eBA}$ *has two composition rules (one if for function variables and the other is for Binding Operators) like in* $\dot{\vdash}_{eBA}$ *instead of one as in* $\vdash_{uEQ}$,

(c) it contains three extra rules of (α), (ξ) *and* (ξ^{-1}). *like in* $\dot{\vdash}_{eBA}$ *compared with* $\dot{\vdash}_{uEQ}$.

Theorem 4.3.5 (soundness and completeness of $\dot{\vdash}^{u}_{eBA}$**):** $\Gamma\dot{\vdash}\Omega$ *iff* $\Gamma\dot{\models}\Omega$.

The proof is similar to Theorem 2.5.5 and is left out.

Part III

Friedman's Approach

In the end of Subsection 1.2.2, we have given a clue of how to achieve the equational completeness $\vdash_{iBA}$, i.e. reducing it to the equational completeness of $\vdash_{EQ}$. So, we are going along this idea. But, first of all, we have to verify the well-definedness of interpretations in iBAs (Section 5.2). Obviously, substitutions have to be tackled before it (Section 5.1). Then, we give a definition for the counterpart of iBAs in many-sorted first order algebras, called b-clones (Section 5.3). The intended reductions is substantiated by discovering two translations, $tr_{\vec{x}}[\![\bullet]\!]$ from binding terms to terms of b-clones (Definition 5.3.5) and $\underline{tr}[\![\bullet]\!]$ from terms of b-clones to binding terms (Definition 5.3.8). Then, we are going to show that

(5.a) for binding terms p and q, $\Gamma \vdash_{iBA} p \simeq q$ implies $tr_{\vec{x}}[\![\Gamma]\!] \cup Ax_b \vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$, where Ax_b is the set of axioms for b-clones, and $(Free(\Gamma) \cup Free(p) \cup Free(q)) \cap V \subseteq \{\vec{x}\}$ (see Theorem 5.5.3.13);

(5.b) for terms $\underline{t}$ and $\underline{u}$ of b-clones, $\underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{t} \simeq \underline{u}$ implies $\underline{tr}[\![\underline{\Gamma}]\!] \vdash_{iBA} \underline{tr}[\![\underline{t}]\!] \simeq \underline{tr}[\![\underline{u}]\!]$ (see Theorem 5.5.2.12);

(5.c) for binding term p, $\vdash_{iBA} p \simeq \underline{tr}[\![tr_{\vec{x}}[\![p]\!]\,]\!]$ where $Free(p) \cap V \subseteq \{\vec{x}\}$ (see Lemma 5.6.2.1);

(5.d) calculus $\vdash_{iBA}$ is sound and complete as a result from (5.a) to (5.c) (see Theorem 5.6.1.1 and Theorem 5.6.2.2).

Next, we extend the results to include dependent Binding Equations (dBEs) in Section 6.1, and quasi-dependent Binding Equations (qBEs) in Section 6.2 together with the extension to the universal Binding Equations (uBEs) in Section 6.3.

Chapter 5

Friedman's Approach

In Chapter 1, Definition 1.2.2.2 gives interpretations in an iBA. But it ran short of an argument for their well-definedness. It is our intention to do so in Section 5.1 and Section 5.2. Then, we follow the idea of exploiting the intuition of binding pre-algebras (see comments before Definition 1.2.2.1) and introduce b-clones formally in Definition 5.3.2. It is followed by discovering syntactic connections between binding terms and terms of b-clones. The connections are two translations $tr_{\vec{x}}$ and $\underline{tr}$. These are also arranged in Section 5.3. Thirdly, the preservations of derivations between $\vdash_{eBA}$ and $\vdash_{EQ}$ with Ax_b by the translations are shown in Section 5.4 (Theorem 5.4.3.2 and Theorem 5.4.3.6) and in Section 5.5 (Theorem 5.5.2.12 and Theorem 5.5.3.13). Finally, the soundness and the completeness of $\vdash_{iBA}$ are achieved in Section 5.6 (Theorem 5.6.1.1 and Theorem 5.6.2.2).

5.0 Relationship between eBAs and iBAs

Our intention of this section is to formally confirm that the relationship between eBAs and iBAs is *Extensionality*. For this purpose, we first introduce *Extensionality* in an iBA $\mathbf{B}$ as follows,

Definition 5.0.1 (Extensionality in iBAs): *Let* $\mathbf{B}$ *be an iBA, i.e. a pair of* $< \vec{B}, \mathcal{B} >$ *such that* $\vec{B} = \{B_m | m \geq 0\}$ *and* $\mathcal{B} = \{pr^{\mathbf{B}}_{k,i}, \circ^{\mathbf{B}}_{k,j}, \sigma^{\mathbf{B}}_{k} | k, j \geq 0 \text{ and } \sigma \in$

$\Sigma_{<\vec{m},n>}$ *and* $1 \leq i \leq k\}$. *Thus,* **B** *is said to be* extensional *if for* $k \geq 0$, *given* $g, h \in B_k$ *and if for all* $b_j \in B_0$, *we have* $go_{k,0} < \vec{b} >= ho_{k,0} < \vec{b} >$, *then* $g = h$.

On the other hand, suppose **A** is an eBA, i.e. it is a pair of $< \mathcal{F}^{\mathbf{A}}, \mathcal{A} >$ such that $\mathcal{F}^{\mathbf{A}} =< A, \{\mathcal{F}_m | m \in Nat\} >$. We understand that A is equivalent to $\mathcal{F}_0$ in the sense that whenever there is an element a in A, correspondingly there is an element $C_{0,a}$ in $\mathcal{F}_0$ and vise verse. Consequently, we can identify these two. Furthermore, we can obtain an eBA from an extensional iBA.

Theorem 5.0.2 (from iBAs to eBAs): *Suppose* **B** *is an iBA with Extensionality. Let* A *be* B_0 *and* $\mathcal{F}_m(A)$ *be* $\{g : A^m \rightarrow A \mid$ *there exists* $h \in B_m$ *such that* $g(\vec{a}) = ho_{m,0} < \vec{a} >$ *for every* $\vec{a} \in A^m\}$ *for each* $m \geq 0$. *Then*

1. $\mathcal{F}^{\mathbf{A}} =< A, \mathcal{F} >$ *is explicitly closed,*

2. $\mathbf{A}_\sigma = \mathbf{B}_{\sigma_0}$ *is uniform over* $\mathcal{F}^{\mathbf{A}}$, *and*

3. **A** *is an eBA.*

Proof

- For *1*, we would have that

  - for $m \geq 0$ and for each $a \in A$, $C_{m,a}$ corresponds to $ao_{0,m} <>$ and it is in $\mathcal{F}_m$.
  - for $m > 0$ and $1 \leq i \leq m$, $\pi_{m,i}$ corresponds to $pr_{m,i}$ and it is in $\mathcal{F}_m$.
  - for $m > 0$, $k \geq 0$, given $h \in \mathcal{F}_m$ and $g_i \in \mathcal{F}_k$, the function defined in (iBA-right-proj) is $ho_{m,k} < \vec{g} >$, i.e. $o_{m,k}$ corresponds to $\odot$ because of (iBA-assoc).

- For *2*, we can get for $k \geq 0$, given $g_i \in \mathcal{F}_{k+m_i}$ and $h_j \in \mathcal{F}_k$, the function defined through (iBA-unif) is $\mathbf{B}_{\sigma_k}(\vec{g}, \vec{h})$.

  To see the uniformity more clearly, we give some details:

(a) $curry_{k,m}(g)(\vec{a})$ can be defined as $g\odot < C_{m,\vec{a}}, \pi_{m,1}, \pi_{m,2}, ..., \pi_{m,m} >$, where $C_{m,\vec{a}}$ is the list of $C_{m,a_1}, C_{m,a_2}, ..., C_{m,a_k}$; and this function corresponds to $g\circ_{k+m,m} < \vec{a}\circ_{0,m} <>, Pr^m_{1,m} >$.

(b)

$$\begin{aligned}
&\mathbf{A}_\sigma\odot < (curry_{k,\vec{m}} \odot \Pi^{\ell+n}_{1,\ell}), \Pi^{\ell+n}_{\ell+1,\ell+n} > \\
&= \mathbf{B}_{\sigma_0}\odot < (curry_{k,\vec{m}} \odot \Pi^{\ell+n}_{1,\ell}), \Pi^{\ell+n}_{\ell+1,\ell+n} > \\
&= \mathbf{B}_{\sigma_k},
\end{aligned}$$

where Π_i is a projection functional such that

$\Pi^{\ell+n}_i(g'_1, ..., g'_{\ell+n}) = g'_i \ (1 \leq i \leq \ell + n)$.

- For *3*, it follows straight from the above two. □

An iBA can be viewed as an eBA if this iBA is extensional. Conversely, an eBA can be thought as an extensional iBA. Subsequently,

Theorem 5.0.3 (from eBAs to iBAs): *Suppose* $\mathbf{A} =< \mathcal{F}^{\mathbf{A}}, \mathcal{A} >$ *is an eBA such that* $\mathcal{F}^{\mathbf{A}} =< A, \{\mathcal{F}^{\mathbf{A}}_m | m \in Nat\} >$. *And, let* $\mathbf{B} =< \vec{B}, \mathcal{B} >$ *such that* $\vec{B} = \{B_m | m \in Nat\}$ *and* $B_m = \mathcal{F}^{\mathbf{A}}_m$ *for* $m \in Nat$ *and* $\mathcal{B} = \{\pi^{\mathbf{A}}_{k,i}, \odot_{m,k}, \mathbf{A}_\sigma\odot < (curry_{k,\vec{m}}\odot \Pi^{\ell+n}_{1,\ell}), \Pi^{\ell+n}_{\ell+1,\ell+n} > | k, m \geq 0$ *and* $1 \leq i \leq k$ *and* $\sigma \in \Sigma_{<\vec{m},n>}$ *and* $\ell = |\vec{m}|\}$. *Thus,* $\mathbf{B}$ *is an extensional iBA.*

Proof

We know that

(i) B_m is $\mathcal{F}_m$ for $m \geq 0$;

(ii) $pr_{k,i}$ is $\pi_{k,i}$ for $1 \leq i \leq k$;

(iii.a) $\circ_{m,k}$ is the composition functional $\odot_{m,k}$ for $m, k > 0$ such that

(iii.b) $g\circ_{0,k} <>=_{df} C_{k,g()}$ if $m = 0$, and

(iii.c) $g\circ_{m,0} < C_{0,\vec{a}} >=_{df} C_{0,g(\vec{a})}$ if $k = 0$;

(iv.a) $\mathbf{B}_{\sigma_0} = \mathbf{A}_\sigma$ when $k = 0$; and

(iv.b) $\mathbf{B}_{\sigma_k} = \mathbf{A}_\sigma\odot < (curry_{k,\vec{m}} \odot \Pi^{\ell+n}_{1,\ell}), \Pi^{\ell+n}_{\ell+1,\ell+n} >$ when $k > 0$;

(v) The rest staff is to check whether the above defined **B** satisfies those equations in Definition 1.2.2.1. □

We conclude that

Corollary 5.0.4 (eBAs and iBAs): *An extensional iBA is equivalent to an eBA.*

Although binding primitives are not explicitly included in signature Σ^{BO}, they are implicitly implied by the indices $< \vec{m}, n >$ of Σ^{BO}. The role of binding (in eBAs) can be viewed as coding a graph into an function object, and interpretations of BOs (in eBAs) can be viewed as having ability of decoding an function object into a graph. The reason we can study BOs in such an algebraic way (i.e. a generality with hidden coding and decoding for function objects) in this thesis is that we have the work of [40,119,134,141,47] and many others.

From the above results, we understand either

(a) that given an extensional iBA **B**, we can obtain an eBA **A'**; or

(b) that given an eBA **A**, we can get an extensional iBA **B'**.

However, we have not checked whether or not these are enough to guarantee the equivalence between interpretations of **B** and ones of **A'** (or between interpretations of **B'** and ones of **A**), i.e. either

(a') $\mathcal{B}_{\vec{x}}[\![t]\!]_{\psi} = \mathcal{A}'[\![(\vec{x} : t)]\!](\rho, \varphi)$ for $\{\vec{x}\} \supseteq (Free(t) \cap V)$ and $\varphi = \psi$; or

(b') $\mathcal{B}'_{\vec{x}}[\![t]\!]_{\psi} = \mathcal{A}[\![(\vec{x} : t)]\!](\rho, \varphi)$ for $\{\vec{x}\} \supseteq (Free(t) \cap V)$ and $\varphi = \psi$.

In other words, whether there is a consistency between interpretations of iBAs (Definition 1.2.2.2) and interpretations of eBAs (Definition 1.2.1.4). This consistency is ensured either by

(a") for $\{\vec{x}\} \supseteq (Free(t) \cap V)$ and $\{\vec{x}\} \supseteq (Free(\langle \vec{y} : t \rangle) \cap V)$,

(a".i) $\mathcal{A}'[\![t]\!](\rho[\vec{a}/\vec{x}], \varphi) = \mathcal{B}_{\vec{x}}[\![t]\!]_{\psi} \circ < \vec{a'} >$

where $\vec{a'}(i) = \vec{a}(i) \circ_{0,0} <>$,

(a".ii) $\mathcal{A}'[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{x}], \varphi) = \mathcal{B}_{\vec{x}}[\![\langle \vec{y} : t\rangle]\!]_{\psi} \circ < \vec{a''}, Pr^{|\vec{y}|}_{1,|\vec{y}|} >$

where $\vec{a''}(i) = \vec{a}(i) \circ_{0,|\vec{y}|} <>$; or by

(b") for $\{\vec{x}\} \supseteq (Free(t) \cap V)$ and $\{\vec{x}\} \supseteq (Free(\langle \vec{y} : t\rangle) \cap V)$,

(b".i) $\mathcal{A}[\![t]\!](\rho[\vec{a}/\vec{x}], \varphi) = \mathcal{B}'_{\vec{x}}[\![t]\!]_{\psi} \circ < \vec{a'} >$

where $\vec{a'}(i) = \vec{a}(i) \circ_{0,0} <>$,

(b".ii) $\mathcal{A}[\![\langle \vec{y} : t\rangle]\!](\rho[\vec{a}/\vec{x}], \varphi) = \mathcal{B}'_{\vec{x}}[\![\langle \vec{y} : t\rangle]\!]_{\psi} \circ < \vec{a''}, Pr^{|\vec{y}|}_{1,|\vec{y}|} >$

where $\vec{a''}(i) = \vec{a}(i) \circ_{0,|\vec{y}|} <>$.

We state the consistency as a theorem below and leave the actual proof to the interested readers.

Theorem 5.0.5 (consistency of interpretations): *Interpretations of iBAs and interpretations of eBAs are consistent with each other.*

Now, we should return from our digression and back to the equality of iBAs.

5.1 Substitutions

To get the well-definedness of interpretations, substitutions of binding terms are unavoidable. But if you have run across Lambda Calculus before and assume the well-definedness of interpretations, you can skip off this section and the next. In case of any doubt in future, you can consult them here later accordingly.

Following [146], we define simultaneous substitution as follows.

Definition 5.1.1 (family of substitution functions $\vec{\varrho}$): *Let $\vec{\varrho}$ be a family of ϱ and $\{\varrho_m \mid m \in Nat\}$ such that $\varrho : V \to T$ and $\varrho_m : FV_m \to FV_m \cup FT_m$ (since we do not have $FV_m \subseteq FT_m$ $(m \geq 0)$),*

1. $\vec{\varrho}[x] =_{df} \varrho(x)$ *for* $x \in V$,

2. $\vec{\varrho}[f] =_{df} \varrho_m(f)$ *for* $f \in FV_m (m \geq 0)$,

3. $\vec{\varrho}[x := t][\bullet] =_{df} \begin{cases} t & if\ x = \bullet \\ \vec{\varrho}[\bullet] & if\ x \neq \bullet \end{cases}$,

   where $x \in V$, $t \in T$ *and* $\bullet \in V \cup (\bigcup_{m \in Nat} FV_m)$.

4. $\vec{\varrho}[f := ft][\bullet] =_{df} \begin{cases} ft & if\ f = \bullet \\ \vec{\varrho}[\bullet] & if\ f \neq \bullet \end{cases}$,

   where $f \in FV_m$ *and* $ft \in FV_m \cup FT_m$ $(m \geq 0)$, $\bullet \in V \cup (\bigcup_{m \in Nat} FV_m)$.

We call $\vec{\varrho}$ a substitution. $\vec{\varrho}$ can be viewed as a product function from $(V \to T) \times (\bigotimes_{m \in Nat} FV_m \to (FV_m \cup FT_m))$, if you wish.

Next, we are dealing with free variables and free function variables in binding terms.

Definition 5.1.2 (free variable *Free*): *We define a function Free on BT inductively as follows:*

1. $Free(x) =_{df} \{x\}$, *where* $x \in V$;

2. $Free(f(\vec{t})) =_{df} \bigcup_{j=1}^{m} Free(t_j) \cup \{f\}$,

   where $f \in FV_m$ *and* $t_j \in T$;

3. $Free(\sigma(\vec{ft}, \vec{t})) =_{df} (\bigcup_{i=1}^{\ell} Free(ft_i)) \cup (\bigcup_{j=1}^{n} Free(t_j))$,

   where $<\vec{m}, n> \in Nat^* \times Nat$ *and* $\sigma \in \Sigma_{<\vec{m},n>}$ *and* $|\vec{m}| = \ell$, $ft_i \in FT_{m_i}$ $(1 \leq i \leq \ell)$, $t_j \in T$ $(1 \leq j \leq n)$.

4. $Free(ft) =_{df} Free(t) - \{\vec{x}\}$,

   where $\langle \vec{x} : t \rangle = ft \in FT_k$ *(and* $k = |\vec{x}|$*)*.

For function terms, we also need some tools to divide them into two parts. One is the binding variables and the other is the term being bound by those variables. For $m \geq 0$, we define two functions $||_m : FT_m \to T$ and $|_m : FT_m \to V^m$ such

that $\langle \vec{x} : t \rangle||_m =_{df} t$ and $\langle \vec{x} : t \rangle|_m =_{df} \vec{x}$. Therefore, we have that for $m \geq 0$, let $ft \in FT_m$. Then $ft = \langle ft|_m : ft||_m \rangle$.

We consider a substitution as applying a family of substitution functions $\vec{\varrho}$ to a binding term as follows.

Definition 5.1.3 (substitution):

1. $x\vec{\varrho} =_{df} \vec{\varrho}[x]$, *where* $x \in V$.

2. $$f(\vec{t})\vec{\varrho} =_{df} \begin{cases} \vec{\varrho}[f](\vec{t}\,\vec{\varrho}) & \text{if } \vec{\varrho}[f] \in FV_m \\ \vec{\varrho}[f]||_m[\vec{x} := \vec{t}\vec{\varrho}] & \text{if } \vec{\varrho}[f] \notin FV_m \end{cases},$$

   where $\vec{x} = \vec{\varrho}[f]|_m$, $\vec{t}\,\vec{\varrho}$ *is the abbreviation for list* $t_1\vec{\varrho}, t_2\vec{\varrho}, ..., t_m\vec{\varrho}$ *and* $f \in FV_m$.

3. $\sigma(\vec{f}t, \vec{t})\vec{\varrho} =_{df} \sigma(\vec{f}t\vec{\varrho}, \vec{t}\,\vec{\varrho})$,

   where $\vec{f}t\vec{\varrho}$ *is short for list* $ft_1\vec{\varrho}, ft_2\vec{\varrho}, ..., ft_\iota\vec{\varrho}$.

4. $\langle \vec{x} : t \rangle\vec{\varrho} =_{df} \langle \vec{y} : t\vec{\varrho}[\vec{\vec{x}} := \vec{y}] \rangle$,

   where y_j *is the least* $y \in V$ *such that*

$$\forall z \in Free(t).y \notin Free(\vec{\varrho}[z]) \cup \{\vec{y}\lceil_{j-1}\}.$$

Later, we will adopt the convention that omits the family $\vec{\imath}$ of idenity substitution functions in substitutions, e.g we will write $p[\vec{f}, \vec{x} := \vec{f}t, \vec{t}]$ for $p\vec{\imath}[\vec{f}, \vec{x} := \vec{f}t, \vec{t}]$.

Before going further, let us look at some properties of substitutions, which convince ourself that substitutions are well-defined (see Theorem 5.1.7).

Property 5.1.4 (renaming): *Let* $rename(\vec{\varrho}) =_{df} (\forall x \in V.\vec{\varrho}[x] \in V) \wedge (\forall m, \forall f \in FV_m.\vec{\varrho}[f] \in FV_m)$. *Then,*

1. *for all* $\vec{\varrho}$, $rename(\vec{\varrho})$ *implies*

$$(\forall x, y \in V.rename(\vec{\varrho}[x := y])) \wedge (\forall m, \forall f, f' \in FV_m.rename(\vec{\varrho}[f := f']))$$

2. *for all* $\vec{\varrho}$, *we have that* $rename(\vec{\varrho})$ *implies*

$$\forall j, \forall t \in T^{(j)}.t\vec{\varrho} \in T^{(j)} \text{ and } \forall m, \forall ft \in FT_m^{(j)}.ft\vec{\varrho} \in FT_m^{(j)};$$

where $BT^{(j)} =< T^{(j)}, FT^{(j)} >$, $FT^{(j)} = \{FT_m^{(j)} \mid m \in Nat\}$, $T^{(j)}$ *and* $FT^{(j)}$ $(j \in Nat)$ *are defined inductively as follows*[1]

1. (a) $T^{(0)} =_{df} \{x|x \in V\} \cup \{f()|f \in FV_0\} \cup \{\sigma()|\sigma \in \Sigma_{<\epsilon,0>}\}$,

   (b) *for* $|\vec{x}| \geq 0$, $FT^{(0)}_{|\vec{x}|} =_{df} \{\langle \vec{x} : t\rangle | t \in T^{(0)} \wedge x_j \in V \wedge x_i \neq x_j (i \neq j)\}$

2. (a)

$$T^{(j+1)} =_{df} T^{(j)} \cup \bigcup_{|\vec{t}|=0}^{\infty} \{f(\vec{t})|f \in FV_{|\vec{t}|} \wedge t_j \in T^{(j)}\}$$
$$\cup \bigcup_{<\vec{m},n> \in Nat^* \times Nat} \{\sigma(\vec{ft}, \vec{t})|\sigma \in \Sigma_{<\vec{m},n>} \wedge ft_i \in FT^{(j)}_{m_i} \wedge t_j \in T^{(j)}\}$$

   where $|\vec{t}| = n$.

   (b) *for* $n \geq 0$, $FT_n^{(j+1)} =_{df} FT_n^{(j)} \cup \{\langle \vec{x} : t\rangle | t \in T^{(j)} \wedge x_j \in V \wedge |\vec{x}| = n\}$, *where* $\vec{x}(i) \neq \vec{x}(j)$ *if* $i \neq j$.

Proof

For the first case, it is easy and is left out.

For the second case, it can be obtained by induction on j with case analysis. □

This property will be used very frequently, especially in proofs involving inductions. Also we can generalize it to Theorem 5.1.7, which says that applying a substitution to a binding term yields a binding term. But we need something else before we can actually prove the generalized theorem.

Lemma 5.1.5 (substitution and terms):

(a) For $t \in T^{(k)}$, *for all* $\{\vec{x}\} \subseteq V$ *and* $t_j \in T^{(j)}$, *we have* $t[\vec{x} := \vec{t}\,] \in T^{(k+j)}$.

---

[1]Let $T^* =_{df} \bigcup_{j \in Nat} T^{(j)}$, and for $k \geq 0$, $FT_k^* =_{df} \bigcup_{j \in Nat} FT_k^{(j)}$. We would have $T = T^*$ and for $k \geq 0$ $FT_k = FT_k^*$, since the following. Firstly, T^* and FT^* satisfy the four conditions in Definition 1.1.3.1, i.e. $T^* \supseteq T$ and for all $k \geq 0$, $FT_k^* \supseteq FT_k$. Secondly, we have that $\forall j.T^{(j)} \subseteq T$. and for all $k \geq 0$ $\forall j.FT_k^{(j)} \subseteq FT_k$. Then, $T^* \subseteq T$ and for all $k \geq 0$, $FT_k^* \subseteq FT_k$. This property mainly enables us to use induction on binding terms in many proofs to follow.

(b) For $ft \in FT_m^{(k)}$, for all $\{\vec{x}\} \subseteq V$ and $t_j \in T^{(j)}$, we have $ft[\vec{x} := \vec{t}] \in FT^{(k+j)}$.

Proof This is done by induction on k. □

We can sharpen the above result to Lemma 5.1.6, since $t \in T^{(j)}$ implies $t \in T^{(j')}$ and $ft \in FT^{(j)}$ implies $ft \in FT^{(j')}$ where $j \leq j'$. The lemma actually describe the impact of substitutions on binding terms. They will be used to convince us that the substitutions are well-defined, see Theorem 5.1.7.

Lemma 5.1.6 (impact of substitution on terms): *Let ν range over $V \cup FV$.*

(a) For $t \in T^{(k)}$, let $\nu \in Free(t)$ and $\vec{\varrho}[\nu] \in T^{(k_\nu)}$ (or $FT_{n_\nu}^{(k_\nu)}$) for some $k_\nu \in Nat$, we have $t\vec{\varrho} \in T^{(k+1+m)}$, where $m = max\{k_\nu | \nu \in Free(t)\}$.

(b) For $ft \in T^{(k)}$, let $\nu \in Free(ft)$ and $\vec{\varrho}[\nu] \in T^{(k_\nu)}$ (or $FT_{m_\nu}^{(k_\nu)}$) for some $k_\nu \in Nat$, we have $ft\vec{\varrho} \in FT_n^{(k+1+m)}$, where $m = max\{k_\nu | \nu \in Free(ft)\}$.

Proof Induction on k with Lemma 5.1.9 and Lemma 5.1.10. □

Because of the above lemmas, the well-definedness of substitutions follows easily.

Theorem 5.1.7 (well-definedness of substitutions): *For all $\vec{\varrho}$, we have that $\forall t \in T. t\vec{\varrho} \in T$ and $\forall m, \forall ft \in FT_m. ft\vec{\varrho} \in FT_m$.*

Proof It follows from Lemma 5.1.5 and Lemma 5.1.6. □

Since there are so many way to express substitutions which have a same effect on binding terms, it is a good idea to classify them into some equivalent classes. We say that two substitutions are *equivalent*, written as $\vec{\varrho} \asymp \vec{\varrho'}$, iff for all $t \in T$, $t\vec{\varrho} = t\vec{\varrho'}$ and for all $m \geq 0$ and for all $ft \in FT_m$, $ft\vec{\varrho} = ft\vec{\varrho'}$. Below we give some properties of equivalent substitutions.

Lemma 5.1.8 (equivalent sustitutions $\asymp$): *For all $\vec{\varrho}$, we have that*

1. *$(\vec{\varrho}[x := t_1])[x := t_2] \asymp \vec{\varrho}[x := t_2]$, where $x \in V$ and $t_1, t_2 \in T$.*
2. *for all $m \geq 0$, $(\vec{\varrho}[f := ft_1])[f := ft_2] \asymp \vec{\varrho}[f := ft_2]$, where $f \in FV_m$ and $ft_1, ft_2 \in FT_m$.*

3. $(\vec{\varrho}[x := t_1])[y := t_2] \asymp (\vec{\varrho}[y := t_2])[x := t_1]$, *where* $x, y \in V$, $x \neq y$ *and* $t_1, t_2 \in T$.

4. *for all* $m \geq 0$, $(\vec{\varrho}[f := ft_1])[f' := ft_2] \asymp (\vec{\varrho}[f' := ft_2])[f := ft_1]$, *where* $f, f' \in FV_m$, $f \neq f'$ *and* $ft_1, ft_2 \in FT_m$.

5. $(\vec{\varrho}[x := t_1])[f := ft_2] \asymp (\vec{\varrho}[f := ft_2])[x := t_1]$, *where* $x \in V$, $f \in FV_m$, $t_1 \in T$ *and* $ft_2 \in FT_m$.

Proof It follows straight by definition. □

Now, we are going to show that the effect of a substitution over free variables and free function variables decides the substitution in the sense of the equivalence. We will not repeat this comment and a later usage of "same" for substitutions will be in this sense of the equivalence.

Lemma 5.1.9 (role of non-free variables):

1. (a) *if* $x \notin Free(t)$, $t\vec{\varrho} = t\vec{\varrho}[x := t']$, *where* $x \in V$ *and* $t' \in T$.

   (b) *if* $x \notin Free(ft)$, $ft\vec{\varrho} = ft\vec{\varrho}[x := t']$, *where* $x \in V$ *and* $t' \in T$.

2. (a) *for all* $m \geq 0$, *if* $f \notin Free(t)$, $t\vec{\varrho} = t\vec{\varrho}[f := ft']$, *where* $f \in FV_m$ *and* $ft' \in FT_m$.

   (b) *for all* $m \geq 0$, *if* $f \notin Free(ft)$, $ft\vec{\varrho} = ft\vec{\varrho}[f := ft']$, *where* $f \in FV_m$ *and* $ft' \in FT_m$.

Proof By structural induction. □

This lemma says that substitutions have no effect on non-free variables. The next lemma will give us a relation between free variables and substitutions.

Lemma 5.1.10 (relation between free variables and substitutions): *Let* $u_j \in T$.

(a) *Given* $t \in T$, *for all* $\{\vec{x}\}$ *if* $\{\vec{x}\} \subseteq Free(t) \cap V$ *and* $\bigwedge_{i \neq j} x_i \neq x_j$ *then*

$$Free(t[\vec{x} := \vec{u}]) = (Free(t) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{x}|} Free(u_j)).$$

(b) For all $k \geq 0$, given $ft \in FT_k$ and for all $\{\vec{x}\}$ if $\{\vec{x}\} \subseteq Free(ft) \cap V$ and $\bigwedge_{i \neq j} x_i \neq x_j$ then $Free(ft[\vec{x} := \vec{u}]) = (Free(ft) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{x}|} Free(u_j))$.

Proof

It is by structural induction. For the case of $|\vec{x}| = 0$, it is trivial, since we have $Free(t\ ) = Free(t)$ and $Free(ft\ ) = Free(ft)$.

The following only deals with $|\vec{x}| > 0$.

case x: since $|\vec{x}| > 0$, i.e. $x = x_1$ and $|\vec{x}| = 1$, then

$$LHS = Free(u_1) = RHS$$

case $f(\vec{t})$: by structural hypothesis,

$$\begin{aligned} LHS &= \bigcup_{j=1}^{m} Free(t_j[\vec{x} := \vec{u}]) \\ &= \bigcup_{j=1}^{m}((Free(t_j) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j))) \\ &= ((\bigcup_{j=1}^{m}(Free(t_j) - \{\vec{x}\})) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) = RHS \end{aligned}$$

case $\langle \vec{y} : t \rangle$: by structural hypothesis,

$$\begin{aligned} LHS &= Free(\langle \vec{z} : t[\vec{x}, \vec{y} := \vec{u}, \vec{z}] \rangle) \\ &= Free(t[\vec{x}, \vec{y} := \vec{u}, \vec{z}]) - \{\vec{z}\} \\ &= ((Free(t) - \{\vec{x}, \vec{y}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) \cup (\bigcup_{i=1}^{|\vec{z}|} Free(z_i))) - \{\vec{z}\} \\ &= ((Free(t) - \{\vec{y}\}) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) = RHS \end{aligned}$$

case $\sigma(\vec{ft}, \vec{t})$: by structural hypothesis,

$$\begin{aligned} LHS &= \bigcup_{i=1}^{\ell} Free(ft_i[\vec{x} := \vec{u}]) \cup \bigcup_{j=1}^{n} Free(t_j[\vec{x} := \vec{u}]) \\ &= (\bigcup_{i=1}^{\ell}(Free(ft_i) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j))) \\ &\quad \cup (\bigcup_{j=1}^{n}(Free(t_j) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) \\ &= ((\bigcup_{i=1}^{\ell} Free(ft_i)) \cup (\bigcup_{j=1}^{n} Free(t_j)) - \{\vec{x}\}) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) = RHS. \square \end{aligned}$$

Extending the above result to include function variables, which will be used in section 5.3, we have the following.

Lemma 5.1.11 (free variables and substitutions): *For $p \in T$ or $p \in FT_k$ of some $k \geq 0$, let $\{\vec{f}\} \subseteq FV$ and $\bigwedge_{i \neq j} f_i \neq f_j$ and $\{\vec{x}\} \subseteq Free(p) \cap V$ and $\bigwedge_{i \neq j} x_i \neq$*

x_j, *we have* $Free(p[\vec{f},\vec{x} := \vec{ft},\vec{u}]) = (Free(p) - \{\vec{f},\vec{x}\}) \cup (\bigcup_{i=1}^{|\vec{ft}|} Free(ft_i)) \cup (\bigcup_{j=1}^{|\vec{u}|} Free(u_j))$, *where* $ft_i \in FV_{k_i} \cup FT_{k_i}$ *and* $u_j \in T$.

Proof By structural induction.□

Now, we are able to give the definition for interpretations of binding terms, which is the subject of the next section.

5.2 Well-definedness of interpretations

Essentially, an interpretation is defined by an arbitrary way of bound all free variables in binding terms. From another point of view, we can think of that interpretations are only provided for closed binding terms. So the well-definedness of interpretations largely depends on the careful manipulations of binding variables in interpretations. The following lemma (Lemma 5.2.1) provides such kind of manipulations. It is followed by the well-definedness of interpretations in iBAs (Lemma 5.2.2).

Lemma 5.2.1 (interpretation and substitution): *Let* **B** *be an iBA. Thus,*

(a) *given* $t \in T$, *for all* $\{\vec{x}\} \subseteq V$ *if* $\{\vec{x}\} \supseteq Free(t) \cap V$ *and* $\bigwedge_{i\neq j} x_i \neq x_j$, *then we have* $\mathcal{B}_{\vec{x}}[\![t[\vec{y} := \vec{z}]\,]\!]_\psi \in B_{|\vec{x}|}$, *where* $Free(t[\vec{y} := \vec{z}]) \cap V \subseteq \{\vec{x}\}$.

(b) *given* $ft \in FT_m$, *for all* $\{\vec{x}\} \subseteq V$ *if* $\{\vec{x}\} \supseteq Free(ft) \cap V$ *and* $\bigwedge_{i\neq j} x_i \neq x_j$, *then we have* $\mathcal{B}_{\vec{x}}[\![ft[\vec{y} := \vec{z}]\,]\!]_\psi \in B_{|\vec{x}|+m}$, *where* $Free(ft[\vec{y} := \vec{z}]) \cap V \subseteq \{\vec{x}\}$.

Proof

We prove this lemma by induction on j of $T^{(j)}$ and $FT_m^{(j)}$ with case analysis.

- For $j = 0$, i.e. $t = x$ and $ft = \langle \vec{y'} : x\rangle$, there are two possibilities for t and three possibilities for ft.

  - Let us consider the two possible cases for t first.

* if $x \in \{\vec{y}\}$, say $x = y_i$ (where $y_j \neq x$ and $1 \leq i < j \leq |\vec{y}|$), then

$$\mathcal{B}_{\vec{x}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]_\psi \in B_{|\vec{x}|}$$

since $Free(x\,[\vec{y} := \vec{z}\,]) = \{z_i\}$.

* if $x \notin \{\vec{y}\}$, then

$$\mathcal{B}_{\vec{x}}[\![x\,[\vec{y} := \vec{z}\,]\,]\!]_\psi = \mathcal{B}_{\vec{x}}[\![x]\!]_\psi \in B_{|\vec{x}|}.$$

– Secondly, three possibilities for ft are

* $x \in \{\vec{y'}\}$,
* $x \notin \{\vec{y'}\}$ and $x \in \{\vec{y}\}$,
* $x \notin \{\vec{y'}\}$ and $x \notin \{\vec{y}\}$.

Let us deal with them one by one.

* for the first possible case,

$$\mathcal{B}_{\vec{x}}[\![ft\,[\vec{y} := \vec{z}\,]\,]\!]_\psi$$
$$= \mathcal{B}_{\vec{x}}[\![\langle \vec{y''} : x\,[\vec{y}, \vec{y'} := \vec{z}, \vec{y''}]\rangle\,]\!]_\psi$$

(where y''_j is the least $y'' \in V$ such that for all $z \in Free(x)$,

$$y'' \notin Free(\vec{v}[\vec{y} := \vec{z}][z]) \cup \{\vec{y''}\lceil_{j-1}\})$$

$$= \mathcal{B}_{\vec{x}}[\![\langle \vec{y''} : y''_j\rangle\,]\!]_\psi$$

(where $x = y'_j \in \{\vec{y'}\}$)

$$= \mathcal{B}_{\vec{x},\vec{z}}[\![y''_j\,[\vec{y''} := \vec{z'}\,]\,]\!]_\psi$$

(where z'_i is the least $z \in V$ such that

$$z \notin (Free(y''_j) - \{\vec{y''}\}) \cup \{\vec{x}\} \cup \{\vec{z'}\lceil_{i-1}\}))$$

$$= \mathcal{B}_{\vec{x},\vec{z'}}[\![z'_j]\!]_\psi$$
$$= pr_{|\vec{x}|+m,|\vec{x}|+j} \in B_{|\vec{x}|+m}$$

* for the second case,

$\mathcal{B}_{\vec{x}}[\![ft[\vec{y} := \vec{z}]\,]\!]_\psi$
$= \mathcal{B}_{\vec{x}}[\![\langle \vec{y''} : x[\vec{y}, \vec{y'} := \vec{z}, \vec{y''}]\rangle\,]\!]_\psi$

(where y''_j is the least $y'' \in V$ such that for all $z \in Free(x)$,

$y'' \notin Free(\vec{t}[\vec{y} := \vec{z}][z]) \cup \{\vec{y''}\lceil_{j-1}\})$

$= \mathcal{B}_{\vec{x}}[\![\langle \vec{y''} : z_i\rangle]\!]_\psi$

(where $x = y_i \in \{\vec{y}\}$)

$= \mathcal{B}_{\vec{x},\vec{z}}[\![z_i[\vec{y''} := \vec{z'}]\,]\!]_\psi$

(where z'_j is the least $z \in V$ such that

$z \notin (Free(z_i) - \{\vec{y''}\}) \cup \{\vec{x}\} \cup \{\vec{z'}\lceil_{j-1}\})$

$= pr_{|\vec{x}|+m,j} \in B_{|\vec{x}|+m}$

(where $z_i = x_j \in \{\vec{x}\}$)

* for the third case, it is similar to the second case. The difference is that instead of having $z_i = x_j \in \{\vec{x}\}$, it directly has $x = x_j \in \{\vec{x}\}$. So, we leave it out.

- For $j > 0$, there are three cases.

  - case $f(\vec{t})$: since $f(\vec{t})[\vec{y} := \vec{z}] = f(\vec{t}[\vec{y} := \vec{z}])$ and induction hypothesis, we have

    $\mathcal{B}_{\vec{x}}[\![f(\vec{t})[\vec{y} := \vec{z}]\,]\!]_\psi = \mathcal{B}_{\vec{x}}[\![f(\vec{t}[\vec{y} := \vec{z}])]\!]_\psi$
    $= \psi_{|\vec{t}|}(f) \circ_{|\vec{t}|,|\vec{x}|} < \mathcal{B}_{\vec{x}}[\![\vec{t}[\vec{y} := \vec{z}]\,]\!]_\psi > \in B_{|\vec{x}|}$

  - case $\langle \vec{y'} : t\rangle$:

    $\mathcal{B}_{\vec{x}}[\![ft[\vec{y} := \vec{z}]\,]\!]_\psi$
    $= \mathcal{B}_{\vec{x}}[\![\langle \vec{y''} : t[\vec{y}, \vec{y'} := \vec{z}, \vec{y''}]\rangle\,]\!]_\psi$

    (where y''_j is the least $y'' \in V$ such that for all $z \in Free(t)$,

    $y'' \notin Free(\vec{t}[\vec{y} := \vec{z}][z]) \cup \{\vec{y''}\lceil_{j-1}\}$.)

$$= \mathcal{B}_{\vec{x}\vec{z}}[\![(t\,[\vec{y},\vec{y'} := \vec{z},\vec{y''}])\,[\vec{y''} := \vec{z'}]\,]\!]_\psi$$

(where z'_j is the least $z \in V$ such that

$$z \notin (Free(t\,[\vec{y},\vec{y'} := \vec{z},\vec{y''}]) - \{\vec{y''}\}) \cup \{\vec{x}\} \cup \{\vec{z'}\lceil_{j-1}\}$$

Since having Lemma 5.1.4, Lemma 5.1.9, Lemma 5.1.11 and induction hypothesis, we have that the above value is in $B_{|\vec{x}|+m}$.

– case $\underline{\alpha}(\vec{ft},\vec{t})$: It follows easily from induction hypothesis. □

With the above lemma, we can say that all values of interpretations of binding terms are in proper domains, i.e. well-definedness of interpretations in iBAs.

Theorem 5.2.2 (well-definedness of interpretations in an iBA): *Let* **B** *be an iBA. Thus,*

(a) given $t \in T$, for all $\{\vec{x}\} \subseteq V$ if $\{\vec{x}\} \supseteq Free(t) \cap V$ and $\bigwedge_{i\neq j} x_i \neq x_j$, then we have $\mathcal{B}_{\vec{x}}[\![t]\!]_\psi \in B_{|\vec{x}|}$.

(b) given $ft \in FT_l$, for all $\{\vec{x}\} \subseteq V$ if $\{\vec{x}\} \supseteq Free(ft) \cap V$ and $\bigwedge_{i\neq j} x_i \neq x_j$, then we have $\mathcal{B}_{\vec{x}}[\![ft]\!]_\psi \in B_{|\vec{x}|+l}$.

Proof

We prove this by structural induction (see Definition 1.2.2.2 for $\mathcal{B}_{\vec{x}}[\![\bullet]\!]_\psi$). Since the other cases are quite easy, we omit them and only give the case for function terms. I.e.

case $\langle \vec{y} : t \rangle$: $\mathcal{B}_{\vec{x}}[\![ft]\!]_\psi = \mathcal{B}_{\vec{x}\vec{z}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]_\psi$, where z_j is the least $z \in V$ such that

$z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$

By Lemma 5.2.1, the above value is in $B_{|\vec{x}|+|\vec{y}|}$. □

5.3 b-clones

As we pointed out before (Section 1.2), a careful reader may already notice that if we consider a natural number k as a sort, then iBAs can be viewed as a certain kind of many-sorted first order algebras, so-called b-clones (b for binding). Such an observation leads to the following.

Let Σ be sorted by $Nat^* \to Nat$, $\underline{f}$ ranges over variables χ. Σ has that

constants: $\underline{pr}_{k,i} \in \Sigma_{\varepsilon \to k}$ $(1 \leq i \leq k)$

operations:

1. $\underline{\circ}_{m,k} \in \Sigma_{m,k^m \to k}$, where $k^m = \underbrace{k, k, ..., k}_{m \ times}$
2. $\underline{\sigma}_k \in \Sigma_{k+m_1,...,k+m_\ell,k^n \to k}$ or simply $\Sigma_{k+\vec{m},k^n \to k}$

Note that there is a correspondence $_$ between Σ^{BO} and Σ , and between FV_m and χ_m such that $f \in FV_m$ iff $\underline{f} \in \chi_m$ and for each $k \in Nat$, $\sigma \in \Sigma^{BO}_{<\vec{m},n>}$ iff $_(\sigma, k) = \underline{\sigma}_k \in \Sigma_{k+\vec{m},k^n \to k}$. This correspondence is a bijection. Because of this, we sometimes omit the underline $_$ of $\underline{t}, \underline{ft}, \underline{f}, \underline{\sigma}_k$ in referring to them in the future.

For convenience, we repeat the definition of first order terms below specially for b-clones.

Definition 5.3.1 (terms of b-clones): *We define b-clone's terms* $T(\vec{X})$ $(\vec{X} \subseteq \vec{\chi})$ *as follows:*

1. *for each* $k \geq 0$,

$$\frac{}{\underline{pr}_{k,i}() \in T_k} \ (\underline{pr}_{k,i} \in \Sigma_{\varepsilon \to k} \text{ and } 1 \leq i \leq k)$$

2.

$$\frac{}{\underline{f} \in T_k} \ (\underline{f} \in X_k \text{ and } k \geq 0)$$

3.

$$\frac{\underline{u} \in T_m;\ \underline{t}_1, \underline{t}_2, ..., \underline{t}_m \in T_k}{\underline{o}_{m,k}(\underline{u}, \underline{t}_1, \underline{t}_2, ..., \underline{t}_m) \in T_k}\ (\underline{o}_{m,k} \in \Sigma_{m,k^m \to k})$$

4.

$$\frac{\underline{t}_1 \in T_{k+m_1}, \underline{t}_2 \in T_{k+m_2}, ..., \underline{t}_\ell \in T_{k+\ell};\ \underline{u}_1, \underline{u}_2, ..., \underline{u}_n \in T_k}{\underline{\sigma}_k(\underline{t}_1, \underline{t}_2, ..., \underline{t}_\ell, \underline{u}_1, \underline{u}_2, ..., \underline{u}_n) \in T_k}\ (\underline{\sigma}_k \in \Sigma_{k+\vec{m},k^n \to k}).$$

From now on, we are only interested in this kind of Σ-algebras, i.e. b-clones, which are intended to be the counterparts of iBA in many-sorted first order algebras. Formally,

Definition 5.3.2 (b-clone): *A Σ-algebra $\mathbf{D}$ is said to be a binding clone (b-clone for short) iff it satisfies the following laws: let $\underline{Pr}^k_{i,j} = \underline{pr}^{\mathbf{D}}_{k,i}, \underline{pr}^{\mathbf{D}}_{k,i+1}, \underline{pr}^{\mathbf{D}}_{k,i+2}, \cdots, \underline{pr}^{\mathbf{D}}_{k,j}$ $(i \leq j)$ and $\underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(\vec{g}, \vec{h}) = \underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(g_1, \vec{h}), \underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(g_2, \vec{h}), ..., \underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(g_{|\vec{g}|}, \vec{h})$ for $g_i \in D_{|\vec{h}|}$;*

1. *(b-cln-left-proj)*

   Let $1 \leq i \leq |\vec{g}|$ and $k \geq 0$, then $\underline{o}^{\mathbf{D}}_{|\vec{g}|,k}(\underline{pr}^{\mathbf{D}}_{|\vec{g}|,i}, \vec{g}) = g_i$,

   where $g_j \in D_k$ $(1 \leq j \leq |\vec{g}|)$.

2. *(b-cln-right-proj)*

   Let $k \geq 0$, then $\underline{o}^{\mathbf{D}}_{k,k}(g, \underline{Pr}^k_{1,k}) = g$,

   where $g \in D_k$;

3. *(b-cln-assoc)*

   Let $k \geq 0$, then $\underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(\underline{o}^{\mathbf{D}}_{|\vec{g}|,|\vec{h}|}(g', \vec{g}), \vec{h}) = \underline{o}^{\mathbf{D}}_{|\vec{g}|,|\vec{h}|}(g', \underline{o}^{\mathbf{D}}_{|\vec{h}|,k}(\vec{g}, \vec{h}))$,

   where $g' \in D_{|\vec{g}|}$, $g_i \in D_{|\vec{h}|}$ $(1 \leq i \leq |\vec{g}|)$, $h_j \in D_k$ $(1 \leq j \leq |\vec{h}|)$;

4. *(b-cln-unif)*

   For $l \geq 0$, $\underline{o}^{\mathbf{D}}_{k,l}(\underline{\sigma}^{\mathbf{D}}_k(\vec{g}, \vec{g'}), \vec{h})) = \underline{\sigma}^{\mathbf{D}}_l(\vec{g''}, \underline{o}^{\mathbf{D}}_{k,l}(\vec{g'}, \vec{h}))$,

   where $\vec{g''}_i = \underline{o}^{\mathbf{D}}_{k+m_i,l+m_i}(g_i, \vec{h'}, \underline{Pr}^{l+m_i}_{l+1,l+m_i})$ $(1 \leq i \leq \ell)$, $h'_{j'} = \underline{o}_{l,l+m_i}(h_{j'}, \underline{Pr}^{l+m_i}_{1,l})$ $(1 \leq j' \leq k)$, $g'_j \in D_k$ $(1 \leq j \leq n)$, $g_i \in D_{k+m_i}$ $(1 \leq i \leq \ell)$, $h_{j'} \in D_l$ $(1 \leq j' \leq k)$, $\underline{\sigma}_k \in \Sigma_{k+\vec{m},k^n \to k}$ and $\underline{\sigma}_l \in \Sigma_{l+\vec{m},l^n \to l}$.

From definitions, we can easily check that

Fact 5.3.3 (equivalence between iBAs and b-clones): **B** *is an iBA iff it is a b-clone (and this is written as* $\mathbf{B} = \mathbf{D}$*).*

This substantiates our previous intuition. On the other hand, the equivalence leads us to think of relating the equality of iBAs to the one of b-clones. With this purpose in mind, we proceed as follows. For convenience, we provide the definition of interpretations in b-clones (Definition 5.3.4), although it is just a special case of the one for many-sorted first order algebras (Definition 2.0.3). With the equivalence in Fact 5.3.3, we would certainly be interested in a syntactic relations between iBAs and b-clones. We discover two translations $tr_{\vec{x}}$ and $\underline{tr}$ between binding terms and the terms of b-clones. They will be defined in Definition 5.3.5 and Definition 5.3.8 and their well-definedness will be verified in Lemma 5.3.7 and Lemma 5.3.9 respectively.

For convenience, we repeat the definition of interpretation for first order algebra below specially for b-clones.

Definition 5.3.4 (interpretation in b-clones): *We follow the standard way of defining an interpretation* $\mathcal{D}$ *over b-clone* **D** *(or* Σ*-algebra) with an environment* $\rho_k : X_k \to D_k$ $(X_k \subseteq \chi_k,\ k \geq 0)$ *such that*

1. $\mathcal{D}[\![\underline{pr}_{k,i}]\!](\rho) =_{df} pr^{\mathbf{D}}_{k,i}$,
2. $\mathcal{D}[\![\underline{f}]\!](\rho) =_{df} \rho_k(\underline{f})$,
3. $\mathcal{D}[\![\underline{\circ}_{|\vec{u}|,k}(\underline{t}, \vec{u}\,)]\!](\rho) =_{df} \mathcal{D}[\![\underline{t}]\!](\rho) \circ^{\mathbf{D}}_{|\vec{u}|,k} < \mathcal{D}[\![\vec{u}]\!](\rho) >$,
4. $\mathcal{D}[\![\underline{\sigma}_k(\vec{t}, \vec{u}\,)]\!](\rho) =_{df} \underline{\sigma}^{\mathbf{D}}_k(\mathcal{D}[\![\vec{t}]\!](\rho), \mathcal{D}[\![\vec{u}]\!](\rho))$.

In what follows, we define two syntactic translations between binding terms and the terms of b-clones. They are the connections to relate Binding Equations, dependent Binding Equations and quasi-dependent Binding Equations to Σ-equations, dependent Σ-equations and quasi-dependent Σ-equations, respectively.

Definition 5.3.5 (translation $tr_{\vec{x}}$**):** *A translation* $tr_{\vec{x}} : BT \to T$ *is defined as below:*

given $t \in T$ and $ft \in FT_k$, for all $\{\vec{x}\} \subseteq V$ such that ($Free(t) \cap V \subseteq \{\vec{x}\}$ and $\bigwedge_{i \neq j} x_i \neq x_j$) and/or ($Free(ft) \cap V \subseteq \{\vec{x}\}$ and $\bigwedge_{i \neq j} x_i \neq x_j$)

1. $tr_{\vec{x}}[\![x]\!] =_{df} \underline{pr}_{|\vec{x}|,i}$, *where* $\vec{x}(i) = x$;

2. $tr_{\vec{x}}[\![f(\vec{t})]\!] =_{df} \underline{o}_{|\vec{t}|,|\vec{x}|}(\underline{f}, tr_{\vec{x}}[\![\vec{t}]\!])$,

   where $tr_{\vec{x}}[\![\vec{t}]\!] = tr_{\vec{x}}[\![t_1]\!], tr_{\vec{x}}[\![t_2]\!], ..., tr_{\vec{x}}[\![t_{|\vec{t}|}]\!]$.

3. $tr_{\vec{x}}[\![\langle \vec{y} : t \rangle]\!] =_{df} tr_{\vec{x},\vec{z}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]$,

   where z_j *is the least* $z \in V$ *such that* $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$;

4. $tr_{\vec{x}}[\![\sigma(\vec{ft}, \vec{t})]\!] =_{df} \underline{o}_{|\vec{x}|}(tr_{\vec{x}}[\![\vec{ft}]\!], tr_{\vec{x}}[\![\vec{t}]\!])$,

   where $tr_{\vec{x}}[\![\vec{ft}]\!] = tr_{\vec{x}}[\![ft_1]\!], tr_{\vec{x}}[\![ft_2]\!], ..., tr_{\vec{x}}[\![ft_{|\vec{ft}|}]\!]$.

The proof of the well-definedness of translation $tr_{\vec{x}}$ is analogous to the ones of Lemma 5.2.1 and Lemma 5.2.2. The lemma below says that the results of translation $tr_{\vec{x}}$ of binding terms being applied to certain substitutions are in proper terms of b-clones.

Lemma 5.3.6 (translation $tr_{\vec{x}}$ and free variables):

(i) if $Free(t\,[\vec{y} := \vec{z}\,]) \cap V \subseteq \{\vec{x}\}$, $tr_{\vec{x}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!] \in T_{|\vec{x}|}$;

(ii) if $Free(ft\,[\vec{y} := \vec{z}\,]) \cap V \subset \{\vec{x}\}$, $tr_{\vec{x}}[\![ft\,[\vec{y} := \vec{z}\,]\,]\!] \in T_{|\vec{x}|+k}$, *where* $ft \in FT_k$.

Proof

It is similar to the proof of Lemma 5.2.1 and is omitted. □

Similar to the relation between Lemma 5.2.1 and Theorem 5.2.2, we have

Lemma 5.3.7 (well-definedness of translation $tr_{\vec{x}}$): *For* $t \in T$, $tr_{\vec{x}}[\![t]\!] \in T_{|\vec{x}|}$; *and for* $ft \in FT_k$, $tr_{\vec{x}}[\![ft]\!] \in T_{|\vec{x}|+k}$.

Proof It follows from Lemma 5.3.6 with structural induction. □

This lemma show that binding terms are translated into proper b-clone terms by $tr_{\vec{x}}$. So, we are going to turn our attention to translation $\underline{tr}$ which is somehow a reversed version of $tr_{\vec{x}}$.

Definition 5.3.8 (translation $\underline{tr}$): *A translation $\underline{tr}$ from terms T of b-clones to binding terms BT is defined as follows:*

1. $\underline{tr}[\![\underline{pr}_{k,i}]\!] =_{df} x_i$, *where x_i is the ith least $x \in V$.*

2. $\underline{tr}[\![\underline{f}]\!] =_{df} f(\vec{x})$, *where x_j is the least $x \in V$ such that $x_j \notin \{\vec{x}\lceil_{j-1}\}$.*

3. $\underline{tr}[\![\underline{o}_{|\vec{t}|,k}(\underline{t}, \vec{u})]\!] =_{df} \underline{tr}[\![\underline{t}]\!][\vec{x} := \underline{tr}[\![\vec{u}]\!]]$, *where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.*

4. $\underline{tr}[\![\underline{\sigma}_k(\vec{t}, \vec{u})]\!] =_{df} \sigma(\vec{fv}, \vec{v})$, *where $\vec{fv}(i) = \langle \vec{x^i}\lfloor_k; \underline{tr}[\![\vec{t}]\!]\rangle$ and $\vec{v}(j) = \underline{tr}[\![\vec{u}(j)]\!]$.*

Translation $\underline{tr}$ is not a direct reversed version of translation $tr_{\vec{x}}$. One obvious sign is that all possible results of translation $\underline{tr}$ are terms, none is a function term. But it still has a nice property relating to free variables, see Lemma 5.3.9. The reason we said that translation $\underline{tr}$ is a reversed version of $tr_{\vec{x}}$ is that they preserve equality and derivability. These are topics of Section 5.4 and Section 5.5, respectively.

Lemma 5.3.9 (well-definedness of translation $\underline{tr}$): *For $\underline{t} \in T_{|\vec{x}|}$, we have $\underline{tr}[\![\underline{t}]\!] \in T$ and $Free(\underline{tr}[\![\underline{t}]\!]) \cap V \subseteq \{\vec{x}\}$, where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.*

Proof By structural induction. □

The above lemma actually expresses that free variables of terms being translated from terms of b-clone by $\underline{tr}$ are among the first a few variables in V. Another thing we should mention here is that the confirmation of that the two translations are the one and the reversed of the other will be given in Subsection 5.6.2, i.e. Theorem 5.6.2.1. This, in turn, shows the well-definedness of one that supports the well-definedness of the other's.

5.4 Preservation of equality

On observation of the similarity between iBAs and b-clones, we obviously hope that the two translations in the previous section (Section 5.3) can preserve semantic equality. With this idea in mind, we would like to know what are

(a) the relationship among $\mathcal{B}_{\vec{x}}$, $tr_{\vec{x}}$ and $\mathcal{D}$, and

(b) the relationship among $\mathcal{B}_{\vec{x}}$, $\underline{tr}$ and $\mathcal{D}$.

For (a), we have Lemma 5.4.1.1. For (b), we have Theorem 5.4.2.4. These two results lead us to expect: "identity" (equality) preservations by the translations.

Next, Subsection 5.4.1 and Subsection 5.4.2 contributes us that

(i) for every binding term p there is a corresponding term $tr_{\vec{x}}[\![p]\!]$ of b-clones which has a same denotation as the binding term (Lemma 5.4.1.1).

(ii) Conversely, for each term $\underline{t}$ of b-clones there is a corresponding binding term $\underline{tr}[\![\underline{t}]\!]$ which has a same denotation as the term of b-clones (Theorem 5.4.2.4).

Therefore, binding terms and terms of b-clones share a same syntactic expressive power. Consequently, Subsection 5.4.3 shows that equalities are preserved under the translations. Syntactically, it implies that the valid equations are preserved under the translations.

5.4.1 Relationship among $\mathcal{B}_{\vec{x}}$, $tr_{\vec{x}}$ and $\mathcal{D}$

Because of Fact 5.3.3, we have the following lemma.

Lemma 5.4.1.1 (interpretation and translation $tr_{\vec{x}}$): *Given an iBA* $\mathbf{B} = \mathbf{D}$, *where* $\mathbf{D}$ *is a b-clone, we have the following:*

(i) For $t \in T^{(j)}$ *and for all* $\{\vec{x}\} \subseteq V$, *if* $Free(t) \cap V \subseteq \{\vec{x}\}$ *and* $\bigwedge_{i \neq j} x_i \neq x_j$, *then* $\mathcal{B}_{\vec{x}}[\![t]\!]_{\psi} = \mathcal{D}[\![tr_{\vec{x}}[\![t]\!]\,]\!](\rho)$, *where* $\psi(f) = \rho(\underline{f})$ *and* $_$ *is a bijection between* FV *and* $\vec{\chi}$.

(ii) For $ft \in FT_k^{(j)}$ *and for all* $\{\vec{x}\} \subseteq V$, *if* $Free(ft) \cap V \subseteq \{\vec{x}\}$ *and* $\bigwedge_{i \neq j} x_i \neq x_j$, *then* $\mathcal{B}_{\vec{x}}[\![ft]\!]_{\psi} = \mathcal{D}[\![tr_{\vec{x}}[\![ft]\!]\,]\!](\rho)$, *where* $\psi(f) = \rho(\underline{f})$.

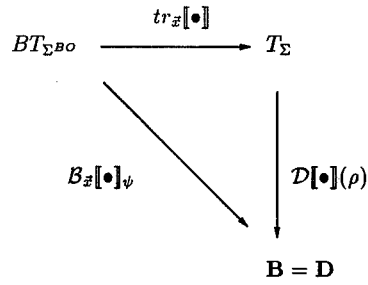

Figure 5–1: translation $tr_{\vec{x}}[\![\bullet]\!]$ and interpretations

Proof

By Fact 5.3.3, Property 5.1.4 and induction on j of $BT^{(j)}$ with case analysis (the reason for not using structural induction is related to the case of function terms $\langle \vec{y} : t \rangle$). □

What this lemma says is that the following diagram (see Figure 5-1) commutes. It can also mean that every binding term has its counterpart in b-clones. The conversion of this is the subject of the next subsection.

5.4.2 Relationship among $\mathcal{B}_{\vec{x}}$, $\underline{tr}$ and $\mathcal{D}$

Similar to Section 5.4.1, we hope the following diagram (see Figure 5-2) commutes. It means that every term of b-clones has its counterpart in BT.

figure 5.4.2.1

This time it can not be achieved as easy as the previous one. Fortunately we do have it commutes (Theorem 5.4.2.4). We proceed as follows.

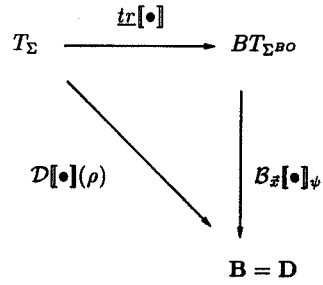

Figure 5–2: translation $\underline{tr}[\bullet]$ and interpretations

Lemma 5.4.2.1 (interpretation and double substitution): *Let* **B** *be an iBA.*

(i) For $t \in T^{(\jmath)}$, $\mathcal{B}_{\vec{x}\vec{z}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]_\psi = \mathcal{B}_{\vec{x}\vec{w}}[\![(t\,[\vec{y} := \vec{y'}])\,[\vec{y'} := \vec{w}\,]\,]\!]_\psi$
where y'_j *is a* $y' \in V$ *such that* $y' \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{y'}\lceil_{j-1}\}$, z_j *is the least* $z \in V$ *such that* $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$, w_j *is the least* $w \in V$ *such that* $w \notin (Free(t\,[\vec{y} := \vec{y'}]) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{w}\lceil_{j-1}\}$ *and* $\vec{y}$ *are distinctive in* V.

(ii) For $ft \in FT_k^{(\jmath)}$ $(k \geq 0)$, $\mathcal{B}_{\vec{x}\vec{z}}[\![ft\,[\vec{y} := \vec{z}\,]\,]\!]_\psi = \mathcal{B}_{\vec{x}\vec{w}}[\![(ft\,[\vec{y} := \vec{y'}])\,[\vec{y'} := \vec{w}\,]\,]\!]_\psi$
where y'_j *is a* $y' \in V$ *such that* $y' \notin (Free(ft) - \{\vec{y}\}) \cup \{\vec{y'}\lceil_{j-1}\}$, z_j *is the least* $z \in V$ *such that* $z \notin (Free(ft) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$, w_j *is the least* $w \in V$ *such that* $w \notin (Free(ft\,[\vec{y} := \vec{y'}]) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{w}\lceil_{j-1}\}$ *and* $\vec{y}$ *are distinctive in* V.

Proof

It is of very length by induction. The proof involves a very careful examing the side condition, e.g. actually $z_j = w_j$, and freely introducing non-free variables and eliminating non-free variables. □

Lemma 5.4.2.1 said that an interpretation of a binding term, which are doubly applied to families of substitution functions, can be replaced by an interpretation

of the same binding term, which are singly applied to single family of substitution functions.

The coming lemma will say that essentially all α-convertible terms (see Definition 5.5.1.1) denote to a same object.

Lemma 5.4.2.2 (interpretation and substitution): *Given an iBA* **B**, *we have the following.*

(i) For $t \in T$ and for all $\{\vec{y}\} \subseteq V$, we have $\mathcal{B}_{\vec{y}}[\![t]\!]_\psi = \mathcal{B}_{\vec{x}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]_\psi$,

provided that $Free(t) \cap V \subseteq \{\vec{y}\}$ and z_j is a $z \in V$ such that $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{z}\lceil_{j-1}\}$.

(ii) For $ft \in FT_k$, for all $\{\vec{y}\} \subseteq V$, we have $\mathcal{B}_{\vec{y}}[\![ft]\!]_\psi = \mathcal{B}_{\vec{x}}[\![ft\,[\vec{y} := \vec{z}\,]\,]\!]_\psi$,

provided that $Free(ft) \cap V \subseteq \{\vec{y}\}$ and z_j is a $z \in V$ such that $z \notin (Free(ft) - \{\vec{y}\}) \cup \{\vec{z}\lceil_{j-1}\}$.

Proof

It follows from Lemma 5.4.1.1 with structural induction and carefully introducing non-free variables to make two sides of equations match to common ones (a similar techniques used in the proof of Lemma 5.4.2.1). □

Lemma 5.4.2.3 (substitution and composition): *Let* **B** *be an iBA. Thus,*

(i) given $t \in T$, for all $\{\vec{y}\} \subseteq V$ and $\bigwedge_{i \neq j} y_i \neq y_j$ and $\{\vec{y}\} \supseteq Free(t) \cap V$, for all $\vec{u} \in T$ and for all $\{\vec{x}\} \subseteq V$ such that $\{\vec{x}\} \supseteq (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) \cap V$ and $|\vec{y}\,| = |\vec{u}|$, we have

$$\mathcal{B}_{\vec{x}}[\![t\,[\vec{y} := \vec{u}\,]\,]\!]_\psi = \mathcal{B}_\varepsilon[\![\langle \vec{y} : t\rangle]\!]_{\psi \circ_{|\vec{u}|,|\vec{x}|}} < \mathcal{B}_{\vec{x}}[\![\vec{u}\,]\!]_\psi >,$$

where $\mathcal{B}_{\vec{x}}[\![\vec{u}\,]\!]_\psi$ means $\mathcal{B}_{\vec{x}}[\![u_1]\!]_\psi$, $\mathcal{B}_{\vec{x}}[\![u_2]\!]_\psi$, ..., $\mathcal{B}_{\vec{x}}[\![u_{|\vec{u}|}]\!]_\psi$.

(ii) given $ft \in FT_k$ ($k \geq 0$), for all $\{\vec{y}\} \subseteq V$ and $\bigwedge_{i \neq j} y_i \neq y_j$ and $\{\vec{y}\} \supseteq Free(ft) \cap V$, for all $\vec{u} \in T$ and for all $\{\vec{x}\} \subseteq V$ such that $\{\vec{x}\} \supseteq (\bigcup_{j=1}^{|\vec{u}|} Free(u_j)) \cap V$ and $|\vec{y}\,| = |\vec{u}|$, we have

$$\mathcal{B}_{\vec{x}}[\![ft\,[\vec{y} := \vec{u}\,]\,]\!]_\psi =$$
$$\mathcal{B}_\varepsilon[\![\langle \vec{y}\,\vec{y'} : ft||_k\,[\vec{z} := \vec{y'}]\rangle]\!]_{\psi \circ_{|\vec{u}|+k,|\vec{x}|+k}} < \mathcal{B}_{\vec{x}}[\![\vec{u}\,]\!]_{\psi \circ_{|\vec{x}|,|\vec{x}|+k}} < Pr^{|\vec{x}|+k}_{1,|\vec{x}|} >, Pr^{|\vec{x}|+k}_{|\vec{x}|+1,|\vec{x}|+k} >,$$

where $\vec{z} = ft|_k$, $|\vec{y'}| = |\vec{z}| = k$,*and* $y'_j \notin \{\vec{y}\} \cup Free(ft) \cup \{\vec{y'}\lceil_{j-1}\}$.

Proof By Lemma 5.4.2.2 with structural induction. □

This lemma told us is that syntactic substitutions can be replaced by semantic compositions in a certain way.

Theorem 5.4.2.4 (interpretation and translation $\underline{tr}$): *Let* **B** *be an iBA and* $\mathbf{B} = \mathbf{D}$ *(***D** *as a b-clone). Thus, for* $\underline{t} \in T_{|\vec{x}|}$, *we have* $\mathcal{D}[\![\underline{t}]\!](\rho) = \mathcal{B}[\![\underline{tr}[\underline{t}]\ ]\!]_\psi$, *where* x_j *is the least* $x \in V$ *such that* $x \notin \{\vec{x}\lceil_{j-1}\}$.

Proof It results from the above lemma and structural induction. □

5.4.3 Intensional equality under translations

With the results of previous subsections, we are ready to show the semantic preservations of identities. First of all, we re-state the satisfaction relation accurately below.

Definition 5.4.3.1 (intensional satisfaction $\models_{iBA}$): *Let* **B** *be an iBA.*

(i) For $t, u \in T$, $\mathbf{B} \models_{iBA} t \simeq u$ *(or* $\mathbf{B} \models t \simeq u$*) iff for all environment* ψ $(= \{\psi_k | k \in Nat\})$ *and for all* $\{\vec{x}\} \subseteq V$ *such that* $\{\vec{x}\} \supseteq (Free(t) \cap V) \cup (Free(u) \cap V)$ *and* ψ_k *are over* $dom(\psi)_k \subseteq FV_k$, *which contains* $(Free(t) \cap FV_k) \cup (Free(u) \cap FV_k)$ $(k \geq 0)$, $\mathcal{B}_{\vec{x}}[\![t]\!]_\psi = \mathcal{B}_{\vec{x}}[\![u]\!]_\psi$.

(ii) For $ft, fu \in T$, $\mathbf{B} \models_{iBA} ft \simeq fu$ *(or* $\mathbf{B} \models_{iBA} ft \simeq fu$*) iff for all environment* ψ *and for all* $\{\vec{x}\} \subseteq V$ *such that* $\{\vec{x}\} \supseteq (Free(ft) \cap V) \cup (Free(fu) \cap V)$ *and* ψ_k *are over* $dom_k(\psi) \subseteq FV_k$, *which contains* $(Free(ft) \cap FV_k) \cup (Free(fu) \cap FV_k)$ $(k \geq 0)$, $\mathcal{B}_{\vec{x}}[\![ft]\!]_\psi = \mathcal{B}_{\vec{x}}[\![fu]\!]_\psi$.

As an easy consequence of Lemma 5.4.1.1, we have that the translation $tr_{\vec{x}}$ does preserve equality of iBAs. Formally

Theorem 5.4.3.2 (equality from iBAs to b-clones): *Let* **B** *be an iBA, both* p *and* q *range over either* T *or* FT_k *for some* $k \in Nat$ *together. Thus,* $\mathbf{B} \models p \simeq q$ *iff* $\mathbf{D} \models tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$ *for all* $\vec{x} \in V$ *and* $\{\vec{x}\} \supseteq (Free(p) \cap V) \cup (Free(q) \cap V)$; *where* $\mathbf{B} = \mathbf{D}$.

Proof It can be proved easily from definitions and Lemma 5.4.1.1. □

Theorem 5.4.3.2 can also be viewed as that translation $tr_{\vec{x}}$ preserves soundness. To get a similar result of this for translation $\underline{tr}$ (see Theorem 5.4.3.6), we need more preparation. Basically, it is the treatment of variable indices. Firstly, let us consider permutations of $\vec{x}$ in $\mathcal{B}_{\vec{x}}$.

Lemma 5.4.3.3 (permutation of binding variables): *Let* **B** *be an iBA.*

(i) For $t \in T$ and $\{\vec{x}\} \supseteq Free(t) \cap V$, let $1 \leq i \leq |\vec{x}|$ and $\vec{x}\lceil_{i-1}$ is the prefix list of list $\vec{x}$ with length $i-1$, and $\vec{x}|_{i+1}$ is the tail list of list $\vec{x}$ obtained by dropping off the first $i+1$ elements in $\vec{x}$, we have

$$\mathcal{B}_{\vec{x}\lceil_{i-1},x_{i+1},x_i,\vec{x}|_{i+1}}[\![t]\!]_\psi = \mathcal{B}_{\vec{x}\,y_1\,y_2\,\vec{z}}[\![t]\!]_{\psi \circ_{|\vec{x}|,|\vec{x}|}} < Pr^{|\vec{x}|}_{1,i-1}, pr_{|\vec{x}|,i+1}, pr_{|\vec{x}|,i}, Pr^{|\vec{x}|}_{i+2,|\vec{x}|} >$$

(ii) For $ft \in FT_m$ and $\{\vec{x}\} \supseteq Free(ft) \cap V$, let $1 \leq i \leq k$, we have

$$\begin{aligned}&\mathcal{B}_{\vec{x}\lceil_{i-1},x_{i+1},x_i,\vec{x}|_{i+1}}[\![ft]\!]_\psi \\ &= \mathcal{B}_{\vec{x}}[\![ft]\!]_{\psi \circ_{|\vec{x}|+m,|\vec{x}|+m}} < Pr^{|\vec{x}|+m}_{1,i-1}, pr_{|\vec{x}|+m,i+1}, pr_{|\vec{x}|+m,i}, Pr^{|\vec{x}|+m}_{i+2,|\vec{x}|+m} >\end{aligned}$$

Proof

It proceeds by structural induction with the laws of b-clones in Definition 5.3.2. □

Next, we are considering dropping off non-free variable y of terms or function terms in $\mathcal{B}_{\vec{x}}$, where y appears in $\vec{x}$.

Lemma 5.4.3.4 (elimination of non-free binding variables): *Let* **B** *be an iBA.*

(i) For $t \in T$ and $y \notin Free(t)$ and $\{\vec{x}\} \supseteq Free(t) \cap V$, we have

$$\mathcal{B}_{y\vec{x}}[\![t]\!]_\psi = \mathcal{B}_{\vec{x}}[\![t]\!]_{\psi \circ_{|\vec{x}|,|\vec{x}|+1}} < Pr^{|\vec{x}|+1}_{2,|\vec{x}|+1} >$$

(ii) For $ft \in FT_m$ and $y \notin Free(ft))$ and $\{\vec{x}\} \supseteq Free(ft) \cap V$, we have

$$\mathcal{B}_{y\vec{x}}[\![ft]\!]_\psi = \mathcal{B}_{\vec{x}}[\![ft]\!]_{\psi \circ_{|\vec{x}|+m,|\vec{x}|+m+1}} < Pr^{|\vec{x}|+m+1}_{2,|\vec{x}|+1} >$$

Proof It is similar to the proof of Lemma 5.4.3.3. □

Correspondingly, we consider adding non-free variable y of terms or function terms to $\mathcal{B}_{\vec{x}}$, where y does not appear in $\vec{x}$.

Corollary 5.4.3.5 (introduction of non-free binding variables): *Let* **B** *be an iBA.*

(i) For $t \in T$ and $y \notin Free(t)$ and $\{\vec{x}\} \supseteq Free(t) \cap V$, we have

$$\mathcal{B}_{\vec{x}}[\![t]\!]_{\psi} = \mathcal{B}_{y\vec{x}}[\![t]\!]_{\psi \circ |\vec{x}|+1,|\vec{x}|} < pr_{|\vec{x}|,1}, Pr^{|\vec{x}|}_{1,|\vec{x}|} >$$

(ii) For $ft \in FT_m$ and $y \notin Free(ft))$ and $\{\vec{x}\} \supseteq Free(ft) \cap V$, we have

$$\mathcal{B}_{\vec{x}}[\![ft]\!]_{\psi} = \mathcal{B}_{y\vec{x}}[\![ft]\!]_{\psi \circ |\vec{x}|+m+1,|\vec{x}|+m} < pr_{|\vec{x}|+m,1}, Pr^{|\vec{x}|+m}_{1,|\vec{x}|+m} >$$

Proof By Lemma 5.4.3.4 with the laws of b-clones in Definition 5.3.2. □

Now, we can prove that translation $\underline{tr}$ preserves the soundness. That is,

Theorem 5.4.3.6 (equality from b-clones to iBAs): *We have that* $\mathbf{D} \models \underline{t} \simeq \underline{u}$ *implies* $\mathbf{B} \models \underline{tr}[\![\underline{t}]\!] \simeq \underline{tr}[\![\underline{u}]\!]$, *where* **B** *is an iBA and* **D** *is a b-clone, i.e.* $\mathbf{B} = \mathbf{D}$.

Proof It is a result of applying Theorem 5.4.2.4 and the above lemmas from Lemma 5.4.3.3, to Corollary 5.4.3.5 with structural induction. □

5.5 Preservation of derivations

With Indistinguishability preservations of the translations, we hope to proceed further to obtain derivation preservations by the two translations. This will be the last step before we reach soundness and completeness of Equational Calculus $\vdash_{iBA}$ for iBAs. The deduction systems of both calculi of iBAs and of b-clones will be presented (in Definition 5.5.1.1 and Definition 5.5.1.4). Then, the preservation of derivations under each translation follows (Theorem 5.5.2.12 and Theorem 5.5.3.13).

The results in this section coupled with the results of previous sections will establish the soundness and completeness of $\vdash_{iBA}$ in next section (Section 5.6).

5.5.1 Calculus $\vdash_{iBA}$ and axioms Ax_b of b-clones

Let Γ_b (or simply Γ) be a set of BEs, where each element in Γ has the form of $p \simeq q$ such that either $p, q \in T$ or $p, q \in FT_k$ for some $k \geq 0$.

A family $\vec{\varrho}$ of substitution functions is *functional* iff $\forall f \in FV_m.\vec{\varrho}[f] \cap V = \emptyset \wedge \forall x \in V.\vec{\varrho}(x) \in V$ for all $m \geq 0$.

Then, calculus $\vdash_{iBA}$ is given below.

Definition 5.5.1.1 (calculus $\vdash_{iBA}$): *Calculus $\vdash_{iBA}$ is defined in the judgement form of*

$$\Gamma \vdash_{iBA} p \simeq q \text{ or } \Gamma \vdash p \simeq q$$

which is almost the same as calculus $\dot{\vdash}_{eBA}$ except the rule for subtitutions where the families of substitution functions can be arbitrary in $\dot{\vdash}_{eBA}$ but they must be functional in $\vdash_{iBA}$.

Since calculus $\vdash iBA$ has been given in Theorem 1.2.3.1, it will not be repeated here.

Similar to Chapter 2, we can introduce a function $\mathcal{M}_b$ such that $p \simeq q \in \mathcal{M}_b(\Gamma)$ iff $\Gamma \vdash_{iBA} p \simeq q$ without using cut rules. Apparently, $\mathcal{M}_b$ is monotonic with the usual set inclusion as the partial order. Furthermore, we can have that a Binding Equation (BE) $p \simeq q$ (or Δ) is derivable from Γ, written as $\Gamma \vdash_{iBA} \Delta$, iff $\Delta \in \sqcup\mathcal{M}_b(\Gamma)$.

Remark 5.5.1.2 ((func-sub)):

(i) A more generalized (func-sub) rule is to drop off the condition $\forall x \in V.\vec{\varrho}(x) \in V$ from functional substitutions. This generalized rule is named as (gen-func-sub).

(ii) A more restricted (func-sub) rule is

$$(pure\text{-}func\text{-}sub) \quad \frac{\Gamma \vdash_{iBA} p \simeq q}{\Gamma \vdash_{iBA} p[\vec{f} := \vec{ft}] \simeq q[\vec{f} := \vec{ft}]},$$

where $f_i \in FV_{m_i}$, $ft_i \in FT_{m_i}$ *and* $Free(ft_i) \cap V = \emptyset$ $(1 \leq i \leq |\vec{ft}|)$; *i.e. the family of substitution functions is also functional.*

(iii) An ordinary (func-sub) rule is

$$(ord\text{-}sub) \quad \frac{\Gamma \vdash_{iBA} t \simeq u}{\Gamma \vdash_{iBA} t[\vec{x} := \vec{v}] \simeq u[\vec{x} := \vec{v}]}$$

where $v_j, t, u \in T$ $(1 \leq j \leq |\vec{v}|)$.

(iv) We understand that (gen-func-sub) rule is derivable from (pure-func-sub) rule coupled with (ord-sub) rule. From Lemma 5.5.1.3 below, we have that (ord-sub) rule is derivable from (pure-func-sub) rule. From this, we have that (gen-func-sub) and (pure-func-sub) are equivalent, since (pure-func-sub) rule is obviously derivable from (gen-func-sub).

For convenience, we will use $\Gamma \vdash_{iBA} p_1 \simeq q_1, p_2 \simeq q_2, ..., p_n \simeq q_n$ as an abbreviation for list $\Gamma \vdash_{iBA} p_1 \simeq q_1$, $\Gamma \vdash_{iBA} p_2 \simeq q_2$, ..., $\Gamma \vdash_{iBA} p_n \simeq q_n$.

Lemma 5.5.1.3 ((ord-sub) rule): *(ord-sub) rule is derivable from (pure-func-sub) rule within* $\vdash_{iBA}$.

Proof

To simplify our proof, we introduce two equivalent rules for (pure-func-sub) and (gen-ord-sub) rule respectively as follows:

$$(gen\text{-}pure\text{-}func\text{-}sub) \quad \frac{\Gamma \vdash p \simeq q,\ ft_i \simeq fu_i\ (1 \leq i \leq j)}{\Gamma \vdash p[\vec{f} := \vec{ft}] \simeq q[\vec{f} := \vec{fu}]}$$

and

$$(gen\text{-}ord\text{-}sub) \quad \frac{\Gamma \vdash_{iBA} t_j \simeq u_j\ (1 \leq j \leq |\vec{t}| = |\vec{u}|), t \simeq u}{\Gamma \vdash_{iBA} t[\vec{x} := \vec{t}] \simeq u[\vec{x} := \vec{u}]}.$$

Thus, let $j = 1$, $p = f(\vec{t}, \vec{x'}\lfloor_k)$, $q = f(\vec{u}, \vec{x'}\lfloor_k)$, $ft_1 = \langle \vec{x'} : t \rangle$ and $fu_1 = \langle \vec{x'} : u \rangle$, where $\vec{x'}\lfloor_{|\vec{x}|} = \vec{x}$, $\{\vec{x'}\} \supseteq Free(t) \cup Free(u)$, in (pure-func-sub) rule. Then, (ord-sub) rule can be derived with (ξ) rule and (b-cmp-1) rule. For instance, $\Gamma \vdash_{iBA} p \simeq q$ by (b-cmp-1) rule and $\vdash_{iBA} ft_1 \simeq fu_1$ by (ξ) rule. □

As a consequence of Lemma 5.5.1.3, we are free to use (ord-sub) rule in $\vdash_{iBA}$. Also we should notice that under certain condition a parallel substitution can be replaced by two sequential ones, e.g. $p[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,] \simeq (p[\vec{f} := \vec{ft}])[\vec{x} := \vec{t}\,]$ is derivable, where

where $f_i \in FV_{m_i}$, $ft_i \in FT_{m_i}$ *and* $Free(ft_i) \cap V = \emptyset$ $(1 \leq i \leq |\vec{ft}|)$; *i.e. the family of substitution functions is also functional.*

(iii) An ordinary (func-sub) rule is

$$(ord\text{-}sub) \quad \frac{\Gamma \vdash_{iBA} t \simeq u}{\Gamma \vdash_{iBA} t[\vec{x} := \vec{v}] \simeq u[\vec{x} := \vec{v}]}$$

where $v_j, t, u \in T$ $(1 \leq j \leq |\vec{v}|)$.

(iv) We understand that (gen-func-sub) rule is derivable from (pure-func-sub) rule coupled with (ord-sub) rule. From Lemma 5.5.1.3 below, we have that (ord-sub) rule is derivable from (pure-func-sub) rule. From this, we have that (gen-func-sub) and (pure-func-sub) are equivalent, since (pure-func-sub) rule is obviously derivable from (gen-func-sub).

For convenience, we will use $\Gamma \vdash_{iBA} p_1 \simeq q_1, p_2 \simeq q_2, ..., p_n \simeq q_n$ as an abbreviation for list $\Gamma \vdash_{iBA} p_1 \simeq q_1$, $\Gamma \vdash_{iBA} p_2 \simeq q_2$, ..., $\Gamma \vdash_{iBA} p_n \simeq q_n$.

Lemma 5.5.1.3 ((ord-sub) rule): *(ord-sub) rule is derivable from (pure-func-sub) rule within* $\vdash_{iBA}$.

Proof

To simplify our proof, we introduce two equivalent rules for (pure-func-sub) and (gen-ord-sub) rule respectively as follows:

$$(gen\text{-}pure\text{-}func\text{-}sub) \quad \frac{\Gamma \vdash p \simeq q,\ ft_i \doteq fu_i\ (1 \leq i \leq j)}{\Gamma \vdash p[\vec{f} := \vec{ft}] \simeq q[\vec{f} := \vec{fu}]}$$

and

$$(gen\text{-}ord\text{-}sub) \quad \frac{\Gamma \vdash_{iBA} t_j \simeq u_j\ (1 \leq j \leq |\vec{t}| = |\vec{u}|), t \simeq u}{\Gamma \vdash_{iBA} t[\vec{x} := \vec{t}] \simeq u[\vec{x} := \vec{u}]}.$$

Thus, let $j = 1$, $p = f(\vec{t}, \vec{x'}\lfloor_k)$, $q = f(\vec{u}, \vec{x'}\lfloor_k)$, $ft_1 = \langle \vec{x'} : t \rangle$ and $fu_1 = \langle \vec{x'} : u \rangle$, where $\vec{x'}\lfloor_{|\vec{x}|} = \vec{x}$, $\{\vec{x'}\} \supseteq Free(t) \cup Free(u)$, in (pure-func-sub) rule. Then, (ord-sub) rule can be derived with (ξ) rule and (b-cmp-1) rule. For instance, $\Gamma \vdash_{iBA} p \simeq q$ by (b-cmp-1) rule and $\vdash_{iBA} ft_1 \simeq fu_1$ by (ξ) rule. □

As a consequence of Lemma 5.5.1.3, we are free to use (ord-sub) rule in $\vdash_{iBA}$. Also we should notice that under certain condition a parallel substitution can be replaced by two sequential ones, e.g. $p[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,] \simeq (p[\vec{f} := \vec{ft}])[\vec{x} := \vec{t}\,]$ is derivable, where

We say that $\underline{\Delta}$ is derivable from $\underline{\Gamma}$ if $\underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{\Delta}$

Remark 5.5.1.5: *From Theorem 2.2.6, we understand that calculus $\vdash_{EQ}$ with axioms $\underline{\Gamma} \cup Ax_b$ is variable index free for arbitrary Γ iff χ_0 can be expressed by other $\chi' \subseteq \chi - \chi_0$.*

With Remark 5.5.1.5, we assume that the signature $\underline{\Sigma}^{BO}$ (or simply Σ) under consideration always satisfies this condition (see Theorem 2.2.6). In turn, Σ^{BO} has to satisfy certain condition, since there is a 1-1 correspondence _ between Σ^{BO} and Σ in the comments before Definition 5.3.2. Actually, for iBAs as long as $FV_0 \neq \emptyset$, everything is going to be all right on this aspect.

5.5.2 Preservation of derivations under $\underline{tr}$

In this subsection, the central consideration is whether

$$\underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{p} \simeq \underline{q} \text{ implies } \underline{tr}[\![\underline{\Gamma}]\!] \vdash_{iBA} \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!],$$

where $\underline{tr}[\![\emptyset]\!] =_{df} \emptyset$ and $\underline{tr}[\![\{\underline{p} \simeq \underline{q}\} \cup \underline{\Gamma}]\!] =_{df} \{\underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!]\} \cup \underline{tr}[\![\underline{\Gamma}]\!]$. For this purpose, we firstly check the preservation of the four axiom schemas in Ax_b (see Lemma 5.5.2.3, Lemma 5.5.2.4, Lemma 5.5.2.5 and Lemma 5.5.2.9). In the entire proof of the preservation, we find out that (func-sub) rule in $\vdash_{iBA}$ is not required except in verifying the preservation of (crs-sub) rule in $\vdash_{EQ}$ (see Lemma 5.5.2.10). However, a weaken substitution rule (ord-sub) is required before Lemma 5.5.2.10. Nevertheless, (pure-func-sub) rule is actually required instead of (func-sub) rule. So, we define the "weaken" derivability without using (func-sub) rule as $\vdash_\alpha$ to distinguish from $\vdash_{iBA}$ as follows. Formally, $\Gamma \vdash_\alpha p \simeq q$ without (pure-func-sub) rule (but with (ord-sub) rule) iff a derivation of $\Gamma \vdash_{iBA} p \simeq q$ only involves (ord-sub) rule whenever substitutions are used in the derivation; i.e. no substitutions of function variables. We will give some necessary properties of "weaken" derivability $\vdash_\alpha$ from Lemma 5.5.2.1 to Lemma 5.5.2.5.

Lemma 5.5.2.1 (identity substitution and $\simeq$): *For $p \in T$ or $p \in FT_k$ for some $k \geq 0$, we have $\vdash_\alpha p \simeq p\vec{\imath}$.*

Proof

By structural induction.

We only show the case for function terms, i.e. $p = \langle \vec{x} : t \rangle$. Hence, $RHS = \langle \vec{y} : t[\vec{x} := \vec{y}] \rangle$, where y_j is the least $y \in V$ such that $\forall z \in Free(t).y \notin Free(\vec{t}[z]) \cup \{\vec{y}\lceil_{j-1}\}$. The side condition is equivalent to that y_j is the least $y \in V$ such that $y \notin Free(t) \cup \{\vec{y}\lceil_{j-1}\}$. Obviously it satisfies the side condition of α-conversion of Definition 5.5.1.1. □

The next lemma says that certain two substitutions can be replaced by a single one.

Lemma 5.5.2.2 (hierarchical reduction of substitutions): *For $p \in T$ or $p \in FT_k$ for some $k \geq 0$, we have that*

$$\vdash_\alpha (p[\vec{x} := \vec{t}\,])[\vec{y} := \vec{u}\,] \simeq p[\vec{x} := \vec{t}\,[\vec{y} := \vec{u}\,]\,],$$

where $\{\vec{y}\} \cap ((Free(t) \cap V) - \{\vec{x}\}) = \emptyset$.

Proof

By structural induction.

case x: there are two possibilities:

(i) either $x = x_i \in \{\vec{x}\}$

(ii) or $x \notin \{\vec{x}\}$.

For (i), we have $LHS = t_i\,[\vec{y} := \vec{u}\,] = RHS$.

For (ii), we have $LHS = x\,[\vec{y} := \vec{u}\,] = x = RHS$.

By reflection rule (in Definition 5.5.1.1), we got what we want.

case $f(\vec{t'})$: by structural hypothesis and (cmp-1) rule.

case $\langle \vec{x'} : t \rangle$: let $x_{i_1}, x_{i_2}, ..., x_{i_j}$ ($\vec{x^i}$ for short), be the string of all free variables of $\langle \vec{x'} : t \rangle$ in string $x_1, x_2, ..., x_m$. Then

$$\langle \vec{x'} : t \rangle[\vec{x} := \vec{t}\,] = \langle \vec{x'} : t \rangle[\vec{x_i} := \vec{t_i}] = \langle \vec{y'} : t[\vec{x_i}, \vec{x'} := \vec{t_i}, \vec{y'}] \rangle$$

where y'_j is the least $y' \in V$ such that for all $z \in Free(t)$,

$$y' \notin Free(\vec{t}[\vec{x_i} := \vec{t_i}][z]) \cup \{\vec{y}\lceil_{j-1}\}).$$

We can conclude that $y'_j \notin \bigcup_j Free(t_{i_j}) \cup \{\vec{y}\lceil_{j-1}\}$.

Let $y_{j_1}, y_{j_2}, ..., y_{j_n}$ ($\vec{y^j}$ for short), be the string of all variables occurring free at least in one term of string $t_{i_1}, t_{i_2}, ..., t_{i_j}$ in $y_1, y_2, ..., y_l$. Then by Lemma 5.1.10, we have

$$LHS = (\langle \vec{x'} : t\rangle[\vec{x_i} := \vec{t_i}])[\vec{y_j} := \vec{u_j}]$$
$$= \langle \vec{y''} : (t[\vec{x_i}, \vec{x'} := \vec{t_i}, \vec{y'}])[\vec{y_j}, \vec{y'} := \vec{u_j}, \vec{y''}]\rangle$$

where y''_j is the least $y'' \in V$ such that

$$\forall z \in Free(t\,[\vec{x_i}, \vec{x'} := \vec{t_i}, \vec{y'}].y'' \notin Free(\vec{t}[\vec{y_j} := \vec{u_j}][z]) \cup \{\vec{y''}\lceil_{j-1}\}$$

On the other hand, we have

$$RHS = \langle \vec{x'} : t\rangle[\vec{x_i} := \vec{t_i}[\vec{y} := \vec{u}]\,]$$
$$= \langle \vec{x'} : t\rangle[\vec{x_i} := \vec{t_i}[\vec{y_j} := \vec{u_j}]\,] = \langle \vec{x''} : t[\vec{x_i}, \vec{x'} := \vec{t_i}[\vec{y_j} := \vec{u_j}], \vec{x''}\,]\rangle$$

where x''_j is the least $x'' \in V$ such that for all $z \in Free(t)$,

$$x'' \notin Free([\vec{t}\vec{x_i} := \vec{t_i}[\vec{y_j} := \vec{u_j}]\,][z]) \cup \{\vec{x''}\lceil_{j-1}\}.$$

By careful examination, we know that $x''_j = y''_j$. Consequently,

$$RHS = \langle \vec{x''} : t[\vec{x_i}, \vec{x'} := \vec{t_i}[\vec{y_j} := \vec{u_j}], \vec{y'}[\vec{y_j}, \vec{y'} := \vec{u_j}, \vec{x''}]\,]\rangle$$

By structural hypothesis, we come to

$$\vdash_\alpha (t[\vec{x_i}, \vec{x'} := \vec{t_i}, \vec{y'}])[\vec{y_j}, \vec{y'} := \vec{u_j}, \vec{y''}] \simeq t[\vec{x_i}, \vec{x'} := \vec{t_i}[\vec{y_j} := \vec{u_j}], \vec{y'}[\vec{y_j}, \vec{y'} := \vec{u_j}, \vec{x''}]\,].$$

And by (ξ) rule, we will have $\Gamma \vdash_\alpha LHS \simeq RHS$.

case $\sigma(\vec{ft}, \vec{t})$: it is a result of using (cmp-2) with structural hypothesis. □

For (Ax_b-left-proj), we have Lemma 5.5.2.3.

Lemma 5.5.2.3 ((Ax_b-left-proj) schema): *For $1 \leq i \leq |\vec{p}|$, we have that*

$$\vdash_\alpha \underline{tr}[\underline{p_i}] \simeq \underline{tr}[\![\underline{o}_{|\vec{p}|,k}(\underline{pr}_{|\vec{p}|,i}, \vec{p})]\!],$$

where $\underline{p}_j \in \underline{T}_k$ $(1 \leq j \leq |\vec{\underline{p}}|)$.

Proof

$RHS = \underline{tr}[\![\, \underline{pr}_{|\vec{\underline{p}}|,i}]\!][\vec{x} := \underline{tr}[\![\, \vec{\underline{p}}]\!]\,]$

(where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.)

$= x_i[\vec{x} := \underline{tr}[\![\, \vec{\underline{p}}]\!]\,]$

(where x_i is the ith least element of V.)

$= \underline{tr}[\![\underline{p}_i]\!] = LHS.$

Then, by (b-rfl) rule, we get what we need. □

For $(Ax_b$-right-proj), we have Lemma 5.5.2.4.

Lemma 5.5.2.4 ($(Ax_b$-right-proj) schema): *For* $\underline{p} \in \underline{T}_k$ *for some* k, *we have*

$$\vdash_\alpha \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{o}_{k,k}(\underline{p}, \underline{Pr}^k_{1,k})]\!].$$

Proof

$RHS = \underline{tr}[\![\underline{p}]\!]\,[\vec{x} := \underline{tr}[\![\underline{Pr}^k_{1,k}]\!]\,]$

(where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.)

$= \underline{tr}[\![\underline{p}]\!]\,[\vec{x} := \vec{x}]$

So, we can get what we need by Lemma 5.5.2.1. □

Lemma 5.5.2.5 below will say that $(Ax_b$-assoc) is preserved too.

Lemma 5.5.2.5 ($(Ax_b$-assoc) schema):

$$\vdash_\alpha \underline{tr}[\![\underline{o}_{|\vec{\underline{q}}|,k}(\underline{o}_{|\vec{\underline{p}}|,|\vec{\underline{q}}|}(\underline{r}, \vec{\underline{p}})\vec{\underline{q}})]\!] \simeq \underline{tr}[\![\underline{o}_{|\vec{\underline{p}}|,|\vec{\underline{q}}|}(\underline{r}, \underline{o}_{|\vec{\underline{q}}|,k}(\vec{\underline{p}}, \vec{\underline{q}}))]\!].$$

Proof

1.

$LHS = \underline{tr}[\![\circ_{|\vec{p}|,|\vec{q}|}(\underline{r}, \vec{\underline{p}})]\!][\vec{x} := \underline{tr}[\![\vec{\underline{q}}]\!]]$

(where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.)

$= (\underline{tr}[\![\underline{r}]\!][\vec{x'} := \underline{tr}[\![\vec{\underline{p}}]\!]])[\vec{x} := \underline{tr}[\![\vec{\underline{q}}]\!]]$

where x'_j is the least $x' \in V$ such that $x' \notin \{\vec{x'}\lceil_{j-1}\}$.

2.

$RHS = \underline{tr}[\![\underline{r}]\!][\vec{y} := \underline{tr}[\![\circ_{|\vec{q}|,k}(\vec{\underline{p}}, \vec{\underline{q}})]\!]]$

(where y_j is the least $y \in V$ such that $y \notin \{\vec{y}\lceil_{j-1}\}$.)

$= \underline{tr}[\![\underline{r}]\!][\vec{y} := \vec{p'}]$

where $p'_i = \underline{tr}[\![\underline{p}_i]\!][\vec{y^i} := \underline{tr}[\![\vec{\underline{q}}]\!]]$ y^i_j is the least $y^i \in V$ such that $y^i \notin \{\vec{y^i}\lceil_{j-1}\}$, $(1 \leq i \leq |\vec{\underline{p}}|\ 1 \leq j \leq |\vec{\underline{q}}|)$.

3. we know that $x'_j = y_j$ for $1 \leq j \leq |\vec{\underline{p}}|$ and $x_j = y^i_j$ for $1 \leq i \leq |\vec{\underline{p}}|, 1 \leq j \leq |\vec{\underline{q}}|$. So, by Lemma 5.5.2.2 we have $\vdash_\alpha LHS \simeq RHS$. □

The "weaken" derivability is a relative derivability, i.e. relative to (ord-sub) rule. We extend this idea of relativity and introduce a "relative derivability" focusing on the substitution rule involved in derivations. Formally,

Definition 5.5.2.6 (relative substitution $\asymp^\Gamma$): $\vec{\varrho} \asymp^\Gamma \vec{\varrho'}$ *iff* $\Gamma \vdash_\alpha \vec{\varrho}[x] \simeq \vec{\varrho'}[x]$ *for each* $x \in V$ *and every* $f \in FV_m$ $(m \in Nat)$, $\vec{\varrho}[f] = \vec{\varrho'}[f] = f$.

Apparently, we have that if $\Gamma'_b \subseteq \Gamma$ then that $\vec{\varrho} \asymp^{\Gamma'_b} \vec{\varrho'}$ implies $\vec{\varrho} \asymp^\Gamma \vec{\varrho'}$. We let $\asymp$ be $\asymp^\emptyset$. The close relationship between $\asymp^\Gamma$ and $\vdash_\alpha$ is exemplified as follows.

Lemma 5.5.2.7 ($\asymp^\Gamma$ and $\vdash_\alpha$): $\vec{\varrho} \asymp^\Gamma \vec{\varrho'}$ *implies that* $\Gamma \vdash_\alpha p\vec{\varrho} \simeq p\vec{\varrho'}$.

Proof

By structural induction.

case x: obvious.

case $f(\vec{t})$: it is by structural hypothesis and compositional rule.

case $\langle \vec{x} : t \rangle$: $LHS = \langle \vec{y} : t\vec{\varrho}[\vec{x} := \vec{y}] \rangle$, where y_j is the least $y \in V$ such that

$$\forall z \in Free(t).y \notin (Free(\vec{\varrho}[z]) - \{\vec{x}\}) \cup \{\vec{y}\lceil_{j-1}\}.$$

and $RHS = \langle \vec{z} : t\vec{\varrho'}[\vec{x} := \vec{z}] \rangle$, where z_j is the least $z \in V$ such that

$$\forall y \in Free(t).z \notin (Free(\vec{\varrho}[y]) - \{\vec{x}\}) \cup \{\vec{z}\lceil_{j-1}\}.$$

Let x'_j be a $x' \in V$ such that

$$x' \notin \ Free(t\vec{\varrho}[\vec{x} := \vec{y}]) \cup Free(t\vec{\varrho'}[\vec{x} := \vec{z}]) \cup \{\vec{x'}\lceil_{j-1}\}$$

and $\vec{\varrho}^*$ be $\vec{\varrho}[\vec{x} := \vec{y}]$ and $\vec{\varrho'}^*$ be $\vec{\varrho'}[\vec{x} := \vec{z}]$. We will have $t\vec{\varrho}^* = t[\vec{z'} := \vec{\varrho}^*[\vec{z'}]\,]$, where $\vec{\varrho}^*[\vec{z'}] = \vec{\varrho}^*[z'_1], \vec{\varrho}^*[z'_2], ..., \vec{\varrho}^*[z'_k]$ and $\{\vec{z'}\} = Free(t) \cap V$.

Similarly, we have $t\vec{\varrho'}^* = t[\vec{z'} := \vec{\varrho'}^*[\vec{z'}]\,]$.

By (α), we get

(a) $\vdash_\alpha LHS \simeq \langle \vec{x'} : (t[\vec{z'} := \vec{\varrho}^*[\vec{z'}])[\vec{y} := \vec{x'}] \rangle$,

(b) $\vdash_\alpha RHS \simeq \langle \vec{x'} : (t[\vec{z'} := \vec{\varrho'}^*[\vec{z'}])[\vec{z} := \vec{x'}] \rangle$.

By Lemma 5.5.2.2 we have

(c) $\vdash_\alpha RHS\ of\ (a) \simeq \langle \vec{x'} : t[\vec{z'} := \vec{\varrho}^*[\vec{z'}][\vec{y} := \vec{x'}]\,] \rangle$, and

(d) $\vdash_\alpha RHS\ of\ (b) \simeq \langle \vec{x'} : t[\vec{z'} := \vec{\varrho'}^*[\vec{z'}][\vec{z} := \vec{x'}]\,] \rangle$.

For z'_j, we have

(e) if $z'_j \in \{\vec{x}\}$, say $z'_j = x_i$, then $\vec{\varrho}^*[z'_j] = y_i$ and $\vec{\varrho'}^*[z'_j] = z_i$. i.e.

$\vec{\varrho}^*[z'_j][\vec{y} := \vec{x'}] = x'_i$ and $\vec{\varrho'}^*[z'_j][\vec{z} := \vec{x'}] = x'_i$.

(f) If $z'_j \in \{\vec{x}\}$, then $\vec{\varrho}^*[z'_j] = \vec{\varrho}[z'_j]$ and $\vec{\varrho'}^*[z'_j] = \vec{\varrho'}[z'_j]$ (they are α-convertable).

Hence, by structural hypothesis we have

$$\Gamma \vdash_\alpha \langle \vec{x'} : t[\vec{z'} := \vec{\varrho}^*[\vec{z'}][\vec{y} := \vec{x'}]\,] \rangle \simeq \langle \vec{x'} : t[\vec{z'} := \vec{\varrho'}^*[\vec{z'}][\vec{z} := \vec{x'}]\,] \rangle.$$

Then, by (ξ) and (b-trs) rule, we arrive at $\Gamma \vdash_\alpha LHS \simeq RHS$.

case $\sigma(\vec{ft}, \vec{t})$: it is a result of using structural hypothesis and compositional rule.

□

To prove the preservation of (Ax_b-unif) schema (see Lemma 5.5.2.9), we need to extend the result of Lemma 5.5.2.2 to include function parts as follows.

Lemma 5.5.2.8 (hierarchical reduction of substitutions): *For $p \in T$ or $p \in FT_k$ ($k \in Nat$),*

$$\vdash_\alpha (p[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,])[\vec{y} := \vec{u}] \simeq p[\vec{f}, \vec{x} := \vec{ft}, \vec{t}[\vec{y} := \vec{u}]\,],$$

where $\{\vec{x}\} \cup \{\vec{f}\} \supseteq Free(p)$ and $Free(ft_i) \cap V = \emptyset$.

Proof

By structural induction.

case x: Assume $x = x_j \in \{\vec{x}\}$. Then $LHS = t_j[\vec{y} := \vec{u}] = RHS$ (i.e. by reflectivity).

case $f(\vec{t'})$: if $f \notin \{\vec{f}\}$, it is obvious by structural hypothesis. So, let $f = f_i \in \{\vec{f}\}$.

$LHS = (ft_i||_{k_i}[\vec{x^i} := \vec{t'}[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,]\,])[\vec{y} := \vec{u}]$

where $\vec{x^i} = ft_i|_{k_i}$ and $k_i = k$.

By Lemma 5.5.2.2, Lemma 5.5.2.7 and (b-trs) rule, we have

$$\Gamma \vdash_\alpha LHS \simeq ft_i||_{k_i}[\vec{x^i} := (\vec{t'}[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,])[\vec{y} := \vec{u}]\,]$$

(Note that $Free(ft_i) \cap V = \emptyset$).

Similarly we can get

$$\Gamma \vdash_\alpha RHS \simeq ft||_{k_i}[\vec{x^i} := \vec{t'}[\vec{f}, \vec{x} := \vec{ft}, \vec{t}[\vec{y} := \vec{u}]\,]\,]$$

By structural hypothesis, we have the following derivable

$$\vec{t}[\vec{x^i} := (\vec{t'}[\vec{f}, \vec{x} := \vec{ft}, \vec{t}\,])[\vec{y} := \vec{u}]\,] \asymp \vec{t}[\vec{x^i} := \vec{t'}[\vec{f}, \vec{x} := \vec{ft}, \vec{t}[\vec{y} := \vec{u}]\,]\,].$$

By Lemma 5.5.2.7, we reach $\Gamma \vdash_\alpha LHS \simeq RHS$.

case $\langle \vec{x'} : t \rangle$: Similar to the proof of Lemma 5.5.2.2, we let $x_{i_1}, x_{i_2}, ..., x_{i_n}$ ($\vec{x_i}$ for short) be the string of all free variables of p in $x_1, x_2, ..., x_k$. Then

$$p[\vec{f}, \vec{x} := \vec{ft}, \vec{t}] = p[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}]$$
$$= \langle \vec{y'} : t[\vec{f}, \vec{x^i}, \vec{x'} := \vec{ft}, \vec{t_i}, \vec{y'}] \rangle$$

where y'_j is the least $y' \in V$ such that for all $z \in Free(t)$,

$$y' \notin Free(\vec{\tau}[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}][z]) \cup \{\vec{y'}\lceil_{j-1}\}.$$

Consequently we have that $y'_j \notin \bigcup_j Free(t_{i_j}) \cup \{\vec{y'}\lceil_{j-1}\}$.

Let $y_{j_1}, y_{j_2}, ..., y_{j_{n'}}$ ($\vec{y^j}$ for short) be the string of variables in $y_1, y_2, ..., y_k$ occurring free in at least one of $t_{i_1}, t_{i_2}, ..., t_{i_n}$. Then

$$LHS = (p[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}])[\vec{y_j} := \vec{u_j}]$$
$$= \langle \vec{y''} : (t[\vec{f}, \vec{x^i}, \vec{x'} := \vec{ft}, \vec{t_i}, \vec{y'}])[\vec{y_j}, \vec{y'} := \vec{u_j}, \vec{y''}] \rangle$$

where y''_j is the least $y'' \in V$ such that

$$\forall z \in Free(t\,[\vec{f}, \vec{x^i}, \vec{x'} := \vec{ft}, \vec{t_i}, \vec{y'}]).y'' \notin Free(\vec{\tau}[\vec{y_j} := \vec{u_j}][z]) \cup \{\vec{y''}\lceil_{j-1}\}.$$

Accordingly we have

$$RHS = p[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}[\vec{y} := \vec{u}]\,]$$
$$= p[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}[\vec{y_j} := \vec{u_j}]\,] = \langle \vec{x''} : t[\vec{f}, \vec{x^i}, \vec{x'} := \vec{ft}, \vec{t_i}[\vec{y_j} := \vec{u_j}], \vec{x''}] \rangle$$

where x''_j is the least $x'' \in V$ such that for all $z \in Free(t)$,

$$x'' \notin Free(\vec{\tau}[\vec{f}, \vec{x^i} := \vec{ft}, \vec{t_i}[\vec{y_j} := \vec{u_j}]\,][z]) \cup \{\vec{x''}\lceil_{j-1}\}.$$

By carefully examining the side conditions, we understand that $x''_j = y''_j$. So, We come to what we need by structural hypothesis, and (b-ξ-rule).

case $\sigma(\vec{ft}, \vec{t'})$: we can obtain what we want by structural hypothesis and compositional rule. □

Lastly, (Ax_b-unif) schema will be preserved. Formally

Lemma 5.5.2.9 ((Ax_b-unif) schema):

$$\vdash_\alpha \underline{tr}[\![\underline{o}_{|\vec{q}|,l}(\underline{o}_{|\vec{q}|}(\underline{\vec{p}}), \underline{\vec{q}})]\!] \simeq \underline{tr}[\![\underline{o}_l(\underline{\vec{p'}}, \underline{\vec{p''}})]\!]$$

where $\underline{p}'_i = \underline{o}_{|\vec{q}|+m_i,l+m_i}(\underline{p}_i, \underline{o}_{l,l+m_i}(\vec{q}, \underline{Pr}_{1,l}^{l+m_i}), \underline{Pr}_{l+1,l+m_i}^{l+m_i})$ *for* $1 \leq i \leq \ell$, $\underline{p}''_j = \underline{o}_{|\vec{q}|,l}(\underline{p}_{\ell+j}, \vec{q})$ *for* $1 \leq j \leq n$.

Proof

Remind you that $\vec{x}\lceil_n = x_1, x_2, ..., x_n$ and $\vec{x}\lfloor_n = x_{n+1}, x_{n+2}, ..., x_{|\vec{x}|}$ $(0 \leq n \leq |\vec{x}|)$. Also, we are going to use a lot of shorthands for those formulars which cannot express inside one line and/or one page if otherwise, especially in this proof. They might be not quite well-defined. But if you read them with the specific context in mind and compatibility in hand, you should be able to read them correctly.

1.

$LHS = \underline{tr}[\![\underline{o}_k(\vec{p})]\!][\vec{x} := \underline{tr}[\![\vec{q}]\!]\,]$

(where x_j is the least $x \in V$ such that $x \notin \{\vec{x}\lceil_{j-1}\}$.)

$= \sigma(\vec{ft}, \vec{t})[\vec{x} := \underline{tr}[\![\vec{q}]\!]\,]$

(where $ft_i = \langle \vec{y^i} : \underline{tr}[\![\underline{p}_i]\!]\rangle$, $t_j = \underline{tr}[\![\underline{p}_{|\vec{m}|+j}]\!]$, and y^i_j is the least y^i such that $y^i \notin \{\vec{y^i}\lceil_{j-1}\}$ and $1 \leq i \leq \ell$ and $1 \leq j \leq |\vec{q}| + m_i$.)

$= \sigma(\vec{ft'}, \vec{t'})$

where $ft'_i = \langle \vec{y^i} : \underline{tr}[\![\underline{p}_i]\!]\rangle[\vec{x} := \underline{tr}[\![\vec{q}]\!]\,]$, $t'_j = \underline{tr}[\![\underline{p}_{|\vec{m}|+j}]\!][\vec{x} := \underline{tr}[\![\vec{q}]\!]\,]$

2.

$RHS = \sigma(\vec{fu}, \vec{u})$

where

$$fu_i = \langle \vec{zi} : \underline{tr}[\![\underline{o}_{|\vec{q}|+\vec{m},l+\vec{m}}(\underline{p}_i, \underline{o}_{l,l+m_i}(\vec{q}, \underline{Pr}_{1,l}^{l+m_i}), \underline{Pr}_{l+1,l+m_i}^{l+m_i})]\!]\rangle,$$

and $u_j = \underline{tr}[\![\underline{o}_{|\vec{q}|,l}(\underline{p}_{\ell+j}, \vec{q})]\!]$, and z^i_j is the least $z^i \in V$ such that $z^i \notin \{\vec{z^i}\lceil_{j-1}\}$.

$= \sigma(\vec{fu'}, \vec{u'})$

where $fu'_i = \langle \vec{z^i} : \underline{tr}[\![\underline{p}_i]\!][\vec{x^i} := \underline{tr}[\![\underline{o}_{l,l+m_i}(\vec{q}, \underline{Pr}_{1,l}^{l+m_i})]\!], \underline{tr}[\![\underline{Pr}_{l+1,l+m_i}^{l+m_i}]\!]\,]\rangle$, $u'_j = \underline{tr}[\![\underline{p}_{\ell+j}]\!][x^{|\vec{m}|+j} := \underline{tr}[\![\vec{q}]\!]\,]$, and x^i_j is the least $x^i \in V$ such that $x^i \notin \{\vec{x^i}\lceil_{j-1}\}$.

$= \sigma(\vec{fu''}, \vec{u''})$

where $fu_i'' = \langle \vec{z^i} : \underline{tr}[\![\underline{p}_i]\!][\vec{x^i} := \vec{q'}, \vec{x^i}\lfloor_{|\vec{q}|}]\rangle$ such that $q_j' = \underline{tr}[\![\underline{q}_j]\!][\vec{x^i} := \underline{tr}[\![\underline{Pr}_{1,l}^{l+\vec{m}}]\!]]$, $u_j'' = u_j'$, and x_j'' is the least $x'' \in V$ such that $x'' \notin \{\vec{x^i}\lceil_{j-1}\}$, x_j is the jth least element of V, and the length of $\vec{z^i}$ is $l + m_i$.

$= \sigma(f\vec{u'''}, \vec{u'''})$

where $fu_i''' = \langle \vec{z^i} : \underline{tr}[\![\underline{p}_i]\!][\vec{x^i} := \vec{q'''}, \vec{x^i}\lfloor_{|\vec{q}|}]\rangle$ such that $q_j''' = \underline{tr}[\![\underline{q}_j]\!][\vec{x^i} := \vec{x}\lceil_l]$, and $u_j''' = u_j'$.

(since $x_j'^i = x_j$ for $1 \le i \le |\vec{q}|$ and $1 \le j \le l$ and $x^i_{|\vec{q}|+j} = x_{|\vec{q}|+j}$ for $1 \le i \le \ell$, $1 \le j \le |\vec{p}| - \ell$)

$= \sigma(f\vec{u^*}, \vec{u^*})$

where $fu_i^* = \langle \vec{z^i} : \underline{tr}[\![\underline{p}_i]\!][\vec{x^i} := \vec{q^*}, \vec{x^i}\lfloor_{|\vec{q}|}]\rangle$ such that $q_j^* = \underline{tr}[\![\underline{q}_j]\!]$, and $u_j^* = u_j'$.

(since $x_j^i = x_j$ for $1 \le i \le |\vec{p}|$, $1 \le j \le |\vec{q}|$,)

$= \sigma(f\vec{u^{*'}}, \vec{u^{*'}})$

where $fu_i^{*'} = \langle \vec{z^i} : \underline{tr}[\![\underline{p}_i]\!][\vec{x} := \vec{q^*}, \vec{x}\lfloor_{|\vec{q}|}]\rangle$ such that $q_j^{*'} = \underline{tr}[\![\underline{q}_j]\!]\vec{i}$, and $u_j^{*'} = u_j'$.

3. By Lemma 5.5.2.7, we get what we require since $y^i_{|\vec{q}|+j} = z^i_{|\vec{q}|+j} = x_{|\vec{q}|+j}$ for $1 \le i \le \ell$ and $1 \le j \le n_i$. □

For (crs-sub) rule, Lemma 5.5.2.10 says that (crs-sub) in b-clones can be replaced by (pure-func-sub) in iBAs. We discover that this replacement does not involve substitutions of function variables. In turn, it justifies of the previous introduction of the relative derivability, say $\vdash_\alpha$ and $\asymp^\Gamma$.

Lemma 5.5.2.10 (substitution replacement): *We have*

$$\underline{tr}[\![\Gamma]\!] \vdash_\alpha \underline{tr}[\![\underline{q}]\!][\vec{f} := \vec{ft}] \simeq \underline{tr}[\![\underline{q}[\vec{p}/\vec{f}]]\!],$$

where $ft_i = \langle \vec{x}^{k_i} : \underline{tr}[\![\underline{p}_i]\!]\rangle$ *and* $\underline{p}_i \in \underline{T}_{k_i}$ $(1 \le i \le |\vec{p}| = |\vec{f}|)$, *and* $\{\vec{f}\} \supseteq Free(\underline{p})$.

Proof

By structural induction on $\underline{q}$.

case $\underline{pr}_{k,i}$: $LHS = x_i = RHS$, where x_i is the ith least element in V.

case $\underline{f} \in \chi_k$: If $\underline{f} \notin \{\vec{\underline{f}}\}$, then it is trivial. So, let $\underline{f} = \underline{f}_i \in \{\vec{\underline{f}}\}$ and $k = k_i$.

$$LHS = f(\vec{x})[\vec{f} := \vec{ft}] = \underline{tr}[\![\underline{p}_i]\!][\vec{x^i} := \vec{x}[\vec{f} := \vec{ft}]\,] = \underline{tr}[\![\underline{p}_i]\!]\vec{i}$$

On the other hand, $RHS = \underline{tr}[\![\underline{p}_i]\!]$. We come to what we need by Lemma 5.5.2.2.

case $\sigma_k(\vec{\underline{q}}^{\ell}, \vec{r}^{k})$: let $fu_i = \langle \vec{x}^{k+m_i}\lfloor_k \colon \underline{tr}[\![\underline{q}_i]\!]\rangle$ $(1 \le i \le \ell)$ and $u_j = \underline{tr}[\![\underline{r}_j]\!]$ $(1 \le j \le l)$.

$$LHS = \sigma(\vec{fu}, \vec{u})[\vec{f} := \vec{ft}] = \sigma(\vec{fu}[\vec{f} := \vec{ft}], \vec{u}[\vec{f} := \vec{ft}])$$

Similarly, we have

$$RHS = \underline{tr}[\![\underline{\sigma}_k(\vec{\underline{q}}[\vec{\underline{p}}/\vec{\underline{f}}], \vec{r}[\vec{\underline{p}}/\vec{\underline{f}}])]\!] = \sigma(\langle \vec{x}^{k+\vec{m}}\lfloor_k \colon \underline{tr}[\![\vec{\underline{q}}[\vec{\underline{p}}/\vec{\underline{f}}]\,]\!]\rangle, \underline{tr}[\![\vec{r}[\vec{\underline{p}}/\vec{\underline{f}}]\,]\!])$$

For each i, we know $\vec{x}^{k+m_i}\lfloor_k = x_{k+1}, x_{k+2}, \ldots, x_{k+m_i}$ and let $\vec{x^i}\lfloor_k = x^i_{k+1}, x^i_{k+2}, \ldots, x^i_{k+m_i}$, by (α) and Lemma 5.5.2.8, we have the following derivable.

$$\vdash_\alpha \langle \vec{x}^{k+m_i}\lfloor_k \colon \underline{tr}[\![\underline{q}_i]\!][\vec{f} := \vec{ft}]\rangle \simeq \langle \vec{x^i}\lfloor_k \colon (\underline{tr}[\![\underline{q}_i]\!][\vec{f} := \vec{ft}])[\vec{x}^{k+m_i}\lfloor_k := \vec{x^i}\lfloor_k]\rangle,$$

where the first k elements of $\vec{x^i}$ are the same as the first k elements of $\vec{x}^{k+m_i}$ and x^i_{k+j} is the least $x^i \in V$ such that for all $z \in Free(\underline{tr}[\![\underline{q}_i]\!])$,

$$x^i \notin Free(\vec{i}[\vec{f} := \vec{ft}][z]) \cup \{\vec{x^i}^{k+j-1}\lfloor_k\}.$$

$\vdash_\alpha RHS$ of the above $\simeq \langle \vec{x^i}\lfloor_k \colon \underline{tr}[\![\underline{q}_i]\!][\vec{f}, \vec{x}^{k+m_i}\lfloor_k := \vec{ft}, \vec{x^i}\lfloor_k]\rangle$

(since we know that for $1 \le j \le |\vec{p}|$, $Free(\langle \vec{x}^{k_j} : \underline{tr}[\![\underline{p}_j]\!]\rangle) = \emptyset$.)

At last, by structural hypothesis, (α) and compositional rules, we obtain what we want $\underline{tr}[\![\Gamma]\!] \vdash_\alpha LHS \simeq RHS$. □

With the above lemma, we are to verify that(crs-sub) rule in Definition 5.5.1.4 is preserved. That is,

Lemma 5.5.2.11 (from (crs-sub) to (pure-func-sub)): *If* $\underline{tr}[\![\Gamma]\!] \vdash_{iBA} \underline{tr}[\![p]\!] \simeq \underline{tr}[\![q]\!]$, *then* $\underline{tr}[\![\Gamma]\!] \vdash_{iBA} \underline{tr}[\![p[\vec{r}/\vec{f}]\,]\!] \simeq \underline{tr}[\![q[\vec{r}/\vec{f}]\,]\!]$, *where* $\{\vec{f}\} \supseteq Free(p) \cup Free(q)$.

Proof

Let $ft_i = \langle x_1, x_2, \ldots, x_{k_i} : \underline{tr}[\![r_i]\!]\rangle$ and Γ be $\underline{tr}[\![\Gamma]\!]$. Thus,

- by (func-b-sub), we know that the precondition implies that

  $\Gamma \vdash_{iBA} \underline{tr}[\![\underline{p}]\!][\vec{f} := \vec{ft}] \simeq \underline{tr}[\![\underline{q}]\!][\vec{f} := \vec{ft}]$

- By Lemma 5.5.2.10 and (b-trs), it is resulted in what we need. □

To sum up the result so far, we come to the point to claim the preservation of derivations for $\underline{tr}$. Formally,

Theorem 5.5.2.12 (derivability from b-clones to iBAs): *Translation* $\underline{tr}$ *preserves derivations, i.e. if* $\underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{p} \simeq \underline{q}$, *then* $\underline{tr}[\![\underline{\Gamma}]\!] \vdash_{iBA} \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!]$.

Proof

By induction with lemmas such as Lemma 5.5.2.1, Lemma 5.5.2.4, Lemma 5.5.2.5, Lemma 5.5.2.10, and Lemma 5.5.2.11, we come to what we want. □

As a summary of this subsection, we have come to

$$(5.5.2.*) \quad \underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{p} \simeq \underline{q} \text{ implies } \underline{tr}[\![\underline{\Gamma}]\!] \vdash_{iBA} \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!]$$

The next subsection is going to attack a kind of the reversed implication of the above, which will involves another translation.

5.5.3 Preservation of derivations under $tr_{\vec{x}}$

In order to get the intended preservation (see Theorem 5.5.3.13), we have to do some delicated work. The most important thing is to show that the derivability is independent from all possible ways of binding ordinary variables (see Lemma 5.5.3.8).

Since the difference between Section 5.4 and here is the difference between semantic one and syntactic one, we need to have a syntactic version of Lemma 5.4.3.3, i.e. to show that derivabilities are invariant under variable permutations.

Lemma 5.5.3.1 (permutaion of variables):

(i) For $t \in T^{(\jmath)}$, *if* $\{\vec{x}\} \supseteq Free(t) \cap V$, *then*

$$\underline{\Gamma} \cup Ax_b \vdash_{\Sigma} tr_{\vec{x}\lceil_{i-1}, x_{i+1}, x_i, \vec{x}\lfloor_{i+1}}[\![t]\!] \simeq \underline{o}_{|\vec{x}|,|\vec{x}|}(tr_{\vec{x}}[\![t]\!], \underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{pr}_{|\vec{x}|,i+1}\underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{|\vec{x}|}_{i+2,|\vec{x}|}).$$

(ii) For $ft \in FT_m^{(j)}$ for some $m \geq 0$, if $\{\vec{x}\} \supseteq Free(ft) \cap V$, then

$$\Sigma \cup Ax_b \vdash_{EQ} tr_{\vec{x}\lceil_{i-1},x_{i+1},x_i,\vec{x}\lfloor_{i+1}}[\![ft]\!]$$
$$\simeq \underline{o}_{|\vec{x}|+m,|\vec{x}|+m}(tr_{\vec{x}}[\![ft]\!], \underline{Pr}_{1,i-1}^{|\vec{x}|+m}, \underline{pr}_{|\vec{x}|+m,i+1}, \underline{pr}_{|\vec{x}|+m,i}, \underline{Pr}_{i+2,|\vec{x}|+m}^{|\vec{x}|+m}).$$

Proof By structural induction.

(i) case x: $LHS = \begin{cases} \underline{pr}_{|\vec{x}|,i+1} & if\ x = x_i \\ \underline{pr}_{|\vec{x}|,i} & if\ x = x_{i+1} \\ \underline{pr}_{|\vec{x}|,j} & if\ x = x_j \in \{\vec{x}\lceil_{i-1}, \vec{x}\lfloor_{i+2}\} \end{cases}$ and

$$RHS = \begin{cases} \underline{o}_{|\vec{x}|,|\vec{x}|}(\underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{1,i-1}^{|\vec{x}|}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{i+2,|\vec{x}|}^{|\vec{x}|}) & if\ x = x_i \\ \underline{o}_{|\vec{x}|,|\vec{x}|}(\underline{pr}_{|\vec{x}|,i+1}, \underline{Pr}_{1,i-1}^{|\vec{x}|}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{i+2,|\vec{x}|}^{|\vec{x}|}) & if\ x = x_{i+1} \\ \underline{o}_{|\vec{x}|,|\vec{x}|}(\underline{pr}_{|\vec{x}|,j}, \underline{Pr}_{1,i-1}^{|\vec{x}|}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{i+2,|\vec{x}|}^{|\vec{x}|}) & if\ x = x_j \in \{\vec{x}\lceil_{i-1}, \vec{x}\lfloor_{i+1}\} \end{cases}$$

Hence, by (Ax_b-left-proj) schema and (eq-trs) we will have

$$\vdash_{EQ} RHS \simeq \begin{cases} \underline{pr}_{|\vec{x}|,i+1} & if\ x = x_i \\ \underline{pr}_{|\vec{x}|,i} & if\ x = x_{i+1} \\ \underline{pr}_{|\vec{x}|,j} & if\ x = x_j \in \{\vec{x}\lceil_{i-1}, \vec{x}\lfloor_{i+1}\} \end{cases}$$

(ii) case $f(\vec{t})$:

$$LHS = \underline{o}_{m,|\vec{x}|}(\underline{f}, tr_{\vec{x}\lceil_{i-1},x_{i+1},x_i,\vec{x}\lfloor_{i+1}}[\![\vec{t}]\!])$$
$$= \underline{o}_{m,|\vec{x}|}(\underline{f}, \underline{o}_{|\vec{x}|,|\vec{x}|}(tr_{\vec{x}}[\![\vec{t}]\!], \underline{Pr}_{1,i-1}^{|\vec{x}|}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{i+2,|\vec{x}|}^{|\vec{x}|}))$$

On the other hand,

$$RHS = \underline{o}_{|\vec{x}|,|\vec{x}|}(\underline{o}_{m,|\vec{x}|}(\underline{f}, tr_{\vec{x}}[\![\vec{t}]\!]), \underline{Pr}_{1,i-1}^{|\vec{x}|}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}_{i+2,|\vec{x}|}^{|\vec{x}|})$$

So, by (Ax_b-assoc) we get $\vdash_{EQ} LHS \simeq RHS$.

(iii) case $\langle \vec{y} : t \rangle$: $LHS = tr_{\vec{x}\lceil_{i-1},x_{i+1},x_i,\vec{x}\lfloor_{i+1},\vec{z}}[\![t[\vec{y} := \vec{z}]\ ]\!]$

(where z_j is the least $z \in V$ such that $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{z\vec{z}\lceil_{j-1}\}$)

and

$$RHS = \underline{o}_{m+|\vec{x}|,|\vec{x}|+m}(tr_{\vec{x},\vec{z}}[\![t[\vec{y} := \vec{z'}]\ ]\!], \underline{Pr}_{1,i-1}^{|\vec{x}|+m}, \underline{pr}_{|\vec{x}|+m,i+1}, \underline{pr}_{|\vec{x}|+m,i}, \underline{Pr}_{i+2,|\vec{x}|+m}^{|\vec{x}|+m})$$

(where z'_j is the least $z' \in V$ such that $z' \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{z'_1, z'_2, ..., z'_{j-1}\}$)

By examining both side conditions, we come to $z'_j = z_j$. Therefore by induction hypothesis we get $\vdash_{EQ} LHS \simeq RHS$.

(iv) case $\sigma(\vec{ft}, \vec{t})$:

$$LHS = \underline{\sigma}_{|\vec{x}|}(tr_{\vec{x}\lceil_{i-1}, x_{i+1}, x_i, \vec{x}\rfloor_{i+1}}[\![\vec{ft}]\!], tr_{\vec{x}\lceil_{i-1}, x_{i+1}, x_i, \vec{x}\rfloor_{i+1}}[\![\vec{t}]\!])$$

and

$$RHS = \underline{\circ}_{|\vec{x}|,|\vec{x}|}(tr_{\vec{x}}[\![\sigma(\vec{ft}, \vec{t})]\!], \underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{|\vec{x}|}_{i+2,|\vec{x}|})$$
$$= \underline{\circ}_{|\vec{x}|,|\vec{x}|}(\underline{\sigma}_{|\vec{x}|}(tr_{\vec{x}}[\![\vec{ft}]\!], tr_{\vec{x}}[\![\vec{t}]\!]), \underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{|\vec{x}|}_{i+2,|\vec{x}|})$$

By induction hypothesis and (eq-cmp) rule, we get $\vdash_{EQ} LHS \simeq \underline{\sigma}_{|\vec{x}|}(\vec{fu}, \vec{u})$,

where $fu_k = \underline{\circ}_{|\vec{x}|+m_k,|\vec{x}|+m_k}(tr_{\vec{x}}[\![ft_k]\!], \underline{Pr}^{|\vec{x}|+m_k}_{1,i-1}, \underline{pr}_{|\vec{x}|+m_k,i+1}, \underline{pr}_{|\vec{x}|+m_k,i}, \underline{Pr}^{|\vec{x}|+m_k}_{i+2,|\vec{x}|+m_k})$
and $u_j = \underline{\circ}_{|\vec{x}|,|\vec{x}|}(tr_{\vec{x}}[\![t_j]\!], \underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{|\vec{x}|}_{i+2,|\vec{x}|})$.

By (Ax_b-unif) and (eq-trs), $\vdash_{EQ} RHS \simeq \underline{\sigma}_{|\vec{x}|}(\vec{fv}, \vec{v})$

where

$$fv_k = \underline{\circ}_{m'_k,m'_k}(tr_{\vec{x}}[\![ft_k]\!], \underline{\circ}_{|\vec{x}|,m'_k}(\underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{Pr}^{m'_k}_{1,|\vec{x}|}), \underline{\circ}_{|\vec{x}|,m'_k}(\underline{pr}_{|\vec{x}|,i+1}, \underline{Pr}^{m'_k}_{1,|\vec{x}|}), \underline{\circ}_{|\vec{x}|,m'_k}(\underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{m'_k}_{1,|\vec{x}|}), \underline{\circ}_{|\vec{x}}$$

and $m'_k = |\vec{x}| + m_k$, and $v_j = \underline{\circ}_{|\vec{x}|,|\vec{x}|}(tr_{\vec{x}}[\![t_j]\!], \underline{Pr}^{|\vec{x}|}_{1,i-1}, \underline{pr}_{|\vec{x}|,i+1}, \underline{pr}_{|\vec{x}|,i}, \underline{Pr}^{|\vec{x}|}_{i+2,|\vec{x}|})$.

By (Ax_b-left-proj), (eq-cmp) and (eq-trs), we reach $\vdash_{EQ} LHS \simeq RHS$. □

Analogous to Lemma 5.4.2.1, we are going to consider the replacement of binding variables, which is a preliminary attempt to attack (α) conversions.

Lemma 5.5.3.2: *For all $j \geq 0$, given $p \in T^{(j)}$ and/or $p \in FT^{(j)}_k$ for some $k \geq 0$, and for all $m \geq 0$, $\vdash_{EQ} tr_{\vec{x},\vec{z}}[\![p[\vec{y} := \vec{z}]\,]\!] \simeq tr_{\vec{x},\vec{z}}[\![(p[\vec{y} := \vec{y'}])[\vec{y'} := \vec{z}]\,]\!]$, where y'_j is a $y' \in V$ such that $y' \notin Free(p) - \{\vec{y}\} \cup \{\vec{y'}\lceil_{j-1}\}$ and z_j is the least $z \in V$ such that $z \notin Free(p) - \{\vec{y}\} \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$.*

Proof

It is proved by induction on j. We only show the case for function terms.

case $\langle \vec{x'} : t \rangle$: let $y_{i_1}, y_{i_2}, ..., y_{i_l}$, $\vec{y_i}$ for short, be all free variable of $y_1, y_2, ..., y_m$ in $\langle \vec{x'} : t \rangle$.

$$LHS = tr_{\vec{x},\vec{z}}[\![\langle \vec{x''} : t[\vec{y_i}, \vec{x'} := \vec{z_i}, \vec{x''}] \rangle]\!]$$

where x''_j is the least $x'' \in V$ such that

$$\forall z \in Free(t) - \{\vec{x'}\}.x'' \notin Free(\vec{i}[\vec{y_i} := \vec{z_i}][z]) \cup \{\vec{x''}\lceil_{j-1}\}$$

$$= tr_{\vec{x},\vec{z},\vec{x^*}}[\![(t\,[\vec{y_i},\vec{x'} := \vec{z_i},\vec{x''}])[\vec{x''} := \vec{x^*}]\,]\!]$$

where x^*_j is the least $x^* \in V$ such that

$$x^* \notin Free(t[\vec{y_i},\vec{x'} := \vec{z_i},\vec{x''}]) - \{\vec{x''}\} \cup \{\vec{x}\} \cup \{\vec{z}\} \cup \{\vec{x^*}\lceil_{j-1}\}$$

Let y^*_j be y_j if $j \in \{i_1, i_2, ..., i_l\}$, otherwise be $y^* \in V$ such that

$$y^* \notin Free(t) \cup \{\vec{x'}\} \cup \{y^*_1, y^*_2, ..., y^*_{j-1}\}$$

and let z^*_j be z_j if $j \in \{i_1, i_2, ..., i_l\}$, otherwise be $z^* \in V$ such that

$$z^* \notin (Free(t) - \{\vec{y^*}\}) \cup \{\vec{x''}\}$$

$$= tr_{\vec{x},\vec{z},\vec{x^*}}[\![(t[\vec{y^*},\vec{x'} := \vec{z^*},\vec{x''}])[\vec{z^*},\vec{x''} := \vec{z},\vec{x^*}]\,]\!]$$

By induction hypothesis, we have $\vdash_{EQ} LHS \simeq tr_{\vec{x},\vec{x},\vec{x^*}}[\![t\,[\vec{y^*},\vec{x'} := \vec{z},\vec{x^*}]\,]\!]$

on the other hand, we have $RHS = tr_{\vec{x},\vec{z}}[\![\langle\vec{w} : t[\vec{y_i},\vec{x'} := \vec{y'_i},\vec{w}]\rangle[\vec{y'_i} := \vec{z_i}]\,]\!]$, where w_j is the least $w \in V$ such that

$$\forall z \in Free(t) - \{\vec{x'}\}.w \notin Free(\vec{i}[\vec{y_i} := \vec{y'_i}][z]) \cup \{\lceil w\lceil_{j-1}\}$$

$$= tr_{\vec{x}\vec{z}}[\![\langle\vec{w'} : (t[\vec{y_i},\vec{x'} := \vec{y'_i},\vec{w}])[\vec{y'_i},\vec{w} := \vec{z_i},\vec{w'}]\rangle\,]\!]$$

where w'_j is the least $w' \in V$ such that

$$\forall z \in Free(t[\vec{y_i},\vec{x'} := \vec{y'_i},\vec{w}]) - \{\vec{w}\}.w' \notin Free(\vec{i}[\vec{y'_i} := \vec{z_i}][z]) \cup \{\vec{w'}\lceil_{j-1}\}$$

$$= tr_{\vec{x},\vec{z},\vec{w''}}[\![((t[\vec{y_i},\vec{x'} := \vec{y'_i},\vec{w}])[\vec{y'_i},\vec{w} := \vec{z_i},\vec{w'}])[\vec{w'} := \vec{w''}]\,]\!]$$

By careful examining, we would discover that $x^*_j = w''_j$ for $1 \leq j \leq k$. Then similar to the technique used on left hand side and by twicely using induction hypothesis, we come to $\vdash_{EQ} LHS \simeq RHS$. □

To replace $\vec{x}$ in $tr_{\vec{x}}$ in certain way, we need that the derivability is invariant under (α) conversions. Formally,

Lemma 5.5.3.3 (replacement of bound variables in tr): *For $p \in T$ or $p \in FT_m$ for some $m \geq 0$, we have $\vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{y}}[\![p[\vec{x} := \vec{y}]\,]\!]$, where $\{\vec{x}\} \supseteq Free(p) \cap V$ and $\{\vec{y}\} \cap (Free(p) - \{\vec{x}\}) = \emptyset$.*

Proof By induction on $\jmath$ of $T^{(\jmath)}$ and $FT^{(\jmath)}$ and the above lemma (Lemma 5.5.3.2). It is worth to point out that only the case for function terms has to use the same technique used in the proof of Lemma 5.5.3.2 and the lemma itself. □

Analogous to Lemma 5.4.3.4, we present that derivability is invariant under eliminating non-free variables. Formally

Lemma 5.5.3.4 (elimination of binding variables):

(i) For $t \in T^{(\jmath)}$, $y \notin Free(t) \cap V$ and $\{\vec{x}\} \supseteq Free(t) \cap V$,

$$Ax_b \vdash_{EQ} tr_{y\vec{x}}[\![t]\!] \simeq \underline{o}_{|\vec{x}|,|\vec{x}|+1}(tr_{\vec{x}}[\![t]\!], \underline{Pr}_{2,|\vec{x}|+1}^{|\vec{x}|+1})$$

(ii) For $ft \in FT_m^{(\jmath)}$, and $y \notin Free(t) \cap V$ and $\{\vec{x}\} \supseteq Free(ft) \cap V$,

$$Ax_b \vdash_{EQ} tr_{y\vec{x}}[\![ft]\!] \simeq \underline{o}_{|\vec{x}|+m,|\vec{x}|+m+1}(tr_{\vec{x}}[\![t]\!], \underline{Pr}_{2,|\vec{x}|+m+1}^{|\vec{x}|+m+1})$$

Proof

By induction on $\jmath$ of $T^{(\jmath)}$ and $FT_m^{(\jmath)}$ with case analysis.

(i) For $\jmath = 0$,

(i.a) case x: say $x = x_j \in \{\vec{x}\}$, $LHS = \underline{pr}_{|\vec{x}|+1,j+1}$ and

$RHS = \underline{o}_{|\vec{x}|,|\vec{x}|+1}(\underline{pr}_{|\vec{x}|,j}, \underline{Pr}_{2,|\vec{x}|+1}^{|\vec{x}|+1})$.

By (Ax_b-left-proj), we have $\vdash_{EQ} RHS \simeq LHS$.

(i.b) case $\langle \vec{y} : z \rangle$: see below case $\langle \vec{y} : t \rangle$ of $\jmath > 0$.

(ii) for $\jmath > 0$

(ii.a) case $f(\vec{t})$: $LHS = \underline{o}_{|\vec{t}|,|\vec{x}|+1}(\underline{f}, tr_{y\vec{x}}[\![\vec{t}\,]\!])$ and

$RHS = \underline{o}_{|\vec{x}|,|\vec{x}|+1}(\underline{o}_{|\vec{t}|,|\vec{x}|}(\underline{f}, tr_{y,\vec{x}}[\![\vec{t}\,]\!])\underline{Pr}_{2,|\vec{x}|+1}^{|\vec{x}|+1})$

By $(Ax_b\text{-assoc})$, we get

$$\vdash_{EQ} RHS \simeq \underline{o}_{|\vec{t}|,|\vec{x}|+1}(\underline{f}, \underline{o}_{|\vec{x}|,|\vec{x}|+1}(tr_{y,\vec{x}}[\![\vec{t}]\!], \underline{Pr}^{|\vec{x}|+1}_{2,|\vec{x}|+1}))$$

By induction hypothesis and composition (cmp), we have $\vdash_{EQ} RHS \simeq LHS$.

(ii.b) case $\langle \vec{y} : t \rangle$: $LHS = tr_{y,\vec{x},\vec{z}}[\![t[\vec{y} := \vec{z}\,]\,]\!]$, where z_j is the least $z \in V$ such that $z \notin (Free(t) - \{\vec{y}\}) \cup \{y, \vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$.

And $RHS = \underline{o}_{|\vec{x}|+m,|\vec{x}|+m+1}(tr_{\vec{x},\vec{z'}}[\![t[\vec{y} := \vec{z'}]\,]\!], \underline{Pr}^{|\vec{x}|+m+1}_{2,|\vec{x}|+m+1})$, where z'_j is the least $z' \in V$ such that $z' \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z'}\lceil_{j-1}\}$. Then,

(ii.b.1) for those $x_j \in \{\vec{y}\}$, we find $x^*_j \in V$ such that

$$x^*_j \in \{\vec{y}\} \cup (Free(t) \cap V) \cup \{y\}$$

(ii.b.2) for those x_j such that $x_j \in \{\vec{y}\}$, we still let $x^*_j = x_j$.

Therefore, $tr_{\vec{x},\vec{z'}}[\![t[\vec{y} := \vec{z'}]\,]\!] = tr_{\vec{x},\vec{z'}}[\![t[\vec{x^*}, \vec{y} := \vec{x}, \vec{z'}]\,]\!]$. By Lemma 5.5.3.3, we come to $\vdash_{EQ} tr_{\vec{x},\vec{z'}}[\![t[\vec{y} := \vec{z'}]\,]\!] \simeq tr_{\vec{x^*},\vec{y}}[\![t]\!]$.

Similarly, we can get $\vdash_{EQ} tr_{y\vec{x}\,\vec{z}}[\![t[\vec{y} := \vec{z}\,]\,]\!] \simeq tr_{y,\vec{x^*},\vec{y}}[\![t]\!]$. I.e., by induction hypothesis we have $\vdash_{EQ} LHS \simeq RHS$.

(ii.c) case $\sigma(\vec{ft}, \vec{t})$: $LHS = \underline{o}_{|\vec{x}|+1}(tr_{y,\vec{x}}[\![\vec{ft}]\!], tr_{y,\vec{x}}[\![\vec{t}\,]\!])$ and

$$RHS = \underline{o}_{|\vec{x}|,|\vec{x}|+1}(\underline{o}_{|\vec{x}|+1}(tr_{\vec{x}}[\![\vec{ft}]\!], tr_{\vec{x}}[\![\vec{t}\,]\!])\underline{Pr}^{|\vec{x}|+1}_{2,|\vec{x}|+1})$$

By $(Ax_b\text{-unif})$, we reach that $\vdash_{EQ} RHS \simeq \underline{o}_{|\vec{x}|+1}(\vec{fu}, \vec{u})$,

where $fu_i = \underline{o}_{|\vec{x}|+\vec{m},|\vec{x}|+\vec{m}+1}(tr_{\vec{x}}[\![ft_i]\!], \underline{o}_{|\vec{x}|+1,|\vec{x}|+m_i+1}(\underline{Pr}^{|\vec{x}|+1}_{2,|\vec{x}|+1})\underline{Pr}^{|\vec{x}|+m_i+1}_{|\vec{x}|+2,|\vec{x}|+m_i+1})$
and $u_j = \underline{o}_{|\vec{x}|,|\vec{x}|+1}(tr_{\vec{x}}[\![t_j]\!], \underline{Pr}^{|\vec{x}|+1}_{2,|\vec{x}|+1})$.

By $(Ax_b\text{-left-proj})$, (eq-cmp), and (eq-trs), we have

$\vdash_{EQ} RHS \simeq \underline{o}_{|\vec{x}|+1}(\vec{fv}, \vec{v})$

where

$$fv_i = \underline{o}_{|\vec{x}|+m_i,|\vec{x}|+m_i+1}(tr_{\vec{x}}[\![ft_i]\!], \underline{Pr}^{|\vec{x}|+m_i+1}_{2,|\vec{x}|+m_i+1})$$

and $v_j = \underline{o}_{|\vec{x}|,|\vec{x}|+1}(tr_{\vec{x}}[\![t_j]\!], \underline{Pr}^{|\vec{x}|+1}_{2,|\vec{x}|+1})$.

By induction hypothesis, we obtain $\vdash_{EQ} LHS \simeq RHS$. □

The above result can be formalized in another way, which is a reversed version of the above, and it is analogous to Corollary 5.4.3.5. Formally,

Corollary 5.5.3.5 (introduction of non-free variables):

(i) For $t \in T^{(j)}$ *and* $y \notin Free(t) \cap V$ *and* $\{\vec{x}\} \supseteq Free(t) \cap V$,

$$Ax_b \vdash_{EQ} tr_{\vec{x}}[\![t]\!] \simeq \underline{o}_{|\vec{x}|,|\vec{x}|+1}(tr_{y\vec{x}}[\![t]\!], \underline{pr}_{|\vec{x}|,1}, \underline{Pr}_{1,|\vec{x}|}^{|\vec{x}|})$$

(ii) For $ft \in FT_m^{(j)}$, *and* $y \notin Free(t) \cap V$ *and* $\{\vec{x}\} \supseteq Free(ft) \cap V$,

$$Ax_b \vdash_{EQ} tr_{\vec{x}}[\![ft]\!] \simeq \underline{o}_{|\vec{x}|+m+1,|\vec{x}|+m}(tr_{y\vec{x}}[\![t]\!], \underline{pr}_{|\vec{x}|+m,1}, \underline{Pr}_{1,|\vec{x}|+m}^{|\vec{x}|+m})$$

Proof By Lemma 5.5.3.4. □

The previous two lemmas allow us to introduce and eliminate non-free variable freely, which are primary tools in proving some results. Nevertheless, we formalize another result about non-free variables as follows.

Lemma 5.5.3.6 (role of non-free variables):

(i) Given $t \in T$, $\{\vec{x}\} \cup \{\vec{x'}\} \supseteq Free(t) \cap V$ *and* $\{\vec{y}\} \cap Free(t) = \emptyset$, *we have*

$$\vdash_{EQ} tr_{\vec{x}\vec{y}\vec{x'}}[\![t]\!] \simeq \underline{o}_{|\vec{x}|+|\vec{x'}|,|\vec{x}|+|\vec{y}|+|\vec{x'}|}(tr_{\vec{x}\vec{x'}}[\![t]\!], Pr_{1,|\vec{x}|}^{|\vec{x}|+|\vec{y}|+|\vec{x'}|}, Pr_{|\vec{x}|+|\vec{y}|+1,|\vec{x}|+|\vec{y}|+|\vec{x'}|}^{|\vec{x}|+|\vec{y}|+|\vec{x'}|}),$$

(ii) Given $ft \in FT_m$ $(m \geq 0)$, $\{\vec{x}\} \cup \{\vec{x'}\} \supseteq Free(ft) \cap V$ *and* $\{\vec{y}\} \cap Free(ft) = \emptyset$, *we have*

$$\vdash_{EQ} tr_{\vec{x}\vec{y}\vec{x'}}[\![\langle \vec{y} : t \rangle]\!] \simeq \underline{o}_{|\vec{x}|+|\vec{x'}|+m,|\vec{x}|+|\vec{y}|+|\vec{x'}|+m}(tr_{\vec{x}\vec{x'}}[\![\langle \vec{y} : t \rangle]\!], Proj)$$

where $ft = \langle \vec{y} : t \rangle$ *and* $Proj = Pr_{1,|\vec{x}|}^{|\vec{x}|+|\vec{y}|+|\vec{x'}|+m}, Pr_{|\vec{x}|+|\vec{y}|+1,|\vec{x}|+|\vec{y}|+|\vec{x'}|+m}^{|\vec{x}|+|\vec{y}|+|\vec{x'}|+m}$.

Proof

By induction on j of $T^{(j)}$ and $FT^{(j)}$ with Lemma 5.5.3.2 and Lemma 5.5.3.3. □

In particular, we have $\vdash_{EQ} tr_{\vec{x}\vec{y}}[\![p]\!] \simeq \underline{o}_{|\vec{x}|,|\vec{x}|+|\vec{y}|}(tr_{\vec{x}}[\![p]\!], \underline{Pr}_{1,|\vec{x}|}^{|\vec{x}|+|\vec{y}|})$. Similar to Lemma 5.5.2.2, we can replace ordinary substitutions by syntactic compositions. On the other hand, this tells us the insight of β-conversions in Lambda Calculus.

Lemma 5.5.3.7 (substitution vs composition):

(i) Given $t \in T$, for every $\{\vec{y}\} \subseteq V$ and $\{\vec{y}\} \supseteq Free(t) \cap V$, for all $\vec{u} \in T$ ($|\vec{u}| = |\vec{y}|$) and for all $\{\vec{x}\} \subseteq V$ such that $\{\vec{x}\} \supseteq \bigcup_{j=1}^{|\vec{u}|} Free(u_j) \cap V$, we have

$$tr_{\vec{x}}[\![t[\vec{y} := \vec{u}]\,]\!] = \underline{o}_{|\vec{y}|,|\vec{x}|}(tr_e[\![\langle \vec{y} : t\rangle]\!], tr_{\vec{x}}[\![\vec{u}]\!])$$

(ii) Given $ft \in FT_k$ ($k \geq 0$), for every $\{\vec{y}\} \subseteq V$ and $\{\vec{y}\} \supseteq Free(ft) \cap V$, for all $\vec{u} \in T$ ($|\vec{u}| = |\vec{t}|$) and for all $\{\vec{x}\} \subseteq V$ such that $\{\vec{x}\} \supseteq \bigcup_{j=1}^{|\vec{u}|} Free(u_j) \cap V$, we have

$$tr_{\vec{x}}[\![ft[\vec{y} := \vec{u}]\,]\!] = \underline{o}_{|\vec{y}|+k,|\vec{x}|+k}(fv, \vec{v}),$$

where $fv = tr_e[\![\langle \vec{y'} : ft||_k[\vec{z} := \vec{y'}\lfloor_{|\vec{y}|}]\rangle]\!]$, $v_j = \underline{o}_{|\vec{x}|,|\vec{x}|+k}(tr_{\vec{x}}[\![u_j]\!], Pr_{1,|\vec{x}|}^{|\vec{x}|+k})$ $v_{|\vec{u}|+i} = pr_{|\vec{x}|+k,|\vec{x}|+i}$, $\vec{z} = ft|_k$, $\vec{y} = \vec{y'}\lceil_{|\vec{y}|}$ and $y'_{|\vec{y}|+j} \notin Free(ft) \cup \{\vec{y'}\lceil_{|\vec{y}|+j-1}\}$.

Proof By structural induction. □

All of the above is a kind of built-up in order to prove the independence of variable indices in the translation of $tr_{\vec{x}}$. This independence can now be achieved. We formalize it below.

Lemma 5.5.3.8 (independence of derivability from binding variables): *For $p, q \in T$ or $p, q \in FT_m$ for some $m \geq 0$, if $\underline{\Gamma} \cup Ax_b \vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$ for some $\{\vec{x}\} \supseteq (Free(p) \cup Free(q)) \cap V$, then $\Gamma \cup Ax_b \vdash_{EQ} tr_{\vec{y}}[\![p]\!] \simeq tr_{\vec{y}}[\![q]\!]$ for all $\{\vec{y}\} \supseteq (Free(p) \cup Free(q)) \cap V$.*

Proof

We can always assume that $\{\vec{x}\} = (Free(p) \cup Free(q)) \cap V$. Otherwise, we can come to this by Lemma 5.5.3.1 and Lemma 5.5.3.4.

Hence, from this $\{\vec{x}\}$, we extend it to any $\vec{y}$ such that $\{\vec{y}\} \supseteq \{\vec{x}\}$ by the same Corollary 5.5.3.5 and Lemma 5.5.3.1. □

As a consequence of Lemma 5.5.3.8, it is more desirable to use notation $tr[\![\Gamma]\!]$ than $tr_{\vec{x}}[\![\Gamma]\!]$ in a sequence of $tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$, where $p \simeq q \in \Gamma$ and $\{\vec{x}\} \supseteq (Free(p) \cup Free(q)) \cap V$. Actually inside the sequence, $\vec{x}$ is just a symbol, therefore has lost its original meaning. Therefore, Γ and $\underline{\Gamma}$ is one corresponding to the other under the translations.

Returning from the digression, we have that (α) conversions are preserved by translation $tr_{\vec{x}}$. Formally,

Lemma 5.5.3.9 ((α) schema): $\vdash_{EQ} tr_{\vec{x}}[\![\langle \vec{x} : t\rangle]\!] \simeq tr_{\vec{x}}[\![\langle \vec{y} : t[\vec{x} := \vec{y}]\rangle]\!]$, *where* $\{\vec{z}\} \supseteq (Free(\langle \vec{x} : t\rangle) \cap V)$ *and* $\{\vec{y}\} \cap Free(\langle \vec{x} : t\rangle) = \emptyset$.

Proof By Lemma 5.5.3.3. □

For the (ξ^{-1}) rule, we have the preservation below.

Lemma 5.5.3.10 (ξ^{-1} rule): *If* $tr[\![\Gamma]\!] \vdash_{EQ} tr_{\vec{y}}[\![\langle \vec{z} : u\rangle]\!] \simeq tr_{\vec{y}}[\![\langle \vec{z'} : u'\rangle]\!]$ *for* $\{\vec{y}\} = (Free(\langle \vec{z} : u\rangle) \cup Free(\langle \vec{z'} : u'\rangle)) \cap V$, *then* $tr[\![\Gamma]\!] \vdash_{EQ} tr_{\vec{y}\vec{x}}[\![u[\vec{z} := \vec{t}]\,]\!] \simeq tr_{\vec{x}}[\![u'[\vec{z'} := \vec{t}]\,]\!]$ *for* $\{\vec{x}\} \cap \{\vec{y}\} = \emptyset$ *and* $\{\vec{y}\} \cup \{\vec{x}\} = ((\bigcup_{j=1}^{|\vec{t}|} Free(t_j)) \cap V)$.

Proof

1. By Lemma 5.5.3.2, we have that the pre-condition implies $tr[\![\Gamma]\!] \vdash_{EQ} tr_{\vec{y},\vec{y'}}[\![u[\vec{z} := \vec{y'}]\,]\!] \simeq tr_{\vec{y},\vec{y'}}[\![u'[\vec{z'} := \vec{y'}]\,]\!]$ where $\{\vec{y'}\} \cap \{\vec{y}\} = \emptyset$, $\{\vec{y'}\} \cap (Free(u) - \{\vec{z}\}) = \emptyset$ and $\{\vec{y'}\} \cap (Free(u') - \{\vec{z'}\}) = \emptyset$.

2. By Lemma 5.5.3.2, we have

   $\vdash_{EQ} tr_{\vec{y},\vec{x}}[\![u[\vec{z} := \vec{t}]\,]\!] \simeq tr_{\vec{y},\vec{x}}[\![(u[\vec{z} := \vec{y'}])[\vec{y}, \vec{y'} := \vec{y}, \vec{t}]\,]\!]$ and

   $\vdash_{EQ} tr_{\vec{y},\vec{x}}[\![u'[\vec{z'} := \vec{t}]\,]\!] \simeq tr_{\vec{y},\vec{x}}[\![(u'[\vec{z'} := \vec{y'}])[\vec{y}, \vec{y'} := \vec{y}, \vec{t}]\,]\!]$.

3. By Lemma 5.5.3.7, we have

   $$\vdash_{EQ} tr_{\vec{y},\vec{x}}[\![(u[\vec{z} := \vec{y'}])[\vec{y}, \vec{y'} := \vec{y}, \vec{t}]\,]\!] \simeq \underline{o}_{j,j'}(tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u[\vec{z} := \vec{y'}]\rangle]\!], tr_{\vec{y},\vec{x}}[\![\vec{t}]\!])$$

   and

   $$\vdash_{EQ} tr_{\vec{y},\vec{x}}[\![(u'[\vec{z'} := \vec{y'}])[\vec{y}, \vec{y'} := \vec{y}, \vec{t}]\,]\!] \simeq \underline{o}_{j,j'}(tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u'[\vec{z'} := \vec{y'}]\rangle]\!], tr_{\vec{y},\vec{x}}[\![\vec{t}]\!]),$$

   where $j = |\vec{y}| + |\vec{y'}|$ and $j' = |\vec{y}| + |\vec{x}|$.

4. By Lemma 5.5.3.3, we have

   $\vdash_{EQ} tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u[\vec{z} := \vec{y'}]\rangle]\!] \simeq tr_{\vec{y},\vec{y'}}[\![u[\vec{z} := \vec{y'}]\,]\!]$ and

   $\vdash_{EQ} tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u'[\vec{z'} := \vec{y'}]\rangle]\!] \simeq tr_{\vec{y},\vec{y'}}[\![u'[\vec{z'} := \vec{y'}]\,]\!]$.

   So, $tr[\![\Gamma]\!] \vdash_{EQ} tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u[\vec{z} := \vec{y'}]\rangle]\!] \simeq tr_{\epsilon}[\![\langle \vec{y}, \vec{y'} : u'[\vec{z'} := \vec{y'}]\rangle]\!]$.

5. By 1, 3 and 4 of the above with composition (cmp), we have

   $tr[\![\Gamma]\!] \vdash_{EQ} tr_{\vec{y},\vec{x}}[\![u[\vec{z} := \vec{t}]\,]\!] \simeq tr_{\vec{x}}[\![u'[\vec{z'} := \vec{t}]\,]\!]$. □

Next, we show that (ξ) rule is preserved.

Lemma 5.5.3.11 ((ξ) rule): *If* $\Gamma \cup Ax_b \vdash_{EQ} tr_{\vec{y}}[\![t]\!] \simeq tr_{\vec{y}}[\![u]\!]$ *for some* $\{\vec{y}\} \supseteq (Free(t) \cup Free(u)) \cap V$, *then* $\Gamma \vdash_{EQ} \cup Ax_b tr_{\vec{z}}[\![\langle \vec{x} : t\rangle]\!] \simeq tr_{\vec{z}}[\![\langle \vec{x} : u\rangle]\!]$ *for all* $\{\vec{z}\}$ *such that* $\{\vec{z}\} \supseteq (Free(\langle \vec{x} : t\rangle) \cup Free(\langle \vec{x} : u\rangle)) \cap V$.

Proof

1. $LHS = tr_{\vec{z}\vec{z'}}[\![t[\vec{x} := \vec{z'}]\,]\!]$ and $RHS = tr_{\vec{z}\vec{z''}}[\![u[\vec{x} := \vec{z''}]\,]\!]$.

   Let z_j^* be $z^* \in V$ such that $z^* \notin \{\vec{z}\} \cup Free(t) \cup Free(u) \cup \{\vec{z'}\} \cup \{\vec{z''}\} \cup \{\vec{z^*}\lceil_{j-1}\}$.

   By Lemma 5.5.3.3, we have $\vdash_{EQ} LHS \simeq tr_{\vec{z},\vec{z^*}}[\![(t[\vec{x} := \vec{z'}])[\vec{z'} := \vec{z^*}]\,]\!]$.

   By Lemma 5.5.3.2, we have $\vdash_{EQ} LHS \simeq tr_{\vec{z},\vec{z^*}}[\![t[\vec{x} := \vec{z^*}]\,]\!]$.

   Similarly, we have $\Gamma \vdash_{EQ} RHS \simeq tr_{\vec{z},\vec{z^*}}[\![u[\vec{x} := \vec{z^*}]\,]\!]$.

2. For x_j which is not in $(Free(t) \cup Free(u)) \cap V$, let $x_j^* \in V$ be such a x^* that $x^* \notin Free(t) \cup Free(u) \cup \{\vec{z^*}\} \cup \{\vec{x^*}\lceil_{j-1}\} \cup \{\vec{z}\}$. If $x_j \in (Free(t) \cup Free(u)) \cap V$, then let $x_j^* = x_j$. So, we have

   $\Gamma \vdash_{EQ} LHS \simeq tr_{\vec{z},\vec{z^*}}[\![t[\vec{x^*} := \vec{z^*}]\,]\!]$ *and* $\Gamma \vdash_{EQ} RHS \simeq tr_{\vec{z},\vec{z^*}}[\![u[\vec{x^*} := \vec{z^*}]\,]\!]$

3. By Lemma 5.5.3.3, if $\{\vec{z}\} \cap \{\vec{x}\} = \emptyset$, we reach

   $\Gamma \vdash_{EQ} LHS \simeq tr_{\vec{z},\vec{x^*}}[\![t]\!]$ *and* $\Gamma \vdash_{EQ} RHS \simeq tr_{\vec{z},\vec{x^*}}[\![u]\!]$

   So, by the pre-condition of (ξ) rule and Lemma 5.5.3.8 we come to

   $\Gamma \vdash_{EQ} LHS \simeq RHS$.

4. For $\{\vec{z}\} \cap \{\vec{x}\} \neq \emptyset$, by Lemma 5.5.3.4 and Lemma 5.5.3.8 we can get

   $\vdash_{EQ} LHS \simeq tr_{\vec{z'}}[\![\langle \vec{x} : t\rangle]\!]$ *and* $\vdash_{EQ} RHS \simeq tr_{\vec{z'}}[\![\langle \vec{x} : u\rangle]\!]$

   where z_j' is z_j if $z_j \notin \{\vec{x}\}$, and otherwise z_j' is a $z' \in V$ such that $z' \notin \{\vec{x}\} \cup Free(t) \cup Free(u)$; thus, the same reasoning of 3 above can be applied again. □

For (pure-func-sub), we have a lemma, see Lemma 5.5.3.12.

Lemma 5.5.3.12 ((pure-func-sub) vs (crs-sub)): $\vdash_{EQ} tr_{\vec{x}}[\![p[\vec{f} := \vec{ft}]\ ]\!] \simeq tr_{\vec{x}}[\![p]\!][tr_{\epsilon}[\![\vec{ft}]\!]/\underline{\vec{f}}]$, *where* $Free(ft_i) \cap V = \emptyset$ $(1 \leq i \leq |\vec{ft}| = |\underline{\vec{f}}|)$.

Proof

By induction on $\jmath$ of $T^{(\jmath)}$ and $FT_k^{(\jmath)}$ $(k \geq 0)$. We only show two cases.

(i) case $f(\vec{t})$: there are two possibilities (i.a) $f \notin \{\vec{f}\}$ and (i.b) $f = f_i \in \{\vec{f}\}$.

For (i.a), we have $LHS = \underline{o}_{|\vec{t}|,|\vec{x}|}(\underline{f}, tr_{\vec{x}}[\![\vec{t}[\vec{f} := \vec{ft}]\ ]\!])$.

By structural hypothesis, we get $\vdash_{EQ} LHS \simeq \underline{o}_{|\vec{t}|,|\vec{x}|}(\underline{f}, tr_{\vec{x}}[\![\vec{t}]\!][tr_{\epsilon}[\![\vec{ft}]\!]/\underline{\vec{f}}])$.

So, $\vdash_{EQ} LHS \simeq RHS$.

For (i.b), we have $LHS = tr_{\vec{x}}[\![ft_i||_k[\vec{y} := \vec{t}[\vec{f} := \vec{ft}]\ ]\ ]\!]$, where $\vec{y} = ft_i|_k$.

By Lemma 5.5.3.7, we have $\vdash_{EQ} LHS \simeq \underline{o}_{k,|\vec{x}|}(tr_{\epsilon}[\![\langle \vec{y} : ft_i||_k \rangle]\!], tr_{\vec{x}}[\![\vec{t}[\vec{f} := \vec{ft}]\ ]\!])$.

So, again by structural hypothesis we come to $\vdash_{EQ} LHS \simeq RHS$.

(ii) case $\langle \vec{y} : t \rangle$: $LHS = tr_{\vec{x}}[\![\langle \vec{z} : t[\vec{f}, \vec{y} := \vec{ft}, \vec{z}]\rangle]\!]$, where z_j is the least $z \in V$ such that for all $z' \in Free(t) - \{\vec{y}\}$, $z \notin Free(\vec{t}[\vec{f} := \vec{ft}][z']) \cup \{\vec{z}\lceil_{j-1}\}$.

We should be aware of that up to now what we actually have is if $\vdash_{\alpha} p \simeq q$ then $\vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$. By Lemma 5.5.2.8, we would have

$$\vdash_{EQ} LHS \simeq tr_{\vec{x}\vec{z}'}[\![t[\vec{y}, \vec{f} := \vec{z}', \vec{ft}]\ ]\!],$$

where z'_j is the least $z' \in V$ such that $z' \notin Free(t[\vec{y}, \vec{f} := \vec{z}, \vec{ft}] - \{\vec{z}\} \cup \{\vec{x}\} \cup \{\vec{z}'\lceil_{j-1}\}$.

Obviously, we know that $p[\vec{y}, \vec{f} := \vec{z}', \vec{ft}] = (p[\vec{y} := \vec{z}'])[\vec{f} := \vec{ft}]$. Then, by Property 5.1.4 and induction hypothesis, we get that $\vdash_{EQ} LHS \simeq tr_{\vec{x}\vec{z}'}[\![t[\vec{y} := \vec{z}']\ ]\!][tr_{\epsilon}[\![\vec{ft}]\!]/\underline{\vec{f}}]$.

On the other hand, we have $RHS = tr_{\vec{x}\vec{z}^*}[\![t[\vec{y} := \vec{z}^*]\ ]\!][tr_{\epsilon}[\![\vec{ft}]\!]/\underline{\vec{f}}]$, where z_j^* is the least $z^* \in V$ such that $z^* \notin Free(t) - \{\vec{y}\} \cup \{\vec{z}^*\lceil_{j_1}\}$.

Hence by Lemma 5.5.3.3, we come to $\vdash_{EQ} LHS \simeq RHS$. □

To sum up the previous results, we will have the preservation of derivability under translation tr. Formally,

Theorem 5.5.3.13 (equality from iBAs to b-clones): *Translation $tr_{\vec{x}}$ preserves derivations of $\vdash_{iBA}$, i.e. if $\Gamma \vdash_{iBA} p \simeq q$, then $\underline{\Gamma} \cup Ax_b \vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$, where $\underline{\Gamma} = tr[\![\Gamma]\!]$.*

As a summary of this subsection (Subsection 5.5.3), we have obtained

$$(5.5.3.*) \quad \Gamma \vdash_{iBA} p \simeq q \text{ implies } tr[\![\Gamma]\!] \cup Ax_b \vdash_{EQ} tr[\![p]\!] \simeq tr[\![q]\!]$$

5.6 Soundness and completeness of $\vdash_{iBA}$

Now we are ready to prove the soundness and completeness of Equational Calculus $\vdash_{iBA}$. Up to now, we know that

(5.6.a) "identity" (equality) are preserved by translations $\underline{tr}$ and $tr_{\vec{x}}$ (Theorem 5.4.3.2 and Theorem 5.4.3.6), i.e. semantic preservation; and

(5.6.b) the derivations are also preserved by these two translations (Theorem 5.5.2.12 and Theorem 5.5.3.13), i.e. syntactic preservation.

In what follows, we are going to show that the soundness and completeness of $\vdash_{iBA}$ can be established from them.

5.6.1 Soundness of $\vdash_{iBA}$ for BEs

In this subsection, we are going to prove the soundness of $\vdash_{iBA}$.

Theorem 5.6.1.1 (soundness of $\vdash_{iBA}$): $\Gamma \vdash_{iBA} p \simeq q$ *implies* $\Gamma \models_{iBA} p \simeq q$.

Proof

- By Theorem 5.5.3.13, we have that $\Gamma \vdash_{iBA} p \simeq q$ implies

$$tr_{\vec{x}}[\![\Gamma]\!] \cup Ax_b \vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!].$$

- By Theorem 2.1.34 and Theorem 2.1.35, we get

$$tr_{\vec{x}}[\![\Gamma]\!] \cup Ax_b \models_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!],$$

since we assume that the signature of b-clones satisfies the condition of eliminating quantifications (see Theorem 2.2.6 and Remark 5.5.1.5).

- By Theorem 5.4.3.2, we will have $\Gamma \models_{iBA} p \simeq q$. □

5.6.2 Completeness of $\vdash_{iBA}$

The proof of the Completeness of $\vdash_{iBA}$ is more complicated than the one of the Soundness in Subsection 5.6.1. Firstly, we will show a syntactic relation between two translations $\underline{tr}$ and $tr_{\vec{x}}$ (Lemma 5.6.2.1). Then, the completeness follows (Lemma 5.6.2.2).

Lemma 5.6.2.1 (syntactic relation between $tr_{\vec{x}}$ and $\underline{tr}$):

1. $\vdash_{iBA} t \simeq \underline{tr}[\![\underline{tr}_{\vec{y}}[\![t]\!]\,]\!]\,[\vec{x} := \vec{y}]$, *where* $\{\vec{y}\} \supseteq Free(t) \cap V$, $|\vec{y}| = |\vec{x}|$ *and* $\vec{x}(j)$ *is the* j*th least element in* V.

2. $\vdash_{iBA} \langle \vec{z} : t \rangle \simeq \langle \vec{x}_{\lfloor |\vec{y}|}, \underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : t \rangle]\!]\,]\!]\rangle\,[\vec{x} := \vec{y}]$, *where* $\{\vec{y}\} \supseteq Free(\langle \vec{z} : t \rangle) \cap V$ *and* $|\vec{x}| = |\vec{y}| + |\vec{z}|$.

Proof

By combining induction on j of $T^{(j)}_{\Sigma^{BO}}$ and $FT^{(j)}_m$ and with case analysis.

(i) For $j = 0$,

(i.a) case x: since $x \in \{\vec{y}\}$, we have $tr_{\vec{y}}[\![x]\!] = pr_{|\vec{y}|,i}$, where $\vec{y}(i) = x$. Then, $\underline{tr}[\![pr_{|\vec{y}|,i}]\!] = x_i$, where x_i is the ith least element in V. So, $x_i\,[\vec{x} := \vec{y}] = x$.

(i.b) case $\langle \vec{z} : x \rangle$: $tr_{\vec{y}}[\![\langle \vec{z} : x \rangle]\!] = tr_{\vec{y},\vec{z}}[\![x\,[\vec{z} := \vec{z'}]\,]\!]$
(where z'_j is the least $z' \in V$ such that $z' \notin (Free(t) - \{\vec{z}\}) \cup \{\vec{y}\} \cup \{\vec{z'}\lceil_{j-1}\}$.)

$$= \begin{cases} pr_{|\vec{y}|+|\vec{z}|,i} & if\ x \notin \{\vec{z}\})\ and\ x = \vec{y}(i) \in \{\vec{y}\} \quad (i.b.1) \\ pr_{|\vec{y}|+|\vec{z}|,|\vec{y}|+j} & if\ x = z_j \in \{\vec{z}\} \quad (i.b.2) \end{cases}$$

For $(i.b.2)$, we can get $\vdash_{iBA} \langle \vec{z} : x \rangle \simeq \langle \vec{x'}\,\vec{x''} : \underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : x \rangle]\!]\,]\!]\rangle\,[\vec{x'} := \vec{y}]$, since $\underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : x \rangle]\!]\,]\!] = \vec{x''}(j)$.

For $(i.b.1)$, We have $\vdash_{iBA} \langle \vec{z} : x \rangle \simeq \langle \vec{x'} \vec{x''} : \underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : x \rangle]\!]\,]\!][\vec{x'} := \vec{y}]\rangle$, since $\underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : x \rangle]\!]\,]\!] = \vec{x'}(i)$.

(ii) For $\jmath > 0$,

(ii.a) case $f(\vec{t})$: since $tr_{\vec{y}}[\![f(\vec{t})]\!] = \circ_{|\vec{t}|,|\vec{y}|}(\underline{f}, tr_{\vec{y}}[\![\vec{t}]\!])$, we would get

$$\underline{tr}[\![tr_{\vec{y}}[\![f(\vec{t})]\!]\,]\!] = \underline{tr}[\![\underline{f}]\!][\vec{x} := \underline{tr}[\![tr_{\vec{y}}[\![\vec{t}]\!]\,]\!]$$

(where x_j is the least element of V such that $x_j \notin \{\vec{x}\lceil_{j-1}\}$.)

$$= f(\underline{tr}[\![tr_{\vec{y}}[\![\vec{t}]\!]\,]\!])$$

So, $\underline{tr}[\![tr_{\vec{y}}[\![f(\vec{t})]\!]\,]\!][\vec{y} := \vec{x}] = f(\underline{tr}[\![tr_{\vec{y}}[\![\vec{t}]\!]\,]\!])[\vec{y} := \vec{x}] = f(\underline{tr}[\![tr_{\vec{y}}[\![\vec{t}]\!]\,]\!][\vec{y} := \vec{x}])$.

By induction hypothesis, we have $\vdash_{iBA} f(\vec{t}) \simeq \underline{tr}[\![tr_{\vec{y}}[\![f(\vec{t})]\!]\,]\!][\vec{y} := \vec{x}]$.

(ii.b) case $\langle \vec{z} : t \rangle$: since $tr_{\vec{y}}[\![\langle \vec{z} : t \rangle]\!] = tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]$, where z'_j is the least $z \in V$ such that $z \notin (Free(t) - \{\vec{z}\}) \cup \{\vec{y}\} \cup \{\vec{z'}\lceil_{j-1}\}$, we would have $\underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : t \rangle]\!]\,]\!] = \underline{tr}[\![tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]$.

On the other hand, by induction hypothesis we have

$$(*) \quad \vdash_{iBA} t\,[\vec{z} := \vec{z'}] \simeq \underline{tr}[\![tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!][\vec{x} := \vec{y}, \vec{z'}].$$

By (ξ) rule we get $\vdash_{iBA} \langle \vec{z'} : t\,[\vec{z} := \vec{z'}]\rangle \simeq \langle \vec{z'} : \underline{tr}[\![tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!][\vec{x} := \vec{y}\vec{z'}]\rangle$.

For the left hand side of $\simeq$ in the above $(*)$, by (α), we have

$$\vdash_{iBA} \langle \vec{z} : t\rangle \simeq \langle \vec{z'} : t\,[\vec{z} := \vec{z'}]\rangle.$$

For the right hand side of $\simeq$ in the above $(*)$, we also have

$$\langle \vec{z'} : \underline{tr}[\![tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!][\vec{x} := \vec{y}, \vec{z'}]\rangle$$
$$= \langle \vec{z'} : (\underline{tr}[\![tr_{\vec{y},\vec{z}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!][\vec{x} := \vec{y}, \vec{z''}])[\vec{z''} := \vec{z'}]\rangle$$

(where z''_j is the least $z'' \in V$ such that

$$\forall z \in Free(\underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : t\rangle]\!]\,]\!]).z'' \notin Free(\vec{v}[\vec{x}\lceil_{|\vec{y}|} := \vec{y}][z]) \cup \{\vec{z''}\lceil_{j-1}\},$$

and $|\vec{z''}| = |\vec{z'}| = |\vec{z}|$.)

And by (α) we have

$$\vdash_{iBA} \langle \vec{z'} : (\underline{tr}[\![tr_{\vec{y},\vec{x}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!]\,[\vec{x} := \vec{y}, \vec{z''}])\,[\vec{z''} := \vec{z'}]\rangle$$
$$\simeq \langle \vec{z''} : \underline{tr}[\![tr_{\vec{y},\vec{x}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!]\,[\vec{x} := \vec{y}, \vec{z''}]\rangle$$

Thus,

$$\vdash_{iBA} \langle \vec{z''} : \underline{tr}[\![tr_{\vec{y},\vec{x}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!]\,[\vec{x} := \vec{y}, \vec{z''}]\rangle$$
$$\simeq \langle \vec{z''} : \underline{tr}[\![tr_{\vec{y},\vec{x}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!]\,[\vec{x}\lceil_{|\vec{y}|}, \vec{x}\lfloor_{|\vec{y}|} := \vec{y}, \vec{z''}]\rangle$$

So, by Definition 5.1.3 (substitution), we have

$$\vdash_{iBA} \langle \vec{z''} : \underline{tr}[\![tr_{\vec{y},\vec{x}}[\![t\,[\vec{z} := \vec{z'}]\,]\!]\,]\!]\,[\vec{x}\lceil_{|\vec{y}|}, \vec{x}\lfloor_{|\vec{y}|} := \vec{y}, \vec{z''}]\rangle$$
$$\simeq \langle x_{|\vec{y}|+1}, x_{|\vec{y}|+2}, ..., x_{|\vec{y}|+|\vec{z}|} : \underline{tr}[\![tr_{\vec{y}}[\![\langle \vec{z} : t\rangle]\!]\,]\!]\rangle\,[\vec{x}\lceil_{|\vec{y}|} := \vec{y}]$$

(ii.b) case $\sigma(\vec{ft}, \vec{t})$: since $tr_{\vec{y}}[\![\sigma(\vec{ft}, \vec{t})]\!] = \underline{\sigma}_{|\vec{y}|}(tr_{\vec{y}}[\![\vec{ft}]\!], tr_{\vec{y}}[\![\vec{t}]\!])$, we get

$$\underline{tr}[\![tr_{\vec{y}}[\![\sigma(\vec{ft}, \vec{t})]\!]\,]\!] = \sigma(\langle \vec{x}^{|\vec{y}|+\vec{m}}\lfloor_{|\vec{y}|} : \underline{tr}[\![tr_{\vec{y}}[\![\vec{ft}]\!]\,]\!]\rangle, \underline{tr}[\![tr_{\vec{y}}[\![\vec{t}]\!]\,]\!])$$

where m_i shares the same arity with ft_i $(1 \leq i \leq \ell)$

Therefore, by induction hypothesis, we have

$$\vdash_{iBA} \underline{tr}[\![tr_{\vec{y}}[\![\sigma(\vec{ft}, \vec{t})]\!]\,]\!]\,[\vec{x} := \vec{y}] \simeq \sigma(\vec{ft}, \vec{t}).\square$$

Consequently, we have the completeness of equational calculus $\vdash_{iBA}$. Formally,

Theorem 5.6.2.2 (completeness of $\vdash_{iBA}$): $\Gamma \models_{iBA} p \simeq q$ *implies* $\Gamma \vdash_{iBA} p \simeq q$.

Proof

(i) By Theorem 5.4.3.2, we can get that $\Gamma \models_{iBA} p \simeq q$ implies

$$Ax_b \cup tr_{\vec{x}}[\![\Gamma]\!] \models_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!].$$

(ii) By Theorem 2.1.35, we have $Ax_b \cup tr_{\vec{x}}[\![\Gamma]\!] \vdash_{EQ} tr_{\vec{x}}[\![p]\!] \simeq tr_{\vec{x}}[\![q]\!]$.

(iii) By Lemma 5.6.2.1, we have $\underline{tr}[\![tr_{\vec{x}}[\![\Gamma]\!]\,]\!] \vdash_{iBA} \underline{tr}[\![tr_{\vec{x}}[\![p]\!]\,]\!] \simeq \underline{tr}[\![tr_{\vec{x}}[\![q]\!]\,]\!]$.

(iv) By Lemma 5.6.2.1, Lemma 5.5.3.10 and transitivity (trs), at last, we have the completeness.$\square$

Note that the reason we need Lemma 5.5.3.10 is as follows. Γ might contain some pairs of function terms, say $ft \simeq fu \in \Gamma$. Then by Lemma 5.6.2.1, we have

(5.6.c) that $ft = \langle \vec{z} : t \rangle \simeq \langle \vec{x'} \vec{x''} : \underline{tr}[tr_{\vec{y}}[\![\langle \vec{z} : t \rangle]\!]\rangle [\vec{x'} := \vec{y}]$, where $\{\vec{y}\} \supseteq Free(\langle \vec{z} : t \rangle) \cap V$, $|\vec{x'}| = |\vec{y}|$, $|\vec{x''}| = |\vec{z}|$ and

(5.6.d) that $fu = \langle \vec{z'} : u \rangle \simeq \langle \vec{x^*} \vec{x^{**}} : \underline{tr}[\![tr_{\vec{y'}}[\![\langle \vec{z'} : u \rangle]\!]\rangle [\vec{x^*} := \vec{y'}]$, where $\{\vec{y'}\} \supseteq Free(\langle \vec{z'} : u \rangle) \cap V$ and $|\vec{x^*}| = |\vec{y'}|$ and $|\vec{x^{**}}| = |\vec{z'}|$ (note : $|\vec{z}| = |\vec{z'}|$ is the arity of ft and fu). Hence, let $\vec{y} = \vec{y'}$, then $ft \simeq fu \vdash_{iBA} \underline{tr}[\![tr_{\vec{y}}[\![ft]\!]]\!] \simeq \underline{tr}[\![tr_{\vec{y}}[\![fu]\!]]\!]$ by (ξ^{-1}) rule.

Looking back to calculus $\vdash_{iBA}$, we understand that the role of binding seems to vanish, since the presence of both (ξ^{-1}) rule and (ξ) rule. This may well be an inappropriate impression, unless we replace Σ^{BO} by Σ^{VBTO}. Therefore, the exact border line between these two deserves a future investigation.

Nevertheless, we point out a condition under which (ξ^{-1}) rule is redundant in the following remark.

Remark 5.6.2.3 (redundancy of (ξ^{-1}) rule): *Overlooking the whole proof of the completeness, we will discover that (ξ^{-1}) rule has not been used until in the proof of Theorem 5.6.2.2. Therefore, this rule is redundant if Γ does not contain BEs between function terms.*

This remark is very important and should be borne in mind, especially in Part V.

Actually, this lengthy chapter was to prove the completeness of $\vdash_{iBA}$. Technically, there is nothing novel once the two translations $tr_{\vec{x}}$ and $\underline{tr}$ being discovered. It seems that the whole proof was a bit too technical and delicate, although the proof was proceeded quite gradually. However, this approach does work (see Chapter 11 for xa possible improvement). This success brings us an intensional characterization of BOs.

Chapter 6

Extended Friedman's Approach

The success of capturing Completeness of $\vdash_{iBA}$ in Friedman's approach in Chapter 5 leads us to consider of extending this approach to include dependent implications (dBEs) and quasi-dependent implications (qBEs). The present chapter is the result of such extensions. The completeness for $\vdash^{d}_{iBA}$ and for $\vdash^{q}_{iBA}$ are established similar to the method of Chapter 5, i.e. reducing them to the corresponding ones of b-clones. Section 6.1 is devoted to $\vdash^{d}_{iBA}$ and $\vdash^{q}_{iBA}$ is the subject of Section 6.2. The universal BEs (uBEs) are treated in Section 6.3. The remaining part of this introduction is to verify that the two translations $tr_{\vec{x}}$ and $\underline{tr}$ preserve the dependent implications, the quasi-dependent implications and the universal equality. In other words, $tr_{\vec{x}}$ and $\underline{tr}$ preserves dBEs, qBEs and uBEs semantically.

Definition 6.0.1 (satisfactions of $\models^{d}_{iBA}$ and $\models^{q}_{iBA}$ and $\models^{u}_{iBA}$): *Let Δ range over BEs, γ range over sets of BEs and $\mathbf{B}$ be an iBA. Then,*

(a) $\mathbf{B} \models^{d}_{iBA} \gamma \mapsto \Delta$ (or $\mathbf{B} \models_{iBA} \gamma \mapsto \Delta$ even $\mathbf{B} \models \gamma \mapsto \Delta$) iff $\mathcal{B}_{\vec{x}}[\![\gamma]\!]_{\psi}$ for each $\{\vec{x}\} \subseteq V$ and $\{\vec{x}\} \supseteq (Free(\gamma) \cup Free(\Delta)) \cap V$ and for all $\psi : FV \to \mathbf{B}$ implies for every $\{\vec{y}\} \subseteq V$ and $\{\vec{y}\} \supseteq (Free(\gamma) \cup Free(\Delta)) \cap V$ $\mathcal{B}_{\vec{y}}[\![\Delta]\!]_{\psi'}$ and for all $\psi' : FV \to \mathbf{B}$,

where $\mathcal{B}_{\vec{x}}[\![p \simeq q]\!]_{\psi}$ means $\mathcal{B}_{\vec{x}}[\![p]\!]_{\psi} = \mathcal{B}_{\vec{x}}[\![q]\!]_{\psi}$ and $\mathcal{B}_{\vec{x}}[\![\gamma]\!]_{\psi}$ is its natural extension to a set of BEs, $Free(\gamma) =_{df} \bigcup_{p \simeq p' \in \gamma}(Free(p) \cup Free(p'))$ and $Free(\Delta) =_{df} \cup(Free(q) \cup Free(q')))$ $(\Delta = q \simeq q')$;

(b) $\mathbf{B} \models^{q}_{iBA} \gamma \hookrightarrow \Delta$ (or $\mathbf{B} \models_{iBA} \gamma \hookrightarrow \Delta$ even $\mathbf{B} \models \gamma \hookrightarrow \Delta$) iff $\mathcal{B}_{\vec{x}}[\![\gamma]\!]_{\psi}$ implies $\mathcal{B}_{\vec{x}}[\![\Delta]\!]_{\psi}$ for every $\{\vec{x}\} \subseteq V$ and $\{\vec{x}\} \supseteq (Free(\gamma) \cup Free(\Delta)) \cap V$ and each ψ.

(c) $\mathbf{B} \models^{u}_{iBA} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$ *(or* $\mathbf{B} \models_{iBA} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$ *even* $\mathbf{B} \models \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$*) iff* $\mathbf{B} \models^{q}_{iBA} \gamma^{(m)} \hookrightarrow \Delta^{(m)}$ *for each* $m \in M$ *implies* $\mathbf{B} \models^{q}_{iBA} \gamma \hookrightarrow \Delta$.

From Chapter 5, especially Lemma 5.4.1.1 and Theorem 5.4.2.4, we understand that when $\mathbf{B} = \mathbf{D}$

(6.0.a) $\mathcal{B}_{\vec{x}}[\![p]\!]_{\psi} = \mathcal{B}_{\vec{x}}[\![q]\!]_{\psi}$ iff $\mathcal{D}[\![tr_{\vec{x}}[\![p]\!]\,]\!](\rho) = \mathcal{D}[\![tr_{\vec{x}}[\![q]\!]\,]\!](\rho)$ and

(6.0.b) $\mathcal{D}[\![\underline{tr}[\![\underline{t}]\!]\,]\!](\rho) = \mathcal{D}[\![\underline{tr}[\![\underline{u}]\!]\,]\!](\rho)$ iff $\mathcal{B}_{\vec{x}}[\![\underline{tr}[\![\underline{t}]\!]\,]\!]_{\psi} = \mathcal{B}_{\vec{x}}[\![\underline{tr}[\![\underline{u}]\!]\,]\!]_{\psi}$,

where $\psi(f) = \rho(\underline{f})$ for $f \in FV$. These two can easily be further exploited; and Theorem 6.0.2 and Theorem 6.0.3 are the results of such exploitation.

Theorem 6.0.2 (dBEs under translations): *Let* $\{\vec{x}\} \supseteq (Free(\gamma) \cup Free(\Delta)) \cap V$. *Then, when* $\mathbf{B} = \mathbf{D}$

1. $\mathbf{D} \models_{dEQ} tr_{\vec{x}}[\![\gamma]\!] \mapsto tr_{\vec{x}}[\![\Delta]\!]$ *iff* $\mathbf{B} \models^{d}_{iBA} \gamma \mapsto \Delta$

2. $\mathbf{B} \models^{d}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \mapsto \underline{tr}[\![\underline{\Delta}]\!]$ *iff* $\mathbf{D} \models_{dEQ} \underline{\gamma} \mapsto \underline{\Delta}$

Similarly,

Theorem 6.0.3 (qBEs under translations): *Let* $\{\vec{x}\} \supseteq (Free(\gamma) \cup Free(\Delta)) \cap V$. *Thus, when* $\mathbf{B} = \mathbf{D}$

1. $\mathbf{D} \models_{qEQ} tr_{\vec{x}}[\![\gamma]\!] \hookrightarrow tr_{\vec{x}}[\![\Delta]\!]$ *iff* $\mathbf{B} \models^{q}_{iBA} \gamma \hookrightarrow \Delta$.

2. $\mathbf{B} \models^{q}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \hookrightarrow \underline{tr}[\![\underline{\Delta}]\!]$ *iff* $\mathbf{D} \models^{q}_{iBA} \underline{\gamma} \hookrightarrow \underline{\Delta}$.

Theorem 6.0.2 and Theorem 6.0.3 say that the two translations $tr_{\vec{x}}$ and $\underline{tr}$ preserve dependent implications and quasi-dependent implications. Obviously, the same treatment can be applied to uBEs.

Theorem 6.0.4 (uBEs under translations): *Let* $\{\vec{x}\} \supseteq Free(\gamma^{(m)}) \cup Free(\Delta^{(m)})$ *for every* $m \in M$, *and* $\{\vec{y}\} \supseteq Free(\gamma) \cup Free(\Delta)$. *Hence, when* $\mathbf{B} = \mathbf{D}$

1. $\mathbf{B} \models^{u}_{iBA} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$ *iff*
   $\mathbf{D} \models_{uEQ} \{tr_{\vec{x}}[\![\gamma^{(m)}]\!] \hookrightarrow tr_{\vec{x}}[\![\Delta^{(m)}]\!] | m \in M\} \mapsto (tr_{\vec{y}}[\![\gamma]\!] \hookrightarrow tr_{\vec{y}}[\![\Delta]\!])$;

2. $\mathbf{B} \models^{u}_{iBA} \{\underline{tr}[\![\underline{\gamma}^{(m)}]\!] \hookrightarrow \underline{tr}[\![\underline{\Delta}^{(m)}]\!] | m \in M\} \mapsto (\underline{tr}[\![\underline{\gamma}]\!] \hookrightarrow \underline{tr}[\![\underline{\Delta}]\!])$ *iff* $\mathbf{D} \models_{uEQ} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} \mid m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$.

6.1 Calculus $\vdash^{d}_{iBA}$ for dBEs

Firstly, we introduce a calculus $\vdash^{d^-}_{iBA}$ for dBEs, which is similar to $\vdash^{-}_{dEQ}$.

Definition 6.1.1 (calculus): *Calculus $\vdash^{d^-}_{iBA}$ is defined in the judgement form of*

$$\Gamma \vdash^{d^-}_{iBA} \gamma \mapsto \Delta \text{ or } \Gamma \vdash^{-}_{iBA} \gamma \mapsto \Delta \text{ even } \Gamma \vdash^{-} \gamma \mapsto \Delta$$

where Γ is a set of dBEs, γ is a set of BEs and Δ is a BE; and $\vdash^{d^-}_{iBA}$ is almost the same as either $\vdash^{-}_{dEQ}$ or $\vdash^{\dot{d}}_{eBA}$ except that

(a) t, u, v in $\vdash^{-}_{dEQ}$ is replaced by p, q, r in $\vdash^{d^-}_{iBA}$

(b) there two composition rules in $\vdash^{d^-}_{iBA}$ (one is for function variables and the other is for Binding Operators) instead of one in $\vdash^{-}_{dEQ}$

(c) there are three extra rules (α), (ξ) and (ξ^{-1}) in $\vdash^{d^-}_{iBA}$ which are not in $\vdash^{-}_{dEQ}$

(d) rule (d-intr) in $\vdash^{d}_{eBA}$ is absent in $\vdash^{d^-}_{iBA}$

(e) families of subtstitution functions involved in substitution rule (d-sub) of $\vdash^{d^-}_{iBA}$ must be limited to functional ones instead of arbitrary ones in either $\vdash^{\dot{d}}_{eBA}$ or $\vdash^{-}_{dEQ}$.

By (5.5.2.$*$) and (5.5.3.$*$), we expect that the following hold.

(6.1.a) $\underline{\Gamma} \cup Ax_b \vdash^{-}_{dEQ} \underline{\gamma} \mapsto \underline{\Delta}$ implies $\underline{tr}[\![\underline{\Gamma}]\!] \vdash^{d^-}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \mapsto \underline{tr}[\![\underline{\Delta}]\!]$ and

(6.1.b) $\Gamma \cup Ax_b \vdash^{d^-}_{iBA} \gamma \mapsto \Delta$ implies $tr[\![\Gamma]\!] \cup Ax_b \vdash^{-}_{dEQ} tr[\![\gamma]\!] \mapsto tr[\![\Delta]\!]$.

For (6.1.a), as a result of Subsection 5.5.2 we have that Ax_b and (d-sub) are preserved by $\underline{tr}$. The preservation of the other rules in $\vdash^{-}_{dEQ}$ can be inductively achieved by $\vdash^{d^-}_{iBA}$ under translation $\underline{tr}$. Therefore, we state this as a theorem (Theorem 6.1.2).

Theorem 6.1.2 (dBE's derivability from b-clones to iBAs): *Given any $\underline{\Gamma}$, we have that*

$$\underline{\Gamma} \cup Ax_b \vdash^{-}_{dEQ} \underline{p} \simeq \underline{q}$$

implies

$$\underline{tr}[\![\underline{\Gamma}]\!] \vdash^{d^-}_{iBA} \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!]$$

where an equation (a BE) $\underline{\Delta}$ *is short for* $\emptyset \mapsto \underline{\Delta}$.

For (6.1.b) as a result of Subsection 5.5.3, we have that (α), (ξ) and (ξ^{-1}) are preserved by translation tr (index $\vec{x}$ of $tr_{\vec{x}}$ is omitted since Lemma 5.5.3.8). The other rules can be easily verified. Thus, we state the result as Theorem 6.1.3.

Theorem 6.1.3 (dBE's derivabiltiy from iBAs to b-clones): *Given an* Γ, *we have that*

$$\Gamma \vdash^{d^-}_{iBA} p \simeq q$$

implies

$$tr[\![\Gamma]\!] \cup Ax_b \vdash^-_{dEQ} tr[\![p]\!] \simeq tr[\![q]\!]$$

where an equation (or a BE) Δ *is an abbrevaition for* $\emptyset \mapsto \Delta$.

In other words, Theorem 6.1.2 and Theorem 6.1.3 together say that

(6.1.a$*$) $\underline{\Gamma} \cup Ax_b \vdash^-_{dEQ} \underline{\gamma} \mapsto \underline{\Delta}$ iff $\underline{tr}[\![\underline{\Gamma}]\!] \vdash^{d^-}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \mapsto \underline{tr}[\![\underline{\Delta}]\!]$ when $\underline{\gamma} = \emptyset$; and

(6.1.b$*$) $\Gamma \vdash^{d^-}_{iBA} \gamma \mapsto \Delta$ iff $tr[\![\Gamma]\!] \cup Ax_b \vdash^-_{dEQ} tr_{\vec{x}}[\![\gamma]\!] \mapsto tr_{\vec{x}}[\![\Delta]\!]$ when $\gamma = \emptyset$.

Since $\vdash^{d^-}_{iBA}$ is sound and complete with respect to BEs. We would have a corresponding result. Formally,

Theorem 6.1.4 (completeness of $\vdash^{d^-}_{iBA}$): $\vdash^{d^-}_{iBA}$ *is sound and complete with respect to BEs.*

Similar to the extension from $\vdash^-_{dEQ}$ to $\vdash_{dEQ}$, $\vdash^{d^-}_{iBA}$ can be extended to $\vdash^{d}_{iBA}$, which is sound and complete. Formally,

Definition 6.1.5 (calculus $\vdash^{d}_{iBA}$): *Calculus* $\vdash^{d}_{iBA}$ *is defined in the judgement form of*

$$\Gamma \vdash^{d}_{iBA} \gamma \mapsto \Delta \text{ or } \Gamma \vdash_{iBA} \gamma \mapsto \Delta \text{ even } \Gamma \vdash \gamma \mapsto \Delta$$

and it contains all rules in $\vdash^{d^-}_{iBA}$ *with one extra rule (d-intr) as in* $\vdash_{dEQ}$.

From this, we have results similar to Theorem 6.1.2 and Theorem 6.1.3 below.

Lemma 6.1.6 (dBE's derivations between iBAs and b-clones): *Replacing* $\vdash^{d^-}_{iBA}$ *and* $\vdash^{-}_{dEQ}$ *by* $\vdash^{d}_{iBA}$ *and* $\vdash_{dEQ}$ *respectively in (6.1.a) and in (6.1.b), we have that both new (6.1.a) and new (6.1.b) are valid.*

Consequently,

Theorem 6.1.7 (completeness of $\vdash^{d}_{iBA}$): $\vdash^{d}_{iBA}$ *is sound and complete for dBEs.*

6.2 Calculus $\vdash^{q}_{iBA}$ for qBEs

Analogous to Section 6.1 and to Section 2.4, we give a calculus $\vdash^{q^-}_{iBA}$ for qBEs. Formally,

Definition 6.2.1 (calculus $\vdash^{q^-}_{iBA}$): *Calculus* $\vdash^{q^-}_{iBA}$ *is defined in the judgement form of*

$$\Gamma \vdash^{q^-}_{iBA} \gamma \hookrightarrow p \simeq q \text{ and } \Gamma \vdash^{q^-}_{iBA} \gamma \mapsto p \simeq q$$

or

$$\Gamma \vdash_{iBA} \gamma \hookrightarrow p \simeq q \text{ and } \Gamma \vdash_{iBA} \gamma \mapsto p \simeq q$$

where Γ *is a set of dBEs and qBEs; and* $\vdash^{q^-}_{iBA}$ *is almost the same as either* $\vdash^{d}_{qEQ}$ *or* $\dot{\vdash}^{q}_{eBA}$ *except that*

(a) t, u, v *in* $\vdash^{d}_{qEQ}$ *are replaced by* p, q, r *in* $\vdash^{q^-}_{iBA}$ *respectively*

(b) two composition rules (one is for function variables and the other is for Binding Operators) are in $\vdash^{q^-}_{iBA}$ *instead of one in* $\vdash^{d}_{qEQ}$

(c) three extra rules (α), (ξ) *and* (ξ^{-1}) *are available in* $\vdash^{q^-}_{iBA}$ *but not in* $\vdash^{d}_{qEQ}$

(d) skolemization rule (q-skm) in $\dot{\vdash}^{q}_{eBA}$ *is not available in* $\vdash^{q^-}_{iBA}$

(e) the families of substitution functions involved in substitute rule (q-sub) of $\vdash^{q^-}_{iBA}$ *are limited to functional ones instead of arbitrary ones as in* $\dot{\vdash}^{q}_{eBA}$.

We should be aware of that (ξ) rule is not like $\{t \simeq u\} \hookrightarrow \langle \vec{x} : t \rangle \simeq \langle \vec{x} : u \rangle$, i.e. (ξ) rule in $\vdash^{q^-}_{iBA}$ is the same as (ξ) rule in $\vdash^{d}_{iBA}$; which is similar to the situation

between $\dot{\vdash}^{d}_{eBA}$ and $\dot{\vdash}^{q}_{eBA}$. So, this is an obvious example to show that the difference between dBEs and qBEs is not just a notational variation, and it is quite subtle.

Remark 6.2.2 (soundness of (ξ) rule in $\vdash^{q^-}_{iBA}$): *The soundness of all rules in $\vdash^{q^-}_{iBA}$ is quite obvious, probably except (ξ) rule. But any suspicion on this will disappear if you check the results of Subsection 5.4.3, i.e. we will have*

$$\mathcal{B}_{\vec{y}}[\![\langle \vec{x} : t\rangle]\!]_{\psi} = \mathcal{B}_{\vec{y}\,\vec{z}}[\![t\,[\vec{x} := \vec{z}\,]\,]\!] \text{ and } \mathcal{B}_{\vec{y}}[\![\langle \vec{x} : u\rangle]\!]_{\psi} = \mathcal{B}_{\vec{y},\vec{z}}[\![u\,[\vec{x} := \vec{z}\,]\,]\!]$$

where $\{\vec{z}\} \cap \{\vec{y}\} = \emptyset$, $\{\vec{z}\} \cap (Free(t) - \{\vec{x}\}) = \emptyset$ *and* $\{\vec{z}\} \cap (Free(u) - \{\vec{x}\}) = \emptyset$.

Each of them can transform to the one with only free variables in both function terms bound in the interpretation. Hence, they are equal because of the precondition of (ξ) rule:

$$(\xi) \quad \vdash^{q^-}_{iBA} \{t \simeq u\} \mapsto \langle \vec{x} : t\rangle \simeq \langle \vec{x} : u\rangle.$$

Analogous to Section 6.1 we hope the following hold:

(6.2.a)

$$\underline{\Gamma} \cup Ax_b \vdash^{d}_{qEQ} \underline{\gamma} \mapsto \underline{p} \simeq \underline{q} \text{ and } \underline{\Gamma} \cup Ax_b \vdash^{d}_{qEQ} \underline{\gamma} \hookrightarrow \underline{p} \simeq \underline{q}$$

implies

$$\underline{tr}[\![\underline{\Gamma}]\!] \vdash^{q^-}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \mapsto \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!] \text{ and } \underline{tr}[\![\underline{\Gamma}]\!] \vdash^{q^-}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \hookrightarrow \underline{tr}[\![\underline{p}]\!] \simeq \underline{tr}[\![\underline{q}]\!];$$

and

(6.2.b)

$$\Gamma \vdash^{q^-}_{iBA} \gamma \mapsto p \simeq q \text{ and } \Gamma \vdash^{q^-}_{iBA} \gamma \hookrightarrow p \simeq q$$

implies

$$tr[\![\Gamma]\!] \cup Ax_b \vdash^{d}_{qEQ} tr[\![\gamma]\!] \mapsto tr[\![p]\!] \simeq tr[\![q]\!] \text{ and } tr[\![\Gamma]\!] \cup Ax_b \vdash^{d}_{qEQ} tr[\![\gamma]\!] \hookrightarrow tr[\![p]\!] \simeq tr[\![q]\!].$$

For (6.2.a), as a consequence of Section 6.1, we have that Ax_b and (q-sub) rule in $\vdash^{q^-}_{iBA}$ are preserved by translation $\underline{tr}$. This is because that all rules in $\vdash^{d}_{iBA}$ are implied by the rules of $\vdash^{q^-}_{iBA}$, and other rules can be inductively shown that they are

preserved by $\vdash^{q^-}_{iBA}$ under translation $\underline{tr}$. For example, we can let $\vec{\varrho}$ be $\underline{tr} \circ \iota \lceil_{\vec{X}} \circ _$ (i.e. $\vec{\varrho}(f) = \underline{tr}[\![\iota(\underline{f})]\!]$), then the substitution rule

$$\frac{\underline{\Gamma} \cup Ax_b \vdash_{qEQ} \underline{\gamma} \hookrightarrow \underline{\Delta}}{\underline{\Gamma} \cup Ax_b \vdash_{qEQ} \iota(\underline{\gamma}) \hookrightarrow \iota(\underline{\Delta})}$$

(where $\iota : T(\vec{X}) \to T(\vec{X})$) is preserved by $\vdash^{q^-}_{iBA}$ under $\underline{tr}$, i.e.

$$\frac{\underline{tr}[\![\underline{\Gamma}]\!] \vdash_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \hookrightarrow \underline{tr}[\![\underline{\Delta}]\!]}{\underline{\Gamma} \vdash_{iBA} \underline{tr}[\![\iota(\underline{\gamma})]\!] \hookrightarrow \underline{tr}[\![\iota(\underline{\Delta})]\!]}$$

So, when $\underline{\gamma}$ is empty, (6.2.a) follows easily. Formally,

Theorem 6.2.3 (qBE's derivability from b-clones to iBAs): *(6.2.a) is valid when* $\underline{\gamma} = \emptyset$.

For (6.2.b), since (α) is the same in both $\vdash^{d}_{iBA}$ and $\vdash^{q^-}_{iBA}$ and since all rules in $\vdash^{d^-}_{iBA}$ are implied by the rules in $\vdash^{q^-}_{iBA}$, (α) is preserved by $\vdash^{d}_{qEQ}$ under translation tr.

For (ξ) rule, i.e.

$$tr[\![\Gamma]\!] \cup Ax_b \vdash^{d}_{qEQ} \{tr_{\vec{y}}[\![t]\!] \simeq tr_{\vec{y}}[\![u]\!]\} \hookrightarrow tr_{\vec{y}}[\![\langle \vec{x} : t \rangle]\!] \simeq tr_{\vec{y}}[\![\langle \vec{x} : u \rangle]\!]$$

for any Γ, Remark 6.2.2 provides a clue for us. As a result of Subsection 5.5.3, we would have

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{y}}[\![\langle \vec{x} : t \rangle]\!] \simeq tr_{\vec{y},\vec{z}}[\![t\,[\vec{x} := \vec{z}\,]\,]\!]$$

and

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{y}}[\![\langle \vec{x} : u \rangle]\!] \simeq tr_{\vec{y},\vec{z}}[\![u\,[\vec{x} := \vec{z}\,]\,]\!],$$

where $\{\vec{z}\} \cap \{\vec{y}\} = \emptyset$, $\{\vec{z}\} \cap (Free(t) - \{\vec{x}\}) = \emptyset$ and $\{\vec{z}\} \cap (Free(u) - \{\vec{x}\}) = \emptyset$. Similarly, we can have that

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{y},\vec{z}}[\![t\,[\vec{x} := \vec{z}\,]\,]\!] \simeq tr_{\vec{y},\vec{z}}[\![u\,[\vec{x} := \vec{z}\,]\,]\!]$$

iff

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{x'}}[\![t]\!] \simeq tr_{\vec{x'}}[\![u]\!]$$

where $\vec{x'} \supseteq \{(Free(t) \cup Free(u) \cap V)\}$. On the other hand, the same treatment can go to the pre-condition of (ξ) rule, i.e.

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{y}}[\![t]\!] \simeq tr_{\vec{y}}[\![u]\!] \quad \text{iff} \quad tr[\![\Gamma]\!] \cup Ax_b \vdash_{qEQ} tr_{\vec{x'}}[\![t]\!] \simeq tr_{\vec{x'}}[\![u]\!].$$

We have that for every $\underline{v}$, $\{\underline{v} \simeq \underline{v}\} \hookrightarrow \underline{v} \simeq \underline{v}$. In turn, we can use (q-trs), (q-wkn) and (q-MP) to obtain what we want. Therefore, (ξ) rule is preserved by $\vdash^{d}_{qEQ}$ under translation tr.

The other rules in $\vdash^{q^-}_{iBA}$ can be inductively proved that they are preserved by $\vdash^{d}_{qEQ}$ with Ax_b under tr. Formally

Theorem 6.2.4 (qBE's derivability from iBAs to b-clones): *(6.2.b) is valid when $\gamma = \emptyset$.*

From Theorem 6.2.3 and Theorem 6.2.4, we can get that $\vdash^{q^-}_{iBA}$ is sound and complete with respect to BEs. Namely

Theorem 6.2.5 (completeness of $\vdash^{q^-}_{iBA}$): *$\vdash^{q^-}_{iBA}$ is sound and complete with respect to BEs.*

Similar to Section 2.4, we can extend the above completeness result to dBEs. That is similar to Lemma 6.1.6, we would have

Lemma 6.2.6 (qBE's derivability between iBAs and b-clones):

1. *When we only consider dBEs, (6.2.a) is valid; and*

2. *when we only consider dBEs, (6.2.b) is valid.*

Succeedingly,

Theorem 6.2.7 (completeness of $\vdash^{q^-}_{iBA}$): *$\vdash^{q^-}_{iBA}$ is sound and complete with respect to dBEs.*

Following Section 2.4, we can extend $\vdash^{q^-}_{iBA}$ to $\vdash^{q}_{iBA}$. Formally,

Definition 6.2.8 (calculus $\vdash^{q}_{iBA}$): *Calculus $\vdash^{q}_{iBA}$ is defined in the judgement form of*

$$\Gamma \vdash^{q}_{iBA} \gamma \hookrightarrow p \simeq q \text{ and } \Gamma \vdash^{q}_{iBA} \gamma \mapsto p \simeq q$$

or

$$\Gamma \vdash_{iBA} \gamma \hookrightarrow p \simeq q \text{ and } \Gamma \vdash^{q}_{iBA} \gamma \mapsto p \simeq q.$$

It contains all rules in $\vdash^{q^-}_{iBA}$ with one extra rule, i.e. skolemization rule (q-skm) in either $\vdash^{q}_{eBA}$ or $\vdash_{qEQ}$.

Analogous to Lemma 6.2.6, we would have a similar lemma. Formally,

Lemma 6.2.9 (qBE's derivations between iBAs and b-clones): *Replacing* $\vdash^{q^-}_{iBA}$ *and* $\vdash^{d}_{qEQ}$ *by* $\vdash^{q}_{iBA}$ *and* $\vdash_{qEQ}$ *respectively, we have that both (6.2.a) and (6.2.b) are valid.*

Therefore, for $\vdash^{q}_{iBA}$, we can have the completeness. That is,

Theorem 6.2.11 (completeness of $\vdash^{q}_{iBA}$): *Calculus* $\vdash^{q}_{iBA}$ *is sound and complete with respect to dBEs and qBEs.*

6.3 Calculus $\vdash^{u}_{iBA}$ for uBEs

This section is investigating a deduction system of uBEs, i.e. Ω of the form

$$\{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\} \mapsto (\gamma \hookrightarrow \Delta).$$

$\mathbf{B}$ is said to be a *model* of it, written as $\mathbf{B} \models^{u}_{iBA} \Omega$ (or $\mathbf{B} \models_{iBA} \Omega$ even $\mathbf{B} \models \Omega$), iff $\mathbf{B} \models^{q}_{iBA} \{\gamma^{(m)} \hookrightarrow \Delta^{(m)} | m \in M\}$ implies $\mathbf{B} \models^{q}_{iBA} \gamma \hookrightarrow \Delta$. Therefore, satisfactions among $\models^{u}_{iBA}$ and $\models_{iBA}$, $\models^{d}_{iBA}$ and $\models^{q}_{iBA}$ coincide with each other under the following conditions: when $M = \emptyset$, $\models^{u}_{iBA}$ is equivalent to $\models^{q}_{iBA}$ (i.e. Ω is a (pure) qBE); further if $\gamma = \emptyset$, $\models^{u}_{iBA}$ is equivalent to $\models_{iBA}$ (i.e. Ω is a BE); when all γ's are empty, $\models^{u}_{iBA}$ is equivalent to $\models^{q}_{iBA}$ (i.e. Ω is a dBE). So, we will adopt similar abbreviations as the ones in Section 4.3 (or Section 2.5).

Obviously, we are investigating a more general format which combines all three formats of BEs, dBEs and qBEs into one.

Analogous to previous sections and Section 2.5 (or Section 4.3), we can naturally define the following.

(i) $\mathcal{K} \models^{u}_{iBA} \Omega$ (or $\mathcal{K} \models_{iBA} \Omega$ even $\mathcal{K} \models \Omega$) is the natural extension of $\mathbf{B} \models^{u}_{iBA} \Omega$ to a class $\mathcal{K}$ of iBAs;

(ii) $\mathbf{B} \models^{u}_{iBA} \Gamma$ (or $\mathbf{B} \models_{iBA} \Gamma$ even $\mathbf{B} \models_{iBA} \Gamma$) is $\mathbf{B} \models_{iBA} \Omega$ for each $\Omega \in \Gamma$, which is a set of Ω's;

(iii) $\mathcal{K} \models^{u}_{iBA} \Gamma$ (or $\mathcal{K} \models_{iBA} \Gamma$ even $\mathcal{K} \models \Gamma$) is the extension of $\mathbf{B} \models^{u}_{iBA} \Gamma$ to a class of iBAs;

(iv) $\Gamma \models^{u}_{iBA} \Omega$ (or $\Gamma \models_{iBA} \Omega$ even $\Gamma \models \Omega$) is $\mathbf{B} \models_{iBA} \Omega$ if $\mathbf{B} \models_{iBA} \Gamma$.

Definition 6.3.1 (calculus $\vdash^{u-}_{iBA}$): *Calculus $\vdash^{u-}_{iBA}$ is defined in the judgement form of*

$$\Gamma \vdash^{u-}_{iBA} \Omega \text{ or } \Gamma \vdash_{iBA} \Omega.$$

It is almost the same as either $\vdash_{uEQ}$ or $\dot{\vdash}^{u}_{eBA}$ except that

(a) t, u, v in $\vdash_{uEQ}$ are replaced by p, q, r respectively in $\vdash^{u-}_{iBA}$

(b) two composition rules (one is for functional variables and the other is for Binding Operators) are in $\vdash^{u-}_{iBA}$ instead of one in $\vdash_{uEQ}$

(c) three extra rules (α), (ξ) and (ξ^{-1}) are available in $\vdash^{u-}_{iBA}$ but not in $\vdash_{uEQ}$

(d) rule (u-q-u) in either $\vdash_{uEQ}$ or $\dot{\vdash}^{u}_{eBA}$ is absent in $\vdash^{u-}_{iBA}$

(e) families of substitution functions involved in substitution rule (u-sub) of $\vdash^{u-}_{iBA}$ are limited to functional ones instead of arbitrary ones as in $\dot{\vdash}^{u}_{eBA}$.

Analogous to Section 6.2, we hope that the following holds

(6.3.a) $\underline{\Gamma} \cup Ax_b \vdash_{uEQ} \underline{\Omega}$ implies $\underline{tr}[\![\underline{\Gamma}]\!] \vdash^{u-}_{iBA} \underline{tr}[\![\underline{\Omega}]\!]$ and

(6.3.b) $\Gamma \vdash^{u-}_{iBA} \Omega$ implies $tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr[\![\Omega]\!]$.

For (6.3.a), as a consequence of Section 6.1, we have that Ax_b and (u-sub) are preserved by translation $\underline{tr}$, since both all rules in $\vdash^{d-}_{iBA}$ and in $\vdash^{q-}_{iBA}$ are implied by the rules in $\vdash^{u-}_{iBA}$. Other rules can be inductively shown to be preserved by $\vdash^{u-}_{iBA}$ under translation $\underline{tr}$. For example, we can let $\vec{\varrho}$ be $\underline{tr} \circ \iota\lceil_{\vec{X}} \circ _$ (i.e. $\vec{\varrho}(f) = \underline{tr}[\![\iota(\underline{f})]\!]$), then the (u-sub) rule

$$\frac{\underline{\Gamma} \vdash_{uEQ} \underline{\gamma} \hookrightarrow \underline{\Delta}}{\Gamma \vdash_{uEQ} \iota(\underline{\gamma}) \hookrightarrow \iota(\underline{\Delta})}$$

(where $\iota : T(X) \to T(Y)$) is preserved by $\vdash^{u-}_{iBA}$ under $\underline{tr}$, i.e.

$$\frac{\underline{tr}[\![\underline{\Gamma}^u]\!] \vdash^{u-}_{iBA} \underline{tr}[\![\underline{\gamma}]\!] \hookrightarrow \underline{tr}[\![\underline{\Delta}]\!]}{\underline{tr}[\![\underline{\Gamma}^u]\!] \vdash^{u-}_{iBA} \underline{tr}[\![\iota(\underline{\gamma})]\!] \hookrightarrow \underline{tr}[\![\iota(\underline{\Delta})]\!]}.$$

So, when M in Ω is empty, (6.3.a) follows easily. Formally,

Theorem 6.3.2 (uBE's derivability from b-clones to iBAs): *(6.3.a) is valid when M in Ω is empty.*

For (6.3.b), since (α) in $\vdash^{u-}_{iBA}$ is essentially the same as (α) in $\vdash^{d-}_{iBA}$, and since $\vdash^{d-}_{iBA}$ is implied by $\vdash^{u-}_{iBA}$, (α) is preserved by $\vdash^{u-}_{iBA}$ under translation tr.

For (ξ) rule, i.e. $tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} \{tr_{\vec{y}}[\![t]\!] \simeq tr_{\vec{y}}[\![u]\!]\} \hookrightarrow tr_{\vec{y}}[\![\langle \vec{x} : t\rangle]\!] \simeq tr_{\vec{y}}[\![\langle \vec{x} : u\rangle]\!]$ for any Γ, Remark 6.2.2 provides a clue for us. As a result of Subsection 5.4.3, we would have

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{y}}[\![\langle \vec{x} : t\rangle]\!] \simeq tr_{\vec{y},\vec{z}}[\![t\,[\vec{x} := \vec{z}]\,]\!]$$

and

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{y}}[\![\langle \vec{x} : u\rangle]\!] \simeq tr_{\vec{y},\vec{z}}[\![u\,[\vec{x} := \vec{z}]\,]\!],$$

where $\{\vec{z}\} \cap \{\vec{y}\} = \emptyset$, $\{\vec{z}\} \cap (Free(t) - \{\vec{x}\}) = \emptyset$ and $\{\vec{z}\} \cap (Free(u) - \{\vec{x}\}) = \emptyset$.

Similarly, we can have that

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{y}\vec{z}}[\![t\,[\vec{x} := \vec{z}]\,]\!] \simeq tr_{\vec{y}\vec{z}}[\![u\,[\vec{x} := \vec{z}]\,]\!]$$

iff

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{x'}}[\![t]\!] \simeq tr_{\vec{x'}}[\![u]\!]$$

where $\{\vec{x'}\} \supseteq (Free(t) \cup Free(u)) \cap V$.

On the other hand, the same treatment can go to the pre-condition of (ξ) rule, i.e.

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{y}}[\![t]\!] \simeq tr_{\vec{y}}[\![u]\!]$$

iff

$$tr[\![\Gamma]\!] \cup Ax_b \vdash_{uEQ} tr_{\vec{x'}}[\![t]\!] \simeq tr_{\vec{x'}}[\![u]\!].$$

We have that for every $\underline{v}$, $\{\underline{v} \simeq \underline{v}\} \hookrightarrow \underline{v} \simeq \underline{v}$. In turn, we can use (u-trs), (u-wkn) and (u-cut) to obtain what we want. Therefore, (ξ) rule is preserved by $\vdash^{u-}_{iBA}$ under translation tr.

The other rules in $\vdash^{q-}_{iBA}$ can be inductively proved that they are preserved by $\vdash^{u-}_{iBA}$ with Ax_b under tr. Formally

Theroem 6.3.3 (uBE's derivability from iBAs to b-clones): *(6.3.b) is valid when M in Ω is empty.*

From Theorem 6.3.2 and Theorem 6.3.3, we can get that $\vdash^{u^-}_{iBA}$ is sound and complete with respect to BEs. Namely, for $\vdash^{u^-}_{iBA}$, we have soundness and completeness with respect to BEs, i.e. completeness up to the form of Δ, which is short for $\emptyset \mapsto (\emptyset \hookrightarrow \Delta)$. Formally,

Theorem 6.3.4 (first completeness of $\vdash^{u^-}_{iBA}$): *Calculus $\vdash^{u^-}_{iBA}$ is sound and complete with respect to BEs.*

Completeness of $\vdash^{u^-}_{iBA}$ up to dBEs follows easily from the above theorem. That is, analogous to the extension from $\vdash^{d^-}_{iBA}$ to $\vdash^{d}_{iBA}$ in Section 6.1, we have soundness and completeness of $\vdash^{u^-}_{iBA}$ with respect to dBEs, i.e. up to the form of $\gamma \mapsto \Delta$, which is an abbreviation for $\{\emptyset \hookrightarrow \Delta' | \Delta' \in \gamma\} \mapsto (\emptyset \hookrightarrow \Delta)$.

Theorem 6.3.5 (second completeness of $\vdash^{u^-}_{iBA}$): *$\vdash^{u^-}_{iBA}$ is sound and complete with respect to dBEs (or more properly dependent uBEs).*

Similar to the extension from $\vdash^{q^-}_{iBA}$ to $\vdash^{q}_{iBA}$ in Section 6.2, we can verify that $\vdash^{u^-}_{iBA}$ is sound and complete with respect to qBEs $\gamma \hookrightarrow \Delta$, which is short for $\emptyset \mapsto (\gamma \hookrightarrow \Delta)$. That is, analogous to Section 6.2, we have a theorem below.

Theorem 6.3.6 (third completeness of $\vdash^{u^-}_{iBA}$): *$\vdash^{u^-}_{iBA}$ is sound and complete with respect to qBEs (or more properly quasi-dependent uBEs).*

To get completeness of uBEs, we need to extend $\vdash^{u^-}_{iBA}$ to $\vdash^{u}_{iBA}$ as follows.

Definition 6.3.7 (calculus $\vdash^{u}_{iBA}$): *Calculus $\vdash^{u}_{iBA}$ is defined in the judgement form of*

$$\Gamma \vdash^{u}_{iBA} \Omega \text{ or } \Gamma \vdash_{iBA} \Omega \text{ even } \Gamma \vdash \Omega$$

It contains all rules in $\vdash^{u^-}_{iBA}$ with one extra rule (u-q-u) in either $\vdash^{u}_{eBA}$ or $\vdash_{uEQ}$.

With previous three completeness of $\vdash^{u^-}_{iBA}$ and the fact that $\vdash^{u}_{iBA}$ contains all rules of $\vdash^{u^-}_{iBA}$, we have the completeness for $\vdash^{u}_{iBA}$ below.

Theorem 6.3.8 (completeness of $\vdash^{u}_{iBA}$): *$\vdash^{u}_{iBA}$ is sound and complete with respect to uBEs, i.e.*

$$\Gamma \models^{u}_{iBA} \Omega \text{ iff } \Gamma \vdash^{u}_{iBA} \Omega$$

where Γ *is a set of uBEs.*

Part IV

Many-Sorted FBO

It is a common practice of introducing higher types (or higher sorts) to deal with non-ordinary objects. However, the Framework for BOs in this thesis tends to introduce "higher order" syntactically (i.e. binding primitive into syntax), instead of into higher sorts. Nevertheless, it does not fail to be a good idea to introduce higher sorts into binding signature Σ^{BO}. But we only show how to introduce multiple sorts into FBO, so-called *many-sorted* Framework for Binding Operators (many-sorted FBO) in this part, i.e. Chapter 7. This extension of from single-sorted to many-sorted is not a trivial extension, see [75,42,164]. The introduction of higher sorts (or higher types) into FBO is briefly discussed in Section 7.3.

Chapter 7

Extensions to Many-Sorted FBO

We are going to extend the Framework for Binding Operators from *single-sorted* to many-sorted. Technically, let Σ^{BO} be a binding signature, then we are considering a FBO with $|Sort(\Sigma^{BO})| > 1$ (sometimes $Sort(\Sigma^{BO})$ is simply written as S) and $Op(\Sigma^{BO})$ (sometimes simply denoted as Σ^{BO}). In many-sorted FBO,

(7.0.a) a constant $\sigma \in \Sigma^{BO}$ is sorted by s where $s \in S = Sort(\Sigma^{BO})$;

(7.0.b) a function $\sigma \in \Sigma^{BO}$ is sorted by $< \vec{s}, s >$ (or $\vec{s} \rightarrow s$) where $\vec{s}$ is a list of sorts in S (sometimes written as $\vec{s} \in S^*$ such that $S^* = \bigcup_{j \in Nat} S^j$;

(7.0.c) a functional $\sigma \in \Sigma^{BO}$ is sorted by $<< \vec{\mathbf{s}}, \vec{s'} >, s'' >$ (or $< \vec{\mathbf{s}}, \vec{s'} > \rightarrow s''$) where sort $s'' \in S$, list $\vec{s'} \in S^{|\vec{s'}|}$, the bold face list $\vec{\mathbf{s}}$ is a list of bold face $\vec{\mathbf{s}}(i) = \mathbf{s}_i$ such that bold face $\mathbf{s}_i$ is $\vec{s^i} \Rightarrow s_i$ $(i = 1, 2, ..., |\vec{\mathbf{s}}|)$, and list $\vec{s^i} \in S^{|\vec{s^i}|}$, sort $s_i \in S$.

We define many-sorted binding terms as follows.

Definition 7.0.1 (many-sorted binding terms BT): *Let V_s be a set of ordinary variables for sort $s \in S$, $FV_{\mathbf{s}}$ be a set of function variables sorted by bold face $\mathbf{s}$ $(= \vec{s} \Rightarrow s')$. The many-sorted binding term BT is a pair $< T, FT >$ of term T and function term FT where T is sorted by S and FT is sorted by $\mathbf{S}$ such that for each $s \in S$ and every $\mathbf{s} \in \mathbf{S}$, ordinary term T_s and function term $FT_{\mathbf{s}}$ are generated using the following formation rules:*

1.

$$\frac{x \in V_s}{x \in T_s}$$

2.

$$\frac{t_1 \in T_{s_1}, t_2 \in T_{s_2}, ..., t_n \in T_{s_n}}{f(t_1, t_2, ..., t_n) \in T_{s'}}$$

where $f \in FV_{\mathbf{s}}$ *and bold face* $\mathbf{s} = \vec{s} \Rightarrow s'$ *and* $|\vec{s}| = n$,

3.

$$\frac{t \in T_{s'}}{\langle \vec{x} : t \rangle \in FT_{\mathbf{s}}}$$

where $x_1 \in V_{s_1}, x_2 \in V_{s_2}, ..., x_n \in V_{s_n}$, *and bold face* $\mathbf{s} = \vec{s} \Rightarrow s'$ *and* $|\vec{s}| = n$,

4. *for* $\sigma \in \Sigma^{BO}_{<<\vec{s},\vec{s'}>,s''>}$,

$$\frac{ft_1 \in FT_{\mathbf{s}_1}, ft_2 \in FT_{\mathbf{s}_2}, ..., ft_\ell \in FT_{\mathbf{s}_\ell};\ t_1 \in T_{s'_1}, t_2 \in T_{s'_2}, ..., t_n \in T_{s'_n}}{\sigma(ft_1, ft_2, ..., ft_\ell, t_1, t_2, ..., t_n) \in T_{s''}}$$

where $\vec{\mathbf{s}}(i) = \mathbf{s}_i$ *and* $|\vec{\mathbf{s}}| = \ell$, *and* $\vec{s'}(j) = s'_j$ *and* $|\vec{s'}| = n$.

For simplicity, we assume that every variable set V_s ($s \in S$) and every function variable set $FV_{\mathbf{s}}$ ($\mathbf{s} = \vec{s} \Rightarrow s'$, $\vec{s} \in S^{|\vec{s}|}$ and $s' \in S$) are disjointed with each other. Then, the results of previous chapters (e.g. Chapter 3 and Chapter 5) follows easily to the many-sorted FBO. Apparently, certain treatment of Chapter 2, say variable index $\simeq_X$, can be adjusted to deal with many-sorted FBO. Perhaps more importantly, the condition for variable index free calculi need to be adjusted, which should essentially be the same.

Section 7.1 will briefly show how to extend the previous work of Friedman's approach to include many-sorted cases. No detailed work will be presented, because that is a kind of easy extensions with more notational modification. Similarly, neither the extension of the previous results of Birkhoff's approach to many-sorted cases will be carried out due to the same reason. However, Section 7.2 is reserved as a reminder for this extension. Higher-sorted FBO will be briefly discussed in Section 7.3.

7.1 Many-sorted Friedman's approach

Let us turn our attention to many-sorted b-clones. Now, $\underline{\Sigma}$ is its signature and $\underline{S} = Sort(\Sigma)$. It contains following operations

(7.1.a) constants: $\underline{pr}_i^{\vec{s}}$ is sorted by $\underline{\mathbf{s}}$, where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s}_i)$ and $\vec{\underline{s}}(i) = \underline{s}_i$;

(7.1.b) functions: $\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}'}$ is sorted by $< \underline{\mathbf{s}}, \vec{\underline{\mathbf{s}}}'' >\!\!\rightarrow \underline{\mathbf{s}}')$, where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$, bold face $\underline{\mathbf{s}}' = (\vec{\underline{s}}' \Rightarrow \underline{s})$, bold face $\vec{\underline{\mathbf{s}}}''(i) = \underline{\mathbf{s}}_i'' = (\vec{\underline{s}}' \Rightarrow \underline{s}_i)$ and $\underline{s}_i = \vec{\underline{s}}(i)$, $(i = 1, 2, ..., |\vec{\underline{s}}| = |\vec{\underline{s}''}|)$;

(7.1.c) functionals: $\underline{\sigma}_{\underline{\mathbf{s}}}$ is sorted by $< \vec{\underline{\mathbf{s}}}', \vec{\underline{\mathbf{s}}}'' >\!\!\rightarrow \underline{\mathbf{s}}$, where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$, bold face $\vec{\underline{\mathbf{s}}}'(i) = \underline{\mathbf{s}}_i' = (\vec{\underline{s}}, \vec{\underline{s}^i} \Rightarrow \underline{s}_i')$ and $|\vec{\underline{s}^i}| = m_i$ $(i = 1, 2, ..., n)$, and bold face $\vec{\underline{\mathbf{s}}}''(j) = \underline{\mathbf{s}}_j'' = (\vec{\underline{s}} \Rightarrow \underline{s}_j'')$ $(j = 1, 2, ..., n)$.

Here, variables range over $\chi_{\underline{\mathbf{s}}}$ by $\underline{f}$, and they are sorted by bold face $\underline{\mathbf{s}}$ for each $\underline{\mathbf{s}} = \vec{s} \Rightarrow s'$.

Definition 7.1.1 (many-sorted terms of b-clones): *Term $\underline{T}_{\underline{\mathbf{s}}}$ with $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$ is generated using the following formation rules:*

1. $\underline{pr}_i^{\vec{s}} \in \underline{T}_{\underline{\mathbf{s}}}$

   where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s}_i)$;

2. $$\frac{\underline{f} \in \chi_{\underline{\mathbf{s}}}}{\underline{f} \in \underline{T}_{\underline{\mathbf{s}}}}$$

3. $$\frac{\underline{u}_i \in \underline{T}_{\underline{\mathbf{s}}_i''};\ \underline{t} \in \underline{T}_{\underline{\mathbf{s}}}}{\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}'}(\underline{t}, \vec{\underline{u}}) \in \underline{T}_{\underline{\mathbf{s}}'}}$$

   where bold face $\underline{\mathbf{s}} = (\vec{s} \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}' = (\vec{\underline{s}}' \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}_i'' = (\vec{\underline{s}}' \Rightarrow \underline{s}_i)$ *and* $\underline{s}_i = \vec{\underline{s}}(i)$ $(1 \leq i \leq |\vec{\underline{s}}|)$;

4. $$\frac{\underline{t}_i \in \underline{T}_{\underline{\mathbf{s}}_i'};\ \underline{u}_j \in \underline{T}_{\underline{\mathbf{s}}_j''}}{\underline{\sigma}_{\underline{\mathbf{s}}}(\vec{\underline{t}}, \vec{\underline{u}}) \in \underline{T}_{\underline{\mathbf{s}}}}$$

where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$, *bold face* $\vec{\underline{\mathbf{s}}'} = \underline{\mathbf{s}}'_i = (\vec{\underline{s}}, \vec{\underline{s}^i} \Rightarrow \underline{s}'_i)$, *and bold face* $\underline{\mathbf{s}}''(j) = \underline{\mathbf{s}}''_j = (\vec{\underline{s}} \Rightarrow \underline{s}''_j)$.

Many-sorted terms of b-clones is being considered to be sorted by $\underline{\mathbf{s}}$ where $\underline{\mathbf{s}}$ is a kind of $(\vec{\underline{s}} \Rightarrow \underline{s}')$. Similarly, many-sorted b-clones can be defined accordingly as follows.

Definition 7.1.2 (many-sorted b-clones): *an* Σ*-algebra* $\mathbf{D}$ *is said to be a b-clones if it satisfies the following laws:*

1. *(m-*Ax_b*-left-proj)*

$$\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}'}(\underline{pr}_i^{\vec{\underline{s}}}, \vec{\underline{t}}) \simeq \underline{t}_i,$$

where $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s}_i)$, $\underline{\mathbf{s}}' = (\vec{\underline{s}'} \Rightarrow \underline{s}_i)$, *and* $\underline{t}_j \in \underline{T}_{(\vec{\underline{s}'} \Rightarrow \underline{s}_j)}$.

2. *(m-*Ax_b*-right-proj)*

$$\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}}(\underline{t}, \underline{Pr}_{1,|\vec{\underline{s}}|}^{\vec{\underline{s}}}) \simeq \underline{t},$$

where $\underline{t} \in \underline{T}_{\underline{\mathbf{s}}}$.

3. *(m-*Ax_b*-assoc)*

$$\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}''}(\underline{o}_{\underline{\mathbf{s}}',\underline{\mathbf{s}}}(\underline{t}, \vec{\underline{v}}), \vec{\underline{u}}) \simeq \underline{o}_{\underline{\mathbf{s}}',\underline{\mathbf{s}}''}(\underline{t}, \underline{o}_{\vec{\underline{\mathbf{s}}},\underline{\mathbf{s}}''}(\vec{\underline{v}}, \vec{\underline{u}})),$$

where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}'' = (\vec{\underline{s}''} \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}'_j = (\vec{\underline{s}} \Rightarrow \underline{s}'_j)$, *bold face* $\underline{\mathbf{s}}''_j = (\vec{\underline{s}''} \Rightarrow \underline{s}'_j)$, $\underline{t} \in \underline{T}_{\underline{s}'}$, $\underline{v}_j \in \underline{T}_{\underline{s}'_j}$, $\underline{u}_i \in \underline{T}_{(\vec{\underline{s}''} \Rightarrow \underline{s}_i)}$, *and* $\underline{o}_{\underline{\mathbf{s}}'_j,\underline{\mathbf{s}}''_j}(\underline{v}_j, \vec{\underline{u}})$;

4. *(m-*Ax_b*-unif)*

$$\underline{o}_{\underline{\mathbf{s}},\underline{\mathbf{s}}'}(\underline{\sigma}_{\underline{\mathbf{s}}}(\vec{\underline{t}}, \vec{\underline{v}}), \vec{\underline{u}}) \simeq \underline{\sigma}_{\underline{\mathbf{s}}'}(\vec{\underline{t}'}, \vec{\underline{v}'})$$

where bold face $\underline{\mathbf{s}} = (\vec{\underline{s}} \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}' = (\vec{\underline{s}'} \Rightarrow \underline{s})$, *bold face* $\underline{\mathbf{s}}_i = (\vec{\underline{s}}, \vec{\underline{s}^i} \Rightarrow \underline{s}^i)$, *bold face* $\underline{\mathbf{s}}'_i = (\vec{\underline{s}}, \vec{\underline{s}^{\prime i}} \Rightarrow \underline{s}^{\prime i})$, $\underline{t}'_i = \underline{o}_{\underline{\mathbf{s}}_i,\underline{\mathbf{s}}'_i}(\underline{t}_i, \vec{\underline{u}'}, \underline{Pr}_{|\vec{\underline{s}'}|+1,|\vec{\underline{s}'}|+|\vec{\underline{s}^i}|}^{\vec{\underline{s}'},\vec{\underline{s}^i}})$ *with* $\underline{u}'_j = \underline{o}_{(\vec{\underline{s}'} \Rightarrow \underline{s}_j),(\vec{\underline{s}'},\vec{\underline{s}^i} \Rightarrow \underline{s}_j)}(\underline{u}_j, \underline{Pr}_{1,|\vec{\underline{s}'}|}^{\vec{\underline{s}'},\vec{\underline{s}^i}})$, $\underline{v}'_j = \underline{o}_{(\vec{\underline{s}} \Rightarrow \underline{s}''_j),(\vec{\underline{s}'} \Rightarrow \underline{s}''_j)}(\vec{\underline{v}}, \vec{\underline{u}})$. $|\vec{\underline{s}^i}| = m_i$, $\underline{t}_i \in \underline{T}_{(\vec{\underline{s}},\vec{\underline{s}^i} \Rightarrow \underline{s}^i)}$ $\underline{v}_j \in \underline{T}_{(\vec{\underline{s}} \Rightarrow \underline{s}''_j)}$, *and* $\underline{u}_j \in \underline{T}_{(\vec{\underline{s}'} \Rightarrow \underline{s}_j)}$ *for* $1 \leq i \leq \ell$ *and* $1 \leq j \leq n$.

Then the relationship between many-sorted b-clones and many-sorted iBAs will be the same between single-sorted b-clones and single-sorted iBAs. The difference is

that instead of sorted by natural number k, they are sorted by some proper $(\vec{s} \Rightarrow s)$ where the length of list $\vec{s}$ is k. So, the results in Chapter 5 and Chapter 6 can be re-done here with some more notational modification. This modification is left out for the interested people. However, later chapters of applications will use the results of this modification.

7.2 Many-sorted Birkhoff's approach

The similar analogy to the last section will be applied to the extensional approach, and this is left out for the interested people.

7.3 Higher-sorted FBO

Referring to Subsection 1.4.1, we understand that there is a distinction between higher orders and higher sorts (or higher types). In this section, we briefly discuss two cases of higher sorts in FBO, i.e.

(7.3.a) *simple typed* FBO and

(7.3.b) *polymorphic typed* FBO.

Let Σ^{BO} be a binding signature for higher-sorted FBO, S be the set of base sorts in $Sort(\Sigma)$, $\otimes$ and $\boxed{\Rightarrow}$ be the constructors for sorts. Then

(7.3.a) Σ^{BO} is *simply-typed* iff $Sort(\Sigma)$ is the least closure of the followings:

(7.3.a.i)

$$\frac{s \in S}{s \in Sort(\Sigma^{BO})},$$

(7.3.a.ii)

$$\frac{s_i \in Sort(\Sigma^{BO})}{s_1 \otimes s_2 \in Sort(\Sigma^{BO})},$$

and

(7.3.a.iii)

$$\frac{s_i \in Sort(\Sigma^{BO})}{s_1 \boxed{\Rightarrow} s_2 \in Sort(\Sigma^{BO})}.$$

For simplicity, we will write $\times$ and $\Rightarrow$ for $\otimes$ and $\boxed{\Rightarrow}$ respectively.

(7.3.b) Σ^{BO} is *polymorphic typed* iff $Sort(\Sigma^{BO})$ is the least closure of the followings:

(7.3.b.i)

$$\frac{s \in S}{s \in Sort(\Sigma^{BO})},$$

(7.3.b.ii)

$$\frac{x \in VS}{x \in Sort(\Sigma^{BO})}$$

where VS is the set of variables for sorts,

(7.3.b.iii)

$$\frac{s_i \in Sort(\Sigma^{BO})}{s_1 \times s_2 \in Sort(\Sigma^{BO})},$$

and

(7.3.b.iv)

$$\frac{s_i \in Sort(\Sigma^{BO})}{s_1 \Rightarrow s_2 \in Sort(\Sigma^{BO})}.$$

The purpose of introducing variable sorts is to consider polymorphism into FBO. So, binding primitives would include binding over sorts as well.

For higher-sorted FBO, we are not clear how to establish connections between "orders" and "sorts", say between arities and $\otimes$, and among $\rightarrow$, $\Rightarrow$ and $\boxed{\Rightarrow}$. With regards to binding over sorts, we are not sure of whether it is fruitful to mix "ordinary" binding with "sort" binding. Anyway, these questions are remained to be answered (see Section 1.4.2 for a possible direction for future research). For ordinary polymorphism, see [107,24,111].

Part V

Applications

In this application part, we demonstrate that the Framework for BOs and its intensional and extensional calculi e.g. $\dot{\vdash}^{u}_{eBA}$ (or $\dot{\vdash}_{eBA}$ even $\dot{\vdash}$) and $\vdash^{u}_{iBA}$ (or $\vdash_{iBA}$ even $\vdash$), can be used

(a) to equationalize First Order Logic,

(b) to characterize λ-algebras and λ-models equationally, and

(c) to capture weak and strong bisimulations of finite CCS with data-dependency. Since from pure syntactic point of view, $\vdash_{iBA}$ is a sub-part of $\dot{\vdash}_{eBA}$ we will present our work through intensional $\vdash_{iBA}$ so that the presented work will also be sound through $\dot{\vdash}_{eBA}$; although by doing so we lose some benefits obtained from $\dot{\vdash}_{eBA}$. For example, for β-conversion

$$(\beta) \quad \mathbf{app}(\lambda x.f(x), y) \simeq f(y)$$

of Lambda Calculus, $\vdash_{iBA}$ has to use (β-schema)

$$(\beta\text{-}schema) \quad \mathbf{app}(\lambda x.t, x) \simeq t$$

instead of (β) as it would be in $\dot{\vdash}_{eBA}$.

Therefore, throughout this part, i.e. from Chapter 8 to Chapter 10, we presume $\vdash$ as $\vdash^{u}_{iBA}$ unless we explicitly say otherwise.

Chapter 8

Equationalization of First Order Logic

As a first example of applications, we choose the most commonly used and well-studied example, i.e. First Order Logic. But our approach is different from the ones in literature, since we have binding primitives available. The idea of using algebraic method to deal with First Order Logic is not new, for examples see [59,66,8,67]. Basically, if you consider logic as meta-laws, then the work of this chapter can be viewed as internalization of logic into algebraic systems. Section 8.1 is going to provide First Order Logic in uBE's form. This explains the practical reason of extending BEs to include dBEs, qBEs and uBEs in this thesis. A Sequent Natural Deduction system of First Order Logic will be given in Section 8.2, which is essentially the system in Hu and Lu's [140]. Section 8.3 is dedicated to complete equationalization of equational presentation of First Order Logic in Section 8.1. Some review with related work is offered in Section 8.4.

8.1 Equational presentation of First Order Logic

Considering signature Σ^{log}, we give its operations in two-sorted case for simplicity, say *bool* and *obj* for boolean and the other sort. They can be easily extended to include other many-sorted cases without higher sorts.

(i) boolean constant $\mathbf{True} \in \Sigma^{log}_{<<\epsilon,s>,bool>}$;

(ii) negation $\neg \in \Sigma^{log}_{<<\epsilon,bool>,bool>}$;

(iii) connectives $\wedge, \vee, \supset \in \Sigma^{log}_{<<\varepsilon,bool^2>,bool>}$;

(iv) m-ary predicates Φ's, for each of these m-ary predicate Φ, there is a obj^m such that $\Phi \in \Sigma^{log}_{<<\varepsilon,obj^m>,bool>}$ where $m \in Nat$;

(v) unary quantifiers $\forall, \exists \in \Sigma^{log}_{<<obj \Rightarrow bool,\varepsilon>,bool>}$.

A formula of Σ^{log} is of the form $t \simeq \mathbf{True}$, or simply t for short. Also, sometimes $\neg t$, $t \supset u$, $t \wedge u$ and $t \vee u$ are adopted for $\neg(t)$, $\supset(t,u)$, $\wedge(t,u)$ and $\vee(t,u)$ respectively as conventions.

Definition 8.1.1 (Ax^{log}): *Let Ax^{log} be a set of following uBEs as follows:*

1. *($\neg$-elim)*

   $\{(\gamma \cup \{\neg(t) \simeq \mathbf{True}\}) \hookrightarrow u \simeq \mathbf{true}, (\gamma \cup \{\neg(t) \simeq \mathbf{True}\}) \hookrightarrow \neg(u) \simeq \mathbf{True}\} \mapsto (\gamma \hookrightarrow t \simeq \mathbf{True})$,

2. *($\supset$-elim)*

   $\{\supset(t,u) \simeq \mathbf{True}, t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}$,

3. *($\supset$-intr)*

   $(\gamma \cup \{t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}) \mapsto (\gamma \hookrightarrow \supset(t,u) \simeq \mathbf{True})$,

4. *(left-$\wedge$-elim)*

   $\wedge(t,u) \simeq \mathbf{True} \hookrightarrow t \simeq \mathbf{True}$,

5. *(right-$\wedge$-elim)*

   $\wedge(t,u) \simeq \mathbf{True} \hookrightarrow u \simeq \mathbf{True}$,

6. *($\wedge$-intr)*

   $\{t \simeq \mathbf{True}, u \simeq \mathbf{True}\} \hookrightarrow \wedge(t,u) \simeq \mathbf{True}$,

7. *($\vee$-elim)*

   $\{t \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True}, u \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True}\} \mapsto (\vee(t,u) \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True})$,

8. *(left-$\vee$-intr)*

$$t \simeq \mathbf{True} \hookrightarrow \vee(t, u) \simeq \mathbf{True},$$

9. *(right-$\vee$-intr)*

$$u \simeq \mathbf{True} \hookrightarrow \vee(t, u) \simeq \mathbf{True},$$

10. *($\forall$-elim)*

$$\forall(\langle x : t \rangle) \simeq \mathbf{True} \hookrightarrow t\,[x := y] \simeq \mathbf{True},$$

11. *($\forall$-intr)*

$$\{\gamma \hookrightarrow t\,[x := y] \simeq \mathbf{True} \mid \textit{for some } y \notin \mathit{Free}(\gamma)\} \mapsto (\gamma \hookrightarrow \forall(\langle x : t \rangle) \simeq \mathbf{True}),$$

12. *($\exists$-elim)*

$$\{t\,[x := y] \simeq \mathbf{True} \hookrightarrow u \simeq \mathbf{True} \mid \textit{for some } y \notin \mathit{Free}(u)\} \mapsto (\exists(\langle x : t \rangle) \simeq \mathbf{True} \hookrightarrow u \simeq \mathbf{True}),$$

where $y \notin \mathit{Free}(u)$;

13. *($\exists$-intr)*

$$t\,[x := y] \simeq \mathbf{True} \hookrightarrow \exists(\langle y : t\,[x := y] \rangle) \simeq \mathbf{True}.$$

We should point out that the side condition of that y is not a free variable of u in the axiomatic schemas of ($\exists$-elim) and ($\forall$-intr) are extremely important. Otherwise, for instance of ($\exists$-elim), we would have

$$(x \leq 3 \hookrightarrow x \leq 4) \mapsto ((\exists x.x \leq 3) \hookrightarrow x \leq 4),$$

where $\exists x.t$ is short for $\exists(< x : t >)$. This is obviously absurd. However, it is a quite common mistake in the reasoning of mathematical logic.

8.2 Sequent natural deduction

We provide a SND system, which is essentially the one in Hu's [140].

Definition 8.2.1 (sequent natural deduction): *Let Γ be a set of terms of First Order Logic, t be a term of First Order Logic (FOL). A derivation relation $\vdash_{snd}$ between Γ and t of First Order Logic is defined as follows:*

1. *(FOL-id)* $\{t\} \vdash_{snd} t$,

2. *(FOL-wkn)*
$$\frac{\Gamma \vdash_{snd} t}{\Gamma, \Gamma' \vdash_{snd} t},$$

3. *(FOL-cut)*
$$\frac{\Gamma \vdash_{snd} t; \Gamma, t \vdash_{snd} u}{\Gamma \vdash_{snd} u},$$

4. *(FOL-$\neg$-elim)*
$$\frac{\Gamma, \neg(t) \vdash_{snd} u; \Gamma \neg(t) \vdash_{snd} \neg(u)}{\Gamma \vdash_{snd} t},$$

5. *(FOL-$\supset$-elim)* $\supset(t, u), t \vdash_{snd} u$,

6. *(FOL-$\supset$-intr)*
$$\frac{\Gamma, t \vdash_{snd} u}{\Gamma \vdash_{snd} \supset(t, u)},$$

7. *(FOL-left-$\wedge$-elim)* $\wedge(t, u) \vdash_{snd} t$,

8. *(FOL-right-$\wedge$-elim)* $\wedge(t, u) \vdash_{snd} u$,

9. *(FOL-$\wedge$-intr)* $t, u \vdash_{snd} \wedge(t, u)$,

10. *(FOL-$\vee$-elim)*
$$\frac{t \vdash_{snd} v; u \vdash_{snd} v}{\vee(t, u) \vdash_{snd} v},$$

11. *(FOL-left-$\vee$-intr)* $t \vdash_{snd} \vee(t, u)$,

12. *(FOL-right-$\vee$-intr)* $u \vdash_{snd} \vee(t,u)$,

13. *(FOL-$\forall$-elim)* $\forall x.t \vdash_{snd} t\,[x := y]$, *where* $\forall x.t$ *stands for* $\forall(\langle x : t\rangle)$;

14. *(FOL-$\forall$-intr)*
$$\frac{\Gamma \vdash_{snd} t\,[x := y]; y \notin Free(\Gamma)}{\Gamma \vdash_{snd} \forall x.t},$$

15. *(FOL-$\exists$-elim)*
$$\frac{t\,[x := y] \vdash_{snd} u; y \notin Free(u)}{\exists x.t \vdash_{snd} u},$$
where $\exists x.t$ *stands for* $\exists(\langle x : t\rangle)$;

16. *(FOL-$\exists$-intr)* $t\,[x := y] \vdash_{snd} (\exists y.t\,[x := y])$.

According to literature, say [140], we have soundness and completeness of $\vdash_{snd}$. We simply state it as follows.

Theorem 8.2.2 (soundness and completeness of SND): *First Order Logic* $\vdash_{snd}$*is sound and complete with respect to standard interpretations.*

The *standard interpretations* mean that there is an iBA $\mathbf{B}'$ such that interpretations of $\wedge$, $\vee$, $\supset$, $\forall$ and $\exists$ are somehow fixed and they correspond to logical "and", "or", "imply", "for all" and "there is" in 2-valued Boolean Algebra of $\mathbf{B}$ respectively). Formally, $B = \{0,1\}$, $\mathbf{True}^{\mathbf{B}} = 1$ and

$\neg^{\mathbf{B}}$	
0	1
1	0

$\wedge^{\mathbf{B}}$	0	1
0	0	0
1	0	1

$\vee^{\mathbf{B}}$	0	1
0	0	1
1	1	1

$\supset^{\mathbf{B}}$	0	1
0	1	1
1	0	1

$$\exists^{\mathbf{B}'}(g) = \begin{cases} 0 & \text{if } g \text{ is } C_0 \\ 1 & \text{otherwise} \end{cases} \qquad \forall^{\mathbf{B}'}(g) = \begin{cases} 1 & \text{if } g \text{ is } C_1 \\ 0 & \text{otherwise} \end{cases}$$

where $g, C_0, C_1 \in B'_{obj} \to B_{bool}$; C_0 and C_1 are the constant functions which map everything to 0 or 1 respectively. From now on, let us fix a $\mathbf{B}'$ for convenience.

Since the proof of Theorem 8.2.2 can be found in many book on First Order Logic, say [139,10,140], we omit it here. However, we should point out that the essential point of the proof is the introduction of "evidences" for the existentially-quantified formulae, so-called Henkin's method. Algebraically, this implicitly implies

a restriction on signatures of algebras for First Order Logic, i.e. the signatures must not have empty sorts. This is a significant point. We will come back to it after the proof of Theorem 8.3.2.

8.3 Complete equationalization

We expect to have that

$$\Gamma \vdash_{snd} t \text{ iff } Ax^{log} \vdash \gamma \hookrightarrow t \simeq \mathbf{True},$$

where $\gamma = \{u \simeq \mathbf{True} | u \in \Gamma\}$. We can called the "if" direction as *soundness* of equationalization of First Order Logic, and the "only if" direction as *completeness* of the equationalization. The following theorem can be viewed as either that axioms Ax^{log} with $\vdash$ (Definition 8.1.1) is complete with respect to $\vdash_{snd}$ (Definition 8.2.1) or that $\vdash_{snd}$ is sound with respect to Ax^{log} with $\vdash$.

Theorem 8.3.1 (complete derivability of $\vdash$ with Ax^{log}): *If $\Gamma \vdash_{snd} t$ then $Ax^{log} \vdash \gamma \hookrightarrow t \simeq \mathbf{True}$ (or $Ax^{log} \vdash^{u}_{iBA} (\gamma \hookrightarrow t \simeq \mathbf{True})$, if you prefer full notation in $\vdash^{u}_{iBA}$).*

Proof

(a) case FOL-id: That is, $t \vdash_{snd} t$ implies $(t \simeq \mathbf{True} \hookrightarrow t \simeq \mathbf{True})$. This is trivial.

(b) case FOL-wkn: That is,

$$\frac{\Gamma \vdash_{snd} t}{\Gamma, \Gamma' \vdash_{snd} t}$$

implies $(\gamma \hookrightarrow t \simeq \mathbf{True}) \mapsto (\gamma \cup \gamma' \hookrightarrow t \simeq \mathbf{True})$. This is trivial, too.

(c) case FOL-cut:

$$\frac{\Gamma \vdash_{snd} t; \Gamma, t \vdash_{snd} u}{\Gamma \vdash_{snd} u}$$

i.e. $\{(\gamma \hookrightarrow t \simeq \mathbf{True}), (\gamma \cup \{t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True})\} \mapsto (\gamma \hookrightarrow u \simeq \mathbf{True})$. By Modus Ponens, we have it hold.

(d) case FOL-¬-elim:

$$\frac{\Gamma, \neg(t) \vdash_{snd} u; \Gamma, \neg(t) \vdash_{snd} \neg(u)}{\Gamma \vdash_{snd} t}$$

i.e. $\{(\gamma \cup \{\neg(t) \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True}), (\gamma \cup \{\neg(t) \simeq \mathbf{True}\} \hookrightarrow \neg(u) \simeq \mathbf{True})\} \mapsto (\gamma \hookrightarrow t \simeq \mathbf{True})$. This is (¬-elim).

(e) case FOL-⊃-elim: $\supset(t,u), t \vdash u$, i.e. $(\{\supset(t,u) \simeq \mathbf{True}, t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True})$. This is (⊃-elim).

(f) case FOL-⊃-intr:

$$\frac{\Gamma, t \vdash_{snd} u}{\Gamma \vdash_{snd} \supset(t,u)}$$

i.e. $\{(\gamma \cup \{t \simeq \mathbf{True}\} \hookrightarrow u \simeq \mathbf{True})\} \mapsto (\gamma \hookrightarrow \supset(t,u) \simeq \mathbf{True})$. This is (⊃-intr).

(g) case FOL-left-∧-elim: $\wedge(t,u) \vdash_{snd} t$, i.e. $(\wedge(t,u) \simeq \mathbf{True} \hookrightarrow t \simeq \mathbf{True})$. This is (left-∧-elim).

(h) case FOL-right-∧-elim: That is, $\wedge(t,u) \vdash_{snd} u$ this is (right-∧-elim).

(i) case FOL-∧-intr: $t, u \vdash_{snd} \wedge(t,u)$, i.e. $(\{t \simeq \mathbf{True}, u \simeq \mathbf{True}\} \hookrightarrow \wedge(t,u) \simeq \mathbf{True})$. This is (∧-intr).

(j) case FOL-∨-elim:

$$\frac{t \vdash_{snd} v; u \vdash_{snd} v}{\vee(t,u) \vdash_{snd} v},$$

i.e. $\{t \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True}; u \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True}\} \mapsto (\vee(t,u) \simeq \mathbf{True} \hookrightarrow v \simeq \mathbf{True})$. This is (∨-elim).

(k) case FOL-left-∨-intr: $t \vdash_{snd} \vee(t,u)$, i.e. $(t \simeq \mathbf{True} \hookrightarrow \vee(t,u) \simeq \mathbf{True})$. This is (left-∨-intr).

(l) case FOL-right-∨-intr: That is $u, \vdash_{snd} \vee(t,u)$. We have $(u \simeq \mathbf{True} \hookrightarrow \vee(t,u) \simeq \mathbf{True})$. This is (right-∨-intr).

(m) case FOL-∀-elim: $\forall x.t \vdash_{snd} t[x := y]$, i.e. $\forall x.t \simeq \mathbf{True} \hookrightarrow t[x := y] \simeq \mathbf{True})$. This is (∀-elim).

(n) case FOL-∀-intr:

$$\frac{\Gamma \vdash_{snd} t[x := y]}{\Gamma \vdash_{snd} \forall x.t} \; (y \notin Free(\Gamma))$$

i.e. $\{\gamma \hookrightarrow t[x := y] \simeq \mathbf{True} \mid y \notin Free(\gamma)\} \mapsto (\gamma \hookrightarrow \forall x.t \simeq \mathbf{True})$. This is ($\forall$-intr).

(o) case FOL-$\exists$-elim:

$$\frac{t[x := y] \vdash_{snd} u}{\exists x.t \vdash_{snd} u} \ (y \notin Free(u))$$

i.e. $\{t[x := y] \simeq \mathbf{True} \hookrightarrow u \simeq \mathbf{True} \mid y \notin Free(u)\} \mapsto (\exists x.t \simeq \mathbf{True} \hookrightarrow u \simeq \mathbf{True})$. This is ($\exists$-elim).

(p) case FOL-$\exists$-intr: $t[x := y] \vdash_{snd} \exists y.(t[x := y])$, i.e. $(t[x := y] \simeq \mathbf{True} \hookrightarrow \exists y.t[x := y] \simeq \mathbf{True})$. This is ($\exists$-intr).

(q) Overall, we can prove this theorem (Theorem 8.3.1) by induction on the length of a proof $\Gamma \vdash_{snd} t$ in First Order Logic. □

What Theorem 8.3.1 tells us is that if $\Gamma \vdash_{snd} t$ is derivable in First Order Logic, then $Ax^{log} \vdash (\gamma \hookrightarrow t \simeq \mathbf{True})$ is derivable. Also, from its proof we notice the following correspondences:

(8.3.a) $\mapsto$ (dependent implication) is corresponding to $\frac{\cdot}{\cdot}$ (inference rule in logic);

(8.3.b) $\hookrightarrow$ (quasi-dependent implication) is corresponding to $\vdash$ (derivability in logic);

Therefore, by (8.3.b) and by (FOL-$\supset$-elim) and (FOL-$\supset$-intr), $\supset$ can be regarded as internalization of $\hookrightarrow$ into logic.

From Theorem 8.2.2 and Theorem 8.3.1, we can infer that $Ax^{log} \vdash (\gamma \hookrightarrow t \simeq \mathbf{True})$ is sound and complete with respect to the standard interpretations as they are defined in Section 8.2. Namely,

Theorem 8.3.2 (soundness of derivability in $\vdash$ with Ax^{log}): *If* $Ax^{log} \vdash (\gamma \hookrightarrow t \simeq \mathbf{True})$, *then* $\Gamma \vdash_{snd} t$.

Proof

This is simply because the followings:

(i) $Ax^{log} \vdash (\gamma \hookrightarrow t \simeq \mathbf{True})$ implies $Ax^{log} \models (\gamma \hookrightarrow t \simeq \mathbf{True})$ (obviously);

(ii) $Ax^{log} \models_{\Sigma BO} (\gamma \hookrightarrow t \simeq \mathbf{True})$ implies $\mathbf{B}' \models (\gamma \hookrightarrow t \simeq \mathbf{True})$, since $\mathbf{B}' \models Ax^{log}$ and (i)

(iii) $\mathbf{B}' \models (\gamma \hookrightarrow t \simeq \mathbf{True})$ is actually $\Gamma \models t$ provided $t \neq \mathbf{True}$.

(iv) $\Gamma \models t$ iff $\Gamma \vdash_{snd} t$ (the usual completeness of First Order Logic),

where $\Gamma \models t$ is defined as: for all environment ψ, $\mathcal{B}'_{\vec{x}}[\![u]\!]_\psi = 1$ (for each $u \in \Gamma$) implies $\mathcal{B}'_{\vec{x}}[\![t]\!]_\psi = 1$. □

As we remark earlier, we should notice that each carrier of a sort in $\mathbf{B}'$ is required to be non-empty in the completeness proof of First Order Logic, say by Henkin's method, since every proper instance of existential quantifier has to have an evidence in $\mathbf{B}'$. This observation implies that the equational system of First Order Logic (in Definition 8.1.1) is *conservative* over First Order Logic in an algebraic sense. That is, the system $\vdash$ with Ax^{log} has no restriction about empty-sorts but $\vdash_{snd}$ does.

Also, we should point out that the equality **eq** in First Order Logic is different from the Indistinguishability (or meta equality) $\simeq$, although $\simeq$ is commonly referred to as "equality" as well. **eq** can be regarded as a special predicate and it is defined inside $\mathbf{B}'$, e.g. $\mathbf{B}' \models \mathbf{eq}(x, y) \simeq \mathbf{True}$ (i.e. $\mathbf{eq}(x, y)$ is true in $\mathbf{B}'$) iff for every interpretation $[\![\bullet]\!]$ in $\mathbf{B}'$, $\mathbf{eq}^{\mathbf{B}'}([\![x]\!], [\![y]\!])$ is $\mathbf{True}^{\mathbf{B}'}$. Of course, **eq** can be viewed in certain sense as an internalization of $\simeq$ into logic.

As a result of Theorem 8.3.1 and Theorem 8.3.2, we have the expected equivalence. That is,

Corollary 8.3.3 (equivalence): *Let* $\gamma = \{u \simeq \mathbf{True} | u \in \Gamma\}$ *and* $\gamma^{(m)} = \{u \simeq \mathbf{True} | u \in \Gamma_m\}$ *for each* $m \in M$. *Thus,*

1. $\Gamma \vdash_{snd} t$ *iff* $Ax^{log} \vdash \gamma \hookrightarrow t \simeq \mathbf{True}$;

2. *further, we have that*

$$\frac{\{\Gamma_m \vdash t_m | m \in M\}}{\Gamma \vdash t}$$

iff $Ax^{log} \vdash \{\gamma^{(m)} \hookrightarrow t_m \simeq \mathbf{True} | m \in M\} \mapsto (\gamma \hookrightarrow t \simeq \mathbf{True})$.

This equivalence also enables us to reduce First Order Logic entirely into a certain deduction system of Binding Algebras (eBAs or iBAs) which satisfy Ax^{log}. However, the same equivalence can be viewed as reducing the deduction system of

many Binding Algebras satisfying Ax^{log} into the deduction system of one certain Binding Algebra $\mathbf{B}'$, i.e. we do not need to consider all possible Binding Algebras but a special one. This special Binding Algebra, say $\mathbf{B}'$ which has Boolean Algebra $\mathbf{B}$ as a sub-algebra, is unlike term Binding Algebra $\mathbf{T}$. It has more meaning to human beings. Nevertheless, this reduction does require certain pay-off, i.e. some extra operations and Binding Operators have to be introduced. They are logical connectives, say negation, universal quantifier and existential quantifier accordingly. But the pay-off has a significant advantage of being intuitively understood by human being.

8.4 Discussion

The essential departing point of the work presented in this chapter from the other's work in literature is that we are trying to reduce First Order Logic to certain algebras rather than to take algebras out of First Order Logic, which is a quite common practice.

Referring to Corollary 8.3.3, we can understand the equivalence as a two-way bridge between model-theoretic approach to logic and proof-theoretic one to logic. So far, we are mainly in model-theoretic approach to logics. There is another way to equationalize First Order Logic, i.e. taking quantification as Adjunction, say

$$\frac{t \preceq y}{\exists x.t \preceq y}$$

where $t \preceq u$ means $Ax^{bool} \cup \{t \preceq \exists x.t,\ \exists x.(t \wedge y) \simeq \exists x.t \wedge y\} \vdash t \simeq t \wedge u$ such that Ax^{bool} is the usual equational axioms for Boolean Algebra $\mathbf{B}$ [60]. There are other references in this approach, see [25,99,12] and especially the bibliography in the end of [12]. Therefore, it is quite reasonable for us to turn our attention to proof-theoretic approach to logics.

From computer science point of view, proof-theoretic approach can be better put in a ring, which is actually a folklore among this society. That is,

$$\begin{array}{ccc} \textit{Propositions} & \Longrightarrow & \textit{Proofs} \\ \Uparrow & & \Downarrow \\ \textit{Types} & \Longleftarrow & \textit{Programs} \end{array}$$

The arrows in the above ring means that one thing leads to another. This ring can be better explained by the following. Informally,

(8.4.a) every formula is a proposition iff there is a proof showing that it is, and every proposition is true iff there is a proof showing that it is as well;

(8.4.b) every proof corresponds to a program;

(8.4.c) every program corresponds to a type; and

(8.4.d) for every type, there is a formula corresponding to it.

For (8.4.a), it corresponds to Intuitionistic Logic which is the same as $\vdash_{snd}$ in Section 8.2 except that (FOL-$\neg$-elim) rule is replaced by (FOL-$\neg$-intr) where (FOL-$\neg$-intr) is

$$\frac{\Gamma, t \vdash u, \neg(u)}{\Gamma \vdash \neg(t)}$$

(see [135,161] for more references to Intuitionistic Logic); considering (8.4.b), it corresponds to logic programming [96]; regarding to (8.4.c), it is a very common practice in programming, say ML typing and others [107,111]; to (8.4.d), there is a formula-as-type correspondence [77,129].

Martin-Löf has exploited the above circle phenomenon to certain extent in his type theory [101]. In his (intuitionistic) type theory, every type is a proposition and every proposition is the collection of all its possible proofs. Apparently, this kind of formalization is in favour of computer science.

The proof-theoretic approach has certain advantages over model-theoretic one. One of them is that we can code proofs of proofs into proofs, and at the meantime, we can extract programs from proofs. This can be exploited and recently demonstrated by Hayashi and by Sato in [64,132]. Hayashi and Sato used LISP language, built up

proof's hierarchy and code proofs of proofs into proofs. Therefore, all meta proofs up to ω level can be realized as an ordinary proof.

Finally, we should point out that the above discussion provides the theoretical reason of considering First Order Logic in this thesis. Another reason for it will be to provide a technical preparation for Chapter 10, where boolean expressions are under consideration.

Chapter 9

Equational Theories of λ-algebras and of λ-models

In this chapter, we relate Lambda Calculus with Combinatory Logic (CL) [32,33] in the Framework for BOs. Recall that we have shown the reason of choosing Lambda Calculus approach in Subsection 1.3.2 instead of Combinatory Logic approach. In this chapter, we would like to give a connection between these two approaches. At the meantime, we relate λ-algebras with λ-models. Standard work on this field can be found in Barendregt's [7, Section 5.2]. We are going to demonstrate this standard work in the present framework. Since we allow function variables appearing in λ-terms, the previous work can not be straightly copied (see Section 9.5 for more discussion on this). Modification is needed. This modification help us in understanding of functionality.

Section 9.1 is to present equational forms of Lambda Calculus and Combinatory Logic. Syntactic connections between Lambda Calculus and Combinatory Logic are established by two translations between the terms of these two calculi, and the well-definedness of these translations is verified in Section 9.2. Section 9.3 is to cover the equational theory of λ-algebras. The equational theory of λ-models will be the subject of Section 9.4. Section 9.5 is to give a review with some related work.

9.1 Equational theories of Lambda Calculus and of Combinatory Logic

The signature Σ^λ for Lambda Calculus consists of **app** and λ where $\mathbf{app} \in \Sigma^\lambda_{<\varepsilon,2>}$ and $\lambda \in \Sigma^\lambda_{<<1>,0>}$.

Let V be a set of ordinary variables, FV_1 be a set of function variables with arity 1, T^λ be a set of λ terms and FT^λ_1 be a set of λ function term with arity 1. For convenience, we sometimes denote $\lambda(\langle x : t\rangle)$ and $\mathbf{app}(t,u)$ as $\lambda x.t$ and $t.u$ (or even $t\,u$) respectively as convetions.

Definition 9.1.1 (equational theory of Lambda Calculus and (β)): *Let Λ be the equational theory of Lambda Calculus. Then, $p \simeq q \in \Lambda$ iff (β-schema) $\vdash_\lambda p \simeq q$ where (β-schema) is* $\mathbf{app}(\lambda x.t, x) \simeq t$.

Note that β-axiom is $\mathbf{app}(\lambda x.f(x), y) \simeq f(y)$. Since function variables are only substituted by semi-closed function terms, Lambda Calculus obtained by β-axiom is *linear* Lambda Calculus, where the "linear" simply means that there is at most one free ordinary variable occurring in (linear) λ-terms. However, whether these two Lambda Calculi can be directly shown equivalent is raised in Chapter 1 (see Subsection 1.4.3). In the present chapter, our main interest is not linear Lambda Calculus. So, later we will simply use (β) to refer *β-schema*.

The signature Σ^c of Combinatory Logic (CL) has (1) constants: $\mathbf{K}, \mathbf{S} \in \Sigma^c_{<\varepsilon,0>}$ and (2) operation: $\mathbf{ap} \in \Sigma^c_{<\varepsilon,2>}$

Let $\vec{\chi}$ be the set of ordinary variables for Combinatory Logic, $\tilde{x}, \tilde{y}, \tilde{z}$ range over $\vec{\chi}$ and T^c for terms of Combinatory Logic. Combinatory Logic can be presented equationally. That is,

Definition 9.1.2 (equational theory of CL, K-axiom and S-axiom): *Let CL be the equational theory of Combinatory Logic. Then, $p \simeq q \in CL$ iff $(\mathbf{K})$ $(\mathbf{S}) \vdash_c p \simeq q$ where* **K**-*axiom, or simply* $(\mathbf{K})$, *and* **S**-*axiom, or simply* $(\mathbf{S})$, *are as follows:*

1. (**K**-*axiom*) $\mathbf{ap}(\mathbf{ap}(\mathbf{K}(), \tilde{x}), \tilde{y}) \simeq \tilde{x}$

2. (**S**-*axiom*) $\mathbf{ap}(\mathbf{ap}(\mathbf{ap}(\mathbf{S}(),\tilde{x}),\tilde{y}),\tilde{z}) \simeq \mathbf{ap}(\mathbf{ap}(\tilde{x},\tilde{z}),\mathbf{ap}(\tilde{y},\tilde{z}))$

Unlike in Lambda Calculus, we do not need to have function variables in CL, which is due to the philosophical argument of Curry, see [32]. Since there is no function variables, (ord-sub) rule comes into the place of (func-sub). The reason for this is that we have no functional substitution rule available to play the role for the ordinary substitution rule in CL. For simplicity, we will use **K** and **S** to refer **K**() and **S**(). Also, it is well-known that there is a close relationship between Lambda Calculus and Combinatory Logic. In order to present this relationship here, let $\vec{\chi} = \underline{V} \cup \underline{FV}_1$ and $\underline{x}$, $\underline{f}$ range over $\underline{V}$ and $\underline{FV}_1$ respectively, where $_$ is a bijection between V (or FV_1) and $\underline{V}$ (or $\underline{FV}_1$). Since $\underline{x}$ and $\underline{f}$ are different by nature, we have to pay special attention to them.

Firstly, we are going to give two translations λ-C and C-λ between T^λ (or FT_1^λ) and T^c (or FT_1^c).

Definition 9.1.3 (translations of C-λ and λ-C):

(i) Translation C-λ from T^c (or FT_1^λ) to T^λ (or FT_λ^c) is defined as

1. $C\text{-}\lambda[\![\underline{x}]\!] =_{df} x$
2. $C\text{-}\lambda[\![\underline{f}]\!] =_{df} \lambda x.f(x)$
3. $C\text{-}\lambda[\![\mathbf{ap}(\underline{t},\underline{u})]\!] =_{df} \mathbf{app}(C\text{-}\lambda[\![\underline{t}]\!], C\text{-}\lambda[\![\underline{u}]\!])$
4. $C\text{-}\lambda[\![\mathbf{K}]\!] =_{df} \lambda x.\lambda y.x$,
   sometimes it will be denoted as $\mathbf{K}_\lambda$ *for short.*
5. $C\text{-}\lambda[\![\mathbf{S}]\!] =_{df} \lambda x.\lambda y.\lambda z.\mathbf{app}(\mathbf{app}(x,z),\mathbf{app}(y,z))$,
   sometimes it will be denoted as $\mathbf{S}_\lambda$ *for short.*
6. $C\text{-}\lambda[\![\langle \underline{x} : \underline{t}\rangle]\!] =_{df} \langle x : C\text{-}\lambda[\![\underline{t}]\!]\rangle$.
   Note that C-λ is only defined over $\langle \tilde{x} : \underline{t}\rangle$ such that $\tilde{x} \in \underline{V} \subseteq \vec{\chi}$.

(ii) Translation λ-C from T^λ (or FT_1^λ) to T^c (or FT_1^c) is defined as following

1. *(a)* $\lambda\text{-}C[\![x]\!] =_{df} \underline{x}$,
   (b) $\lambda\text{-}C[\![f(t)]\!] =_{df} \mathbf{ap}(\underline{f}, \lambda\text{-}C[\![t]\!])$,

(c) $\lambda\text{-}C[\![\mathbf{ap}(t,u)]\!] =_{df} \mathbf{ap}(\lambda\text{-}C[\![t]\!], \lambda\text{-}C[\![u]\!])$,

(d) $\lambda\text{-}C[\![\langle x:t\rangle]\!] =_{df} \langle \underline{x} : \lambda\text{-}C[\![t]\!]\rangle$,

(e) $\lambda\text{-}C[\![\lambda(\langle x:t\rangle)]\!] =_{df} \mathbf{ab}(\lambda\text{-}C[\![\langle x:t\rangle]\!])$.

2. *where* **ab** *is defined as follows*

(a) $\mathbf{ab}(\langle \underline{x} : \underline{x}\rangle) =_{df} \mathbf{ap}(\mathbf{ap}(\mathbf{S},\mathbf{K}),\mathbf{K})$, *it will be denoted as* $\mathbf{I}$ *for short.*

(b) $\mathbf{ab}(\langle \underline{x} : \underline{y}\rangle) =_{df} \mathbf{ap}(\mathbf{K},\underline{y})$, *where* $\underline{x} \neq \underline{y}$.

(c) $\mathbf{ab}(\langle \underline{x} : \underline{f}\rangle) =_{df} \mathbf{ap}(\mathbf{K},\underline{f})$,

(d) $\mathbf{ab}(\langle \underline{x} : \mathbf{K}\rangle) =_{df} \mathbf{ap}(\mathbf{K},\mathbf{K})$,

(e) $\mathbf{ab}(\langle \underline{x} : \mathbf{S}\rangle) =_{df} \mathbf{ap}(\mathbf{K},\mathbf{S})$,

(f) $\mathbf{ab}(\langle \underline{x} : \mathbf{ap}(\underline{t},\underline{u})\rangle) =_{df} \mathbf{ap}(\mathbf{ap}(\mathbf{S},\mathbf{ab}(\langle \underline{x} : \underline{t}\rangle)),\mathbf{ab}(\langle \underline{x} : \underline{u}\rangle))$.

Note that **ab** *is only defined over* $\langle \tilde{x} : \underline{t}\rangle$ *such that* $\tilde{x} \in \underline{V}$.

Intuitively, **ab** is the counterpart of λ-abstraction in Combinatory Logic, where **ab** is often referred as λ^* in literature (see [7] for instance). Also, there is another definition for **ab** which has a slight difference on $\mathbf{ab}(\langle \underline{x} : \underline{t}\rangle)$ for $x \notin \mathit{Free}(\underline{t})$, i.e. $\mathbf{ab}(\langle \underline{x} : \underline{t}\rangle) =_{df} \mathbf{ap}(\mathbf{K},\underline{t})$. These two definitions are equivalent under $\simeq$ if we accept that for all $\underline{t},\underline{u} \in T^c$, $\mathbf{ap}(\mathbf{K},\mathbf{ap}(\underline{t},\underline{u})) \simeq \mathbf{ap}(\mathbf{ap}(\mathbf{S},\mathbf{ap}(\mathbf{K},\underline{t})),\mathbf{ap}(\mathbf{K},\underline{u}))$; which is the **KS**-schema to be introduced in Lemma 9.3.5. We will turn to this point later (see Section 9.3).

Before proceeding further, we have to make sure the definitions of both translations λ-C and C-λ are well-defined. Actually, the well-definedness means inter-well-definedness between the two translations.

9.2 Well-definedness of C-λ and of λ-C

With regard to well-definedness of the translations in Definition 9.1.3, the easy one to check seems to be C-λ, since combinators **K** and **S** can be regarded as $\lambda x.\lambda y.x$ and $\lambda x.\lambda y.\lambda z.xz(yz)$ respectively. This observation leads to the following expectation, i.e. if $(\mathbf{K})\ (\mathbf{S}) \vdash_c p \simeq q$ then $(\beta) \vdash_\lambda C\text{-}\lambda[\![p]\!] \simeq C\text{-}\lambda[\![q]\!]$. However, we do not have this

expected result in general. A weaker result is obtained, and it is very reasonable as you will see. So, we state the weaker result as a theorem below. Its proof will be postponed until some technical preparations are ready (after Lemma 9.2.4).

Theorem 9.2.1 (well-definedness of C-λ): *Let CL^- be (almost) CL except that the following two conditions.*

1. *for (α) and for (ξ), $\tilde{x}$ and $\tilde{y}$ in $\langle \tilde{x} : \underline{t} \rangle \in FT_1^c$ and $\langle \tilde{y} : t[\tilde{x} := \tilde{y}] \rangle \in FT_1^c$, $\langle \tilde{x} : \underline{u} \rangle \in FT_1^c$, must be in $\underline{V}$;*

2. *for (b-sub) rule, $\varrho(\underline{f})$ must be in $Ab(T^c)$ for $\underline{f} \in \underline{FV}_1$,*

   where $Ab(T^c) = \{\mathbf{ab}(\langle \underline{x} : \underline{t} \rangle) | \underline{x} \in \underline{V} \wedge \underline{t} \in T^c\}$.

Then, we have that $\underline{p} \simeq \underline{q} \in CL^-$ (or $(\mathbf{K})$ $(\mathbf{S}) \vdash_{c^-} \underline{p} \simeq \underline{q}$) implies $(\beta) \vdash_\lambda C\text{-}\lambda[\![\underline{p}]\!] \simeq C\text{-}\lambda[\![\underline{q}]\!]$.

Note that CL^- is weaker than CL and the restriction of variables in Theorem 9.2.1 is reasonable, since $\underline{f} \in \underline{FV}_1$ and $\underline{x} \in \underline{V}$ have different nature in functionality. More discussion on the difference is in Section 9.5. We will proceed the proof of Theorem 9.2.1 gradually through Lemma 9.2.2, Lemma 9.2.3 and Lemma 9.2.4.

Lemma 9.2.2 (λ-abstraction and ab): *For $t \in T^c$ and $\underline{x} \in \underline{V}$, we have that*

$$(\beta) \vdash_\lambda C\text{-}\lambda[\![\mathbf{ab}(\langle \underline{x} : \underline{t} \rangle)]\!] \simeq \lambda(\langle x : C\text{-}\lambda[\![\underline{t}]\!] \rangle).$$

Proof By structural induction on $\underline{t}$. □

The above lemma confirms our belief that **ab** is the counterpart of λ-abstraction in Combinatory Logic.

Lemma 9.2.3 (substitution exchange): *For $p \in T^c$ or $p \in FT_1^c$, we have that*

$$(\beta) \vdash_\lambda C\text{-}\lambda[\![\underline{p}[\vec{f}, \vec{x} := \underline{\vec{t}}, \underline{\vec{u}}]]\!] \simeq C\text{-}\lambda[\![\underline{p}]\!][\vec{f}, \vec{x} := C\text{-}\lambda[\![\underline{\vec{t}}]\!], C\text{-}\lambda[\![\underline{\vec{u}}]\!]],$$

where $t_i = \mathbf{ab}(\langle \underline{y}_i : \underline{t'}_i \rangle)$, and $\langle \underline{y}_i : \underline{t'}_i \rangle$ must not have free variables in $\underline{V} \subseteq \vec{\chi}$, i.e. $Free(\langle \underline{y}_i : \underline{t'}_i \rangle) \cap \underline{V} = \emptyset$.

Proof

By structural induction and Lemma 9.2.3 and be aware of that $Free(\mathbf{ab}(\langle \underline{x} : \underline{t} \rangle)) = Free(\underline{t}) - \{\underline{x}\}$. □

By Lemma 9.2.4, we understand that substitutions can be swapped with the translation C-λ.

Lemma 9.2.4 (substitution): *For $p, q \in T^c$ or $p, q \in FT_1^c$, we have that if $(\beta) \vdash_\lambda C\text{-}\lambda[\![p]\!] \simeq C\text{-}\lambda[\![q]\!]$, then $\vdash_\lambda C\text{-}\lambda[\![p[\vec{f}, \vec{x} := \vec{t}, \vec{u}]]\!] \simeq C\text{-}\lambda[\![q[\vec{f}, \vec{x} := \vec{t}, \vec{u}]]\!]$;*

where $t_i = \mathbf{ab}(\langle \underline{y}_i : \underline{t}'_i \rangle)$, and $\langle \underline{y}_i : \underline{t}'_i \rangle$ must not have free variables in $\underline{V} \subseteq \vec{\chi}$, i.e. $Free(\langle \underline{y}_i : \underline{t}'_i \rangle) \cap \underline{V} = \emptyset$.

Proof by Lemma 9.2.2 and Lemma 9.2.3. □

Lemma 9.2.4 says that substitutions can not destroy equality of Lambda Calculus. Therefore, we can safely get a proof for Theorem 9.2.1, i.e. the well-definedness of translation C-λ. Formally,

Proof (of **Theorem 9.2.1**)

By induction on the length of proof $\vdash_c p \simeq q$ and Lemma 9.2.2, Lemma 9.2.3 and Lemma 9.2.4. □

Now, we have established the well-definedness of C-λ. We turn our attention to the other translation λ-C. Apparently, the two well-definedness are inter-linked with each other. We can exploit this observation and acquire the well-definedness of λ-C through the well-definedness of C-λ. Formally,

Theorem 9.2.5 (well-definedness of λ-C): *For $p \in T^\lambda$ or $p \in FT_1^\lambda$, we have $(\beta) \vdash_\lambda C\text{-}\lambda[\![\lambda\text{-}C[\![p]\!]]\!] \simeq p$.*

Proof by structural induction with Lemma 9.2.2. □

Therefore, intuitively Theorem 9.2.1 and Theorem 9.2.5 demonstrate that translations C-λ and λ-C are (somehow) faithful to each other. In next section, we further pursue the faithfulness to a certain extent and establish the equational theory of λ-algebras, which are *Combinatory Algebras* satisfying all equations that the counterpart of them are derivable in Lambda Calculus.

9.3 Equational theory for λ-algebras

Before introducing λ-algebras, we first look at the following result. That is,

Theorem 9.3.1 (from Combinatory Logic to Lambda Calculus): *For* $p, q \in T^{\lambda}$ *or* $p, q \in FT_1^{\lambda}$, *we have that* (**K**) (**S**) $\vdash_{c-} \lambda\text{-}C[\![p]\!] \simeq \lambda\text{-}C[\![q]\!]$ *implies* $(\beta) \vdash_{\lambda} p \simeq q$.

Proof

By Theorem 9.2.5, we have $\vdash_{\lambda} C\text{-}\lambda[\![\lambda\text{-}C[\![p]\!]]\!] \simeq p$ and $\vdash_{\lambda} C\text{-}\lambda[\![\lambda\text{-}C[\![q]\!]]\!] \simeq q$.

By Lemma 9.2.1, we get that (**K**) (**S**) $\vdash_{c-} \lambda\text{-}C[\![p]\!] \simeq \lambda\text{-}C[\![q]\!]$ implies $(\beta) \vdash_{\lambda} C\text{-}\lambda[\![\lambda\text{-}C[\![p]\!]]\!] \simeq C\text{-}\lambda[\![\lambda\text{-}C[\![q]\!]]\!]$.

By transitivity rule, we come to the required result. □

If we accept that $\lambda\text{-}C$ is a faithful translation, we can infer that Lambda Calculus has no weaker proof power than Combinatory Logic from Theorem 9.3.1. But what about the other way around? That is, whether Combinatory Logic and Lambda Calculus have a same proof power. This question leads to the introduction of λ-algebras (also see [7, Definition 5.2.2]).

Definition 9.3.2 (λ-algebras): *An iBA* **B** *of Combinatory Logic is a λ-algebra iff for all* $\underline{p}, \underline{q} \in T^c$ *or* $\underline{p}, \underline{q} \in FT_1^c$, *we have that* $(\beta) \vdash_{\lambda} C\text{-}\lambda[\![\underline{p}]\!] \simeq C\text{-}\lambda[\![\underline{q}]\!]$ *implies* $\mathbf{B} \models \underline{p} \simeq \underline{q}$,

where an iBA **B** *of Combinatory Logic means that it satisfies* **K***-axiom and* **S***-axiom in Definition 9.1.2.*

Therefore, although we are actually looking for an equational theory, written as CL_{λ_a}, for λ-algebras, we can name it as $CL(A_a)$. More formally,

Definition 9.3.3 (equational theory $CL(A_a)$ of λ-algebras): *An equational theory* $CL(A_a)$ *is said to be the theory for λ-algebras provided that* $A_a \vdash_{c-} \underline{p} \simeq \underline{q}$ *(or* $\underline{p} \simeq \underline{q} \in CL(A_a)$*) iff for all λ-algebra* **B**, $\mathbf{B} \models \underline{p} \simeq \underline{q}$.

We know that for an iBA **B** of Combinatory Logic, if $\mathbf{B} \models A_a$ then $\mathbf{B} \models p \simeq q$; where $A_a \vdash_c p \simeq q$. This implication will remain its truth if we replace $\vdash_c$ by

$\vdash_{c^-}$ in the side condition. Consquently, if $\mathbf{B} \models A_a$ is *equivalent* to that $\mathbf{B}$ is a λ-algebra, then the remaining job is to find a proper A_a such that the pre-condition (equivalence) holds.

Therefore, We need to find a A_a such that $\mathbf{B} \models A_a$ implies that $\mathbf{B}$ is a λ-algebra. And then we show that the reversed implication holds as well. Another way to understand the equivalence is to see whether λ-algebras are equationally definable. The obvious way to start the finding is to consider whether we can swap the positions of C-λ and λ-C in Theorem 9.2.5, and the same result holds. The same result does hold provided that there are some extra axioms. Formally,

Lemma 9.3.4 (inter-relation between λ-C and C-λ): *For $\underline{p} \in T^c$ or $\underline{p} \in FT_1^c$, we have* $(\mathbf{K})\,(\mathbf{S})\,(Ab_f)\,(Ab'_{\mathbf{K}})\,(Ab'_{\mathbf{S}}) \vdash_{c^-} \lambda\text{-}C[\![C\text{-}\lambda[\![\underline{p}]\!]]\!] \simeq \underline{p}$,

where

(a) (Ab_f) $\mathbf{ab}(\langle \underline{x} : \mathbf{ap}(\underline{f}, \underline{x})\rangle) \simeq \underline{f}$ *(or* $\mathbf{S}(\mathbf{K}\underline{f})\mathbf{I} \simeq \underline{f}$*),*

supposing that $t\,u$ stands for $\mathbf{ap}(t, u)$*;*

(b) $(Ab'_{\mathbf{K}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \underline{x}\rangle)\rangle) \simeq \mathbf{K}$ *(or* $\mathbf{S}(\mathbf{KK})\mathbf{I} \simeq \mathbf{K}$*);*

(c) $(Ab'_{\mathbf{S}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ab}(\langle \underline{z} : \mathbf{ap}(\mathbf{ap}(\underline{x}, \underline{z}), \mathbf{ap}(\underline{y}, \underline{z}))\rangle)\rangle)\rangle)\rangle) \simeq \mathbf{S}$*;*

(or $\mathbf{S}(\mathbf{S}(\mathbf{KS})(\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{KS})))(\mathbf{S}(\mathbf{S}(\mathbf{KS})(\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{KS})))(\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{KK})))\mathbf{K})))(\mathbf{K}(\mathbf{KI})))))(\mathbf{K}(\mathbf{S}(\mathbf{S}(\mathbf{KS})\mathbf{K})(\mathbf{KI}))) \simeq \mathbf{S}$*).*

Proof By structural induction. □

This lemma coupled with Lemma 9.2.5 convinces us that λ-C is a reversed version of C-λ, and vice versa.

Lemma 9.3.5 (*ab* and bound variable): *For $\underline{t} \in T^c$, we have*

$$(\mathbf{K})\,(\mathbf{S})\,(\mathbf{KS}) \vdash_{c^-} \mathbf{ab}(\langle \underline{x} : \underline{t}\rangle) \simeq \mathbf{ap}(\mathbf{K}, \underline{t})$$

provided $\underline{x} \notin Free(\underline{t})$, and where $(\mathbf{KS})$ *is* $\mathbf{ap}(\mathbf{K}, \mathbf{ap}(\tilde{x}, \tilde{y})) \simeq \mathbf{ap}(\mathbf{ap}(\mathbf{S}, \mathbf{ap}(\mathbf{K}, \tilde{x})), \mathbf{ap}(\mathbf{K}, \tilde{y}))$ *for $\tilde{x}, \tilde{y} \in \vec{\chi}$.*

Proof By structural induction on $\underline{t}$. □

Lemma 9.3.5 can also be regarded as explaining the necessity of $(\mathbf{KS})$.

Lemma 9.3.6 (*ab* **and substitutions**): *For* $t \in T^c$, *we have*

$$(\mathbf{K})\ (\mathbf{S})\ (\mathbf{KS}) \vdash_{c^-} \mathbf{ab}(\langle \underline{x} : \underline{t}\rangle)[\vec{\underline{f}}, \vec{\underline{x}} := \vec{\underline{u}}, \vec{\underline{v}}] \simeq \mathbf{ab}(\langle \underline{y} : \underline{t}[\vec{\underline{f}}, \vec{\underline{x}}, \underline{x} := \mathbf{ab}(\langle \vec{\underline{y}} : \vec{\underline{v}}\rangle), \vec{\underline{v}}, \underline{y}]\rangle),$$

where

(i) $\vec{\underline{u}}(i) = \mathbf{ab}(\langle \underline{y}_i : \underline{v}_i\rangle)$;

(ii) $\underline{y} \in \underline{V}$ *such that* $\forall \tilde{z} \in Free(\underline{t}) - \{\underline{x}\}.\underline{y} \notin Free(\vec{\imath}[\vec{\underline{f}}, \vec{\underline{x}} := \vec{\underline{u}} : \vec{\underline{v}}]), \vec{\underline{v}}][\tilde{z}])$;

(iii) $Free(\mathbf{ab}(\langle \underline{y}_i : \underline{v}_i\rangle)) \cap \underline{V} = \emptyset$ $(1 \leq i \leq k = |\vec{\underline{v}}|)$.

Proof By structural induction with Lemma 9.3.5. □

Lemma 9.3.7 (**β-conversion in Combinatory Logic**): *Given* $t \in T^c$, *for all* $u \in T^c$, *we have*

$$(\mathbf{K})\ (\mathbf{S}) \vdash_{c^-} \mathbf{ap}(\mathbf{ab}(\langle \underline{x} : \underline{t}\rangle), u) \simeq t[\underline{x} := u].$$

Proof by structural induction on t. □

It is hard to show that **ab** preserves derivations directly, and especially when it is related to substitutions (ord-sub), we are thinking of eliminating substitution rule completely in Combinatory Logic. For this reason, we introduce a new concept *pseudo semi-closedness* for CL-terms, i.e. a CL-term $\underline{p} \in T^c$ (or a CL function term $\underline{p} \in FT_1^c$) is *pseudo semi-closed* iff $Free(\underline{p}) \cap \underline{V} = \emptyset$; since $\underline{x}$ and $\underline{f}$ play different roles anyway. Similarly, an equation $p \simeq q$ is said to be *pseudo semi-closed* iff $(Free(\underline{p}) \cup Free(\underline{q})) \cap \underline{V} = \emptyset$. Thus, we have an easy verified fact by Lemma 9.3.5. Namely,

Fact 9.3.8 (*ab* **and pseudo semi-closedness**): *If* $\vdash_{c^-} \underline{t} \simeq \underline{u}$ *and* $\underline{t} \simeq \underline{u}$ *is pseudo semi-closed, then for* $\underline{x} \in \underline{V}$, $\vdash_{c^-} \mathbf{ab}(\langle \underline{x} : \underline{t}\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{u}\rangle)$.

Therefore, with the pseudo semi-closedness **ab** does preserve derivability.

To eliminate the need for substitution rule in proof, we have to find some extra axioms (or schemas) which, as a whole, have the same proof power, (i.e derivability) as the original CL with substitution rule. Obviously, the proof power of the new collection of axioms (or schemas) has to be closed under substitutions.

In what follows, we proceed our search for the new collection of axioms gradually. Most technical verifications are left out. However, if you are interested in such details, you should have no difficulty in carrying them out by yourself or you can look them up in [7] with slight notational modification.

Lemma 9.3.9 (search step 1): *With substitution rule, the set of axioms* $\{(\mathbf{K}'), (\mathbf{S}'), (\mathbf{KS}')\}$ *has the same derivability as the original set of axioms* $\{(\mathbf{K}), (\mathbf{S}), (\mathbf{KS})\}$ *respectively; where*

1. $(\mathbf{K}')$ $\mathbf{K}\,\underline{x}\,\underline{y} \simeq \underline{x}$ *for* $\underline{x}, \underline{y} \in \underline{V}$,

2. $(\mathbf{S}')$ $\mathbf{S}\,\underline{x}\,\underline{y}\,\underline{z} \simeq \underline{x}\,\underline{z}(\underline{y}\,\underline{z})$ *for* $\underline{x}, \underline{y}, \underline{z} \in \underline{V}$,

3. $(\mathbf{KS}')$ $\mathbf{K}(\underline{x}\,\underline{y}) \simeq \mathbf{S}(\mathbf{K}\,\underline{x})(\mathbf{K}\,\underline{y})$.

The difference between $\{(\mathbf{K}'), (\mathbf{S}'), (\mathbf{KS}')\}$ and $\{(\mathbf{K}), (\mathbf{S}), (\mathbf{KS})\}$ is quite a minor one. It is just the variables ranging over different domains, say over $\underline{V}$ instead of over $\vec{\chi} = \underline{V} \cup \underline{FV}_1$.

Lemma 9.3.10 (search step 2): $\{(\mathbf{K}'), (\mathbf{S}'), (\mathbf{KS}'), (Ab_f)\}$ *with the substitution rule has the same proof power as the set of schemas* $\{(\mathbf{K}''), (\mathbf{S}''), (\mathbf{SK_{ab}}), (Ab_f)\}$ *without the substitution rule;*

1. $(\mathbf{K}'')$ $\mathbf{K}\,\underline{t}\,\underline{u} \simeq \underline{t}$,

2. $(\mathbf{S}'')$ $\mathbf{S}\,\underline{t}\,\underline{u}\,\underline{v} \simeq \underline{t}\,\underline{v}(\underline{u}\,\underline{v})$,

3. $(\mathbf{KS}'')$ $\mathbf{K}\,\underline{t}\,\underline{u} \simeq \mathbf{S}(\mathbf{K}\,\underline{t})(\mathbf{K}\,\underline{u})$,

4. $(\mathbf{SK_{ab}})$ $\mathbf{S}(\mathbf{Kab}(\langle \underline{x} : \underline{t} \rangle))\mathbf{I} \simeq \mathbf{ab}(\langle \underline{x} : \underline{t} \rangle)$, *and*

5. (Ab_f) $\mathbf{S}(\mathbf{K}\underline{f})\mathbf{I} \simeq \underline{f}$ *(where* $\underline{f}$ *is viewed as a special constant).*

Lemma 9.3.11 (search step 3): $(\mathbf{SK}_{ab})$, *i.e.* $\mathbf{S}(\mathbf{Kab}(\langle \underline{x} : \underline{t} \rangle))\mathbf{I} \simeq \mathbf{ab}(\langle \underline{x} : \underline{t} \rangle)$, *is implied by the following three axioms:*

(i) $(Ab_{\mathbf{I}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ap}(\mathbf{I}, \underline{x}) \rangle) \simeq \mathbf{I}$,

(ii) $(Ab_{\mathbf{S}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ap}(\mathbf{ap}(\mathbf{S}, \mathbf{ap}(\mathbf{K}, \mathbf{ap}(\mathbf{ap}(\mathbf{S}, \underline{x}), \underline{y}))), \mathbf{I})\rangle)\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ap}(\mathbf{ap}(\mathbf{S}, \underline{x})\underline{y})\rangle)\rangle)$,

(iii) $(Ab_{\mathbf{K}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ap}(\mathbf{ap}(\mathbf{K}, \underline{x})\underline{y})\rangle)\rangle) \simeq \mathbf{K}$.

Lemma 9.3.12 (search step 4): $(\mathbf{K}'')$, *i.e.* $\mathbf{ab}(\langle \underline{x} : \mathbf{K}\,\underline{t}\,\underline{u}\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{t}\rangle)$, *is implied by*

(a) $(\mathbf{K}_{\mathbf{ab}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{S}(\mathbf{S}(\mathbf{KK})\underline{x})\underline{y}\rangle)\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{S}(\mathbf{K}\,\underline{x})\mathbf{I}\rangle)\rangle)$ *and*

(b) $(\mathbf{SK}_{\mathbf{ab}})$ $\mathbf{S}(\mathbf{Kab}(\langle \underline{x} : \underline{t}\rangle))\mathbf{I} \simeq \mathbf{ab}(\langle \underline{x} : \underline{t}\rangle)$

Lemma 9.3.13 (search step 5): $(\mathbf{S}'')$, *i.e.* $\mathbf{ab}(\langle \underline{x} : \mathbf{S}\,\underline{t}\,\underline{u}\,\underline{v}\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{t}\,\underline{v}(\underline{u}\,\underline{v})\rangle)$, *is implied by* $(\mathbf{S}_{\mathbf{ab}})$, *i.e.* $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ab}(\langle \underline{z} : \mathbf{S}(\mathbf{S}(\mathbf{S}(\mathbf{KS})\underline{x})\underline{y})\underline{z}\rangle)\rangle)\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{ab}(\langle \underline{z} : \mathbf{S}(\mathbf{S}\,\underline{x}\,\underline{z})(\mathbf{S}\,\underline{y}\,\underline{z})\rangle)\rangle)\rangle)$.

Lemma 9.3.14 (search step 6): $(\mathbf{KS}'')$, *i.e.* $\mathbf{ab}(\langle \underline{x} : \mathbf{K}(\underline{t}\,\underline{u})\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{S}(\mathbf{K}\,\underline{t})(\mathbf{K}\,\underline{u})\rangle)$, *is implied by* $(\mathbf{KS}_{\mathbf{ab}})$, *i.e.* $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{S}(\mathbf{KK})(\mathbf{S}\,\underline{x}\,\underline{y})\rangle)\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{S}(\mathbf{S}(\mathbf{KS})(\mathbf{S}(\mathbf{KK})\underline{x}))(\mathbf{S}(\mathbf{KK})\underline{y})\rangle)\rangle)$.

By the previous six search steps (from Lemma 9.3.9 to Lemma 9.3.14), we are finally reaching our goal, i.e. elimination of the substitution rule.

Note that the six previous lemmas can be proved by structural induction or by induction on the length of proof $\underline{t} \simeq \underline{u}$. Basically, we can introduce *"contexts"*, i.e. let [ ] be a special (meta-)variable, being viewed as the identity context, and structurally define context $Cont^c$ as

1. $[\,] \in Cont^c$
2. $\mathbf{K}, \mathbf{S}, \underline{x}, \underline{f} \in Cont^c$
3. $$\frac{C_i \in Cont^c}{\mathbf{ap}(C_1, C_2) \in Cont^c}.$$

Then by induction on the length of proof $\underline{t} \simeq \underline{u}$, we have the property that if $t \simeq u$ is derivable then so is $C[\underline{t}] \simeq C[\underline{u}]$. Subsequently, we define $\mathbf{ab}(\langle \underline{x} : C[\,]\rangle)$ over context $Cont^c$ as

- $\mathbf{ab}(\langle \underline{x} : [\,] \rangle) =_{df} [\,]$

- $\mathbf{ab}(\langle \underline{x} : \mathbf{ap}(C_1, C_2) =_{df} \mathbf{S}(\mathbf{ab}(\langle \underline{x} : C_1 \rangle))(\mathbf{ab}(\langle \underline{x} : C_2 \rangle))$.

Therefore, we can use $C[\,]$ to show that $[\,]$ may or may not be present, and it is more meaningful to denote $\mathbf{ab}(\langle \underline{x} : C[\,] \rangle)$ as $\mathbf{ab}(\langle \underline{x} : C \rangle)[\,]$. We know that "abstraction" over "contexts" has the following property, i.e. if $\underline{t} \simeq \underline{u}$ is derivable, so is $\mathbf{ab}(\langle \underline{x} : C[\underline{t}] \rangle) \simeq \mathbf{ab}(\langle \underline{x} : C[\underline{u}] \rangle)$, since $\mathbf{ab}(\langle \underline{x} : C \rangle)[\mathbf{ab}(\langle \underline{x} : \underline{t} \rangle)] = \mathbf{ab}(\langle \underline{x} : C[\underline{t}] \rangle)$.

In summary, we have that

Theorem 9.3.15 (A'_a set of axiom schemas):

1. *If $\underline{p} \simeq \underline{q}$ is derivable from $\{(\mathbf{K}'), (\mathbf{S}'), (\mathbf{KS}'), (Ab_f)\}$, then so is $\underline{p} \simeq \underline{q}$ from A'_a without the substitution rule, where A'_a contains the following*

   (a) $(Ab_{\mathbf{I}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{I}\,\underline{x} \rangle) \simeq \mathbf{I}$

   (b) (Ab_f) $\mathbf{ab}(\langle \underline{x} : \mathbf{ap}(\underline{f}, \underline{x}) \rangle) \simeq \underline{f}$.

   (c) $(Ab_{\mathbf{K}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{K}\,\underline{x}\,\underline{y}) \rangle)) \simeq \mathbf{K}$

   (d) $(\mathbf{K_{ab}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{S}(\mathbf{S}(\mathbf{KK})\underline{x})\underline{y}) \rangle)) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\langle \underline{y} : \mathbf{S}(\mathbf{K}\,\underline{x})\mathbf{I}) \rangle))$

   (e) $(Ab_{\mathbf{S}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{S}\,\underline{x}\,\underline{y}) \rangle)) \simeq \mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{S}(\mathbf{K}(\mathbf{S}\,\underline{x}\,\underline{y}))\mathbf{I}) \rangle))$

   (f) $(\mathbf{S_{ab}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{ab}(\langle \underline{z} : \mathbf{S}(\mathbf{S}(\mathbf{KS})\underline{x})\underline{y})\underline{z}) \rangle))) \simeq$
   $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{ab}(\langle \underline{z} : \mathbf{S}(\mathbf{S}\,\underline{x}\,\underline{z})(\mathbf{S}\,\underline{y}\,\underline{z}) \rangle)) \rangle))$

   (g) $(\mathbf{KS_{ab}})$ $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{S}(\mathbf{KK})(\mathbf{S}\,\underline{x}\,\underline{y}) \rangle)) \simeq$
   $\mathbf{ab}(\langle \underline{x} : \mathbf{ab}(\underline{y} : \mathbf{S}(\mathbf{S}(\mathbf{KS})(\mathbf{S}(\mathbf{K}$
   $\mathbf{K})\underline{x}))(\mathbf{S}(\mathbf{KK})\underline{y}) \rangle)) \rangle)$.

2. *The derivability (or proof power) of A'_a (or coupled with $\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}$) with the substitution rule is the same as the one of A'_a (or coupled with $\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}$) without the substitution rule.*

Finally, we reach that the deduction closure of $A'_a(\cup\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\})$ is closed under (Ab_ξ) where (Ab_ξ) is $\{\underline{t} \simeq u\} \mapsto \mathbf{ab}(\langle \underline{x} : \underline{t} \rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{u} \rangle)$ and which means that whenever $\underline{t} \simeq \underline{u}$ is derivable, so is $\mathbf{ab}(\langle \underline{x} : \underline{t} \rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{u} \rangle)$. Formally,

Theorem 9.3.16 (derivation closure over *ab*): *For $\underline{t}, \underline{u} \in T^c$, we have that $A'_a(\cup\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}) \vdash_{c-} \underline{t} \simeq \underline{u}$ implies that $A'_a(\cup\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}) \vdash_{c-} \mathbf{ab}(\langle \underline{x} : \underline{t} \rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{u} \rangle)$ for $\underline{x} \in \underline{V}$.*

Proof

By induction on the length of proof $A'_a \vdash_{c-} \underline{t} \simeq \underline{u}$ with the lemmas from Lemma 9.3.5 to Lemma 9.3.14. □

Lemma 9.3.17 (substitution and λ-*C*): *For $p \in T^\lambda$ or $p \in FT_1^\lambda$, we have $A'_a(\cup\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}) \vdash_{c-} \lambda\text{-}C[\![p[\vec{f}, \vec{x} := \vec{v}, \vec{t}\,]\,]\!] \simeq \lambda\text{-}C[\![p]\!][\underline{\vec{f}}, \vec{\underline{x}} := \vec{\underline{v}}, \lambda\text{-}C[\![\vec{\underline{t}}\,]\!]\,]$,*

where $\vec{v}(i) = \langle y_i : u_i \rangle$, $\vec{\underline{v}}(i) = \mathbf{ab}(\langle \vec{\underline{y}}_i : \lambda\text{-}C[\![\vec{u}_i]\!] \rangle)$ and $Free(\langle y_i : u_i \rangle) \cap V = \emptyset$ $(1 \leq i \leq k)$.

Proof

By structural induction on p with Lemma 9.3.6, Lemma 9.3.7 and Lemma 9.3.16. □

Lemma 9.3.18 (λ-*C* and A'_a): *For $p, q \in T^\lambda$ or $p, q \in FT_1^\lambda$, we have that $(\beta) \vdash_\lambda p \simeq q$ implies $A'_a(\cup\{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}) \vdash_{c-} \lambda\text{-}C[\![p]\!] \simeq \lambda\text{-}C[\![q]\!]$.*

Proof

By induction on the length of proof $(\beta) \vdash_\lambda p \simeq q$ with Lemma 9.3.16 and Lemma 9.3.17. □

Therefore, we come to

Theorem 9.3.19 (A'_a and λ-algebras): *For $\underline{p}, \underline{q} \in T^c$ or $\underline{p}, \underline{q} \in FT_1^c$, if $(\beta) \vdash_\lambda C\text{-}\lambda[\![\underline{p}]\!] \simeq C\text{-}\lambda[\![\underline{q}]\!]$, then $A'_a \cup \{(Ab'_{\mathbf{K}})(Ab'_{\mathbf{S}})\} \vdash_{c-} \underline{p} \simeq \underline{q}$.*

Proof

By induction on the length of proof $(\beta) \vdash_\lambda C\text{-}\lambda[\![\underline{p}]\!] \simeq C\text{-}\lambda[\![\underline{q}]\!]$, with Lemma 9.3.4, Lemma 9.3.18, i.e. $\underline{p} \simeq \lambda\text{-}C[\![C\text{-}\lambda[\![\underline{p}]\!]\,]\!] \simeq \lambda\text{-}C[\![C\text{-}\lambda[\![\underline{q}]\!]\,]\!] \simeq \underline{q}$.□

Up to this stage, we know that for an iBA **B** of Combinatory Logic, if $\mathbf{B} \models A'_a \cup \{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}$ then **B** is an λ-algebra. The reversed implication holds as well. Consequently, we get that

Theorem 9.3.20 (λ-algebras and A_a): *Let* $\mathbf{B}$ *be an algebra of Combinatory Logic and* $A_a = A'_a \cup \{(Ab'_{\mathbf{K}}), (Ab'_{\mathbf{S}})\}$ *Then,* $\mathbf{B}$ *is a λ-algebra iff* $\mathbf{B} \models A_a$.

Proof

For "⇐" (if direction), we have Theorem 9.3.19.

For "⇒" (only if direction), we just need to check that $(\beta) \vdash_\lambda C\text{-}\lambda[\![A_a]\!]$. □

9.4 Equational theory for λ-models

From Section 9.3, we understand that λ-algebras are equivalent to β-algebras in the sense of derivability where β-algebras are iBAs satisfying (β). There is another kind of semantic models similar to λ-algebras, i.e. λ-models (see [7, Definition 5.2.7]). They are of the subject of this section. λ-models are basically λ-algebras with Extensionality. We define λ-models first.

Definition 9.4.1 (λ-models): *A λ-algebra* $\mathbf{B}$ *is said to be a λ-model iff* $\mathbf{B} \models (ext_c\text{-}axiom)$, *where* $(ext_c\text{-}axiom)$ *is* $\{\mathbf{ap}(\underline{y}, \underline{x}) \simeq \mathbf{ap}(\underline{z}, \underline{x})\} \mapsto \underline{y} \simeq \underline{z}$.

There is another equivalent way of defining λ-models, e.g. let $\mathbf{B}$ be a λ-algebra, then whenever $\mathbf{B} \models \mathbf{ap}(\underline{y}, \underline{x}) \simeq \mathbf{ap}(\underline{z}, \underline{x})$ holds, $\mathbf{B} \models \underline{y} \simeq \underline{z}$ holds as well. Also, $(ext_c\text{-}axiom)$ can be equivalently replaced by $(ext_c\text{-}schema)$ where $(ext_c\text{-}schema)$ is $\{\mathbf{ap}(\underline{t}, \underline{x}) \simeq \mathbf{ap}(\underline{u}, \underline{x})\} \mapsto \underline{t} \simeq \underline{u}$.

Let

(a) ext_λ be Extensionality rule in Lambda Calculus, i.e.

$$(ext_\lambda) \quad \{\mathbf{ap}(y, x) \simeq \mathbf{ap}(z, x)\} \mapsto y \simeq z;$$

(b) $\eta\text{-}axiom$ be:

$$(\eta) \quad \lambda x.yx \simeq y;$$

(c) (Ab_η) be

$$(Ab_\eta) \quad \mathbf{ab}((\underline{x} : \mathbf{S}(\mathbf{K}\,\underline{x})\mathbf{I})) \simeq \mathbf{I}.$$

We should notice that $(ext_c\text{-}axiom)$ (or simply ext_c) and (ext_λ) are dBEs.

Then, we can easily get that

Theorem 9.4.2 (**λ-models and** Ab_η): *The following three statements are equivalent:*

1. $\mathbf{B} \models \underline{p} \simeq \underline{q}$ *for all* λ*-model* $\mathbf{B}$.

2. $A_a \vdash_{c^- + ext_c} \underline{p} \simeq \underline{q}$ *(since the derivability is closed under BEs in derivations of* $\vdash^d_{iBA}$*)*

3. $A_a \cup \{(Ab_\eta)\} \vdash_{c^-} \underline{p} \simeq \underline{q}$.

Proof

(i) For *1* $\Leftrightarrow$ *2*, it is obvious.

(ii) For *2* $\Leftrightarrow$ *3*,

(ii.a) let us look at *2* $\Rightarrow$ *3* first. applying $\underline{t}$ to both sides of (Ab_η), we have $\mathbf{ab}(\langle \underline{x} : \underline{t}\,\underline{x}\rangle) \simeq \underline{t}$, where $\underline{x} \notin Free(\underline{t})$.

Similarly, we have $\mathbf{ab}(\langle \underline{x} : \underline{u}\,\underline{x}\rangle) \simeq \underline{u}$, where $\underline{x} \notin Free(\underline{u})$.

Now, let us assume that $A_a \cup \{(Ab_\eta)\} \vdash_{c^-} \mathbf{ap}(\underline{t}, \underline{x}) \simeq \mathbf{ap}(\underline{u}, \underline{x})$, where $\underline{x} \notin Free(\underline{t}) \cup Free(\underline{u})$. Then by Lemma 9.3.16 we have $A_a \cup \{(Ab_\eta)\} \vdash_{c^-} \mathbf{ab}(\langle \underline{x} : \underline{t}\,\underline{x}\rangle) \simeq \mathbf{ab}(\langle \underline{x} : \underline{u}\,\underline{x}\rangle)$. So, $A_a \cup \{(Ab_\eta)\} \vdash_{c^-} \underline{t} \simeq \underline{u}$. i.e. (ext_c) is valid in $A_a \cup \{(Ab_\eta)\}$ deduction system. Thus, (ext_c) is valid in $A_a \cup \{(Ab_\eta)\}$ deduction system (since $(ext_c\text{-}schema)$ without substitution is equivalent to (ext_c) with substitution). Therefore, by induction on the length of proofs of $A_a \vdash_{c^- + (ext_c)} \underline{p} \simeq \underline{q}$, we come to *2* $\Rightarrow$ *3*.

(ii.b) Next, for *3* $\Rightarrow$ *2*, we know that $A_a \vdash_{c^- + (ext_c\text{-}schema)} \mathbf{ab}(\langle \underline{x} : \underline{t}\,\underline{x}\rangle)\,\underline{x} \simeq \underline{t}\,\underline{x}$. Then by $(ext_c\text{-}schema)$, we get $A_a \vdash_{c^- + (ext_c\text{-}schema)} \mathbf{ab}(\langle \underline{x} : \underline{t}\,\underline{x}\rangle) \simeq \underline{t}$.

So, let $\underline{t}$ be $\underline{y}$ $(\not\equiv \underline{x})$, we get $A_a \vdash_{c^- + (ext_c\text{-}schema)} \mathbf{ab}(\langle \underline{x} : \underline{y}\,\underline{x}\rangle) \simeq \underline{y}$.

To put it another way, this is

$$A_a \vdash_{c^- + (ext_c\text{-}schema)} \mathbf{ab}(\langle \underline{y} : \mathbf{ab}(\langle \underline{x} : \underline{y}\,\underline{x}\rangle)\rangle)\underline{y} \simeq \mathbf{ab}(\langle \underline{y} : \underline{y}\rangle)\underline{y}.$$

Again by (ext_c-*schema*), we come to

$$A_a \vdash_{c^-+(ext_c\text{-}schema)} \mathbf{ab}(\langle \underline{y} : \mathbf{ab}(\langle \underline{x} : \underline{y}\,\underline{x}\rangle)\rangle) \simeq \mathbf{ab}(\langle \underline{y} : \underline{y}\rangle).$$

i.e. $A_a \vdash_{c^-+(ext_c\text{-}schema)} (Ab_\eta)$.

Note that $A_a \vdash_{c^-+(ext_c\text{-}schema)} \underline{p} \simeq \underline{q}$ without substitutions iff $A_a \vdash_{c^-+(ext_c)} \underline{p} \simeq \underline{q}$ with substitutions.

Therefore, by induction on the length of proof $A_a \cup \{(Ab_\eta)\} \vdash_{c^-} \underline{p} \simeq \underline{q}$, we have that *2* $\Leftarrow$ *3*. □

So, we know that the equational theory of λ-models is almost the same as the one of λ-algebras but with an extra axiom (Ab_η). Naturally, we would like to know the corresponding theory of λ-models in Lambda Calculus. This correspondence is provided as follows and the details are omitted. In case of any doubt, you can check those in [7].

Lemma 9.4.3 (ext_λ, ext_c **and** λ-C): $(\beta) \vdash_{\lambda+(ext_\lambda)} p \simeq q$ *iff* $A_a \vdash_{c^-+(ext_c)} \lambda\text{-}C[\![p]\!] \simeq \lambda\text{-}C[\![q]\!]$.

Lemma 9.4.4 (ext_λ, ext_c **and** C-λ): $A_a \vdash_{c^-+(ext_c)} \underline{p} \simeq \underline{q}$ *iff* $(\beta) \vdash_{\lambda+(ext_\lambda)} C\text{-}\lambda[\![\underline{p}]\!] \simeq C\text{-}\lambda[\![\underline{q}]\!]$.

Lemma 9.4.5 (η **and** ext_λ): $\{(\beta),(\eta)\} \vdash_\lambda p \simeq q$ *iff* $(\beta) \vdash_{\lambda+(ext_\lambda)} p \simeq q$.

In summary, we have the following.

Theorem 9.4.6 (η, ext_λ, ext_c **and** Ab_η): *The following deductive systems share proof power: (i)* $\{(\beta),(\eta)\}$, *(ii)* $\{(\beta) \cup (ext_\lambda)\}$, *(iii)* $A_a \cup \{(ext_c)\}$, *and (iv)* $A_a \cup \{(Ab_\eta)\}$.

9.5 Discussion

9.5.1 Curry's A_β and λ-algebra's A_a

Comparing the results in this chapter with some results in literature, for example let us relate A_a with A_β (see [105]), we have that

(9.5.a) $(Ab_{\mathbf{K}})$, $(\mathbf{K_{ab}})$ and $(\mathbf{S_{ab}})$ are (7.1), (7.4) and (7.5) in [105] respectively.

(9.5.b) $(Ab'_{\mathbf{K}})$ implies $(\mathbf{K}')$ and $(Ab'_{\mathbf{S}})$ implies $\mathbf{S}$ in [105].

(9.5.c) $(Ab_{\mathbf{S}}) + (Ab'_{\mathbf{S}})$ implies (7.2) in [105].

(9.5.d) $(\mathbf{KS_{ab}})$ implies (7.3) in [105].

(9.5.e) $(Ab_{\mathbf{I}})$ and (Ab_f) do not appear in A_β. We can say that the disappearance of (Ab_f) is due the non-existence of $\underline{f}$ in the CL of [105]. But this can not applied to the case for $(Ab_{\mathbf{I}})$. So, the reason for the disappearance of $(Ab_{\mathbf{I}})$ in [105] is not clear to us.

9.5.2 Function variables and definable elements

In section 9.2, we make a correspondence of function variables in Lambda Calculus to "*definable elements*" in Combinatory Logic. Accordingly, this divides CL-terms into two categories. One is "ordinary" terms (say T^c) and the other is "*definable*" terms (say $Ab(T^c)$). The collection of the "definable" terms is a sub-collection of the "ordinary" terms. This difference is represented in the deduction system of the equational theory in the following ways,

(9.5.f) by the "ordinary" variables (say $\underline{x}$) can be substituted by both "ordinary" and "definable" terms;

(9.5.g) by the "definable" variables (say $\underline{f}$) can only be substituted by the "definable" terms; and

(9.5.h) more significantly by an extra axiom (say (Ab_f)) for the "definable" elements (the "definable" makes more sense if we only consider the counterpart of $(\beta) \vdash_\lambda$ in CL).

Interestingly, this difference essentially disappears in the equational theory for λ-models, because of the presence of axiom (Ab_η). This means that the "ordinary" elements and "definable" elements are inter-substitutable. This also implies that it is no longer necessary to distinguish the variables like $\underline{x}$ from the variables like $\underline{f}$. However, this situation can not be applied to Lambda Calculus part, i.e. so-called $\lambda\beta\eta$-calculus (see [7]). The difference between the ordinary variables and the function variables still remains. It is worth to mention that η conversion (i.e. $\lambda x.yx \simeq y$) for function variables (i.e. $\lambda x.(\lambda y.f(y))x \simeq \lambda x.f(x)$), is derivable from (β). And it is believed to correspond to (Ab_f) in Combinatory Logic.

9.5.3 λ-family and extensional β-algebra

According to [7, p.127], we can also establish a connection between Volken's *λ-family* and an eBA satisfying β, i.e. λ-families and eBAs satisfying β are essentially the same. More specifically, let $\mathcal{F} =< X, \cdot >$ be an applicative structure and $\mathcal{F}^* = \bigcup_{\ell \in Nat} \mathcal{F}_\ell$ be a family of functions such that for every $f \in \mathcal{F}_\ell, f : X^\ell \to X$. $\mathcal{F}^*$ is called a λ-family on $\mathcal{F}$ iff

(9.5.1) $\mathcal{F}^*$ contains all *algebraic* functions[1] where $g : X^\ell \to X$ is algebraic (or definable) if there is a term t and $Free(t) \subseteq \{\vec{x}\}$ and $|\vec{x}| = \ell$ such that

$$\forall \vec{a}.g(\vec{a}) = [\![t]\!](\rho[\vec{x} := \vec{a}]),$$

(9.5.2) $\mathcal{F}^*$ is closed under substitution of constants (e.g. if $g \in \mathcal{F}_2$ and $a \in X$, then $[\![\langle y : f(x,y)\rangle]\!](\rho[x := a][f := g]) \in \mathcal{F}_1$),

(9.5.3) there is a map $G : \mathcal{F}_1 \to X$ such that $\forall g \in \mathcal{F}_1, \forall a \in X.G(g) \cdot a = g(a)$, and

---

[1]The definition for algebraic functions can be found in [7, Definition 5.1.6].

(9.5.4) $\forall g \in \mathcal{F}_{k+1}$, $\forall \vec{a} \in X^k$, if $h(\vec{a}) = G([\![(x : f(\vec{y}, x))]\!](\rho[f := g][\vec{y} := \vec{a}]))$ then $h \in \mathcal{F}_k$ where $|\vec{y}| = k$.

Note that the above definition is almost the same as in [7, p.127] except of some notational modification.

Obviously, if $\mathbf{A}$ is an eBA satisfying (β), $\mathcal{F}^{\mathbf{A}}$ satisfies (9.5.1) and (9.5.2) above and (9.5.4) corresponds to (eBA-unif). Moreover, the interpretation $\lambda^{\mathbf{A}}$ of λ-abstraction operator corresponds to G in (9.5.2) and that $\mathbf{A}$ satisfies β means (9.5.2) is true for $\mathbf{A}$. In the other words, $\mathcal{F}^{\mathbf{A}}$ is a λ-family.

Conversely, if $\mathcal{F}^*$ is a λ-family, Conditions (9.5.1) and (9.5.2) guarantee that it is explicitly closed (i.e. closed under constant functions[2], projections[3] and function compositions). G can be taken as the interpretation of λ-abstraction operator, and condition (9.5.4) guarantees that G is uniform over $\mathcal{F}^*$. So, $< X, \mathcal{F}^* >$ can be regarded as an eBA. Furthermore, condition (9.5.3) guarantees that $< X, \mathcal{F}^* >$ satisfies (β). Hence, we have briefly established the connection between Volken's λ-families and eBAs satisfying (β).

Similar to Volken's connection between λ-families and λ-models, we can have an easy correspondence between *syntactical* λ-models and eBAs satisfying (β) (i.e. λ-families), where $\mathbf{M} =< X, \cdot, [\![\bullet]\!] >$ is a *syntactical* λ-model if it is a *syntactical applicative structure* (see [7, Section 5.3]); i.e.

(9.5.i) $[\![x]\!](\rho) = \rho(x)$,

(9.5.ii) $[\![C_{\mathbf{a}}]\!](\rho) = a$,

(9.5.iii) $[\![\mathbf{app}(M, N)]\!](\rho) = [\![M]\!](\rho) \cdot [\![N]\!](\rho)$,

(9.5.iv) $[\![\lambda x.M]\!](\rho) \cdot a = [\![M]\!](\rho[x := a])$

(9.5.v) $\rho\lceil Free(M) = \rho'\lceil Free(M)$ implies $[\![M]\!](\rho) = [\![M]\!](\rho')$ and that

---

[2]This is actually implied by condition (9.5.2).

[3]This is implied by condition (9.5.1), e.g. $\forall \vec{a}.a_j = \pi_{\ell,j}(\vec{a}) = [\![x_j]\!](\rho[\vec{x} := \vec{a}])$, where $|\vec{x}| = \ell$.

(9.5.vi) for all a $[\![M]\!](\rho[x := a]) = [\![N]\!](\rho[x := a])$ implies $[\![\lambda x.M]\!](\rho) = [\![\lambda x.\dot{N}]\!](\rho)$.

Similar to showing the connection between λ-families and eBAs satisfying (β), the link from eBAs satisfying (β) to syntactical λ-models is straightforward and is left out. For the reversed link, following Section 3.4 (Chapter 3), especially from Lemma 3.4.6 to Lemma 3.4.9, we can build a term eBA from $\boldsymbol{M}$ which is an eBA satisfying (9.5.i), (9.5.ii), (9.5.iii), (9.5.v) and (9.5.vi). In other words, the term eBA is a conservative extension of $\boldsymbol{M}$ satisfying (9.5.i), (9.5.ii), (9.5.iii), (9.5.v) and (9.5.vi). For (9.5.iv), we know that

$$[\![\lambda x.M]\!](\rho) \cdot a = [\![\lambda x.M]\!](\rho) \cdot [\![C_{\mathbf{a}}]\!](\rho) = [\![\mathbf{app}(\lambda x.M, C_{\mathbf{a}})]\!](\rho).$$

Since quotient algebras preserve satisfaction (see the comment after Definition 3.7.7), we can get a quotient eBA from the above term eBA over the least congruence containing (β). Apparently, this quotient term algebra satisfies (9.5.iv) besides the others. Therefore, we have completed the reversed link. Also, we should point out that the correspondence between syntactical λ-models and eBAs satisfying (β) is *functorial* (i.e. preserving homomorphisms).

With regards to the connection between λ-models and syntactical λ-models, see [7, Theorem 5.3.6]. We shall not repeat this here.

9.5.4 Intensional β-algebra and Cartesian Closed Category

Another thing we should point out here is that our attempt to connect an intensional *β-algebra* (of Lambda Calculus), which is an iBA satisfies (β), to a Cartesian Closed Category (CCC) is not as successful as in [93,7].

Basically, what we do is to construct a CCC $\mathcal{C}$ from an intensional β-algebra $\mathbf{B}$ and to derive another intensional β-algebra $\mathbf{B}'$ from such a $\mathcal{C}$ by following the approach taken in [93]. But we are not able to show that these two β-algebras are isomorphic. In what follows, we demonstrate our unsuccessful attempt. We are able to construct both $\mathcal{C}$ from $\mathbf{B}$ and $\mathbf{B}'$ from $\mathcal{C}$. Thus, the question becomes whether we can still have the isomorphism between $\mathbf{B}$ and $\mathbf{B}'$.

Let $\mathbf{B}$ be an intentional β-algebra (an iBA satisfying β). We define $u * v =_{df} \mathcal{B}_\varepsilon[\![\langle x : f(f'(x))\rangle]\!](\psi[u, v/f, f'])$ for $u, v \in B_1$, where $f \neq f'$. It is easy to show that this definition is independent of the environment ψ. It is also not difficult to check that $*$ is associative.

Next, we construct a category $\mathcal{C}$ from $\mathbf{B}$. Let $Obj(\mathcal{C}) =_{df} \{g \in B_1 | g * g = g\}$ and $Hom(u, v) =_{df} \{w \in B_1 | v * w * u = w\}$ for $u, v \in Obj(\mathcal{C})$. The composition ";" is defined as $u; v =_{df} v * u$ where $u \in Hom(w_1, w_2)$, $v \in Hom(w_2, w_3)$ and $w_i \in Obj(\mathcal{C})$. So, $u; v \in Hom(w_1, w_3)$. For each $u \in Obj(\mathcal{C})$, the identity $id_u =_{df} u$. We can easily check that $id_u; v = v$ and $w; id_u = w$ where $v \in Hom(u, ?)$ and $w \in Hom(?, u)$.

To see the constructed $\mathcal{C}$ is a CCC, we proceed as usual, see [93] for a reference:

1. terminal object: $T =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.y\rangle]\!](\psi)$.

2. (a) product: $u \times v =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.y \cdot f(x \cdot \mathbf{K}_\lambda) \cdot g(x \cdot \mathbf{K}'_\lambda)\rangle]\!](\psi[u, v/f, g])$, where $t \cdot t' = \mathbf{app}(t, t')$, $\mathbf{K}_\lambda = \lambda x \lambda y.x$ and $\mathbf{K}'_\lambda = \lambda x \lambda y.y$. We always assume the usual association of $\cdot$ (dot) to the left.

   (b) projections: $\pi_u^{u \times v} =_{df} \mathcal{B}_\varepsilon[\![\langle x : f(x \cdot \mathbf{K}_\lambda)\rangle]\!](\psi[u/f])$ and $\pi_v^{u \times v} =_{df} \mathcal{B}_\varepsilon[\![\langle x : g(x \cdot \mathbf{K}'_\lambda)\rangle]\!](\psi[v/g])$

   (c) $< u, v >=_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.y \cdot f(x) \cdot g(x)\rangle]\!](\psi[u, v/f, g])$.

3. (a) exponents: $v^u =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.g(x \cdot f(y))\rangle]\!](\psi[u, v/f, g])$

   (b) $u_1 \times u_2 =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.y \cdot f_1(f(x \cdot \mathbf{K})) \cdot f_2(g(x \cdot \mathbf{K}'))\rangle]\!](\rho, \phi[u_1, u_2, u, v/f_1, f_2, f, g])$, where $u_1 \in Hom(u, u')$, $u_2 \in Hom(v, v')$.

   (c) $ev =_{df} \mathcal{B}_\varepsilon[\![\langle x : g((x \cdot \mathbf{K}_\lambda) \cdot f(x \cdot \mathbf{K}'_\lambda))\rangle]\!](\psi[u, v/f, g])$.

   (d) $\Lambda(u') =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.f'(\lambda z.z \cdot f(x) \cdot g(x))\rangle]\!](\psi[u', u, v/f', f, g])$

We can check the followings:

(i) (terminal object) T is a terminal object in $\mathcal{C}$;

(ii.a) (projections) $\pi_u^{u \times v} \in Hom(u \times v, u)$ and $\pi_v^{u \times v} \in Hom(u \times v, v)$;

(ii.b) (pairs) $< u_1, u_2 > \in Hom(w, u \times v)$

(ii.c) (pairs and projections) $u_1 =< u_1, u_2 >; \pi_u^{u\times v}$ and $u_2 =< u_1, u_2 >; \pi_v^{u\times v}$,

where $u_1 \in Hom(w, u)$ and $u_2 \in Hom(w, v)$;

(iii.a) (products) $u_1 \times u_2 \in Hom(u \times v, u' \times v')$

(iii.b) (evaluation) $ev \in Hom(v^u \times u, v)$

(iii.c) (currying) $\Lambda(u') \in Hom(u, w^v)$, where $u' \in Hom(u \times v, w)$;

(iii.d) (projections and products 1) $\pi_u^{u\times v}; u_1 = (u_1 \times u_2); \pi_{u'}^{u'\times v'}$

(iii.e) (projections and products 2) $\pi_v^{u\times v}; u_1 = (u_1 \times u_2); \pi_{v'}^{u'\times v'}$

(iii.f) (currying and evaluation) $u' = \Lambda(u') \times id_v; ev$.

There is a special object U in $\mathcal{C}$ with the retraction pair e and p :

1. reflexive object $U =_{df} \mathcal{B}_\varepsilon[\![\langle x : x\rangle]\!](\psi)$,

2. embedding $e =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.x \cdot y\rangle]\!](\psi)$,

3. projection $p =_{df} \mathcal{B}_\varepsilon[\![\langle x : \lambda y.x \cdot y\rangle]\!](\psi)$.

We can also verify that $U \in Obj(\mathcal{C})$, $e \in Hom(U^U, U)$ and $p \in Hom(U, U^U)$.

Therefore, the above construction yields a CCC $\mathcal{C}$ with a reflexive object U from $\mathbf{B}$. Now, we come to the questional part, i.e. from $\mathcal{C}$ to an intentional β-algebra $\mathbf{B}'$, which is supposed to be isomorphic to the original $\mathbf{B}$.

Let $\mathcal{C}$ be a CCC with a reflexive object U and retraction pair e and p. For any object v, let $\cdot_v : Hom(v, U)^2 \to Hom(v, U)$ be defined by $v_1 \cdot_v v_2 =_{df} < v_1; p, v_2 >; ev$. It is easy to check that for any $w' \in Hom(w, v)$, $w'; (v_1 \cdot_v v_2) = (w'; v_1) \cdot_w (w'; v_2)$ and $\mathbf{app}^{\mathbf{B}'} = \cdot_T$. Let us forget every thing (say how to extend this to an intentional β-algebra $\mathbf{B}'$) except just look at a possible bijection ι between $\mathbf{B}$ and $\mathbf{B}'$. It is clear to us that such a map ι to be a morphism has to preserve compositionality (or to preserve application **app** operator), i.e. $\iota(t \cdot^{\mathbf{B}} t') = \iota(t) \cdot^{\mathbf{B}'} \iota(t')$. On the other hand, we know there is a Koymans' isomorphism ι in [93], where $\iota(a)$ is defined as $\mathbf{K}_\lambda a$ (also see the proof of Theorem 5.5.13 in [7]). Therefore, we naturally think of translating his bijection into the present framework. However, there is no trivial

way to translate his bijection. Nevertheless, there are two obvious ways of defining such a "bijection" ι by translating the one in [93]. They are ι_1 and ι_2 as follows:

1. $\iota_1(v) =_{df} \mathcal{B}_\epsilon[\![\langle x : \lambda y.f(y)\rangle]\!](\psi[v/f])$ for $v \in B_1$,

2. $\iota_2(v) =_{df} \mathcal{B}_\epsilon[\![\langle x : f(\lambda y.y)\rangle]\!](\psi[v/f])$ for $v \in B_1$.

With some effort, we would know that

(a) ι_1 is injective;

(b) ι_2 is surjective, since $Hom(T, U)$ collects all constant functions in B_1,

(c) ι_1 does not preserves compositionality, since

$$\begin{aligned}
&\iota_1(u) \cdot_1^{\mathbf{B}'} \iota_1(v)\\
&= \mathcal{B}_\epsilon[\![\langle x : f(\lambda y.g(y))\rangle]\!](\psi[u, v/f, g])\\
&\neq \mathcal{B}_\epsilon[\![\langle x : \lambda y.f(g(y))\rangle]\!](\psi[u, v/f, g])\\
&= \iota_1(u \cdot_1^{\mathbf{B}} v);
\end{aligned}$$

(d) ι_2 preserves compositionality, since

$$\begin{aligned}
&\iota_2(u) \cdot_1^{\mathbf{B}'} \iota_2(v)\\
&= \mathcal{B}_\epsilon[\![\langle x : f(\lambda y.y)\rangle \cdot_1 \langle x : g(\lambda y.y)\rangle]\!](\psi[u, v/f, g])\\
&= \mathcal{B}_\epsilon[\![\langle x : f(\lambda y.y) \cdot g(\lambda y.y)\rangle]\!](\psi[u, v/f, g])\\
&= \mathcal{B}_\epsilon[\![\langle x : (\lambda z.f(z) \cdot g(z)) \cdot \lambda y.y\rangle]\!](\psi[u, v/f, g])\\
&= \mathcal{B}_\epsilon[\![\langle x : h(\lambda y.y)\rangle]\!](\psi[u \cdot^{\mathbf{B}} v/h]) = \iota_2(u \cdot^{\mathbf{B}} v);
\end{aligned}$$

(e) $\iota_2(v) = \mathbf{app}_1^{\mathbf{B}}(\iota_1(v), T)$, since $\langle x : \mathbf{app}(\lambda y.f(y), \lambda z.z)\rangle \simeq \langle x : f(\lambda z.z)\rangle$.

Because the images of ι_1 are constant functions, we immediately realize that $\iota_1(B_1) \subseteq \iota_2(B_1)$. Thus, $card(B_1) = card(\iota_1(B_1)) = card(\iota_2(B_1))$, where $card(B)$ means the cardinal number of B. But we have no way to tell whether or not ι_2 is injective at this stage. Therefore, neither ι_1 nor ι_2 is an isomorphism between $\mathbf{B}$ and $\mathbf{B}'$ (unless these two ι_1 and ι_2 collapse with each other).

However, we believe that the above problem can be gotten around if we modify the above definitions to the following: (a) for $\ell > 0$, $B'_\ell = Hom(U^\ell, U)$ (where $U^0 = T$ and $U^{\ell+1} = U^\ell \times U$); (b) compositions and projections have evident

definitions; (c) for application operator, we have that $\mathbf{app}_\ell^{\mathbf{B}'} \in B'_{\ell+2}$ is given by $\mathbf{app}_\ell^{\mathbf{B}'} = eval \circ (((p \circ \pi_2) \times id) \circ ((e \circ \Lambda_\ell) \times (e \circ \Lambda_\ell)))$ and (d) $\lambda_\ell^{\mathbf{B}'} : B'_{\ell+1} \to B'_\ell$ is given by $\lambda_\ell^{\mathbf{B}'}(v) = e \circ \Lambda_\ell^{\mathcal{C}}(v)$ where $v \in Hom(U^{\ell+1}, U)$, and $\Lambda_\ell^{\mathcal{C}}$ is *currying* and $\Lambda_\ell^{\mathcal{C}} : Hom(U^{\ell+1}, U) \to Hom(U^n, U^U)$.

Nevertheless, we are not very clear of the reason why this is the case at present stage. But, it does suggest us to connect iBAs (and/or eBAs) directly with Category Theory instead of going through intentional (and/or extensional) β-algebras and/or CCCs. By doing so, it may be helpful in understanding the reason of the above breakdown (see Subsection 1.4.2 for a possible future research direction).

Chapter 10

Equational Theories of Finite CCS with Data-Dependency

In this chapter, we are going to present equational characterizations of strong bisimulation and weak bisimulation of finite CCS with data-dependency in Framework for BOs. Similar work but without *data-dependency* have been done in [69,68,162].

The main difference between the work here and [68,162] is the presence of value-passing and **if**-statements. Perhaps, some people might think that **if**-statements can be reduced to an *infinite sum*. This idea turns out to be naive, see section 10.2 (although an infinite sum is not allowed syntactically). Therefore, the new feature (value-passing with **if**-statements) is not a trivial extension of the work in [68,162] because the feature makes bisimulations depending on data (data-dependency).

The difference between the work here and [69] is the definition of observational equivalence. For example, let $\sim^{hp}$ be Hennessy and Plotkin's observational equivalence generated by $\sqsubseteq$ in [69] and $\sim$ be the observational equivalence defined in Section 10.1 below, then we would have that

(a) $(\alpha?x.t + \alpha?x.u) \sim^{hp} \alpha?x.(\mathbf{eq}(x,0) \supset (t,u)) + \alpha?x.(\mathbf{eq}(x,0) \supset (u,t))$,

(b) but $(\alpha?x.t + \alpha?x.u) \not\sim \alpha?x.(\mathbf{eq}(x,0) \supset (t,u)) + \alpha?x.(\mathbf{eq}(x,0) \supset (u,t))$;

where **eq** is a predicate which means "being equal" and $b \supset (t,u)$ is an abbreviation of **if** b **then** t **else** u.

Philosophically, (a) means that not only the behaviour on communicating ports is observable, but also the messages (or values) carried through the ports are observable. And the observability is based on both with no priority. (b) has observability on both as well, but it puts a priority on the former. That is, the behaviour on ports has the decisive factor in influencing observations. The associated values passing through ports have a secondary influence on observations. This can be better phrased as "observability + causality".

Since (a) ignores causality completely, (b) implies (a), i.e. for every two terms t and u, if $t \sim u$ then $t \sim^{hp} u$. Syntactically, the negligence of causality has to be achieved by an ω-rule (for $\sim^{hp}$) as demonstrated in [69].

Interestingly, for $\sim$ we do not have (a) but have (c) below.

(c) $(\alpha?x.t + \alpha?x.u) \sim \alpha?x.((\mathbf{eq}(x,0) \supset (t,u)) + (\mathbf{eq}(x,0) \supset (u,t)))$.

Finite CCS will be introduced briefly in the first section (Section 10.1). Then, we will provide an equational characterization of strong bisimulation (with silent τ actions observable) in Section 10.2. Weak bisimulation (with silent τ action unobservable) is complicated, but we are able to give an equational characterization in Section 10.3. A review is provided in Section 10.4.

10.1 CCS

The signature Σ^{ccs} of pure CCS contains a collection of sorts and a collection of operations (i.e. BOs). More specifically, Σ^{ccs} has the following.

(i) sorts:

(i.a) m — the sort for *messages* in communication;

(i.b) a — the sort for *agents* in communication;

(i.c) *bool* — the sort for truth values. And

(ii) operations:

(ii.a) $\tau \in \Sigma_{\varepsilon,a\to a}$;

(ii.b) $Nil \in \Sigma_{\varepsilon\to a}$ (or Σ_a);

(ii.c) $Plus, Par \in \Sigma_{\varepsilon,a^2\to a}$ (or $+, | \in \Sigma_{a\times a\to a}$ respectively);

(ii.d) **out**-$\alpha \in \Sigma_{\varepsilon,m\times a\to a}$, where $\alpha \in Lab$ and Lab is the collection of communicating ports;

(ii.e) **in**-$\alpha \in \Sigma_{(m\Rightarrow a)\to a}$, where $\alpha \in Lab$;

(ii.f) **if** $\in \Sigma_{\varepsilon,bool\times a^2\to a}$ (or **if**$_{-1}$**then**$_{-2}$**else**$_{-3}$, and $_{-1} \supset (_{-2}, _{-3})$).

Let x, y, z range over V_m, f range over $FV_{m\Rightarrow a}$ and w range over V_a. Also, let us fix a boolean language for the present chapter, i.e. we consider a toy boolean language $E_{\mathbf{eq}}$ with one predicate **eq**. Its abstract syntax is given in BNF as following.

(iii) $E ::= \mathbf{True} \mid \mathbf{False} \mid \mathbf{eq}(x, y)$,

where $x, y \in V_m$. Sometimes we denote E as $E_{\mathbf{eq}}$ for convenience. That is, Σ^{bool} contains $\mathbf{True}, \mathbf{False} \in \Sigma^{bool}_{\varepsilon\Rightarrow bool}$ and $\mathbf{eq} \in \Sigma^{bool}_{\varepsilon,m^2\Rightarrow m}$. With regards to the expressive power of **if**-statements and E, we do not need logical connectives like negation ($\neg$), conjunction ($\wedge$) and disjunction ($\vee$) since

(iv.a) $(\mathbf{if}\ \neg b\ \mathbf{then}\ t\ \mathbf{else}\ u) = (\mathbf{if}\ b\ \mathbf{then}\ u\ \mathbf{else}\ t)$,

(iv.b) $(\mathbf{if}\ b_1 \wedge b_2\ \mathbf{then}\ t\ \mathbf{else}\ u) = (\mathbf{if}\ b_1\ \mathbf{then}(\mathbf{if}\ b_2\ \mathbf{then}\ t\ \mathbf{else}\ u)\mathbf{else}\ u))$, and

(iv.c) $(\mathbf{if}\ b_1 \vee b_2\ \mathbf{then}\ t\ \mathbf{else}\ u) = (\mathbf{if}\ b_1\ \mathbf{then}\ t\ \mathbf{else}(\mathbf{if}\ b_2\ \mathbf{then}\ t\ \mathbf{else}\ u))$.

Similar to [69], we consider communication capabilities (or communication messages). For this purpose, let us assume Nat as the values of proper messages, where $\perp$ means the empty message and set $Nat_\perp = Nat \cup \{\perp\}$, and we extend the above signature to include $\mathbf{n} \in \Sigma^{msgs}_{\varepsilon\to m}$ (or Σ_m) for all natural number $n \in Nat$ and $\perp \in \Sigma^{msgs}_{\varepsilon\to m}$ (i.e. every message is definable including the empty message). So, the complete signature of CCS language being considered in this chapter is $\Sigma^{ccs} \cup \Sigma^{bool} \cup \Sigma^{msgs}$.

For readability, we write $\alpha!e.t$ instead of $\mathbf{out}\text{-}\alpha(e,t)$, and $\alpha?x.t$ instead of $\mathbf{in}\text{-}\alpha(\langle x : t\rangle)$, where e is a data-expression, say simply assuming $e \in \mathbf{Nat}_\perp \cup V_m$, where $\mathbf{Nat}_\perp = \{\ \mathbf{n} \mid n \in Nat\} \cup \{\perp\}$.

Later we refer t as a *pure* CCS term if $Free_a(t) = \emptyset$. We also define a "*depth*" function *depth* over terms and function terms in the obvious way.

In what follows, we are going to give an operational semantics for CCS by following Plotkin's [122]. Since we are interested not only in agents' behaviour over communicating ports but also in the messages carried through the ports, the operational semantics to be presented has to take evaluation environments into account. Informally, we are to define a transition system $\longrightarrow$ on the terms $T_a(V_m \cup FV_{m\Rightarrow a})$, or T_a for short, of CCS over evaluation *Eval*, i.e.

$$\longrightarrow : (T_a \times Eval) \times (T_a \times Eval),$$

where *Eval* contains the function space $V_m \rightarrow Nat_\perp$ and interpretations for all operations in E. According to convention, we write

$$t\ [\![\bullet]\!]^{src} \xrightarrow{\Xi} [\![\bullet]\!]^{tgt}\ u$$

for

$$< t, [\![\bullet]\!]^{src} > \xrightarrow{\Xi} < u, [\![\bullet]\!]^{tgt} >,$$

and it means that under a *source evaluation* $[\![\bullet]\!]^{src}$, t can be transformed to u with a *target evaluation* $[\![\bullet]\!]^{tgt}$ by performing an action Ξ, where Ξ is an instance of either $\alpha!n$, or $\alpha?x$, or τ. Since under normal circumstances source evaluations are always known by contexts, we will often omit them and only write down target evaluations, even sometimes neglect the target evaluations if they happen to be the same as the corresponding source evaluations. Of course, we will not make such omissions when they lead to confusion.

In what follows, we provide an operational semantics for CCS with data-dependency by following Plotkin's [122] (this operational semantics has a denotational semantics in [152], which is based on self-independent Petri Nets).

Definition 10.1.1 (operational semantics): *Given a (source) evaluation* $[\![\bullet]\!]$ *on* e *(expression for output) and* b *(boolean expression) such that* $[\![\mathbf{n}]\!] = n$ *for*

$\mathbf{n} \in \mathbf{Nat}$, $[\![\bot]\!] = \bot$, $[\![x]\!] \in Nat_\bot$ *for* $x \in V_m$, $[\![\mathbf{True}]\!] = true$ *and* $[\![\mathbf{False}]\!] = false$, *and we assume that for all* b *either* $[\![b]\!] = true$ *or* $[\![b]\!] = false$. *Then, we have the following:*

1. *Nil has no transition*

2. $\alpha!e.t \xrightarrow{\alpha![\![e]\!]} t$,

   where source and target evaluations share a same $[\![\bullet]\!]$;

3. $\alpha?x.t \longrightarrow^{\alpha?y}_{[\![\bullet]\!][\bot/y]} t\,[x := y]$,

   where y *is not in* $Free(\alpha?x.t)$;

4. $\tau.t \longrightarrow^{\tau}_{[\![\bullet]\!]} t$,

   where source and target evaluations are the same $[\![\bullet]\!]$;

5. *(sum)*

$$\frac{t_i \longrightarrow^{\Xi}_{[\![\bullet]\!]} t_i'}{(t_1 + t_2) \longrightarrow^{\Xi}_{[\![\bullet]\!]} t_i'},$$

   where $(t_1 + t_2)$ *is short for* $Plus(t_1, t_2)$, *both of the top transition and the bottom transition share the same source evaluation (omitted), and a same target evaluation* $[\![\bullet]\!]$; $\Xi \in Com_\tau$, *and* $Com_\tau =_{df} Com \cup \{\tau\}$ *in which* $Com = Input \cup Output$, $Input = \{\alpha?y | \alpha \in Lab \,\&\, y \in V_m\}$ *and* $Output =_{df} \{\alpha!n | \alpha \in Lab \,\&\, n \in Nat_\bot\}$;

6. *(par)*

   (a)

$$\frac{t \xrightarrow{\alpha![\![e]\!]}_{[\![\bullet]\!]} t'}{(t|u) \xrightarrow{\alpha![\![e]\!]}_{[\![\bullet]\!]} (t'|u)} \quad and \quad \frac{u \xrightarrow{\alpha![\![e]\!]}_{[\![\bullet]\!]} u'}{(t|u) \xrightarrow{\alpha![\![e]\!]}_{[\![\bullet]\!]} (t|u')}$$

   where both of the top transition and the bottom transition shares a same source evaluation (omitted) and a same target evaluation $[\![\bullet]\!]$ *for each inference rule;*

(b)

$$\frac{t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'}{(t|u) \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} (t'[y := z]|u)}, \text{ and } \frac{u \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} u'}{(t|u) \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} (t|u'[y := z])},$$

where $(t|u)$ means $Par(t,u)$, z is not free in t and u, and $[\![\bullet]\!]$ is the source evaluation.

(c)

$$\frac{t \xrightarrow{\tau}_{[\bullet]} t'}{(t|u) \xrightarrow{\tau}_{[\bullet]} (t'|u)} \text{ and } \frac{u \xrightarrow{\tau}_{[\bullet]} u'}{(t|u) \xrightarrow{\tau}_{[\bullet]} (t|u')}$$

(d)

$$\frac{t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t';\ u \xrightarrow{\alpha!n}_{[\bullet]} u'}{(t|u) \xrightarrow{\tau}_{[\bullet]} (t'[y := \mathbf{n}]|u')} \text{ and } \frac{u \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} u';\ t \xrightarrow{\alpha!n}_{[\bullet]} t'}{(t|u) \xrightarrow{\tau}_{[\bullet]} (t'|u'[y := \mathbf{n}])}$$

where $[\![\bullet]\!]$ is the source evaluation, $n = [\![e]\!]$ ($\in Nat_\perp$, notice that n can be $\perp$ in here).

7. **(if)**

$$\frac{t \xrightarrow{\Xi}_{[\bullet]'} t';\ [\![b]\!] = true}{(\textbf{if } b \textbf{ then } t \textbf{ else } u) \xrightarrow{\Xi}_{[\bullet]'} t'} \text{ and } \frac{u \xrightarrow{\Xi}_{[\bullet]'} u';\ [\![b]\!] = false}{(\textbf{if } b \textbf{ then } t \textbf{ else } u) \xrightarrow{\Xi}_{[\bullet]'} u'}$$

where $[\![\bullet]\!]$ is the source evaluation and $[\![\bullet]\!]'$ is the target evaluation.

Note that 6(d) of Definition 10.1.1 is not defined as below, and readers are invited to think of the reason why

$$\frac{t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t';\ u \xrightarrow{\alpha!n}_{[\bullet]} u'}{(t|u) \xrightarrow{\tau}_{[\bullet][n/y]} (t'|u')} \text{ and } \frac{u \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} u';\ t \xrightarrow{\alpha!n}_{[\bullet]} t'}{(t|u) \xrightarrow{\tau}_{[\bullet][n/y]} (t'|u')},$$

where $n \in Nat_\perp$ (hint: variable clashing and considering the 3rd item and the 6(d) together with possible substitution in mind).

The above transition system has a property of non-changing evaluation environments by all actions perhaps except input actions. This property will be exploited in definitions of strong and weak observational equivalences (or bisimulations).

Since we have to consider the value of a free variable, the original observational equivalence (or *bisimulation*), which does not take messages into account, has to be modified. We offer a new definition. Define a functional $\mathcal{O}_\tau$

$$\mathcal{O}_\tau : (Eval \to \mathcal{P}(T_a \times T_a)) \to (Eval \to \mathcal{P}(T_a \times T_a))$$

so that for any $R : Eval \to \mathcal{P}(T_a \times T_a)$, $t\mathcal{O}_\tau(R)([\![\bullet]\!])u$ iff

1. • if $t \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t'$ then $\exists u'.u \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} u'$ and $\exists x \notin (Free(t') - \{y\}) \cup (Free(u') - \{z\}), \forall c \in Nat_\perp.t'[y := x]R([\![\bullet]\!][c/x])u'[z := x]$
   • if $t \xrightarrow{\alpha![c]}_{[\bullet]} t'$ then $\exists u'.u \xrightarrow{\alpha![c]}_{[\bullet]} u'$ and $t'R([\![\bullet]\!])u'$
   • if $t \longrightarrow_{[\bullet]} t'$ then $\exists u'.u \longrightarrow_{[\bullet]} u'$ and $t'R([\![\bullet]\!])u'$.

2. • if $u \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} u'$ then $\exists t'.t \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} t'$ and $\exists x \notin (Free(t') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.t'[z := x]R([\![\bullet]\!][c/x])u'[y := x]$
   • if $u \xrightarrow{\alpha![c]}_{[\bullet]} u'$ then $\exists t'.t \xrightarrow{\alpha![c]}_{[\bullet]} t'$ and $t'R([\![\bullet]\!])u'$
   • if $u \longrightarrow_{[\bullet]} u'$ then $\exists t'.t \longrightarrow_{[\bullet]} t'$ and $t'R([\![\bullet]\!])u'$

It is easy to check that $\mathcal{O}$ is monotonic and so it has a (maximal) fixed point.

Definition 10.1.2 (observational equivalences $\sim_\omega$, and first bisimulation $\sim^1$ and second bisimulation $\sim^2$): *Given $[\![\bullet]\!]$, let $\sim_0 ([\![\bullet]\!])$ be $T_a \times T_a$. Then, observational equivalence $\sim_\omega ([\![\bullet]\!])$ is $\bigcap_{k \in Nat} \sim_k ([\![\bullet]\!])$, where $\sim_{k+1} ([\![\bullet]\!]) =_{df} \mathcal{O}_\tau(\sim_k)([\![\bullet]\!])$.*

1. *$t \sim^1 u$ iff for every $[\![\bullet]\!]$, $t \sim_\omega ([\![\bullet]\!])u$; sometimes we will simply denote $\sim^1$ as $\sim_\omega$.*

2. *$t \sim^2 u$ iff for all $[\![\bullet]\!]$,*

   (a) • *if $t \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t'$ then $\exists u'.u \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} u'$ and $\exists x \notin (Free(t') - \{y\}) \cup (Free(u') - \{z\}), \forall c \in Nat_\perp.t'[y := x] \sim_\omega ([\![\bullet]\!][c/x])u'[z := x]$*
   • *if $t \xrightarrow{\alpha![c]}_{[\bullet]} t'$ then $\exists u'.u \xrightarrow{\alpha![c]}_{[\bullet]} u'$ and $t' \sim_\omega ([\![\bullet]\!])u'$*
   • *if $t \longrightarrow_{[\bullet]} t'$ then $\exists u'.u \longrightarrow_{[\bullet]} u'$ and $t' \sim_\omega ([\![\bullet]\!])u'$.*

   (b) • *if $u \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} u'$ then $\exists t'.t \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} t'$ and $\exists x \notin (Free(t') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.t'[z := x] \sim_\omega ([\![\bullet]\!][c/x])u'[y := x]$*
   • *if $u \xrightarrow{\alpha![c]}_{[\bullet]} u'$ then $\exists t'.t \xrightarrow{\alpha![c]}_{[\bullet]} t'$ and $t' \sim_\omega ([\![\bullet]\!])u'$*
   • *if $u \longrightarrow_{[\bullet]} u'$ then $\exists t'.t \longrightarrow_{[\bullet]} t'$ and $t' \sim_\omega ([\![\bullet]\!])u'$*

Since we are not considering infinite sum terms, and then the existential quantifier in above definition (Definition 10.1.2) has no infinite possible choices available,

the two equivalences $\sim^1$ and $\sim^2$ have to be the same by the syntactic restriction of finiteness in this chapter. That is,

Fact 10.1.3 (coincidence of $\sim^1$ and $\sim^2$): *$t \sim^1 u$ iff $t \sim^2 u$, under the assumption of no infinite sum terms.*

Fact 10.1.3 is important in axiomatizing strong bisimulation since it allows us to do structural induction over terms. This fact also explains why we need $\sim^1$ and $\sim^2$. However, whether this fact is valid only because of the finiteness is remained to be seen. Nevertheless, we are interested in the terms which can not be observed differently in any context. So, we introduce *contexts* first as the following.

Definition 10.1.4 (contexts): *Let [ ] be a special variable (i.e. identity context) and we define contexts $Cont^{ccs}$ for CCS as below*

1. $[\,] \in Cont_a^{ccs}$

2. $$\frac{C_i \in Cont_a^{ccs}}{(C_1 + C_2) \in Cont_a^{ccs}}$$

3. $$\frac{C_i \in Cont_a^{ccs}}{(C_1 | C_2) \in Cont_a^{ccs}}$$

4. $$\frac{C \in Cont_a^{ccs};\ x \in V}{\mathbf{out}\text{-}\alpha(x, C) \in Cont_a^{ccs}} \ and \ \frac{C \in Cont_a^{ccs};\ \mathbf{n} \in \mathbf{Nat}}{\mathbf{out}\text{-}\alpha(\mathbf{n}, C) \in Cont_a^{ccs}}$$

5. $$\frac{C \in Cont_a^{ccs}; x \in V_m}{\langle x : C \rangle \in Cont_{m \Rightarrow a}}$$

6. $$\frac{fC \in Cont_{m \Rightarrow a}}{\mathbf{in}\text{-}\alpha(fC) \in Cont_a^{ccs}}$$

Since a free variable may be bound in a context, substitution $C[t/[\,]\,]$, simply written as $C[t]$, can not be anything else but a textual substitution. Thus, strong bisimulation can be defined as follows.

Definition 10.1.5 (strong bisimulation $\sim_c^2$): *We say that t and u are strongly bisimulated, written as $t \sim_c^2 u$ or simply $t \sim_c u$, iff for all $C \in Cont_a^{ccs}$, $C[t] \sim^2 C[u]$, where $C[t]$ means $C[\,t/[\,]\,]$, textual substitution.*

Another informal way to say strong bisimulation is to suppose that all silent action τ's are observable.

Since strong bisimulation itself is complicated enough, we will restrict our attention to "closed" CCS terms. Whether "open" terms (i.e. there is free ordinary variable in the terms to be bound by contexts) have an influence on the result obtained from closed terms is subject to future research.

10.2 Equational characterization of strong bisimulation $\sim_c$

Our purpose is to find a collection $Eq_{\sim_c}$ of equations such that

$$t \sim_c u \quad \text{iff} \quad Eq_{\sim_c} \vdash t \simeq u.$$

Of course, for each $p \simeq q \in Eq_{\sim_c}$ it does not need to be closed, although we are only interested in semi-closed t, u, where p is said to be *semi-closed* iff $Free_a(p) = \emptyset$, i.e. no free agent variable at all.

By definition, we can easily get that $t \sim_c u$ implies $t \sim^2 u$ (i.e. under the identity context). To see the reversed implication hold, we need to show that $\sim^2$ is a congruence relation. This is confirmed by the following lemma.

Lemma 10.2.1 (congruence relation $\sim^2$): *$\sim^2$ is a congruence relation on the terms, i.e.*

1.

$$\frac{t \sim^2 u}{\alpha!e.t \sim^2 \alpha!e.u)}$$

2.

$$\frac{t \sim^2 u}{\alpha?x.t \sim^2 \alpha?x.u}$$

3.

$$\frac{t_i \sim^2 u_i}{(t_1 + t_2) \sim^2 (u_1 + u_2)}$$

4.

$$\frac{t_i \sim^2 u_i}{(t_1|t_2) \sim^2 (u_1|u_2)}$$

5.

$$\frac{t_i \sim^2 u_i}{(\textbf{if } b \textbf{ then } t_1 \textbf{ else } t_2) \sim^2 (\textbf{if } b \textbf{ then } u_1 \textbf{ else } u_2)}$$

Proof

By structural induction on terms. To facilitate our proof, we can introduce *Subterm* for terms as follows

1. $Subterm(Nil) =_{df} \emptyset$

2. $Subterm(\alpha!e.t) =_{df} \{t\} \cup Subterm(t)$

3. $$Subterm(\alpha?x.t) =_{df} (\bigcup_{y \in V \text{ and } y \notin Free(t)-\{x\}} (\{t\,[x := y]\} \cup Subterm(t\,[x := y])) \cup (\bigcup_{n \in Nat_\perp} (\{t\,[x := \mathrm{n}]\} \cup Subterm(t\,[x := \mathrm{n}])))$$

4. $Subterm((t + u)) =_{df} Subterm(t) \cup Subterm(u)$

5. $Subterm((t|u)) =_{df} Subterm(t) \cup Subterm(u)$

6. $Subterm((\textbf{if } b \textbf{ then } t \textbf{ else } u)) =_{df} Subterm(t) \cup Subterm(u)$

We also need to show that

(10.2.1.a) for some $[\![\bullet]\!]$, if $t \xrightarrow{\Xi}_{[\![\bullet]\!]} t'$ then $t' \in Subterm(t)$, and

(10.2.1.b) if $t \sim u$ then $t\,[y := e] \sim u\,[y := e]$ for all $e \in \mathbf{Nat}_\perp \cup V_m$.

Let us demonstrate a proof for the 2nd item and the 4th item of Lemma 10.2.1. Given $[\![\bullet]\!]$, for the 2nd item, i.e. $\alpha?x.t \xrightarrow{\alpha?y}_{[\![\bullet]\!][\perp/y]} t[x := y]$, where $y \notin Free(t) - \{x\}$, from the first item in 2.(a) of Definition 10.1.2 we need to have that there is a u' such that $\alpha?x.u \xrightarrow{\alpha?z}_{[\![\bullet]\!][\perp/z]} u'$ and $\exists x \notin (Free(t) - \{y\}) \cup (Free(u') - \{z\}), \forall c \in Nat_\perp . t[y := x] \sim_\omega ([\![\bullet]\!][c/x])u'[z := x]$. Apparently, by letting $u' = u[x := z]$ we get what we want.

For the 4th item, there are three possibilities, i.e.

(10.2.1.i) $(t_1|t_2) \xrightarrow{\alpha?z}_{[\![\bullet]\!][\perp/z]} (t_1'|t_2')$,

(10.2.1.ii) $(t_1|t_2) \xrightarrow{\alpha![e]}_{[\![\bullet]\!]} (t_1'|t_2')$, and

(10.2.1.iii) $(t_1|t_2) \xrightarrow{\tau}_{[\![\bullet]\!]} (t_1'|t_2')$.

For instance, considering (10.2.1.i), we know that there two possible cases corresponding to which rule has been used to derive it (see the 7th item in Definition 10.1.1). Either of them, i.e. $t_i \xrightarrow{\alpha?y}_{[\![\bullet]\!][\perp/y]} t_i'$, leads to $u_i \xrightarrow{\alpha?y}_{[\![\bullet]\!][\perp/y]} u_i'$. Since $t_i \sim u_i$ and (10.2.1.a), we can apply induction hypothesis on $(t_1'|t_2')$ and $(u_1'|u_2')$. When we consider (10.2.1.iii), there are four cases (see the 8th item and the 9th item in Definition 10.1.1) and we need (10.2.1.b) for the two cases corresponding to the 9th item in Definition 10.1.1. Summing up the three possibilities, by induction we get what we want. □

Therefore, from Lemma 10.2.1 we can easily verify that $\sim_c$ is the same as $\sim^2$. Further, we have that

Lemma 10.2.2 (properties of $\sim_c$): *For t, u, v, the following hold*

1. *(a) (+-ass)* $(t + (u + v)) \sim_c ((t + u) + v)$

   (b) (+-cmm) $(t + u) \sim_c (u + t)$

   (c) (+-rdc) $(t + t) \sim_c t$

   (d) (Nil-+-elim) $(t + Nil) \sim_c t$

2. *(a) (|-ass)* $(t|(u|v)) \sim_c ((t|u)|v)$

   (b) (|-cmm) $(t|u) \sim_c (u|t)$

(c) *(Nil-|-elim)* $(t|Nil) \sim_c t$

3. *(|-to-+) If* t *is* $\sum_i \alpha_i \nabla_i.t_i$ *and* u *is* $\sum_j \beta_j \#_j.u_j$, *where* $\alpha_i, \beta_j \in Lab \cup \{\tau\}$, $\nabla_i = !e_i$ *or* $?x_i$ *or* ε *(empty), and* $\#_j = !e'_j$ *or* $?y_j$ *or* ε, *then*

$$\begin{array}{rcl} & & \sum_i \alpha_i \nabla'_i.(t'_i|u) \\ (t|u) & \sim_c & +\sum_j \beta_j \#'_j.(t|u'_j) \\ & & +\sum_{\alpha_i=\beta_j \ \&\ (\nabla_i=!e_i \ \&\ \#_j=?y_j \ \vee\ \nabla_i=?x_i \ \&\ \#_j=!e'_j)} \tau.(t'_i|u'_j), \end{array}$$

where

(a) $\sum_{i=1}^{1} t_i = t_1$ *and* $\sum_{i=1}^{k+1} t_i = (\sum_{i=1}^{k} t_i + t_{k+1})$ *for* $k > 0$

(b)
- $\nabla'_i = \nabla_i$ *and* $t'_i = t_i$ *if* $\nabla_i = !e_i$ *or* ε,
- $\nabla'_i = ?x'_i$, $t'_i = t_i[x_i := x'_i]$ *and* $x'_i \notin Free(u) \cup (Free(t_i) - \{x_i\})$ *if* $\nabla_i = ?x_i$,

(c)
- $\#'_j = \#_j$ *and* $u'_j = u_j$ *if* $\#_j = !e'_j$ *or* ε,
- $\#'_j = ?y'_j$, $u'_j = u_j[y_j := y'_j]$ *and* $y'_j \notin Free(t) \cup (Free(u_j) - \{y_j\})$ *if* $\#_j = ?y_j$,

(d)
- $t'_i = t_i$ *and* $u'_j = u_j[y_j := e_i]$ *if* $\nabla_i = !e_i$ *and* $\#_j = ?y_j$,
- $t'_i = t_i[x_i := e'_j]$ *and* $u'_j = u_j$ *if* $\nabla_i = ?x_i$ *and* $\#_j = !e'_j$.

4. (a) **(if-then)** (**if** b **then** t **else** u) $\sim_c t$ *or* $(b \supset (t, u)) \sim_c t$,
where $Free_m(b) = \emptyset$ *and* $[\![b]\!] = true$.

(b) **(if-else)** (**if** b **then** t **else** u) $\sim_c u$ *or* $(b \supset (t, u)) \sim_c u$,
where $Free_m(b) = \emptyset$ *and* $[\![b]\!] = false$.

(c) *(***if***-both)* $(b \supset (t, t)) \sim_c t$

(d) **(then-***forward***)** $(b_1 \supset (b_2 \supset (u_1, u_2), t)) \sim_c (b_2 \supset (b_1 \supset (u_1, t), b_1 \supset (u_2, t)))$

(e) **(else-***forward***)** $(b_1 \supset (t, (b_2 \supset (u_1, u_2)))) \sim_c (b_2 \supset (b_1 \supset (t, u_1), b_1 \supset (t, u_2)))$

(f) **(then-***cut***)** $(b \supset (b \supset (t_1, t_2), u)) \sim_c (b \supset (t_1, u))$

(g) **(else-***cut***)** $(b \supset (t, b \supset (u_1, u_2))) \sim_c (b \supset (t, u_2))$

5. *(**if**-forward)*

   (a) *(!-forward)* $\alpha!e.(b \supset (u_1, u_2)) \sim_c (b \supset (\alpha!e.u_1, \alpha!e.u_2))$

   (b) *(τ-forward)* $\tau.(b \supset (u_1, u_2)) \sim_c (b \supset (\tau.u_1, \tau.u_2))$

   (c) *(?-forward)* $\alpha?x.(b \supset (u_1, u_2)) \sim_c (b \supset (\alpha?x.u_1, \alpha?x.u_2))$, *where* $x \notin Free_m(b)$

   (d) *(+-forward)* $(t + (b \supset (u_1, u_2)) \sim_c (b \supset (t + u_1, t + u_2))$

   (e) *(|-forward)* $(t|(b \supset (u_1, u_2)) \sim_c (b \supset (t|u_1, t|u_2))$

Proof By structural induction on context $C \in Cont$ □

If you observe the above lemma (Lemma 10.2.2), especially the collection of pairs in item 5 of the above, you may notice that **if**-statements can universally move forward syntactically without changing $\sim_c$, perhaps with a slight side-condition on (?-forward). We exploit this phenomenon below and introduce a concept of *normal forms* for CCS terms. To simplify the matter, we can repeatly apply laws in (**if**-forward), which will moves every **if**-statement forward as much as possible in CCS terms and this movement will preserve strong bisimulation. And then, we also repeatly apply (|-to-+) to eliminate the appearances of | operations in CCS terms, and this elimination will preserve strong bisimulation as well. Hence, we reach a collection of CCS terms which are |-free (i.e. there is no appearance of | operations in CCS terms). Now, for these |-free terms, we introduce semi-normal form as follows.

Definition 10.2.3 (semi-normal form or semi-nf): *For |-free CCS terms, we define the semi-normal forms (or semi-nf) as follows*

1. *Nil is semi-nf;*

2. *$\alpha!e.t$ is semi-nf if t is semi-nf and $t \notin IF$,*

   *where $t \in IF$ iff there are t_1, t_2, b such that $t =$ (**if** b **then** t_1 **else** t_2);*

3. *$\tau.t$ is semi-nf if t is semi-nf and $t \notin IF$;*

4. *$\alpha?x.t$ is semi-nf if t is semi-nf, and $t \in IF(x)$ or $t \notin IF$,*

where $t \in IF(x)$ *iff there are* t_1, t_2, b *such that* $t =$ (**if** b **then** t_1 **else** t_2), $x \in Free_m(b)$;

5. $(t_1 + t_2)$ *is semi-nf if* t_i *is semi-nf* $(i = 1,2)$, $t_1 \neq t_2$, $t_i \neq Nil$ *and* $t_i \notin IF$ $(i = 1,2)$;

6. (**if** b **then** t_1 **else** t_2) *is semi-nf if* t_i *is semi-nf* $(i = 1,2)$, $t_1 \neq t_2$, *and* $Free_m(b) \neq \emptyset$.

As a summary of Definition 10.2.3, we understand that if t is semi-nf then either t is Nil, a sum-form $\sum_{i=1}^{k} \alpha_i \nabla_i t_i$, or a (nested) **if**-statement (**if** b **then** t **else** u), where $Free_m(b) \neq \emptyset$. We can further define a *reduction system* on semi-closed CCS terms as follows.

(i) reductions obtained from 1(c), 1(d), 2(c), 3, 4(a) to 4(g), and 5(a) to 5(e) in Lemma 10.2.2 by replacing $\sim_c$ with $\rhd$; and

(ii) reductions obtained from 1(a), 1(b), 2(a), and 2(b) by replacing $\sim_c$ with $\bowtie$; where $\bowtie$ means $\rhd$ for both directions.

Observing the reduction related to (τ-forward) we will learn that **if-then-else** is quite different from $+$ operator. For example, we never have $\tau.(t+u) \sim^2 (\tau.t + \tau.u)$ in general. This makes it clear that a **if-then-else** can never be reduced to a sum expression in general, even if an infinite sum is introduced.

The definition of semi-normal forms is not usual, i.e. it does not designate as that t is a normal form iff there is no redex in t. In other words, so-called *strong normalization* property does not hold for such a reduction system. This non-strong normalization property is a simple result of the reductions related from (**if**-both) to (**else**-cut). For example, we would have

$$(10.2.*)\quad b_1 \supset (b_2 \supset (t_1, t_2), b_2 \supset (u_1, u_2)) \bowtie b_2 \supset (b_1 \supset (t_1, u_1), b_2 \supset (t_2, u_2))$$

That is, $\rhd$ is not *noetherian* (or not strong normalizable).

Apparently, if $\rhd$ (and/or $\bowtie$) is replaced by $\sim_c$, then the reductions above are valid.

Lemma 10.2.4 (semi-normalizability): *For each |-free semi-closed t, there exists |-free u such that $t \rhd^* u$ and u is semi-nf.*

Proof

By induction on ℓ of $T_a^{(\ell)}$ with case analysis and be aware of that if $t \rhd t'$ and $t \in T_a^{(\ell)}$ implies $t' \in T_a^{(\ell)}$, where t and t' are |-free.

For $\ell = 0$, there is only one case, i.e. $t = Nil$ and it is semi-nf.

For $\ell > 0$, there are six cases for t:

(10.2.4.i) $\alpha!e.t'$,

(10.2.4.ii) $\alpha?x.t'$,

(10.2.4.iii) $\tau.t'$,

(10.2.4.iv) $(t_1 + t_2)$,

(10.2.4.v) **if** b **then** t_1 **else** t_2.

We are only to show three cases (10.2.4.ii) and (10.2.4.v) and the rest cases are left out.

For (10.2.4.ii), by induction hypothesis we have $t' \rhd^* u'$ and u' is semi-nf. If u' is not an **if**-statment, then $t \rhd^* \alpha?x.u'$ and $\alpha?x.u'$ is semi-nf. Suppose u' is an **if**-statment, i.e. $u' =$ **if** b' **then** u'_1 **else** u'_2. If $u' \in IF(x)$, i.e. $x \in Free_m(b')$, then $t \rhd^* \alpha?x.u'$ and $\alpha?x.u'$ is semi-nf. If $u' \notin IF(x)$, then by reduction 5(c) in Lemma 10.2.2 we have that $\alpha?x.u' \rhd$ **if** b' **then** $\alpha?x.u'_1$ **else** $\alpha?x.u'_2$. Further since $Free_m(b')$ is not $\emptyset$ (otherwise u' is not a semi-nf), by induction hypothesis $\alpha?x.u'_i \rhd^* u''_i$ and u''_i is semi-nf. Now, if $u''_1 = u''_2$ then by 4(c) in Lemma 10.2.2, we have $\alpha?x.u' \rhd^* \alpha?x.u''_1$ which is a semi-nf. Further suppose $u''_1 \neq u''_2$, we have that **if** b' **then** $\alpha?x.u'_1$ **else** $\alpha?x.u'_2 \rhd^*$ **if** b' **then** $\alpha?x.u''_1$ **else** $\alpha?x.u''_2$ which is semi-nf.

For (10.2.4.v), if $Free_m(b) = \emptyset$, then either by reduction 4(a) in Lemma 10.2.2 we have $t \rhd t_1$ and apply induction hypothesis to t_1 or by reduction 4(b) in Lemma 10.2.2 we have $t \rhd t_2$ and apply induction hypothesis to t_2. □

We understand that **if**-statements will cluster together in semi-nf, which may be prefixed by relevant inputs (i.e. a certain kind of binding). But for nested **if**-statments, this need further clarification.

if-statements are neither just syntactic sugar nor easy to tackle with in semi-nf, without mention of that binding happens at the same time as well. The reason for this can be found in some work relating to axiomatization of **if**-statements (without binding) in [17,57,104]. Their work are not data-dependent, which is the major difference of our work below with theirs on axiomatization of **if**-statements.

Firstly, we define the collection $path([\![\bullet]\!], t)$ of all possible path for **if**-statements of a (semi-nf) term t under an evaluation environment $[\![\bullet]\!]$. Formally, given an evaluation $[\![\bullet]\!]$,

$$path([\![\bullet]\!], t) =_{df} \begin{cases} \varepsilon & if\ t \notin IF \\ 1\, path([\![\bullet]\!], t_1) & if\ t = \textbf{if}\ b\ \textbf{then}\ t_1\ \textbf{else}\ t_2\ and\ [\![b]\!] = true \\ 2\, path([\![\bullet]\!], t_2) & if\ t = \textbf{if}\ b\ \textbf{then}\ t_1\ \textbf{else}\ t_2\ and\ [\![b]\!] = false. \end{cases}$$

Secondly, we need to know all possible path $paths(t)$ of a term t without involving an evaluation environment as well, i.e.

$$paths(t) =_{df} \{\varepsilon\} \cup \{1\, l_1, 2\, l_2 |\, t = (\textbf{if}\ b\ \textbf{then}\ t_1\ \textbf{else}\ t_2)\ and\ l_i \in paths(t_i)\}$$

Thirdly, we are interested in knowing what is the subterm, $subterm(p, t)$, of a term t under its path p. Namely, for $p \in paths(t)$,

$$subterm(p, t) =_{df} \begin{cases} t & if\ p = \varepsilon (and\ t \notin IF) \\ subterm(p_1, t_1) & if\ p = 1\, p_1\ and\ t = (\textbf{if}\ b\ \textbf{then}\ t_1\ \textbf{else}\ t_2) \\ subterm(p_2, t_2) & if\ p = 2\, p_2\ and\ t = (\textbf{if}\ b\ \textbf{then}\ t_1\ \textbf{else}\ t_2) \end{cases}$$

Lastly, we have to be aware of all possible boolean expressions $premise(p, t)$ involved in a term t along its path p. Precisely, let $p \in paths(t)$,

$$premise(p, t) =_{df} \begin{cases} \emptyset & if\ p = \varepsilon (and\ t \notin IF) \\ \{b\} \cup subterm(p_1, t_1) & if\ p = 1\, p_1\ and\ t = b \supset (t_1, t_2) \\ \{\neg b\} \cup premise(p_2, t_2) & if\ p = 2\, p_2\ and\ t = b \supset (t_1, t_2) \end{cases}$$

With these, we can introduce *concise* semi-nf below. A semi-nf t is said to be *concise* either if it does not contain if-statment or if it contains a subterm $\alpha?x.\mathbf{if}\ b\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2$ then $x \in Free_m(b)$ and both $\alpha?x.t_1$ and $\alpha?x.t_2$ are concise.

Clearly, the above definitions are more interesting if we can connect them with strong bisimulation $\sim_c$ (or $\sim^2$). We give one as follows.

Property 10.2.5 ($\sim_\omega$ **and** *if*-**statements**): *For CCS terms t and u, when $t \sim^2 u$ we have that for all $[\![\bullet]\!]$,*

$$subterm(path([\![\bullet]\!], t), t) \sim_\omega ([\![\bullet]\!])subterm(path([\![\bullet]\!], u), u).$$

Proof

$t \sim_\omega ([\![\bullet]\!])subterm(path([\![\bullet]\!], t), t)$ and $u \sim_\omega ([\![\bullet]\!])subterm(path([\![\bullet]\!], u), u)$. □

Informally, considering semi-normal forms without loss of generality, we understand that Property 10.2.5 gives a nice evidence of showing the role of **if**-statements without binding, i.e. if no binding is available then the role of **if**-statements can be eliminated.

Now, we are considering boolean expressions, since some expressions are semantic equivalent to each other. We will take certain restriction on these expressions. For example, no such b_1 and b_2 is allowed to appear in a semi-normal CCS term like $b_1 \supset (b_2 \supset (t_1, t_2), u)$ where $[\![b_1]\!] = true$ always implies $[\![b_2]\!] = false$ for all evaluation $[\![\bullet]\!]$, i.e. the branch t_2 in $b_1 \supset (b_2 \supset (t_1, t_2), u)$ is not *reachable* in any way. Therefore, a CCS term containing non-reachable sub-terms is not desirable. Also, for boolean expressions, the *independency* of their representatives would help us to obtain a proof system with syntactic compositionality. So, we formalize the above reachability and independency as a theorem as follows.

Theorem 10.2.6 ($\mathcal{E}_\tau$): *Let $\alpha?x.t$ be $|$-free semi-nf. Thus, if t is an* **if**-*statement then we have the following.*

1. *(reachability) There is another concise semi-nf t' such that $t \sim_c t'$ and for $p \in paths(t')$, and for all $[\![\bullet]\!]$, we have that $p \neq \varepsilon$ implies $x \in \bigcap_{b \in premise(p,t')} Free_m(b)$*

and $\exists c \in Nat_\perp$, $\forall b \in premise(p, t').[\![b]\!]' = true$, *where* $[\![\bullet]\!]' = [\![\bullet]\!][c/x]$. *Sometimes, we said that* $subterm(path([\![\bullet]\!][c/x], t'), t')$ *is* x*-reachable.*

2. *(independency) Let* u *be another concise semi-nf and* $t \sim_c u$. *If* $t = \mathbf{if}\ b\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2$ *and* $u = \mathbf{if}\ b'\ \mathbf{then}\ u_1\ \mathbf{else}\ u_2$, *then either that* b' *is (semantic) equivalent representative of* b *(written as* $b \leftrightarrow b'$*) or for some* i *there is a* $p_i \in paths(u_i)$ *and* $b \in premise(p_i, u_i)$.

The proof of Theorem 10.2.6 is postponed to Section 10.4 (see Theorem 10.4.6).

Note that combined Lemma 10.2.4 and Theorem 10.2.6, we can improve the result of Lemma 10.2.4, i.e. we can have a refined version of Lemma 10.2.4 as "for every term t there is a semi-nf u such that $t \rhd^* u$ and u has the reachable property and independence property in Theorem 10.2.6". For such kind of u, we will refer it as *refined* semi-nf.

With this Theorem 10.2.6, we are able to isolate CCS terms from **if**-statements. By doing so, we are able to yield a complete characterization of $\sim_c$. Basically, the completeness will be independent of individual boolean language as long as the language satisfies the properties in Theorem 10.2.6.

Theorem 10.2.7 ($\sim^1$): *Let* t, u *be concise refined semi-nfs and given* $[\![\bullet]\!]$, *we have that if* $t \sim_\omega ([\![\bullet]\!])u$ *then*

$$subterm(path([\![\bullet]\!], t), t) \sim^1 subterm(path([\![\bullet]\!], u), u).$$

Proof

By induction on the sum of depths of t and u with Fact 10.1.3.

(i) case $depth(t) + depth(u) = 0$: it is a trivial case.

(ii) case $depth(t) + depth(u) > 0$: let $t' = subterm(path([\![\bullet]\!], t), t)$ and $u' = subterm(path([\![\bullet]\!], u), u)$,

- if $t' \xrightarrow{\alpha![\epsilon]}_{[\![\bullet]\!]} t''$, $\exists u''.u' \xrightarrow{\alpha![\epsilon]}_{[\![\bullet]\!]} u''$ and $t'' \sim_\omega ([\![\bullet]\!])u''$, then $t'', u'' \notin IF$ and by induction hypothesis, we have that $t'' \sim^1 u''$.

- if $t' \xrightarrow{\tau}_{[\bullet]} t'', \exists u''.u' \xrightarrow{\tau}_{[\bullet]} u''$ and $t'' \sim_\omega ([\![\bullet]\!])u''$, then $t'', u'' \notin IF$ and by induction hypothesis, we have that $t'' \sim^1 u''$.

- if $t' \xrightarrow{\alpha?y}_{[\bullet]} t'', \exists u''.u' \xrightarrow{\alpha?z}_{[\bullet]} u''$ and $\exists x \notin Free(t'') - \{y\} \cup Free(u'') - \{z\}$ such that for all $c \in Nat_\perp$, $t''[y := x] \sim_\omega ([\![\bullet]\!][c/x])u''[z := x]$, then by the first part (reachability) of Theorem 10.2.6, we have that

  (a) $paths(t''[y := x]) = prefix(\{path([\![\bullet]\!][c/x], t''[y := x])|c \in Nat_\perp\})$ and similarly

  (b) $paths(u''[y := x]) = prefix(\{path([\![\bullet]\!][c/x], u''[y := x])|c \in Nat_\perp\})$,

  where $prefix(S) =_{df} \{\varepsilon, s', s'', s|s \in S \wedge s'1 \in S \wedge s''2 \in S\}$.

  Since $prefix(\{path([\![\bullet]\!]', t''[y := x])|\ [\![\bullet]\!]' \in Eval\}) = paths(t''[y := x])$ and $prefix(\{path([\![\bullet]\!]', u''[z := x])|\ [\![\bullet]\!]' \in Eval\}) = paths(u''[z := x])$, we have $t''[y := x] \sim_\omega u''[z := x]$. That is, we get what we want. □

Note that $\sim^1$ is equivalent to $\sim^2$, So Theorem 10.2.7 is a very nice result. It says that if two terms in concise and refined semi-nf are observably-equivalent up to ω-level under an evaluation then the subterms of them correspondingly obtained through the evaluation are bisimulated with each other independent of any evaluation. That is, with regard to strong bisimulation the influence of evaluation (or messages communicated through ports) is restricted to (nested) **if**-statements. This observation leads to a new definition for strong bisimulation. Formally,

Definition 10.2.8 ($\sim^3$ third bisimulation): *For concise and refined semi-nfs t and u, we define $t \sim^3 u$ iff for all $[\![\bullet]\!]$,*

1. (a) *if* $t \xrightarrow{\alpha?y}_{[\bullet]} t'$ *then* $\exists u'.u \xrightarrow{\alpha?z}_{[\bullet]} u'$ *and* $\forall c \in Nat_\perp.t'[y := \mathbf{c}] \sim^2 u'[z := \mathbf{c}]$
   (b) *if* $t \xrightarrow{\alpha![\mathbf{e}]}_{[\bullet]} t'$ *then* $\exists u'.u \xrightarrow{\alpha![\mathbf{e}]}_{[\bullet]} u'$ *and* $t' \sim^2 u'$
   (c) *if* $t \xrightarrow{\tau}_{[\bullet]} t'$ *then* $\exists u'.u \xrightarrow{\tau}_{[\bullet]} u'$ *and* $t' \sim^2 u'$.

2. (a) *if* $u \xrightarrow{\alpha?y}_{[\bullet]} u'$ *then* $\exists t'.t \xrightarrow{\alpha?z}_{[\bullet]} t'$ *and* $\forall c \in Nat_\perp.t'[z := \mathbf{c}] \sim^2 u'[y := \mathbf{c}]$
   (b) *if* $u \xrightarrow{\alpha![\mathbf{e}]}_{[\bullet]} u'$ *then* $\exists t'.t \xrightarrow{\alpha![\mathbf{e}]}_{[\bullet]} t'$ *and* $t' \sim^2 u'$
   (c) *if* $u \xrightarrow{\tau}_{[\bullet]} u'$ *then* $\exists t'.t \xrightarrow{\tau}_{[\bullet]} t'$ *and* $t' \sim^2 u'$.

What the above definition (Definition 10.2.7) is trying is to bring the semantic version of strong bisimulation $\sim^2$ up to a syntactic level. Apparently, we would be interested in $\sim^3$ only if this syntactic version of strong bisimulation is tightly linked to the previous semantic one. Obviously, we have that $t \sim^3 u$ implies $t \sim^2 u$. For the reversed implication, we provide a lemma (Lemma 10.2.9) as follows.

Lemma 10.2.9 ($\sim^2$ and $\sim^3$): *Let t and u be concise and refined semi-nfs and $t \sim^2 u$. We have that given $[\![\bullet]\!]$,*

- *if $t \xrightarrow{\alpha![\![\bullet]\!]} t'$, $u \xrightarrow{\alpha![\![\bullet]\!]} u'$ and $t' \sim_\omega ([\![\bullet]\!])u'$, then $t' \sim^2 u'$;*
- *if $t \xrightarrow{\tau}_{[\![\bullet]\!]} t'$, $u'.u \xrightarrow{\tau}_{[\![\bullet]\!]} u'$ and $t' \sim_\omega ([\![\bullet]\!])u'$, then $t' \sim^2 u'$;*
- *if $t \xrightarrow{\alpha?y}_{[\![\bullet]\!][\perp/y]} t'$, $t \xrightarrow{\alpha?z}_{[\![\bullet]\!][\perp/z]} t'$ and $x \notin (Free(t') - \{z\}) \cup (Free(u') - \{y\})$ such that $\forall c \in Nat_\perp . t'[z := x] \sim_\omega ([\![\bullet]\!][c/x])u'[y := x]$, then $t'[z := x] \sim^2 u'[y := x]$.*

Proof

By induction on the sum of depths of t and u with Lemma 10.2.7. □

As a result of Lemma 10.2.9, we know that $\sim^2$ is the same as $\sim^3$. Coupled with Fact 10.1.3, we have that $t \sim^1 u$ iff $t \sim^2 u$ iff $t \sim^3 u$ for concise and refined semi-nfs t and u. This equivalence link enables us to do induction over CCS terms and to get the equational characterization of strong bisimulation $\sim_c$. Formally,

Theorem 10.2.10 (equational characterization of strong bisimulation $\sim_c$): *For terms t and u, $t \sim_c u$ iff $Eq_\tau \vdash_{ccs} t \simeq u$ without substitutions, where Eq_τ^{ccs} contains all instances of $t \simeq u$ which are instances of $t \sim_c u$ in Lemma 10.2.2 replacing $\sim_c$ by $\simeq$.*

Proof

Since every t can be reduced to a concise and refined semi-nf, we can concentrate on such concise and refined semi-nf's t and u such that $t \sim_c u$.

We assume that $t \sim_c u$ and t, u are concise and refined semi-nfs, and the proof of this theorem is based on induction at the sum of the depths of t and u with case analysis.

It is trivial if $depth(t) + depth(u) = 0$. So, let us assume $depth(t) + depth(u) > 0$ and consider all possible cases for t.

(10.2.10.a) case $t = Nil$: we know that u cannot be Nil. So, if $u = u_1 + u_2$, then $Nil \sim_c u_i$ $(i = 1, 2)$. By induction hypothesis we get $Nil \simeq u_i$ $(i = 1, 2)$. Thus, $t \simeq u$ by (Nil-elim).

(10.2.10.b) case $\alpha!e.t'$: If u is an **if**-statment, say $u = \mathbf{if}\ b\ \mathbf{then}\ u_1\ \mathbf{else}\ u_2$; by reachability of Theorem 10.2.6 we know that both u_1 and u_2 are reachable; by Theorem 10.2.7 we have $t \sim u_i$ $(i = 1, 2)$; by induction hypothesis, we have $t \simeq u_i$ $(i = 1, 2)$, i.e. $u \simeq \mathbf{if}\ b\ \mathbf{then}\ t\ \mathbf{else}\ t$;; by (**if**-both), we have $t \simeq u$.

If u is not an **if**-statment, then it must be a form of $\sum \alpha!e_j.u_j$ such that for all j, $t' \sim_c u_j$. By induction hypothesis we come to $t' \simeq u_j$ for all j. Hence, $\alpha!e.t' \simeq u$ by (+-rdc).

(10.2.10.c) case $\tau.t'$: it is similar to the above case.

(10.2.10.d) case $\alpha?x.t'$: u must be a form of $\sum \alpha?x_j.u_j$ such that $t'[x := y] \sim_c u_j[x_j := y]$ for all j where $y \notin Free(t') - \{x\} \cup \bigcup_j (Free(t_j) - \{x_j\})$ (see Lemma 10.2.9). Consequently, by induction hypothesis we have $t'[x := \chi] \simeq u_j[x_j := \chi]$ for all j. So, $\alpha?x.t' \simeq u$ by (+-rdc).

(10.2.10.e) case $\sum_i \alpha_i \nabla_i.t_i$: similar to the above three cases (i.e. from (10.2.10.b) to (10.2.10.d)), for each i, there exists $\#_{j_i}$ and u_{j_i} such that $\alpha_i \nabla_i.t_i \simeq \alpha_i \#_i.u_i$ by induction hypothesis. Then, we are considering all those possible $\#_{j_i}$ and u_{j_i} as a group. Obviously, this group can be reduced to a single one by applying (+-rdc). Similarly, after finishing this kind of reductions on u's side, we can in turn do the same kind of reductions on t's side. Consequently, we can put $\alpha_i \nabla_i.t_i$ and $\alpha_j \#_j.u_j$ into an 1-1 correspondence. Therefore, for each i and the corresponding j, we can get $\alpha_i \nabla_i.t_i \simeq \alpha_j \#_j.u_j$. So, $t \simeq u$ is the natural result of applying (b-cmp).

(10.2.10.f) case $(\mathbf{if}\ b\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2)$: there are two possible cases for u, i.e. either (10.2.10.f.i) $u \in IF$ or (10.2.10.f.ii) $u \notin IF$.

For (10.2.10.f.ii), by Lemma 10.2.7, Fact 10.1.3 and Lemma 10.2.1 with the

reachability of Theorem 10.2.6, we can have $t_i \sim_c u$ $(i = 1, 2)$. Hence, by induction hypothesis, we can get $t_i \simeq u$. Thus, by (**if**-both), we have $t \simeq u$.

For (10.2.10.f.i), let us suppose (**if** b' **then** u_1 **else** u_2). If $b = b'$ then by induction hypothesis we have $t_i \simeq u_i$ for $i = 1, 2$. Thus, $t \simeq u$ is naturally derived by (b-cmp). If $b \neq b'$ then by the independency of Theorem 10.2.6, we can get $u \simeq$ **if** b **then** u_1' **else** u_2' by the reductions from 5(c) to 5(g) in Lemma 10.2.2, which all preserve semi-nfs. This time, we return to the case $b = b'$ above.

(10.2.10.g) All rules involved function terms do not creat more equations on terms (since **in**-α is an unary Binding Operator).

As a result of the above from (10.2.10.a) to (10.2.10.g), we have the completeness of the equational characterization of strong bisimulation. □

Note that the derivability $\vdash_{ccs}$ without substitutions is equivalent to the derivability of $\vdash_{ccs}$ for semi-closed CCS terms. Whether the semi-closedness is removable or not is interesting to see.

10.3 Equational characterization of weak bisimulation $\approx_c$

Unlike strong bisimulation $\sim_c$, as pointed out by Milner in [108], weak bisimulation $\approx$ (i.e. silent action τ's are not observable) is not preserved by $+$; i.e. that $t_i \approx u_i$ $(i = 1, 2)$ does not necessarily imply $t_1 + t_2 \approx u_1 + u_2$. A well-quoted example of this kind is $Nil + a!e.Nil \not\approx \tau.Nil + a!e.Nil$. That is, $\approx$ does not preserve compositionality. On the other hand, $\approx_c$ is obviously an equivalence relation which preserves compositionality (i.e. a congruence). Therefore, the relationship between $\approx$ and $\approx_c$ becomes very interesting. For instance, $\tau.Nil \approx Nil$ but $\tau.Nil \not\approx_c Nil$. Of course, we have that $t \approx_c u$ implies $t \approx u$. So, it appears that the techniques used in last section (Section 10.2), say reducing $\sim_c$ to $\sim$, can not be used in here (Section 10.3) for weak bisimulation although we understand that $t \sim_c u$ implies $t \approx_c u$ and $t \approx u$; i.e. $\approx_c$ can not be reduced to $\approx$. Other techniques are indispensable.

Milner shows that $t \approx_c u$ iff $t \approx^+ u$, where $t \approx^+ u$ is defined as $\forall v.(t+v) \approx (u+v)$. So, $\approx^+$ is a congruence, see [108, Theorem 10.8]. This result locates the trouble spot for $\approx_c$ at $+$ operator only.

Further, Walker finds a new definition of an equivalence relation $\approx^*$. In fact, he proves that this $\approx^*$ is equivalent to $\approx^+$, where $\approx^*$ can be considered as an equivalence relation generated from $\sqsubseteq^*$ in [162]. This new definition of $\approx^*$ has an advantage over $\approx^+$ that $\approx^*$ has no relationship with contexts. It is independent of contexts.

Basically, the work to be presented in this section (Section 10.3) can be viewed of borrowing their results and developing our own techniques to tackle data-dependency.

Similar to Section 10.1, we make $\approx$ more precisely as follows. We define a functional $\mathcal{O}$ (without an index τ unlike $\mathcal{O}_\tau$ in Section 10.2).

$$\mathcal{O} : (Eval \to \mathcal{P}(T_a \times T_a)) \to (Eval \to \mathcal{P}(T_a \times T_a))$$

such that given $[\![\bullet]\!]$ and $R : Eval \to \mathcal{P}(T_a \times T_a)$, $t\mathcal{O}(R)([\![\bullet]\!])u$ iff

1. (a) if $t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$ then $\exists u'''.u \overset{\xi}{\Longrightarrow}_{[\bullet]} u'''$ and $u''' \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} u''$ and $\exists x \notin (Free(t') - \{y\}) \cup (Free(u'') - \{z\}), \forall c \in Nat_\perp.\exists u'.u'' \overset{\epsilon}{\Longrightarrow}_{[\bullet][c/z]} u'$ and $t'[y := x]R([\![\bullet]\!][c/x])u'[z := x]$

   (b) if $t \xrightarrow{\alpha![e]}_{[\bullet]} t'$ then $\exists u'.u \overset{\alpha![e]}{\Longrightarrow}_{[\bullet]} u'$ and $t'R([\![\bullet]\!])u'$

   (c) if $t \xrightarrow{\tau}_{[\bullet]} t'$ then $\exists u'.u \overset{\xi}{\Longrightarrow}_{[\bullet]} u'$ and $t'R([\![\bullet]\!])u'$.

2. (a) if $u \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} u'$ then $\exists t'''.t \overset{\xi}{\Longrightarrow}_{[\bullet]} t'''$ and $t''' \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} t''$ and $\exists x \notin (Free(t'') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.\exists t'.t'' \overset{\epsilon}{\Longrightarrow}_{[\bullet][c/z]} t'$ and $t'[y := x]R([\![\bullet]\!][c/x])u'[z := x]$

   (b) if $u \xrightarrow{\alpha![e]}_{[\bullet]} u'$ then $\exists t'.t \overset{\alpha![e]}{\Longrightarrow}_{[\bullet]} t'$ and $t'R([\![\bullet]\!])u'$

   (c) if $u \xrightarrow{\tau}_{[\bullet]} u'$ then $\exists t'.t \overset{\xi}{\Longrightarrow}_{[\bullet]} t'$ and $t'R([\![\bullet]\!])u'$

where transition system $\Longrightarrow$, say $t \overset{\Xi}{\Longrightarrow}_{[\bullet]'} u$, is defined through transition system $\longrightarrow$ as follows:

(10.3.i) $t \overset{\epsilon}{\Longrightarrow}_{[\bullet]} t$;

(10.3.ii)

$$\frac{t \ [\bullet] \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} t'; t' \ [\bullet] \stackrel{\tau}{\longrightarrow}_{[\bullet]} t''}{t \ [\bullet] \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} t''};$$

(10.3.iii)

$$\frac{t \ [\bullet] \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} t'; t' \ [\bullet] \stackrel{\Xi}{\longrightarrow}_{[\bullet]'} t''}{t \ [\bullet] \stackrel{\Xi}{\Longrightarrow}_{[\bullet]'} t''},$$

where $\Xi \neq \tau$;

(10.3.iv)

$$\frac{t \ [\bullet] \stackrel{\Xi}{\Longrightarrow}_{[\bullet]'} t'; t' \ [\bullet]' \stackrel{\tau}{\longrightarrow}_{[\bullet]'} t''}{t \ [\bullet] \stackrel{\Xi}{\Longrightarrow}_{[\bullet]'} t''},$$

where $\Xi \neq \tau$.

We can easily check that given a source $[\![\bullet]\!]$, $\mathcal{O}$ is monotonic and it has a (maximal) fixed point.

Definition 10.3.1 (first weak observational equivalence $\approx^1$ and second weak observational equivalence $\approx^2$): *Let $\approx_0 ([\![\bullet]\!])$ be $T_a \times T_a$ for each $[\![\bullet]\!]$. Then,*

(i) observational equivalence $t \approx^1 u$ (with τ unobservable) is that for all $[\![\bullet]\!]$ $t \approx_\omega ([\![\bullet]\!])u$, where $\approx_\omega ([\![\bullet]\!]) =_{df} \bigcap_{k \in Nat} \approx_k ([\![\bullet]\!])$ and $\approx_{k+1} ([\![\bullet]\!]) =_{df} \mathcal{O}(\approx_k)([\![\bullet]\!])$;

(ii) $t \approx^2 u$ iff for all $[\![\bullet]\!]$,

1. (a) *if $t \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t'$ then $\exists u'''.u \Longrightarrow^{\epsilon}_{[\bullet]} u'''$ and $u''' \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} u''$ and $\exists x \notin (Free(t') - \{y\}) \cup (Free(u'') - \{z\}), \forall c \in Nat_\perp.\exists u'.u'' \ {}_{[\bullet][c/x]} \Longrightarrow^{\epsilon}_{[\bullet][c/x]} u'$ and $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$*

   (b) *if $t \stackrel{\alpha![e]}{\longrightarrow}_{[\bullet]} t'$ then $\exists u'.u \stackrel{\alpha![e]}{\Longrightarrow}_{[\bullet]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*

   (c) *if $t \stackrel{\tau}{\longrightarrow}_{[\bullet]} t'$ then $\exists u'.u \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$.*

2. (a) *if $u \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} u'$ then $\exists t'''.t \Longrightarrow^{\epsilon}_{[\bullet]} t'''$ and $t''' \longrightarrow^{\alpha?z}_{[\bullet][\perp/z]} t''$ and $\exists x \notin (Free(t'') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.\exists t'.t'' \ {}_{[\bullet][c/x]} \Longrightarrow^{\epsilon}_{[\bullet][c/x]} t'$ and $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$*

   (b) *if $u \stackrel{\alpha![e]}{\longrightarrow}_{[\bullet]} u'$ then $\exists t'.t \stackrel{\alpha![e]}{\Longrightarrow}_{[\bullet]} t'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*

   (c) *if $u \stackrel{\tau}{\longrightarrow}_{[\bullet]} u'$ then $\exists t'.t \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} t'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*

Similar to Fact 10.1.3, we have a similar result for $\approx^1$ and $\approx^2$ below. Namely,

Fact 10.3.2 (coincidence of $\approx^1$ and $\approx^2$): *$t \approx^1 u$ iff $t \approx^2 u$, under the assumption of no infinite sum terms.*

Following [162], we introduce a notation $\stackrel{\tau}{\Longrightarrow}$ which means the transitive closure of $\stackrel{\tau}{\longrightarrow}$, or $\stackrel{\tau^+}{\longrightarrow}$. We define a new observational equivalence $\approx^*$ as follows, which is actually a congruence (to be verified in Lemma 10.3.5). Formally,

Definition 10.3.3 (weak observational congruence $\approx^*$): *$t \approx^* u$ iff for all $[\![\bullet]\!]$,*

1. - *if $t \stackrel{\alpha?y}{\longrightarrow}_{[\![\bullet]\!][\perp/y]} t'$ then $\exists u'''.u \stackrel{\xi}{\Longrightarrow}_{[\![\bullet]\!]} u'''$ and $u''' \stackrel{\alpha?z}{\longrightarrow}_{[\![\bullet]\!][\perp/z]} u''$ and $\exists x \notin (Free(t') - \{y\}) \cup (Free(u'') - \{z\}), \forall c \in Nat_\perp.\exists u'.u'' \,{}_{[\![\bullet]\!][c/z]} \stackrel{\epsilon}{\Longrightarrow}_{[\![\bullet]\!][c/z]} u'$ and $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$*
   - *if $t \stackrel{\alpha![c]}{\longrightarrow}_{[\![\bullet]\!]} t'$ then $\exists u'.u \stackrel{\alpha![c]}{\Longrightarrow}_{[\![\bullet]\!]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*
   - *if $t \stackrel{\tau}{\longrightarrow}_{[\![\bullet]\!]} t'$ then $\exists u'.u \stackrel{\tau}{\Longrightarrow}_{[\![\bullet]\!]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$.*

2. - *if $u \stackrel{\alpha?y}{\longrightarrow}_{[\![\bullet]\!]'} u'$ then $\exists t'''.t \stackrel{\xi}{\Longrightarrow}_{[\![\bullet]\!]} t'''$ and $t''' \stackrel{\alpha?z}{\longrightarrow}_{[\![\bullet]\!]''} t''$ and $\exists x \notin (Free(t'') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.\exists t'.t'' \stackrel{\epsilon}{\Longrightarrow}_{[\![\bullet]\!]''[c/z]} t'$ and $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$*
   - *if $u \stackrel{\alpha![c]}{\longrightarrow}_{[\![\bullet]\!]} u'$ then $\exists t'.t \stackrel{\alpha![c]}{\Longrightarrow}_{[\![\bullet]\!]} t'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*
   - *if $u \stackrel{\tau}{\longrightarrow}_{[\![\bullet]\!]} u'$ then $\exists t'.t \stackrel{\tau}{\Longrightarrow}_{[\![\bullet]\!]} t'$ and $t' \approx_\omega ([\![\bullet]\!])u'$*

Note that the difference between $\approx^*$ of Definition 10.3.3 and $\approx^2$ of Definition 10.3.1 is whether an unobservable change can be identifiable. The former can identify the change but not the latter. Also, there is a close relationship between the former and the latter, i.e. we can easily arrive at the conclusion of $\approx^* \subseteq \approx^2$ by checking their definitions, i.e. $t \approx^* u$ implies $t \approx^2 u$. About the reversed implication, we have the following.

Lemma 10.3.4 (equivalences between $\approx^*$ and $\approx^+$): *$t \approx^+ u$ iff $t \approx^* u$, where $t \approx^+ u$ means that for all v, $(t + v) \approx^2 (u + v)$.*

Proof

(i) For "$\Rightarrow$" (only-if direction), assuming $t \not\approx^* u$ we expect that $\exists v.(t + v) \not\approx (u + v)$, i.e. $t \approx^+ u$ implies $t \approx^* u$, where $\approx$ means $\approx^2$.

Suppose $t \not\approx^* u$, there exists a source $[\![\bullet]\!]$ such that we have the following cases

(i.a) case $\exists \alpha, y, t'.t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$ and for all u''' and u'' such that either $u \not\stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'''$ or $u''' \not\stackrel{\alpha?z}{\Longrightarrow}_{[\bullet][\perp/z]} u''$ or for all $x \notin Free(t') - \{y\} \cup Free(u') - \{z\}$ and there exists $c \in Nat_\perp$ such that there are two possible cases, i.e. either $u'' \stackrel{\epsilon}{\Longrightarrow}_{[\bullet][c/z]} u'$ or $t'[y := x] \not\approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$:

let $v = Nil$, we have $(t + v) \not\approx (u + v)$.

(i.b) case $\exists \alpha, t'.t \xrightarrow{\alpha![e]}_{[\bullet]} t'$ and for all u' either $u \stackrel{\alpha![e]}{\Longrightarrow}_{[\bullet]} u'$ or $t' \not\approx_\omega ([\![\bullet]\!])u'$:

let $v = Nil$.

(i.c) case $\exists t'.t \xrightarrow{\tau}_{[\bullet]} t'$ and for all u' either $u \not\stackrel{\tau}{\Longrightarrow}_{[\bullet]} u'$ or $t' \not\approx_\omega ([\![\bullet]\!])u'$: let $v = Nil$.

(i.d) case $t \xrightarrow{\tau}_{[\bullet]} t'$ and for all u' either $u \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'$ or $t' \not\approx_\omega ([\![\bullet]\!])u'$:

there are two subcases for the former and the latter can be regarded as the second subcase.

(i.d.1) subcase $u = u'$ and u does not need a τ action to reach u' :

Let $v = \alpha!e.Nil$ where α does not appear in t and u. Then, since $(t+v) \xrightarrow{\tau}_{[\bullet]} t'$ and since there is no t'' such that $t \not\xrightarrow{\alpha![e]}_{[\bullet]} t''$, $t' \not\approx_\omega ([\![\bullet]\!])(u + v)$.

(i.d.2) subcase $u \stackrel{\tau}{\Longrightarrow}_{[\bullet]} u'$ and $t' \not\approx_\omega ([\![\bullet]\!])u'$:

let $v = Nil$.

(ii) For "$\Leftarrow$" (if direction), we want to show that $t \approx^* u$ implies $t \approx^+ u$, i.e. $\forall v.(t + v) \approx (u + v)$.

(ii.a) case $(t + v) \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$:

there are two subcases, whether the transition is contributed by v or not, as follows.

(ii.a.1) subcase $v \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$:

we would have $(u+v) \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$. The remaining is trivial.

(ii.a.2) subcase $t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$:

we would have that there exist u''' and u'' such that $u \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'''$ and $u''' \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} u''$ and $\exists x \in Free(t') - \{y\} \cup Free(u'') - \{z\}, \forall c \in Nat_\perp, \exists u'.u \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'$ and $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$. So, $(u+v) \stackrel{\alpha?x}{\Longrightarrow}_{[\bullet][\perp/x]} u'$.

(ii.b) case $(t+v) \xrightarrow{\alpha![c]}_{[\bullet]} t'$: similar to the above case (ii.a), we have to consider whether the transition is contributed by summand v.

If this is the former case, it would be trivial. So, we suppose the latter. From this, we can derive that there exists u' such that $u \stackrel{\alpha![c]}{\Longrightarrow}_{[\bullet]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$ by $t \approx^* u$. Then, $(u+v) \stackrel{\alpha![c]}{\Longrightarrow}_{[\bullet]} u'$ and $t' \approx_\omega ([\![\bullet]\!])u'$.

(ii.c) case $(t+v) \xrightarrow{\tau} t'$: it is similar to the above case (ii.b).

(ii.d) The remaining arguments will be similar to the above cases from (ii.a) to (ii.c). The only difference is to exchange the roles of t and of u. We omit them. □

As we point out previously, the above result suggest that $\approx^*$ is a congruence relation. The following lemma is actually a verification of this suggestion.

Lemma 10.3.5 (congruence $\approx^*$): *$\approx^*$ is a congruence relation on CCS terms, i.e.*

1.
$$\frac{t \approx^* u}{\alpha!e.t \approx^* \alpha!e.u}$$

2.
$$\frac{t \approx^* u}{\alpha?x.t \approx^* \alpha?x.u}$$

3.
$$\frac{t_i \approx^* u_i}{(t_1 + t_2) \approx^* (u_1 + u_2)}$$

4.
$$\frac{t_i \approx^* u_i}{(t_1|t_2) \approx^* (u_1|u_2)}$$

5.

$$\frac{t_i \approx^* u_i}{(\textbf{if } b \textbf{ then } t_1 \textbf{ else } t_2) \approx^* (\textbf{if } b \textbf{ then } u_1 \textbf{ else } u_2)}$$

The proof of Lemma 10.3.5 is a simple definitional check and is left out.

As a result of Lemma 10.3.4 and Lemma 10.3.5, we sense that $\approx_c^1$, $\approx^+$ and $\approx^*$ are actually the same relation, as the case with data-dependency for CCS. We present this as a theorem (Theorem 10.3.7) below. Before doing so, we give another handy lemma (Lemma 10.3.6).

Lemma 10.3.6 (congruences and $\approx_c^1$): *If θ is a congruence and $t\theta u$ implies $t \approx^1 u$, then $t\theta u$ implies $t \approx_c^1 u$.*

Proof

Since θ is a congruence, $C[t]\theta C[u]$ for all context $C[\,]$. Then, $\forall C[\,].C[t] \approx^1 C[u]$, i.e. $t \approx_c^1 u$. □

Finally, we have

Theorem 10.3.7 (equivalence among $\approx_c^1$, $\approx^+$ and $\approx^*$): *$t \approx_c^1 u$ iff $t \approx^+ u$ iff $t \approx^* u$.*

Proof

It is obvious that $t \approx_c^1 u$ implies $t \approx^+ u$, i.e. $\approx_c^1 \subseteq \approx^+$. Then, $\approx_c \subseteq \approx^+ = \approx^*$. On the other hand, $t \approx^* u$ implies $t \approx^1 u$. By Lemma 10.3.5, $\approx^*$ is a congruence. So, by Lemma 10.3.6 we come to the conclusion of that $t \approx^* u$ implies $t \approx_c^1 u$, i.e. $\approx^* \subseteq \approx_c^1$. Overall, $\approx_c^1 = \approx^+ = \approx^*$. □

Lemma 10.3.8 (soundness of Eq): *If $Eq^{ccs} \vdash_{ccs} t \simeq u$, then $t \approx_c^1 u$, where Eq contains Eq_τ^{ccs} and $Eq_{\tau^{-1}}^{ccs}$. $Eq_{\tau^{-1}}^{ccs}$ only contains all instances of following*

1. (a) *(?-τ-elim)* $\alpha?x.\tau.t \simeq \alpha?x.t$

   (b) *(!-τ-elim)* $\alpha!e.\tau.t \simeq \alpha!e.t$

   (c) *(τ-elim)* $\tau.\tau.t \simeq \tau.t$

2. *(+-τ-elim)* $(t + \tau.t) \simeq \tau.t$

3. *(a) (?-τ-+-elim)* $(\alpha?x.(t + \tau.u) + \alpha?x.u) \simeq \alpha?x.(t + \tau.u)$

   (b) (!-+-τ-elim) $(\alpha!e.(t + \tau.u) + \alpha!e.u \simeq \alpha!e.(t + \tau.u)$

   (c) (τ-+-τ-elim) $(\tau.(t + \tau.u) + \tau.u) \simeq \tau.(t + \tau.u)$

4. *(a) (?-**then**-τ-elim)* $\alpha?x.t \simeq \alpha?x.u \mapsto \alpha?x.(b \supset (t, v)) \simeq \alpha?x.(b \supset (u, v))$

   *(b) (?-**else**-τ-elim)* $\alpha?x.t \simeq \alpha?x.u \mapsto \alpha?x.(b \supset (v, t)) \simeq \alpha?x.(b \supset (v, u))$.

Proof

Replacing $\approx_c^1$ by $\approx^*$, we only need to check the instances not in Eq_τ. The actual verification is omitted. □

The result of Lemma 10.3.8 leads us to wonder whether weak bisimulation can be characterized purely by equations (or more precisely, BEs).

Following Section 10.2, we only need to consider |-free CCS terms and extend the previous reduction system in Section 10.2 (replacing $\rhd$ by $\trianglerighteq$) to include the reductions which are gotten by replacing $\simeq$ with $\trianglerighteq$ in $Eq_{\tau-1}^{ccs}$. Similarly, we introduce a new definition for *semi-Normal forms* (semi-Nf in short, not semi-nf as in Section 10.2)

Definition 10.3.9 (semi-Nfs): *We define semi-Normal forms (semi-Nfs) for* $\trianglerighteq$ *as follows*

(a) Nil is a semi-Nf;

(b) $\tau.t$ is semi-Nf if t is semi-Nf, $t \notin IF$ and there does not exist t' such that $\tau.t' = t$;

(c) $\alpha!e.t$ is semi-Nf if t is semi-Nf, $t \notin IF$ and there does not exist t' such that $\tau.t' = t$;

(d) $\alpha?x.t$ is semi-Nf if t is semi-Nf and there does not exist t' such that $\tau.t' = t$, and further either $t \in IF(x)$ or $t \notin IF$;

(e) $\sum_{i=1}^{k} t_i$ $(k > 1)$ is semi-Nf if t_i is semi-Nf $(1 \leq i \leq k)$, $t_i \neq t_j$ $(i \neq j)$, $t_i \neq \tau.t_j$, $t_i \notin IF$, and $t_i \neq Nil$;

(f) $(\textbf{if } b \textbf{ then } t_1 \textbf{ else } t_2)$ *is semi-Nf if* t_i *is semi-Nf* $(i = 1, 2)$, $t_1 \neq t_2$, *and* $Free^{\cdot}_m(b) = \emptyset$.

Analogous to previous section, we need to know about normalizability. We state it as follows.

Property 10.3.10 (semi-Normalizability): *Let* $\trianglerighteq^*$ *be the (reflexive and) transitive closure of* $\trianglerighteq$. *Then, for all* t, *there exists a* t' *such that* $t \trianglerighteq^* t'$ *and* t' *is a semi-Nf.*

We leave the proof out, since it is similar to the proofs of Lemma 10.2.4. Also, $\trianglerighteq$ inherits the non-strong-normalizability of $\triangleright$. The problem comes from the same source, i.e. **if**-statements. We firstly locate the scope of the influence of **if**-statements in following Lemma 10.3.11. Similar to Section 10.2, we also introduce *concise* semi-Nfs and *refined* semi-Nfs. Since their definitions can be easily copied over from the previous ones with terminological modification, we omit them.

Lemma 10.3.11 (weak bisimulation and *if*-statements): *Let* t *and* u *be (|-free and) concise and refined semi-Nfs. Thus, given* $[\![\bullet]\!]$, *if* $t \approx_\omega ([\![\bullet]\!])u$ *then*

$$subterm(path([\![\bullet]\!], t), t) \approx^1 subterm(path([\![\bullet]\!], u), u).$$

Proof Note that for each $[\![\bullet]\!]'$,

$t \approx_\omega ([\![\bullet]\!]')subterm(path([\![\bullet]\!]', t), t)$ *and* $u \approx_\omega ([\![\bullet]\!]')subterm(path([\![\bullet]\!]', u), u)$.□

Since the above lemma, we can define $\approx^3$ similar to $\sim^3$ in previous section (see Definition 10.2.8) as follows.

Definition 10.3.12 (third weak bisimulation $\approx^3$): *Let* t *and* u *be concise and refined semi-Nfs, then* $t \approx^3 u$ *iff for all* $[\![\bullet]\!]$,

1. - *if* $t \xrightarrow[[\bullet][\perp/y]]{\alpha?y} t'$ *then* $\exists u'''.u \overset{\xi}{\Longrightarrow}_{[\bullet]} u'''$ *and* $u''' \xrightarrow[[\bullet][\perp/z]]{\alpha?z} u''$ *and* $\exists x \notin (Free(t') - \{y\}) \cup (Free(u'') - \{z\}), \forall c \in Nat_\perp . \exists u'.u'' \overset{\xi}{\Longrightarrow}_{[\bullet][c/z]} u'$ *and* $t'[y := c] \approx^2 u'[z := c]$
   - *if* $t \xrightarrow[[\bullet]]{\alpha![e]} t'$ *then* $\exists u'.u \overset{\alpha![e]}{\Longrightarrow}_{[\bullet]} u'$ *and* $t' \approx^2 u'$

- *if* $t \xrightarrow{\tau}_{[\bullet]} t'$ *then* $\exists u'.u \stackrel{\tau}{\Longrightarrow}_{[\bullet]} u'$ *and* $t' \approx^2 u'$.

2. - *if* $u \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} u'$ *then* $\exists t'''.t \stackrel{\xi}{\Longrightarrow}_{[\bullet]} t'''$ *and* $t''' \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} t''$ *and* $\exists x \notin (Free(t'') - \{z\}) \cup (Free(u') - \{y\}), \forall c \in Nat_\perp.\exists t'.t''_{[\bullet][c/z]} \stackrel{\xi}{\Longrightarrow}_{[\bullet][c/z]} t'$ *and* $t'[y := \mathbf{c}] \approx^2 u'[z := \mathbf{c}]$
   - *if* $u \xrightarrow{\alpha![c]}_{[\bullet]} u'$ *then* $\exists t'.t \stackrel{\alpha![c]}{\Longrightarrow}_{[\bullet]} t'$ *and* $t' \approx^2 u'$
   - *if* $u \xrightarrow{\tau}_{[\bullet]} u'$ *then* $\exists t'.t \stackrel{\tau}{\Longrightarrow}_{[\bullet]} t'$ *and* $t' \approx^2 u'$

The above definition is to bring a semantic version of $\approx^2$ up to syntactic level. This is somehow accomplished by a technique, so-called call-by-name.

Similarly we bring a new version of Theorem 10.2.6 to here, and state it as follows. Formally,

Theorem 10.3.13 ($\mathcal{E}$): *Let* $\alpha?x.t$ *be (|-free) semi-Nf.*

1. *(reachability) There is another concise semi-Nf* t' *such that* $t \approx^2 t'$ *and for* $p \in paths(t')$ *and for all* $[\![\bullet]\!]$, *we have that* $p \neq \varepsilon$ *implies* $x \in \bigcap_{b \in premise(p,t')} Free_m(b)$ *and* $\exists c \in Nat_\perp, \forall b \in premise(p,t').[\![b]\!]' = true$, *where* $[\![\bullet]\!]' = [\![\bullet]\!][c/x]$.

2. *(independency) Let* u *be another semi-Nf and* $t \approx^2 u$. *If* $t = b \supset (t_1, t_2)$ *and* $u = b' \supset (u_1, u_2)$, *then either* $b \leftrightarrow b'$ *or for some* i *there is* $p_i \in paths(u_i)$ *and* $b \in premise(p_i, u_i)$.

The proof of Theorem 10.3.13 is postponed to Section 10.4, i.e. Theorem 10.4.7.

Of course, we can easily get $t \approx^3 u$ implies $t \approx^2 u$. The reversed implication can be derived from the following lemma, Lemma 10.3.14.

Lemma 10.3.14 ($\approx^2$ and $\approx^3$): *Let* t *and* u *be semi-Nfs, and* $t \approx^2 u$. *We have that given* $[\![\bullet]\!]$,

- *if* $t \xrightarrow{\alpha![c]}_{[\bullet]} t'$, $\exists u'.u \stackrel{\alpha![c]}{\Longrightarrow}_{[\bullet]} u'$ *and* $t' \approx_\omega ([\![\bullet]\!])u'$, *then* $t' \approx^2 u'$;

- *if* $t \xrightarrow{\tau}_{[\bullet]} t'$, $\exists u'.u \stackrel{\tau}{\Longrightarrow}_{[\bullet]} u'$ *and* $t' \approx_\omega ([\![\bullet]\!])u'$, *then* $t' \approx^2 u'$;

- *if* $t \xrightarrow{\alpha?y}_{[\bullet][\perp/y]} t'$, $\exists u''', u''.u \stackrel{\epsilon}{\Longrightarrow}_{[\bullet]} u'''$ *and* $u''' \xrightarrow{\alpha?z}_{[\bullet][\perp/z]} u''$ *and* $\exists x \notin Free(t') - \{y\} \cup Free(u') - \{z\}$, $\forall c \in Nat_\perp$, *there exists a* u' *such that* $u'' \stackrel{\perp}{\Longrightarrow}_{[\bullet][c/x]} u'$ *and and* $t'[y := x] \approx_\omega ([\![\bullet]\!][c/x])u'[z := x]$, *then* $t'[y := x] \approx^2 u'[z := x]$;

Proof It is similar to the one of Lemma 10.2.9. □

As a consequence of the above lemma, we know that $t \approx^2 u$ implies $t \approx^3 u$, where t and u are semi-Nfs. So, we have an equivalence link that $t \approx^1 u$ iff $t \approx^2 u$ iff $t \approx^3 u$. However, this link has no much value in itself if there is no good relation between $\approx^2$ and $\approx^*$, since $\approx^*$ is equivalent to $\approx_c$. The following lemma (Lemma 10.3.15) is to prove such a good relationship between $\approx^2$ and $\approx^*$.

Lemma 10.3.15 ($\approx^2$ **and** $\approx^*$): *Let* t *and* u *be semi-Nfs, and* $t, u \notin IF$. *If* $t \approx^2 u$ *then either* $t \approx^* u$, $\tau.t \approx^* u$ *or* $t \approx^* \tau.u$.

Proof

We prove this lemma by case analysis.

(a) case of there exists $[\![\bullet]\!]$ such that $\exists t'.t \xrightarrow{\tau}_{[\bullet]} t'$ and $t' \approx_\omega ([\![\bullet]\!])u$:

by the definition of $\approx^*$, we have $t \approx^* \tau.u$.

(b) case of there exists $[\![\bullet]\!]$ such that $\exists u'.u \xrightarrow{\tau}_{[\bullet]} u'$ and $t \approx_\omega ([\![\bullet]\!])u'$:

it is similar to the above case (a) that $\tau.t \approx^* u$.

(c) case of neither of the above cases (a) and (b) hold: we would have $t \approx^* u$. □

Since we are reasoning with weak bisimulation, a target term of a weak transition, i.e. the term on the right hand side of a transition $\Longrightarrow$, plays an interesting role in an inductive reasoning. On this observation, we are led to consider all directly reachable target terms of a term by $\Longrightarrow$, and see whether the reachable target terms can somehow be regarded as summands of the term. Therefore, a weak transition $\Longrightarrow$ of a term can be replaced by a strong transition $\longrightarrow$ of the term. The following lemma is to exploit the above idea. Formally,

Lemma 10.3.16 (reachable terms and summands): *Let* t *be semi-Nf and* $t \notin IF$. *Thus,*

(i) if there exists $[\![\bullet]\!]$ *such that* $\exists t'.t \stackrel{\alpha![e]}{\Longrightarrow}_{[\bullet]} t'$, *either* $\alpha!e.t'$ *is summand of* t *or* t *can be extended to include* $\alpha!e.t'$ *as its summand, i.e.* $t \simeq (t + \alpha!e.t')$ *is derivable from* Eq^{ccs};

(ii) if there exists $[\![\bullet]\!]$ *such that* $\exists t'.t \stackrel{\tau}{\Longrightarrow}_{[\bullet]} t'$, *either* $\tau.t'$ *is summand of* t *or* t *can be extended to include* $\tau.t'$ *as its summand, i.e.* $t \simeq (t + \tau.t')$ *is derivable from* Eq^{ccs};

(iii) if there exists $[\![\bullet]\!]$ *such that* $\exists t', t'', c \in Nat_\perp.t \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t''$, *and* $t'' \ {}_{[\bullet][c/y]} \stackrel{\epsilon}{\Longrightarrow}_{[\bullet][c/y]} t'$, *then we have either* $t' = t''$ *or* $t' \neq t''$. *For the former, we have that* $\alpha?x.t'[y := x]$, *where* $x \notin Free(t')$, *is summand of* t *up to* α*-conversions, i.e.* $t \simeq (t + \alpha?x.t'[y := x])$ *is derivable. For the latter, there are two subcases:*

(iii.a) if $\exists t''.t \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t''$, $t'' \ {}_{[\bullet][c/y]} \stackrel{\tau}{\Longrightarrow}_{[\bullet][c/y]} t'$ *and* $t'' \notin IF$, *then* t *can be extended to include* $\alpha?x.t'[y := x]$ *as its summand, i.e.* $t \simeq (t + \alpha?x.t'[y := x])$ *is derivable;*

(iii.b) if $\exists t''.t \stackrel{\alpha?y}{\longrightarrow}_{[\bullet][\perp/y]} t''$, $t'' \ {}_{[\bullet][c/y]} \stackrel{\tau}{\Longrightarrow}_{[\bullet][c/y]} t'$ *and* $t'' \in IF$, *then there exists* t''' *such that* $t \simeq (t + \alpha?x.t'''[y := x])$,

where for all (other) evaluation $[\![\bullet]\!]'$, $paths(t'') = paths(t''')$, $premise(p, t'') = premise(p, t''')$ *for each* $p \in paths(t'')$, *and* $subterm(path([\![\bullet]\!]', t'''), t''')$

$$= \begin{cases} t' & if\ path([\![\bullet]\!]', t''') = path([\![\bullet]\!]', t'') \\ subterm(path([\![\bullet]\!]', t''), t'') & otherwise \end{cases}$$

(iv) if there exists $[\![\bullet]\!]$ *such that* $\exists t', t'', t''', c \in Nat_\perp.t \stackrel{\tau}{\Longrightarrow}_{[\bullet]} t'''$, $t''' \longrightarrow^{\alpha?y}_{[\bullet][\perp/y]} t''$, *and* $t'' \ {}_{[\bullet][c/y]} \stackrel{\epsilon}{\Longrightarrow}_{[\bullet][c/y]} t'$, *then* t *can be extended to include* $\alpha?x.t'[y := x]$ *as its summand, i.e.* $t \simeq t + \alpha?x.t'[y := x]$ *is derivable.*

Proof

By combining all possible cases with structural induction.

For case (i), it is trivial for $\alpha!e.t'$ to a summand of t. So, we are considering two subcases (not this one), i.e. either (i.a) $\exists t''.t \stackrel{\alpha![e]}{\longrightarrow}_{[\bullet]} t''$ and $t'' \stackrel{\tau}{\Longrightarrow}_{[\bullet]} t'$, or (i.b) $\exists t''.t \stackrel{\tau}{\Longrightarrow}_{[\bullet]} t''$ and $t'' \stackrel{\alpha![e]}{\longrightarrow} t'$.

Considering (i.a), i.e. $\exists t''.t \xrightarrow{\alpha![e]}_{[\bullet]} t''$ and $t'' \overset{\tau}{\Longrightarrow}_{[\bullet]} t'$, we have $t'' \notin IF$ and by structural hypothesis, $t'' \simeq (t'' + \tau.t')$. Hence,

$$t \simeq (t + \alpha!e.t'') \simeq (t + \alpha!e.(t'' + \tau.t'))$$
$$\simeq (t + (\alpha!e.(t'' + \tau.t') + \alpha!e.t'))$$
$$\simeq (t + (\alpha!e.t'' + \alpha!e.t')) \simeq (t + \alpha!e.t')$$

Considering (i.b), i.e. $\exists t''.t \overset{\tau}{\Longrightarrow}_{[\bullet]} t''$ and $t'' \xrightarrow{\alpha![e]} t'$, we have $t'' \notin IF$ and by structural hypothesis, $t'' \simeq (t'' + \alpha!e.t')$. Therefore,

$$t \simeq (t + \tau.t'') \simeq (t + (\tau.t'' + t''))$$
$$\simeq (t + (\tau.t'' + (t'' + \alpha!e.t')))$$
$$\simeq (t + (\tau.t'' + \alpha!e.t')) \simeq (t + \alpha!e.t').$$

For case (ii), it is similar to the case (i) above.

For case (iii), it is trivial if $\alpha?x.t'[y := x]$ is a summand of t up to a α-conversion. So, we are considering the other possibilities of (iii.a) and (iii.b).

For (iii.a), it is similar to the case (i) above.

For (iii.b), we firstly introduce a technique to deal with nested if-statements with silent action τ's. For example, we have

$$(b \supset (t_1, t_2)) + \tau.(b \supset (u, t_2)) \simeq (b \supset (t_1 + \tau.u, \tau.t_2));$$

and also, we have

$$\alpha?x.((b \supset (t_1, t_2)) + \tau.(b \supset (u, t_2))) \simeq \alpha?x.(b \supset (t_1 + \tau.u, t_2)).$$

Therefore, by ($?$-τ-+-elim), ($?$-**else**-τ-elim) and ($?$-τ-elim), we get

$$\alpha?x.(b \supset (t_1 + \tau.u, t_2)) \simeq \alpha?x.(b \supset (t_1 + \tau.u, t_2)) + \alpha?x.(b \supset (u, t_2)).$$

The above technique can be applied repeatedly. Consequently, we will have what we claim, i.e. $t \simeq (t + \alpha?x.t'''[y := x])$.

For (iv), we understand that (1) there is only one τ action prefixing a term which is in semi-Nf, and (2) there is only one τ action prefixing each summand of

a term which is in semi-Nf. Because of case (ii) above, we can bring all reachable subterms of a term by silent action τ's up to the level of direct summands with only one τ action prefixing by repeatedly applying (+-τ-+-elim). The same technique is also used to eliminate the by-products of the previous procedure in order to get the right summand. After finishing these two procedures, we will have $t \simeq t + \tau.t'''$ where t''' has a summand prefixed by an input on port α, i.e. the summand is identical to $\alpha?x.t''[y := x]$ with subject to α-conversion. By (+-τ-elim), we arrive at $\tau.t''' \simeq \tau.t''' + t'''$. Considering $\tau.t'''$ in the above sum term, it can be eliminated by the previous technique in elimination; considering t''' in the above sum term, it is actually the case (iii) above. Therefore, what we claim holds. □

Note that the new extended summands appearing in the proof of Lemma 10.3.16 are semi-Nfs, as well. It does not fail to remind us of that the above lemma (Lemma 10.3.16) plays a significant role in the proof of completeness for Eq^{ccs} of Theorem 10.3.17. The following is my account on key ideas of the completeness proof.

(10.3.a) Firstly, we locate the breakdown of compositionality at + (sum) operator, and introduce $\approx^+$ (see [108, Theorem 10.8]).

(10.3.b) Secondly, the new definition of $\approx^*$ substitutes for $\approx^+$. This replacement wipes out all possibility of depending on contexts (see Lemma 10.3.4). Now, we are only needed to reason straightly from the definition of $\approx^*$.

(10.3.c) Thirdly, the reduction of weak transitions (double-arrow like $\Rightarrow$) to strong transitions (single-arrow like $\rightarrow$) enables us to reason weak bisimulation just like strong bisimulation. This task is accomplished by extending a term to include all reachable subterms as its summands (see Lemma 10.3.16).

(10.3.d) Lastly, the extension of a term as in (10.3.c) above can be viewed as an opposite of the processing, i.e. contraction of the term. More specifically, a term t can be extended to $t + \Sigma\alpha_i\nabla_i t_i$ and they are weakly bisimulated with each other, where t_i's exhaust all possible reachable subterms without duplications. Also, $\Sigma\alpha_i\nabla_i t_i$ is actually weakly bisimulated with t. So, if u is weakly bisimulated with t, then both of t and u can be extended to $t+\Sigma\alpha_i\nabla_i t_i$ and $u+\Sigma\alpha_j\nabla_j u_j$. By an inductive

reasoning, we can prove that $\Sigma\alpha_i\nabla_i t_i \simeq \Sigma\alpha_j\nabla_j u_j$ since there is a 1-1 correspondence between i's and j's. Then, by applying the same lemma (Lemma 10.3.16) but in an opposite way, i.e. contracting, we get back to $t + \Sigma\alpha_i\nabla_i t_i \simeq u + \Sigma\alpha_j\nabla_j u_j$. So, $t \simeq u$.

The last point in (10.3.d) finally brings us the completeness as follows.

Theorem 10.3.17 (completeness of Eq^{ccs}): *If $t \approx^* u$ then $Eq^{ccs} \vdash_{ccs} t \simeq u$ (without substitutions).*

Proof We should understand that substitution rule is not essential (or eliminable) since Eq^{ccs} contains axiomatic schemas only. Secondly, without loss of generality, we can restrict our attention to concise and refined semi-Nfs. For each concise and refined semi-Nf term t, we can get a term $t' + \Sigma\alpha_i\nabla_i t_i$ such that $\Sigma\alpha_i\nabla_i t_i$ is bisimulated with t and it completely reduces the weak transitions to strong transitions. We can name $\Sigma\alpha_i\nabla_i t_i$ as the *standard semi-Nf* of t, i.e a semi-Nf term t is a *standard* one if its weak bisimulation coincides with its strong bisimulation.

In what follows, we are only interested in all possible standard terms (in semi-Nfs), written as $Standard(T_a)$ if you prefer.

Similar to the proof of Lemma 10.2.9, let t and u be two standard semi-Nfs such that $t \approx^* u$. Let us consider all possible cases for t with induction over the sum of the depth of t and u defined in Section 10.1.

(a) For $depth(t) + depth(u) = 0$, it is trivial.

(b) For $depth(t) + depth(u) > 0$, there are several cases of t to be considered below.

(b.i) case Nil: we know that u cannot be Nil, $\tau.u'$, $\alpha!e.u'$ or $\alpha?x.u'$ for some u'. So, if $u = (u_1 + u_2)$ or if $u = (\textbf{if } b \textbf{ then } u_1 \textbf{ else } u_2)$, then by induction hypothesis, we have $t \simeq u_i$ for all i. Thus, $t \simeq u$.

(b.ii) case $\alpha!e.t'$: we would have that u cannot be Nil. And u can not be in IF since it is in semi-Nf. So, u must be $\sum_j \alpha!e_j.u_j$ (actually $[\![e]\!] = [\![e_j]\!]$ has to hold for all evaluations $[\![\bullet]\!]$, then we can replace e_j by e from a point of view of representing

an equivalent class), Since $\alpha!e.t'$ and u are in standard forms, t' must be bisimulated with u_j. By inductive hypothesis, we get $t' \simeq u_j$, therefore $t \simeq u$ holds.

(b.iii) case $\tau.t'$: it is similar to the case $t = \alpha!e.t'$.

(b.iv) case $\alpha?x.t'$: We have that u cannot be Nil. Also, if $u \in IF$, say $u = (\mathbf{if}\ b\ \mathbf{then}\ u_1\ \mathbf{else}\ u_2)$, then by induction hypothesis, we have $t \simeq u_i$. Thus, $t \simeq u$.

So, suppose $u \notin IF$, say $u = \sum_j \alpha?y_j.u_j$. Since t and u are in standard forms, t' is bisimulated with u_j. By inductive hypothesis, $t' \simeq u_j$ is derivable, so is $t \simeq u$.

(b.v) case $\sum_i \alpha_i?x_i.t_i + \sum_j \alpha_{n+j}!e_j.t_{n+j} + \sum_k \tau.t_{n+m+k}$: u cannot be a Nil. If $u \in IF$, say $u = (\mathbf{if}\ b\ \mathbf{then}\ u_1\ \mathbf{else}\ u_2)$, then by induction hypothesis, we have $t \simeq u_i$. Thus, $t \simeq u$ is derivable.

If $u \notin IF$, say $u = \sum_{i'} \alpha_{i'}?y_{i'}.u_{i'} + \sum_{j'} \alpha_{n'+j'}!e_{j'}.u_{n'+j'} + \sum_{k'} \tau.u_{n'+m'+k'}$, then since both t and u are in standard forms, there be a 1-1 correspondence between the collection of t_i, t_{n+j}, and t_{n+m+k} and the collection of $u_{i'}$, $u_{n'+j'}$ and $u_{n'+m'+k'}$. For each pair of such corresponding terms, say t_i and $u_{i'}$, $t_i \simeq t_{i'}$ is derivable by inductive hypothesis. So is $t \simeq u$.

(b.vi) case $t = (\mathbf{if}\ b\ \mathbf{then}\ t_1\ \mathbf{else}\ t_2)$: there two possibilities for u, i.e. either $u \in IF$ or $u \notin IF$.

(b.vi.1) For the latter, by Lemma 10.3.14, we can have $t_i \approx_c u$ $(i = 1, 2)$. Hence, by induction hypothesis, we can get that $t \simeq u$ is derivable.

(b.vi.2) For the former, if $b = b'$, where $u = (\mathbf{if}\ b'\ \mathbf{then}\ u_1\ \mathbf{else}\ u_2)$, then by induction hypothesis $t_i \simeq u_i$ holds for $i = 1, 2$. Thus, $t \simeq u$ is derivable.

If $b \neq b'$, then by the independency part of Theorem 10.3.13, we can get $u \simeq (\mathbf{if}\ b\ \mathbf{then}\ u_1'\ \mathbf{else}\ u_2')$ by reductions from 4(c) to 4(g) in Eq_τ^{ccs}. which preserve semi-Nfs. Now, we return to the case of $b = b'$ as the above again. □

10.4 Discussion of boolean languages

In Section 10.2 and Section 10.3, we have presented two equational characterizations of weak bisimulations and strong bisimulations of finite CCS with data-dependency. But the completeness proofs are relied on Theorem 10.2.6 and Theorem 10.3.11. The purpose of this section is to discuss the generality of Theorem 10.2.6 and Theorem 10.3.13 and show that there are some boolean languages which satisfy the theorems, i.e. the extra conditions of $\mathcal{E}$ and $\mathcal{E}_\tau$ are satisfiable. However, how far we can go to extend the limited boolean languages to more general one is remained to be investigated.

For each theorem there are two parts of it. (10.4.a) One is reachable part and (10.4.b) the other is independency part. For (10.4.a), it is easy to verify that the variable bound by an input appears free in boolean expressions. We state this as a lemma (Lemma 10.4.1) below.

Lemma 10.4.1 (variables bound by an input): *Let t be a closed semi-Nf (or semi-Nf) and for all u, if there is a $[\![\bullet]\!]$, $t \xrightarrow{\vec{a}}_{[\bullet]} \alpha?x.t'$, and $t' \in IF$, then $x \in \bigcap_{b \in premise(p,t')} Free_m(b)$, where $p \in paths(t')$.*

Proof

It is obvious, otherwise there will be some instances of the reductions from 5(a) to 5(e) in Eq_τ^{ccs} □

For (10.4.b), there is difficulty in proving the most general case. The reason comes from the fact that in First Order Logic, $\forall x.A(x) \vee B(x)$ is not always equivalent to $\forall x.A(x)B(x) \vee \forall x.A(x)\neg B(x) \vee \forall x.\neg A(x)B(x)$. Apparently, the latter implies the former but not necessarily another way around. And each of $A(x)B(x)$, $A(x)\neg B(x)$ and $\neg A(x)B(x)$ is independent of each other by the obvious sense. In what follows, we are only trying to demonstrate that the independency part of the theorems is satisfiable.

Obviously, in general we understand that

Fact 10.4.2 (implication and *if*-statements):

(a) $b_1 \supset b_2$ *implies* $(b_1 \supset (b_2 \supset (t_1, t_2), u)) \simeq (b_1 \supset (t_1, u))$;

(b) $b_1 \supset \neg b_2$ *implies* $(b_1 \supset (b_2 \supset (t_1, t_2), u)) \simeq (b_1 \supset (t_2, u))$;

(c) $\neg b_1 \supset b_2$ *implies* $(b_1 \supset (t, (b_2 \supset (u_1, u_2)))) \simeq (b_1 \supset (t, u_1))$;

(d) $\neg b_1 \supset \neg b_2$ *implies* $(b_1 \supset (t, (b_2 \supset (u_1, u_2)))) \simeq (b_1 \supset (t, u_2))$.

Firstly, we need to have some extra equations which serve to treat boolean expression and **if**-statements. Bearing this purpose in mind, we can denote $[__1, __2, __3, __4]$ as **if** $\mathbf{eq}(__1, __2)$ **then** $__3$ **else** $__4$ for simplicity. Thus, there are some obvious laws:

Property 10.4.3 (E_{eq} and if-statements):

(i) $[x, x, t, u] \simeq t$,

(ii) $[x, y, t, u] \simeq [y, x, t, u]$,

(iii) $[x, y, t, u] \simeq [x, y, t\,[x := y], u]$.

From the above, we would have

Fact 10.4.4 (forwarding variables in if-statements):

1.

$$\begin{aligned}
&[x, y, [x, z, t_1, t_2], u] \\
&\simeq [x, y, [y, z, t_1, [x, z, t_1, t_2]\,], u] \\
&\simeq [y, z, [x, y, t_1, u], [x, y, [x, z, t_1, t_2], u]\,]
\end{aligned}$$

2.

$$\begin{aligned}
&[x, y, t, [x, z, u_1, u_2]\,] \\
&\simeq [x, y, t, [y, z, u_2, [x, z, u_1, u_2]\,]\,] \\
&\simeq [y, z, [x, y, t, u_2], [x, y, t, [x, z, u_1, u_2]\,]\,].
\end{aligned}$$

If a nested **if**-statement involves x, y, z then there are three kind of relations, (i) between x and y, (ii) between x and z and (iii) between y and z.

For relation (iii), we can further exploit it. Suppose variable x is prefixed and bound by an input $\alpha?x$. Both of the equations in Lemma 10.4.4 show that unbound variables can be move forward from nested **if**-statements. This observation leads to the following property (Property 10.4.3).

Property 10.4.5 (forwarding free variables in prefixed *if*-statements):

(a) $\alpha?x.[x,y,[x,z,t_1,t_2],u] \simeq [y,z,\alpha?x.[x,y,t_1,u],\alpha?x.[x,y,[x,z,t_1,t_2],u]]$ *and*

(b) $\alpha?x.[x,y,t,[x,z,u_1,u_2]] \simeq [y,z,\alpha?x.[x,y,t,u_2],\alpha?x.[x,y,t,[x,z,u_1,u_2]]]$.

The above properties will be used in the proof of Theorem 10.4.6, i.e. it enables us to insert some proper boolean expressions as we wish. Consequently, if boolean expressions in a nested **if**-statement involve three variables x, y, z as our previous analysis, then the relationship between y and z must be clarified (or sorted out) before the input action $\alpha?x$ taking place.

Before proceeding further, we would like to do some preparation.

Firstly, let us extend the paths to include the case of non-**if**-statements, i.e.

$$Paths(t) =_{df} \begin{cases} \{\epsilon\} & if\ t = Nil \\ \{\epsilon, 0, 0\,l_0 |\ l_o \in Paths(t')\} & if\ t = \alpha!e.t', \alpha?x.t', or\ \tau.t' \\ \{\epsilon, 1, 2, 1\,l_1, 2\,l_2 |\ l_i \in Paths(t_i)\} & if\ t = (t_1 + t_2)\ or \\ & (b \supset (t_1, t_2)); \end{cases}$$

Secondly, we extend the definition of subterms under a path. Formally,

$$Subterm(p,t) =_{df} \begin{cases} t & if\ p = \epsilon \\ Subterm(p_0, t') & if\ p = 0\,p_0\ and\ t = \alpha!e.t'\ or\ \tau.t' \\ Subterm(p_0, t^*) & if\ p = 0\,p_0\ and\ t = \alpha?x.t' \\ & y \notin Free_m(t') - \{x\}\ and\ t^* = t'[x := y] \\ Subterm(p_i, t_i) & if\ p = i\,p_i\ and\ t = (t_1 + t_2)\ or\ (b \supset (t_1, t_2)) \end{cases},$$

where $p \in Paths(t)$;

By extending the above definition, we can have the collection of all possible subterms of a term. Namely, $Sterm(t) =_{df} \bigcup_{p \in Paths(t)} Subterm(p,t)$.

Lastly, we need to know of the condition to reach a branch under a path. For-

mally,

$$Cond(p,t) =_{df} \begin{cases} \emptyset & if\ t = Nil\ (and\ p = varepsilon) \\ Cond(p_0, t') & if\ p = 0\,p_0\ and\ t = \tau.t', \alpha!e.t'\ or\ \alpha?x.t' \\ Cond(p_i, t_i) & if\ t = (t_1 + t_2)\ and\ p = i\,p_i \\ \{b\} \cup Cond(p_1, t_1) & if\ p = 1\,p_1\ and\ t = b \supset (t_1, t_2) \\ \{\neg b\} \cup Cond(p_2, t_2) & if\ p = 2\,p_2\ and\ t = b \supset (t_1, t_2) \end{cases}$$

Now, we are ready to prove $E_{\mathbf{eq}}$ satisfies the theorems. It might appear that the above definitions are not necessary but we need them, at least in stating the theorems below.

Theorem 10.4.6 ($E_{\mathbf{eq}}$ and Theorem 10.2.6): *Suppose t is a semi-closed semi-nf. Then there is another semi-nf u such that $t \simeq u$ and for all $\alpha?x.u' \in Sterm(u)$ u' satisfies the conditions of Theorem 10.2.6.*

Proof

This proof is actually based on an algorithm. The principle behind the algorithm is to exploit Property 10.4.5 to certain extent so that the relationship among variables other than x (which is bound by an input $\alpha?x$) has been sorted out along the path of the term before reaching the input action $\alpha?x$.

That is, given a semi-nf t, we can always obtain a corresponding semi-nf u, by repeatedly applying Property 10.4.5, such that the relationship among variables other than the variable bound by an input has been established. The termination of this algorithm comes when there is no case available to apply Property 10.4.5. The *termination* of the algorithm is guaranteed by the fact that only finite terms are under our consideration (in this chapter). Therefore, we claim that the reachability part of Theorem 10.4.6 is proved.

For the independency part of Theorem 10.4.6, we only need to remind you of the fact that **eq** is independent in boolean language E_{eq} and every distinct variable forms a syntactic representative of a semantic equivalence class of message expressions. □

Apparently, we can name a term in semi-nf (or semi-Nf), which has the property of the relationship among variables other than the variable bound by an input has

been established by the corresponding path, as boolean *standard semi-nf* (or semi-Nf). Also, Property 10.4.5 preserves such normality of a term. So, the previous theorem can be viewed as saying that every semi-nf (or semi-Nf) term t has a corresponding boolean standard semi-nf (or semi-Nf) term u such that $t \simeq u$ is derivable.

For Theorem 10.3.13, we can prove a similar result and by a similar way except that we need t to be a boolean standard semi-Nf rather than just a semi-Nf. So, we state the result only as follows.

Theorem 10.4.7 ($E_{\mathbf{eq}}$ and Theorem 10.3.13): *Suppose t is a semi-closed standard semi-Nf. Then there is a boolean standard semi-Nf u such that $t \simeq u$ and for all $\alpha?x.u' \in Sterm(u)$ u' satisfies the conditions of Theorem 10.3.13.*

By the above two theorems, we have shown that $E_{\mathbf{eq}}$ satisfies Theorem 10.2.6 and Theorem 10.3.13.

By the remark between Theorem 10.4.6 and Theorem 10.4.7, we understand that the crucial point in proving both theorems is (a) to establish the property of the relationship among variables other than the variable bound by an input before the input can be perform, i.e. the relationship has been sorted out by the corresponding path to the input; and (b) to have total independent predicates in boolean languages.

It seems that as long as all "primitive" predicates in a boolean language are independent with each other, i.e. none of them can be expressible by the others through logical connectives, we will have our two theorems holds. However, we wonder whether this is true in general case. We expect that this is the case and state it as an open problem.

Open Question 10.4.8: *Let E be an boolean language, if none of predicates in E are expressible by other predicates in E through logical connectives, then both of Theorem 10.2.6 and Theorem 10.3.11 hold for this E.*

Part VI

Comparision and future development

This is the last part of this thesis. We will compare satisfactions between $\models_{eBA}$ and $\models_{iBA}$ as well as the proof powers between calculus $\vdash_{eBA}$ and calculus $\vdash_{iBA}$. Then, a discussion on a possible improvement of $\vdash_{iBA}$ follows.

Chapter 11

Birkhoff's approach vs Friedman's approach

This thesis has so far presented a Framework for Binding Operators and their algebraic characterizations with some applications. As we pointed out in Chapter 1, such a framework has been foreseen by many people. There is no originality due to me on this aspect. What I have done in this thesis is working out the details. Obviously, there are many problems I run across while I work on the thesis. Some problems are unexpected, e.g. the breakdown of commutativity for Binding Homomorphisms which leads to a weaker result than the expected. However, we would like to know whether $\models_{eBA}$ and $\dot{\models}_{eBA}$ coincide with each other. We have some discussion of this in Section 3.13, and we do not resume them again.

In this thesis, we essentially present two calculi $\vdash_{iBA}$ and $\dot{\vdash}_{eBA}$ along with their Completeness and Admissible Completeness, which are achieved through Friedman approach and Birkhoff approach. In this chapter, we intend to clarify on the relationship between these two approaches, especially between $\models_{eBA}$ and $\models_{iBA}$ as well as between $\dot{\vdash}_{eBA}$ and $\vdash_{iBA}$. The former is to be discussed in Section 11.1; and Section 11.2 is given to the subject of the latter. Finally, a possible improvement on Friedman approach is examined in Section 11.3.

Before we actually get to the fore mentioned subjects, there are some comments need to be made.

Question 3.11.11 - 3.11.12: *(a) Under what condition, admissible equality of a class $\mathcal{K}$ is closed under direct product of eBAs?*

(b) Whether there is a constructible (or definable) operator Υ over classes $\mathcal{K}$ of eBAs such that $Mod(\dot{I}_{\mathcal{K}}) = \Upsilon(\mathcal{K})$?

Question 10.4.7: Let E be a boolean language. Thus, if none of predicates in E are expressed by other predicates available in E through logical connectives, then whether both Theorem 10.2.6 and Theorem 10.3.13 hold.

It is a little surprise for us to understand that First Order Logic is the same as others, i.e. it can be equationalized in such a way as in Chapter 8. More importantly, this equationalization convinces us that many more theories can be equationalized similarly as long as the following key points of the equationalization is valid for those theories:

(11.0.i) reduce a deduction $\Gamma \vdash t$ to a quasi-dependent Binding Equation $\gamma \hookrightarrow t \simeq \mathbf{True}$, where $\gamma = \{u \simeq \mathbf{True} | u \in \Gamma\}$, and

(11.0.ii) reduce an inference rule

$$\frac{\Gamma_i \vdash t_i \, (i \in I)}{\Gamma \vdash t}$$

to an universal Binding Equation $\{\gamma_i \hookrightarrow t_i \simeq \mathbf{True} | i \in I\} \mapsto (\gamma \hookrightarrow t \simeq \mathbf{True})$.

11.1 Relation between $\models_{eBA}$ and $\models_{iBA}$

With regards to the extensional satisfaction $\models_{eBA}$ and the intentional satisfaction $\models_{iBA}$, from Theorem 5.0.5, we have both (11.1.a) and (11.1.b) as follows:

(11.1.a) $\mathbf{B} \models_{iBA} t \simeq u$ implies that there exists an eBA $\mathbf{A}'$ such that $\mathbf{A}' \models_{eBA} \langle \vec{x} : t \rangle \simeq \langle \vec{x} : u \rangle$, where $\{\vec{x}\} \supseteq (Free(t) \cap V) \cup (Free(u) \cap V)$;

(11.1.b) $\mathbf{A} \models_{eBA} \langle \vec{x} : t \rangle \simeq \langle \vec{x} : u \rangle$ implies that there exists an iBA $\mathbf{B}'$ such that $\mathbf{B}' \models_{iBA} t \simeq u$, where $\{\vec{x}\} \supseteq (Free(t) \cap V) \cup (Free(u) \cap V)$.

From these, we have the following theorem.

Theorem 11.1.1: *If p and q are semi-closed binding terms, i.e. $Free(p) \cap V = \emptyset$ and $Free(q) \cap V = \emptyset$, then $\Gamma \models_{iBA} p \simeq q$ iff $\Gamma \models_{eBA} p \simeq q$.*

Proof Let $Mod_{iBA}(\Gamma)$ be the class of iBAs satisfying Γ, and $Mod_{eBA}(\Gamma)$ be the class of eBAs satisfying Γ. Hence, this lemma can be rephrased as that $Mod_{iBA}(\Gamma) \models_{iBA} p \simeq q$ iff $Mod_{eBA}(\Gamma) \models_{eBA} p \simeq q$, where p and q are semi-closed.

By the proof of Theorem 5.0.5, we can get $\mathcal{K}_{iBA}$ of eBAs, and $\mathcal{K}_{eBA}$ of iBAs such that they correspond to $Mod_{iBA}(\Gamma)$ and $Mod_{eBA}(\Gamma)$ respectively. Furthermore, we know that

(i) $\mathcal{K}_{iBA} \subseteq Mod_{eBA}(\Gamma)$ and

(ii) $\mathcal{K}_{eBA} \subseteq Mod_{iBA}(\Gamma)$.

From (i), we have $Mod_{eBA}(\Gamma) \models_{eBA} p \simeq q$ implies $Mod_{iBA}(\Gamma) \models_{iBA} p \simeq q$; i.e. $\Gamma \models_{eBA} p \simeq q$ implies $\Gamma \models_{iBA} p \simeq q$.

From (ii), we have $Mod_{iBA}(\Gamma) \models_{iBA} p \simeq q$ implies $Mod_{eBA}(\Gamma) \models_{eBA} p \simeq q$; i.e. $\Gamma \models_{iBA} p \simeq q$ implies $\Gamma \models_{eBA} p \simeq q$. □

11.2 Relation between $\vdash_{iBA}$ and $\dot{\vdash}_{eBA}$

Previously, we implicitly claim that calculus $\vdash_{iBA}$ has equivalent proof power with calculus $\dot{\vdash}_{eBA}$ if we use axiomatic scheme in $\vdash_{iBA}$ instead of axioms in $\dot{\vdash}_{eBA}$. In this section, we are going to formally justify such a claim with a slight modification.

That is, let $cl_{\alpha}(\Gamma)$ be the closure of Γ under (full) substutions, (α) and (id). Then,

Theorem 11.2.1: *Given Γ, $\Gamma \dot{\vdash}_{eBA} p \simeq q$ iff $cl_{\alpha}(\Gamma) \vdash_{iBA} p \simeq q$.*

Before we formally proceed to the proof of the theorem, we introduce *derivations* for an arbitrary calculus $\vdash$ as follows.

We say that $\Gamma \vdash p \simeq q$ is derivable in $\vdash$ if there is a sequence $\Gamma_1 \vdash p_1 \simeq q_1$, $\Gamma_2 \vdash p_2 \simeq q_2$, ..., $\Gamma_\ell \vdash p_\ell \simeq q_\ell$ such that $\Gamma_\ell = \Gamma$, $p_\ell = p$, $q_\ell = q$ and $\Gamma_j \vdash p_j \simeq q_j$

$(1 \leq j \leq \ell)$ is either an axiom of $\vdash$ or a result of applying a rule in $\vdash$ to some previous $\Gamma_{i_1} \vdash p_{i_1} \simeq q_{i_1}$, $\Gamma_{i_2} \vdash p_{i_2} \simeq q_{i_2}$, ..., $\Gamma_{i_n} \vdash p_{i_n} \simeq q_{i_n}$ where $1 \leq i_k \leq j$ for $1 \leq k \leq n$. Also, ℓ is called the length of the derivation $\Gamma \vdash p \simeq q$.

Let $\dot{\vdash}^k_{eBA}$ be almost the same calculus of $\dot{\vdash}_{eBA}$ except that

(a) substitution rule (ord-sub), see Remark 5.5.1.2, is always allowed; and

(b) substitution rule (func-sub) is only allowed to be used up to the length of k in derivations.

For simplicity, we abbreviate $\dot{\vdash}^k_{eBA}$ as $\dot{\vdash}^k$.

So, obviously if $\Gamma\dot{\vdash}_{eBA}p \simeq q$, then there is a k such that $\Gamma\dot{\vdash}^k p \simeq q$. Conversely, if $\Gamma\dot{\vdash}^k p \simeq q$ for some k, then $\Gamma\dot{\vdash}_{eBA}p \simeq q$.

On the other hand, we apparently have the following lemma.

Lemma 11.2.2: *$cl_\alpha(\Gamma)\dot{\vdash}_{eBA}p \simeq q$ iff $\Gamma\dot{\vdash}_{eBA}p \simeq q$.*

Next,

Lemma 11.2.3: *For $k \geq 1$, if $cl_\alpha(\Gamma)\dot{\vdash}^{k+1} p \simeq q$, then $cl_\alpha(\Gamma)\dot{\vdash}^k p \simeq q$.*

Proof

Let $\Gamma_1\dot{\vdash}p_1 \simeq q_1$, $\Gamma_2\dot{\vdash}p_2 \simeq q_2$, ..., $\Gamma_\ell\dot{\vdash}p_\ell \simeq q_\ell$ be a derivation of $\Gamma\dot{\vdash}^{k+1}p \simeq q$, and $\Gamma_{k+1}\dot{\vdash}p_{k+1} \simeq q_{k+1}$ is a result of applying substitution rule (func-sub) to a previous $\Gamma_j\dot{\vdash}p_j \simeq q_j$ $(1 \leq j \leq k)$.

We will prove this lemma by induction on k.

case $k = 1$: j must be 1. In this case, $\Gamma_1\dot{\vdash}p_1 \simeq q_1$ must be one instance of (id), (refl) and (α). We only show the subcase for (α) since the other two subcases can be shown similarily.

For subcase (α), we do not need using substitution rule (func-sub) since premises $cl_\alpha(\Gamma)$ already contains all possible results of applying (func-sub) rule.

induction hypothesis $k \leq m$: That is, for $k \leq m$, if $cl_\alpha(\Gamma)\dot{\vdash}^{k+1} p \simeq q$, then $cl_\alpha(\Gamma)\dot{\vdash}^k p \simeq q$.

case $k = m+1$: Assuming that $\Gamma_{m+2} \dot{\vdash} p_{m+2} \simeq q_{m+2}$ is a result of applying (func-sub) to $\Gamma_j \dot{\vdash} p_j \simeq q_j$ and $1 \leq j \leq m+1$.

(a) If $j < m+1$, we can exchange $\Gamma_{m+2} \dot{\vdash} p_{m+2} \simeq q_{m+2}$ with $\Gamma_{m+1} \dot{\vdash} p_{m+1} \simeq q_{m+1}$ in the derivation, and obtain a new derivation which is then a derivation for $cl_\alpha(\Gamma) \dot{\vdash}^{m+1} p \simeq q$.

(b) If $j = m+1$, there are several subcases which result in $\Gamma_{m+1} \dot{\vdash} p_{m+1} \simeq q_{m+1}$.

subcases (α), (id) and (refl): These are the same as in case $k = 1$.

subcases (sym), (trs), (wkn), (cut) and (cmp-2): We can obtain a new derivation simply by moving the substitution to apply on its premises instead. The new derivation is actually a derivation for $cl_\alpha(\Gamma) \dot{\vdash}^{m+1} p \simeq q$.

subcase (cmp-1): Let $p = f(t_1, t_2, ..., t_n)$ and $q = f(u_1, u_2, ..., u_n)$. Thus, if the substitution does not involve f, then we can obtain a new derivation just like the above subcases.

If the substitution do involve f, say $\langle x_1, x_2, ..., x_n : v \rangle$ to substitute for f, we can obtain a new derivation, i.e. to replace $\Gamma_{m+2} \dot{\vdash} p_{m+2} \simeq q_{m+2}$ in the derivation by a derivation of

$$(*) \quad \Gamma_{m+1} \dot{\vdash} v\,[\vec{x} := \vec{t'}\,] \simeq v\,[\vec{x} := \vec{u'}\,],$$

where $\vec{t'}$ and $\vec{u'}$ are new lists of term lists $\vec{t}$ and $\vec{u}$ obtained from them by applying the substitution. It is easy to verify that a derivation of $(*)$ do not involve new substitutions besides the one for derivations of each $t'_1 \simeq u'_1$, $t'_2 \simeq u'_2$, ..., and $t'_n \simeq u'_n$. On the other hand, by repeatly applying the induction hypothesis, each derivation of $t'_j \simeq u'_j$ can be reduced to a derivation involved none of (func-sub) rules. Therfore, a new derivation is obtained for $\Gamma \dot{\vdash}^{m+1} p \simeq q$.

subcases (ξ) and (ξ^{-1}): These are left out for interested readers. □

Proof of Theorem 11.2.1 Because of Lemma 11.2.3, by induction on k we have that for any $k > 0$ $cl_\alpha(\Gamma) \dot{\vdash}^k p \simeq q$ iff $cl_\alpha(\Gamma) \dot{\vdash}^1 p \simeq q$. Actually, $\dot{\vdash}^1$ is identical with $\vdash_{iBA}$, i.e. $cl_\alpha(\Gamma) \dot{\vdash}^1 p \simeq q$ iff $cl_\alpha(\Gamma) \vdash_{iBA} p \simeq q$. So, by Lemma 11.2.2 we have that $\Gamma \dot{\vdash}_{eBA} p \simeq q$ iff $cl_\alpha(\Gamma) \vdash_{iBA} p \simeq q$. □

11.3 Possible improvement on Friedman approach

Lastly, whether the result of Chapter 5 can be improved, i.e. whether there exists another proof for soundness and completeness of $\vdash_{iBA}$ such that substitution $\vec{\varrho}$ does not require to be functional.

Following the above question, we know that functional substitutions come into the present place because of Lemma 5.5.3.12. If we can replace this lemma by a more general result, we may be able to improve our present result. However, to do so we need a significantly different proof than the one presented in Chapter 5. The reason for this is simply as follows.

From the role of Lemma 5.5.3.12 in the proof of completeness, we would lead to introduce more general translations between binding terms and terms of b-clones. For this, we need to have a technical extension of binding terms such that each function variable $f \in FV_m$ would be extended to $f^{(k)} \in FV^{ext}_{k+m}$ for every $k \in Nat$ and there is no such extension for ordinary variables. Then, we build up extended binding terms BT^{ext} by following Definition 1.1.3.1 and let BT^{ext}_0 be the extended binding terms such that all function variables appear in it must be $f^{(0)}$ for some f. Then, BT^{ext}_0 can be regarded as identical to BT. For this BT^{ext}, we can build up terms $\underline{T}^{ext}$ of b-clones by following Definition 5.3.1. Hence, we can introduce new translations between BT^{ext} and $\underline{T}^{ext}$ as below.

Definition 11.3.1 (extended $tr^{ext}_{k,\vec{x}}$): *Given $k \in Nat$, we are to define a (new) translation $tr^{ext}_{k,\vec{x}}[\![\bullet]\!]$ from binding terms to terms of b-clones such that $tr^{ext}_{k,\vec{x}}[\![t]\!] : \underline{T}_{k+|\vec{x}|}$ and $tr^{ext}_{k,\vec{x}}[\![ft]\!] : \underline{T}_{k+m+|\vec{x}|}$ for $t \in T$ and $ft \in FT_m$ respectively as follows.*

1. $tr^{ext}_{k,\vec{x}}[\![y]\!] = \underline{pr}_{k+|\vec{x}|,k+j}$ *for* $y = \vec{x}(j)$

2. $tr^{ext}_{k,\vec{x}}[\![f(\vec{t})]\!] = \underline{o}(\underline{f}^{(k)}, \underline{Pr}^{k+|\vec{x}|}_{1,k}, tr^{ext}_{k,\vec{x}}[\![\vec{t}\,]\!])$

3. $tr^{ext}_{k,\vec{x}}[\![\langle \vec{y} : t \rangle]\!] = tr^{ext}_{k,\vec{x}\vec{z}}[\![t\,[\vec{y} := \vec{z}\,]\,]\!]$ *where z_j is the least $z \in V$ such that $z \notin (Free(t) - \{\vec{y}\}) \cup \{\vec{x}\} \cup \{\vec{z}\lceil_{j-1}\}$*

4. $tr^{ext}_{k,\vec{x}}[\![\sigma(\vec{f}t,\vec{t})]\!] = \underline{o}_{k+|\vec{x}|}(tr^{ext}_{k,\vec{x}}[\![\vec{f}t]\!], tr^{ext}_{k,\vec{x}}[\![\vec{t}]\!])$

Note that $tr_{\vec{x}}$ defined in Definition 5.3.5 is the same as $tr^{ext}_{k,\vec{x}}$ (in Definition 11.3.1) when $k = 0$. For the reversed translation, we have the following.

Definition 11.3.2 (extended $\underline{tr}^{ext}$): *We are to define a (new) translation $\underline{tr}^{ext}$ from terms of b-clones to binding terms as follows.*

1. $\underline{tr}^{ext}[\![\underline{pr}_{k,j}]\!] = x_j$ *where* x_j *is the* j*th least* $x \in V$;

2. $\underline{tr}^{ext}[\![\underline{f}^{(k)}]\!] = f^{(k)}(\vec{x})$ *where* $\vec{x}(j)$ *is the* j*th least* $x \in V$ *and for some* $m \in Nat$ $|\vec{x}| = k + m$ *and* $f \in FV_m$;

3. $\underline{tr}^{ext}[\![\underline{o}(\underline{t},\vec{u})]\!] = \underline{tr}^{ext}[\![\underline{t}]\!]\,[\vec{x} := \underline{tr}^{ext}[\![\vec{u}]\!]\,]$;

4. $\underline{tr}^{ext}[\![\underline{o}_k(\vec{t},\vec{u})]\!] = \sigma(\vec{f}v,\vec{v})$ *where* $\vec{f}v(i) = \langle \vec{x^i} : \underline{tr}^{ext}[\![\vec{t}(i)]\!] \rangle$ *such that* $\vec{x^i}(j)$ *is the* $(k+j)$*th least* $x \in V$ *and* $|\vec{x^i}| = m_i$ *and where* $\vec{v}(j) = \underline{tr}^{ext}[\![\vec{u}(j)]\!]$.

With these extended translations, we seems to be able to replace the crucial substitution lemma (Lemma 5.5.3.12) by the following

$$(11.3.a) \quad tr^{ext}_{k,\vec{y}\vec{x}}[\![p\,[\vec{g} := \vec{f}t]\,]\!] \simeq tr^{ext}_{k+|\vec{y}|,\vec{x}}[\![p]\!]\,[\vec{g'} := tr^{ext}_{k,\vec{y}}[\![\vec{f}t]\!]\,]$$

where $\vec{g}(i) = f^{(k_i)}$, $\vec{g'}(i) = \underline{f}^{(k_i+k+|\vec{y}|)}$ for some $f \in FV$, $(Free(\vec{f}t) \cap V) \subseteq \{\vec{y}\}$, $(Free(p) \cap V) \subseteq \{\vec{x}\}$ and $\{\vec{x}\} \cap \{\vec{y}\} = \emptyset$. This attempt seems to work but it does not. The breakdown comes when we consider a similar result of Lemma 5.6.2.1, i.e.

$$(11.3.b) \quad \underline{tr}^{ext}[\![tr^{ext}_{k,\vec{x}}[\![\bullet]\!]\,]\!]\,[\vec{y} := \vec{y}\lceil_k, \vec{x}] \simeq \bullet$$

where $\vec{y}(j)$ is the jth least $y \in V$.

This time, in order to have the equality (11.3.b), we are required to have some extra laws like

$$(11.3.c) \quad f^{(k)}(\vec{x},\vec{y}\,) \simeq f^{(0)}(\vec{y}\,)$$

where $|\vec{x}| = k$ and $|\vec{y}|$ is the same as the arity of $f^{(0)}$. This (11.3.c) further would impose certain requirement on the substitution rule (e.g on (11.3.a)).

For example, we expect something like

$$f(x_1)\,[f := \langle x_3 : g(x_3, x_2)\rangle] \simeq g(x_1, x_2),$$

or

$$f^{(0)}(x_1)\,[f^{(0)} := \langle x_3 : g^{(0)}(x_3, x_2)\rangle] \simeq g^{(0)}(x_1, x_2).$$

With respect to (11.3.a) and $tr^{ext}_{k,x_2x_1}[\![\bullet]\!]$, we have

$$\begin{aligned}
&tr^{ext}_{k,x_2x_1}[\![f(x_1)\,[f := \langle x_3 : g(x_3, x_2)\rangle]\,]\!] \\
&\simeq tr^{ext}_{k+1,x_1}[\![f(x_1)]\!]\,[f^{(k+1)} := tr^{ext}_{k,x_2}[\![\langle x_3 : g(x_3, x_2)\rangle]\,]\!] \quad by\ (11.3.a) \\
&= \underline{o}(\underline{f}^{(k+1)}, \underline{Pr}^{k+2}_{1,k+1}, tr^{ext}_{k+1,x_1}[\![x_1]\!])\,[\underline{f}^{(k+1)} := tr^{ext}_{k,x_2x_3}[\![g(x_3, x_2)]\!]\,] \\
&= \underline{o}(\underline{f}^{(k+1)}, \underline{Pr}^{k+2}_{1,k+1}, \underline{pr}_{k+2,k+2})\,[f^{(k+1)} := \underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, tr^{ext}_{k,x_2x_3}[\![x_3]\!], tr^{ext}_{k,x_2x_3}[\![x_2]\!])] \\
&\qquad by\ (Ax_b\text{-}right\text{-}proj) \\
&\simeq \underline{f}^{(k+1)}\,[f^{(k+1)} := \underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, \underline{pr}_{k+2,k+2}, \underline{pr}_{k+2,k+1}) \\
&= \underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, \underline{pr}_{k+2,k+2}, \underline{pr}_{k+2,k+1})
\end{aligned}$$

and

$$\begin{aligned}
&tr^{ext}_{k,x_2x_1}[\![g(x_1, x_2)]\!] \\
&= \underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, tr^{ext}_{k,x_2x_1}[\![x_1]\!], tr^{ext}_{k,x_2x_1}[\![x_2]\!]) \\
&= \underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, \underline{pr}_{k+2,k+2}, \underline{pr}_{k+2,k+1}).
\end{aligned}$$

With respect to (11.3.b), we have

$$\begin{aligned}
&(\underline{tr}^{ext}[\![\underline{o}(\underline{g}^{(k)}, \underline{Pr}^{k+2}_{1,k}, \underline{pr}_{k+2,k+2}, \underline{pr}_{k+2,k+1})]\!])\,[\vec{y} := \vec{y}\lceil_k x_2x_1] \\
&= (g^{(k)}(\vec{y})\,[\vec{y} := \underline{tr}^{ext}[\![\underline{Pr}^{k+2}_{1,k}]\!], \underline{tr}^{ext}[\![\underline{pr}_{k+2,k+2}]\!], \underline{tr}^{ext}[\![\underline{pr}_{k+2,k+1}]\!]\,])\,[\vec{y} := \vec{y}\lceil_k x_2x_1] \\
&= (g^{(k)}(\vec{y})\,[\vec{y} := \vec{y}\lceil_k \vec{y}(k+2)\vec{y}(k+1)])\,[\vec{y} := \vec{y}\lceil_k x_2x_1] \\
&= g^{(k)}(\vec{y}\lceil_k \vec{y}(k+2)\vec{y}(k+1))\,[\vec{y} := \vec{y}\lceil_k x_2x_1] \\
&= g^{(k)}(\vec{y}\lceil_k, x_1x_2).
\end{aligned}$$

Then, by (11.3.b) we would have

$$g^{(k)}(\vec{y}\lceil_k, x_1, x_2) \simeq g^{(0)}(x_1, x_2)$$

which is an example of (11.3.c).

Therefore, we are lead to a circular argument of improving substitution rule i.e. for an arbitrary substitution $\vec{\varrho}$, we need $\vec{\varrho}(f^{(m)}) \simeq \vec{\varrho}(f^{(0)})$ (for each $m \in Nat$) in order to be used in proofs. A conclusion from the above analysis is that a completely

different proof than the one presented in Chapter 5 is needed in order to improve the results of that chapter. Subsequently, $\vdash_{iBA}$ has to use axiomatic schemas instead of axioms in order to share a same proof power with $\dot{\vdash}_{eBA}$. For example, in Lambda Calculus we need (β-schema) for $\vdash_{iBA}$ to share the same proof power of $\dot{\vdash}_{eBA}$ with (β-axiom). So, what is the essential difference between Birkhoff's approach (in Chapter 3) and Friedman's approach (in Chapter 5) remained to be seen.

Part VII

Bibliography and indices

Bibliography

[1] C. A. A. P. Abar. *Descricao e Paraconsistencia.* PhD thesis, Pontifical Catholic University of Sao Paulo, 1985.

[2] C. A. A. P. Abar and M. Yamashita. Remarks on variable binding term operators. *Polish Acad. Sci. Inst. Philos. Social. Bull. SECT LOGIC*, 15(4):145–151, 1986.

[3] P. Aczel. A general church-rosser theorem. unpublished, September 1978.

[4] P. Aczel. Frege structures and notations of propositions, truth and set. In *The Kleene Symposium*, Studies in Logic and the foundation of mathematics, pages 31–59. North-Holland, 1980.

[5] P. Aczel. Abstract models of λ-calculus. unpublished, October 1981.

[6] M. A. Arbib and E. G. Manes. *Arrows, Structures and Functors : categorical imperitives.* Academic Press, 1975.

[7] H. P. Barendregt. *The Lambda Calculus : its Syntax and Semantics.* Studies in Logic and the foundation of mathematics. North-Holland, revised ed edition, 1984.

[8] D. W. Barnes and J. M. Mack. *An Algebraic Introduction to Mathematical Logic*, volume 22 of *Graduate Text in Mathematics.* Springer-Verlag, 1975.

[9] M. Barr and C. Wells. *Topos, Triples and Theories.* Springer-Verlag, 1985.

[10] J. Barwise. An introduction to first order logic. In *Handbook of Mathematical Logic*, Studies in Logic and the foundation of mathematics. North-Holland, 1977.

[11] J. Barwise. The situation in logic, I, II, III. Technical Report 2, 21 and 26, Center for Study of Language and Information, Stanford University, 1984.

[12] J. Barwise and S. Feferman. *Model-Theoretic Logics.* Perspectives in Mathematical Logics. Springer-Verlag, 1985.

[13] M. J. Beeson. *Foundation of Constructive Mathematics.* Springer-Verlag, 1985.

[14] J. A. Bergstra and J. V. Tucker. Algebraic specification of computable and semicomputable data types. *Theoretical Computer Science*, 50:137–181, 1987.

[15] G. Berry, P. L. Curien, and J.-J. Lévy. Full abstraction for sequential languages : the state of the art. In *Algebraic Methods in Semantics*, pages 89–132. Cambridge University Press, 1985.

[16] G. Birkhoff. On the structure of abstract algebras. *Proceedings of Cambridge Philosophy Society*, 31:433–454, 1935.

[17] S. Bloom and R. Tindell. Varieties of if ... then ... else. *SIAM Journal on Computing*, 12:677–707, 1983.

[18] T. Bolognesi and E. Brinskma. Intorduction to the iso specification language lotos. *Computer Networks & ISDN Systems*, 14(1):25–29, January 1987.

[19] G. S. Boolos and R. C. Jeffrey. *Computability and Logic.* Cambridge University Press, 2nd edition, 1980.

[20] V. Breazu-Tannen. Combining algebra and higher-order types. Technical Report MS-CIS-88-21, Department of Computer and Information Science, School of Engineering and Applied Science, University of Pennsylvania, March 1988.

[21] M. W. Bunder. Predicate calculus of arbitrary high finite order. *Arch. Math. Logik*, 23:1–10, 1983.

[22] M. W. Bunder, J. R. Hindley, and J. P. Seldin. On adding (ξ) to weak equality in combinatory logic. *Journal of symbolic logic*, 54(2):590–607, 1989.

[23] S. Burris and H. P. Sankappanavar. *A Course in Universal Algebra.* Graduate Text in Mathematics. Springer-Verlag, 1981.

[24] L. Cardelli and P. Wegner. On understanding types, data abstraction and polymorphism. *ACM Computing Surveys*, 17(4), 1985.

[25] C. C. Chang and M. J. Keisler. *Model Theories*, volume 73 of *Studies in Logic and the foundation of mathematics.* North-Holland, 1973.

[26] A. Church. *The Calculi of Lambda-conversion*, volume 6 of *AMS Annals of Mathematics Studies.* American Mathematical Society, 1941.

[27] A. Church. *Introduction to Mathematical Logic.* Princeton University Press, 1956.

[28] P.M. Cohn. *Universal Algebra.* Harper and Row, New York, 1980.

[29] T. Coquand and G. Huet. The calculus of constructions. *Information and Computation (or Information and Control)*, 76:95–120, 1988.

[30] J. Corcoran, W. Hatcher, and J. Herring. Variable binding term operators. *Zeitschr. f. Math. Logik und Grundlagen d. Math.*, 18:177–182, 1972.

[31] H. B. Curry. Some philosophical aspects of combinatory logic. In *The Kleene Symposium*, Studies in Logic and the foundation of mathematics. North-Holland, 1980.

[32] H. B. Curry and R. Feys. *Combinatory Logic, vol. I.* Studies in Logic and the foundation of mathematics. North-Holland, 1958.

[33] H. B. Curry, J. R. Hindley, and J. P. Seldin. *Combinatory Logic vol.II.* Studies in Logic and the foundation of mathematics. North-Holland, 1972.

[34] N. C. A. da Costa. A model-theoretical approach to variable binding operators. In *Mathematical Logic in Latin American*, Studies in Logic and the foundation of mathematics. North-Holland, 1980.

[35] M. Davis, Y. Matijasevic, and J. Robinson. Hilbert's 10th problem, diophantine equations : positive aspects of a negative solution. In *AMS Proceedings of Symposiums in Pure Mathematics*, volume 28. American Mathematical Society, 1976.

[36] N. Dershowitz and J. P. Jouannaud. Rewrite systems. Technical Report rapport de recherche n° 478, L.R.I. Center d'Orsay, Universite de Paris-Sud, France, Avril 1989.

[37] H. Ehrig, J. Loeckx, and B. Mahr. A remark on the equational calculus for many-sorted equational logic. *EATCS Bulletin*, 30, November 1986.

[38] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1 : Equations and initial semantics*, volume 6 of *EATCS Monographs on Theoretical Computer Science.* Springer-Verlag, 1985.

[39] A. P. Ershov. Algorithmatic operators. *Probl. Kib.*, 3:5 - 48, 1960. transl. vol.3 (1960) pp.696 - 763.

[40] Iu. L. Ershov. Theorie der numerierungen. *Zeitschr. f. Math. Logik und Grundlagen d. Math.*, 19, 21 and 23, 1973, 1975 and 1977.

[41] J. Farres-Casals. Proving correctness of constructor implementation. Technical Report 72, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989.

[42] S. Feferman. Lectures on proof theory. In *Lecture Notes in Mathematics*, volume 70, pages 1–108. Springer-Verlag, 1968.

[43] S. Feferman. Theories of finite types related to mathematical practice. In *Handbook of Mathematical Logic*, Studies in Logic and the foundation of mathematics, pages 913–971. North-Holland, 1977.

[44] J. E. Fenstad. *Axiomatic approach to general recursive theory.* Perspectives in Mathematical Logics. Springer-Verlag, 1980.

[45] J. E. Fenstad, P. Halvorsen T. Langholm, and J. von Benthem. Equations, schemata and situations : A framework for linguistic semantics. Technical Report 29, Center for Study of Language and Information, Stanford University, 1985.

[46] H. Friedman. Equality between functionals. In *Lecture Notes in Mathematics*, volume 453, pages 22–37. Springer-Verlag, 1975.

[47] J. Y. Girard. System F of variable types. *Theoretical Computer Science*, 45:159–192, 1986.

[48] K. Gödel. On undecidable propositions of formal mathematical systems (1934). In *The Undecidable*, pages 39–74. Raven Press Book, ltd, 1965.

[49] J. Goguen and R. Burstall. Introducing institutions. In *Lecture Notes in Computer Science*, volume 164. Springer-Verlag, 1984.

[50] J. Goguen and R. Burstall. Institutions : Abstract model theory for specification and programming. Technical Report 30, Center for Study of Language and Information, Stanford University, 1985.

[51] J. A. Goguen and J. Meseguer. Completeness of many-sorted equational logic. *ACM SigPlan Notice*, 1981.

[52] J. A. Goguen and J. Meseguer. Universal realization, persistent interconnection and implementation of abstract modules. In *Lecture Notes in Computer Science*, volume 140. Springer-Verlag, 1982.

[53] J. A. Goguen and J. Meseguer. Completeness of many-sorted equational logic. *Houston Journal of Mathematics*, 1985. (also in Report No. 15, CSLI, Stanford University, 1984).

[54] J. A. Goguen and J. Meseguer. Remark on a remark on the equational calculus for many-sorted equational logic. *EATCS Bulletin*, 30, November 1986.

[55] M. J. C. Gordon, R. Milner, and C. Wadsworth. *Edinburgh LCF : a mechanical logic of computation*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[56] G. Grätzer. *Universal Algebra*. Springer-Verlag, 2nd edition, 1979.

[57] I. Guessarian and J. Meseguer. On the axiomatization of 'if-then-else'. *SIAM Journal on Computing*, 16:332–357, 1987.

[58] P. R. Halmos. Polyadic boolean algebras. *Proceedings of National Academy of Sciences*, 40, 1954.

[59] P. R. Halmos. *Algebraic Logic*. Chelsea, New York, 1962.

[60] P. R. Halmos. *Lectures on Boolean Algebras*, volume 1 of *Van Nostrand Mathematical Studies*. Springer-Verlag, 1963.

[61] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. In *proceedings of the 2nd logic in computer science*, Ithaca, New York, June 1987. IEEE.

[62] R. Harper, R. Milneer, and M. Tofts. The definition of standard ml : version 2. Technical Report ECS-LFCS-88-62, Laboratory for Foundation of Computer Science, University of Edinburgh, August 1988.

[63] W. Hatcher. *The Logical Foundations of Mathematics*. Pergamon Press, 1982.

[64] S. Hayashi and H. Nakano. *PX : a Computational Logic*. Foundation of Computing. MIT Press, 1989.

[65] L. Henkin. Logic of equalities. *AMS Mathematical Monthly*, 1977.

[66] L. Henkin, J. D. Monk, and A. Tarski. *Cylindric Algebras : Part I*, volume 64 of *Studies in Logic and the foundation of mathematics*. North-Holland, 1971.

[67] L. Henkin, J. D. Monk, and A. Tarski. *Cylindric Algebras : Part II*, volume 115 of *Studies in Logic and the foundation of mathematics*. North-Holland, 1985.

[68] M. Hennessy and R. Milner. Algebraic laws for nondeterministism and concurrency. *Journal of the Association of Computing Machinery*, 32(1), January 1985.

[69] M. Hennessy and G. Plotkin. A term model for ccs. In *Lecture Notes in Computer Science*, volume 88. Springer-Verlag, 1979.

[70] H. Herrlich and G. E. Strecker. *Category Theory*. Allyn and Bacon, Inc., 1973.

[71] J. R. Hindley and J. P. Seldin. *Introduction to Combinators and lambda-calculus*, volume 1 of *London Mathematical Society Student Texts*. Cambridge University Press, 1986.

[72] P. G. Hinman. *Recursive-theoretical Hierarchies*. Perspectives in Mathematical Logics. Springer-Verlag, 1978.

[73] C.A.R. Hoare. Communicating sequential processes. *the Communicaton of the Assocation of Computing Machinery*, 28, 1978.

[74] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[75] Julia L. Hook. A note on interpretations of many-sorted theories. *Journal of symbolic logic*, 50(2):372–374, 1985.

[76] W. A. Howard. Hereditarily majorizable functionals of finite type. In *Lecture Notes in Mathematics*, volume 344, pages 454–461. Springer-Verlag, 1973.

[77] W. A. Howard. The formulae-as-types notion of construction. In *To H. B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism.* Academic Press, 1980.

[78] G. Huet and D. C. Oppen. Equations and rewrite rules : a survey. In *Formal Language Theory : Perspective and Open Problems*, pages 349–405. Academic Press, 1980.

[79] ISO. LOTOS – a formal description technique based on the temporal ordering of observational behaviour. Technical Report ISO DIS 8807, ISO/TC 97SC 21 N, International Organization for Standardization, July 1987.

[80] T. Jech. *Set theory.* Academic Press, 1978.

[81] J. P. Jones. Universal diophantine equations. *Journal of symbolic logic*, 47(3):549–571, 1982.

[82] A. Kanamori and M. Magidor. The evolution of large cardinal axioms in set theory. In *Higher set theory*, volume 669 of *Lecture Notes in Mathematics*, pages 99–275. Springer-Verlag, 1978.

[83] A. S. Kechris and Y. N. Moschovakis. Recursion in higher types. In *Handbook of Mathematical Logic*, Studies in Logic and the foundation of mathematics, pages 681–737. North-Holland, 1977.

[84] G. M. Kelly and R. Street. Review of the elements of 2-categories. In *Lecture Notes in Mathematics*, volume 420, pages 75–103. Springer-Verlag, 1974.

[85] S. C. Kleene. A theory of positive integers in formal logic. *American Journal of Mathematics*, 57:153–173, 1935.

[86] S. C. Kleene. *Introduction to Metamathematics.* D. Van Nostrand Company, Inc, 1952.

[87] S. C. Kleene. Recursive functionals and quantifiers of finite types I. *Transaction of American Mathematical Society*, 91, 1959.

[88] S. C. Kleene. Lambda-definable functions of finite types. *Fundamenta Mathematicae*, 520:281–303, 1962.

[89] J. W. Klop. *Combinatory Reduction System.* PhD thesis, CWI, Amsterdam, The Netherland, 1980.

[90] J. W. Klop. Term rewriting sytems : from church-rosser to knuth-bendix and beyond. In *Automata, Languages and Programming*, volume 443 of *Lecture Notes in Computer Science*, pages 350–369. Springer-Verlag, 1990.

[91] A. Knobel. *Full abstraction of Intutionistic λ-models (?).* PhD thesis, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989(?).

[92] P. G. Kolaitis. Canonical forms and hierarchies in generalized recursion theory. In *AMS Proceedings of Symposiums in Pure Mathematics*, volume 42. American Mathematical Society, 1985.

[93] C. P. J. Koymans. Models of the lambda calculus. *Information and Computation (or Information and Control)*, 52, 1982.

[94] G. Kreisel and W. Tait. Finite definability of number-theoretic functions and parametri completeness of equational calculus. *Zeitschr. f. Math. Logik und Grundlagen d. Math.*, 7:28–38, 1961.

[95] M. Li and P. M. B. Vitányi. Two decades of applied kolmogorov complexity. In *Proceedings of the 3rd IEEE Conference on Structure in Complexity Theory.* IEEE, 1988.

[96] J. W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, 1984.

[97] S. MacLane. *Categories for the Working Mathematician*, volume 5 of *Graduate Text in Mathematics.* Springer-Verlag, 1971.

[98] S. MacLane. Part I history of algebra, history of abstract algebra : origin and decline of a movement. *American Mathematical Herritage : Algebra and Applied Mathematics*, 13, 1981.

[99] A. I. Mal'cev. *Algebraic Systems*, volume 192 of *Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen Band.* Springer-Verlag, 1973.

[100] R. L. Martin. *Recent Essays on Truth and the Liar Paradox.* Oxford University Press, 1984.

[101] Per Martin-Löf. *Intuitionistic Type Theory.* Bibliopolis, 1984.

[102] Y. V. Matijasevic. Diophantine representation of recursively enumerable predicates. In *Proceedings of the 2nd Scandinavian Logic Symposium*, volume 63 of *Studies in Logic and the foundation of mathematics.* North-Holland, 1971.

[103] Y. V. Matijasevic and Julia Robison. Reduction of an arbitrary diophantine equations to one in 13 unknowns. *Acta Arithematica*, 27:521–553, 1975.

[104] A. H. Mekler and E. M. Nelson. Equational bases for if-then-else. *SIAM Journal on Computing*, 16(3), June 1987.

[105] A. R. Meyer. What is a model of the lambda calculus? *Information and Computation (or Information and Control)*, 52, 1982.

[106] R. Milner. Fully abstract models of typed λ-calculus. *Theoretical Computer Science*, 4:1–22, 1977.

[107] R. Milner. A theory of type polymorphism in programming. *Journal of Computer and System Science*, 17:348–375, 1978.

[108] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, 1980.

[109] R. Milner. *Concurrency and communication.* Prentice Hall, 1989.

[110] J. Mitchell and A. R. Meyer. Second-order logical relations (extended abstract). In *Lecture Notes in Computer Science*, volume 193, pages 225–236. Springer-Verlag, 1985.

[111] J. C. Mitchell. Polymorphic type inference and containment. *Information and Computation (or Information and Control)*, 76, 1988.

[112] E. Moggi. *The Partial Lambda Calculus.* PhD thesis, Laboratory for Foundation of Computer Science, University of Edinburgh, 1988.

[113] J. D. Monk. *Mathematical Logic.* Graduate Text in Mathematics. Springer-Verlag, 1976.

[114] K. Mulmuley. *Fully Abstraction and Semantics Equivalence.* PhD thesis, Computer Science Department, Carnegie-Mellon University, 1985.

[115] M. J. O'Donnell. *Equational logic as a programming language.* Foundation of Computing. MIT Press, 1985.

[116] P. Padawitz. *Computing in Horn Clause Theories*, volume 16 of *EATCS Monographs on Theoretical Computer Science.* Springer-Verlag, 1988.

[117] C. Paul-Mohring. Extracting f^{ω} program from proofs in the calculus of constructions. In *ACM proceedings of Principles of Programming Languages.* Association of Computing Machinery, 1989.

[118] G. D. Plotkin. A set-theoretical definition of application. Technical Report Memo MIP-R-95, School of AI, University of Edinburgh, 1972.

[119] G. D. Plotkin. A powerdomain construction. *SIAM Journal on Computing*, 5(3):452–487, 1976.

[120] G. D. Plotkin. Lcf considered as a programming language. *Theoretical Computer Science*, 5, 1977.

[121] G. D. Plotkin. Lambda-definability in the full type hierachy. In *To H. B. Curry : Esseys on Combinatory Logic, Lambda Calculus and Formalism.* Academic Press, 1980.

[122] G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer Science Department, Aarhus University, September 1981.

[123] G. D. Plotkin. Domain theory. lecture notes for PG course, 1985/6.

[124] G. D. Plotkin. A illative theory of relations. to appear in "Situation Theory and its application" (a volume in lecture series of CSLI, Stanford University), 1990.

[125] J. C. Reynolds. Variable binding. Chapter 4 in "Semantics as a Design Tool" (advanced topics in theory), January 1989.

[126] Julia Robinson. General recursive functions. *Proceedings of American Mathematical Society*, 1:703–718, 1950.

[127] Julia Robinson. Recursive functions of one variables. *Proceedings of American Mathematical Society*, 19:815–820, 1968.

[128] D. E. Rydeheard and R. M. Burstall. Monads and theories : a survey for computation. In *Algebraic Methods in Semantics*, pages 575–605. Cambridge University Press, 1985.

[129] T. Sakurai. Theory of formulas-as-types (preliminary version 1). (Japan), 1988.

[130] D. Sannella and A. Tarlecki. Extended ml : an institution-independent framework for formal program development. In *Lecture Notes in Computer Science*, volume 240. Springer-Verlag, 1985.

[131] D. Sannella and A. Tarlecki. Toward formal development of ml programs : foundations and methodology. Technical Report 71, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989.

[132] M. Sato. Symbolic proof theory – sets, propositions and truths. presented in European workshop on typed lambda calculus (Jumelage meeting) in Edinburgh 1989, 1989.

[133] D. Scott. Lambda calculus and recursion theory. In *Proceedings of the 3rd Scandinavian Logic Symposium*, Studies in Logic and the foundation of mathematics, pages 154–193. North-Holland, 1975.

[134] D. Scott. Data types as lattices. *SIAM Journal on Computing*, pages 522–587, 1976.

[135] D. Scott. Identity and existence in intuitionistic logic. In *Applications of Sheaves*, Lecture Notes in Mathematics. Springer-Verlag, 1979.

[136] D. Scott. Lambda calculus : some models, some philosophy. In *The Kleene Symposium*, Studies in Logic and the foundation of mathematics. North-Holland, 1980.

[137] R. A. G. Seely. Modeling computations : A 2-categorical framework. In *proceedings of the first IEEE annual conference on logic in computer science.* IEEE, 1987.

[138] S. Shapiro. Second-order languages and mathematical practice. *Journal of symbolic logic*, 50(3):714–742, 1985.

[139] J. Sheonfield. *Mathematical Logic.* Reading : Addison-Wesley, 1967.

[140] Hu Shi-hua and Lu Zong-wan. *Introduction to Mathematical Logic*, volume 1 and 2 of *introductions to foundations of modern mathematics.* Science Press, Beijing, 1983. (in Chinese).

[141] M. Smith and G. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM Journal on Computing*, 11(4), 1982.

[142] R. Statman. Completeness, invariance and λ-definability. *Journal of symbolic logic*, 47(1):17–26, 1982.

[143] R. Statman. Equality between functionals revisited. In *Harvey Friedman's Research on Foundations of Mathematics*, volume 117 of *Studies in Logic and the foundation of mathematics*, pages 331–338. North-Holland, 1985.

[144] R. Statman. Logical relations and the typed lambda calculus. *Information and Computation (or Information and Control)*, 1985.

[145] A. Stoughton. *Fully Abstract Models of Programming Languages.* PhD thesis, Department of Computer Science, University of Edinburgh, November 1986.

[146] A. Stoughton. Substitution revisited. (Department of Computer Science, University of Edinburgh), 1987.

[147] J. Stoy. *Denotational Semantics : the Scot-Strachey Approach to Programming Language Theory.* The MIT Press, 1977.

[148] H. R. Strong. Translating recursive equations into flow charts. *Journal of Computer and System Science*, 5:254–285, 1971.

[149] Y. Sun. Specification of protocols and its independence of communication mechanisms. In *Informatik-Fachberichte*, volume 130. Springer-Verlag, 1987.

[150] Y. Sun. Equational characterizations of binding. Technical Report 94, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989. presented in the European workshop on typed lambda calculi (Jumelage meeting), Edinburgh (1989).

[151] Y. Sun. Logical design of vlsi circuit with extension of uncertainty (or monotonic functional completeness of kleene ternary logic). Technical Report 95, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989.

[152] Y. Sun. Self-independent petri-nets (or a deadlock free paradigm). Technical Report 98, Laboratory for Foundation of Computer Science, University of Edinburgh, 1989.

[153] Y. Sun. Axiomatization of calculus of constructions. In *preliminary proceedings of the symposium on Constructivity in Computer Science*, pages 106–126. Trinity University, San Antonio, Texas, June 1991.

[154] W. Tait. Intentional interpretations of functionals of finite types. *Journal of symbolic logic*, 32:198–212, 1967.

[155] G. Takeuti and W. M. Zaring. *Introduction to axiomatic set theory*, volume 1 of *Graduate Text in Mathematics*. Springer-Verlag, 1971.

[156] A. Tarski. Equational logic and equational theories of algebras. In *Contribution to Mathematical Logic*, Studies in Logic and the foundation of mathematics, pages 275–288. North-Holland, 1968.

[157] A. Tarski and S. Givant. *A Formalization of Set Theory without Variables*, volume 41. American Mathematical Society Colloquium Publications, 1987.

[158] W. Taylor. Equational logic. In *Universal Algebras*. Springer-Verlag, 2nd edition, 1979. (in appendix).

[159] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics : an introduction (I and II)*, volume 121 and 123 of *Studies in Logic and the foundation of mathematics*. North-Holland, 1988.

[160] D. van Dalen. *Logic and Structure*. Springer-Verlage, 2nd edition, 1983.

[161] D. van Dalen. Intuitionistic logic. In *Handbook of Philosophical Logic vol.III : alternatives in classical logic*, pages 225–339. D. Reidel Publishing Company, 1986.

[162] D. Walker. Bisimulation equivalence and divergence in ccs. Technical Report ECS-LFCS-87-29, Laboratory for Foundation of Computer Science, University of Edinburgh, June 1987.

[163] H. Wang. A variant of turing's theory of calculating. *Journal of the Association of Computing Machinery*, 4:63–92, 1957.

[164] H. Wang. Logic of many sorted theories. *Journal of symbolic logic*, 17:105–116, 1959.

[165] D. Westerstahl. Quantifiers in formal and natural languages. In *Handbook of Philosophical Logic*, volume IV. D. Reidel, 1989.

[166] Chaochen Zhou and C. A. R. Hoare. Partial correctness of communicating sequential processes. In *proceedings of the 2nd International Conference on Distributed Computing Systems*, Paris, April 1981.

Index of Formulae and Symbols

Symbol	Definition	Page
$\langle \underbrace{_, _, \ldots, _}_{n \text{ times}} : _ \rangle$	n-ary binding primitive	22
n	natural number	22
ω	the least limit ordinal	22
Σ^{BO}	binding signature	23
S	set of sorts	23
$\times$	(set) cartesian product	23
*	Kleene star operator	23
$Sort(\Sigma^{BO})$	set of sorts for binding signatures	23
$\exists$	existential quantifier	23
λ	λ-abstraction operator in Lambda Calculus	23
let	**let**-operator in ML	23
in-α	input operator in CCS	23
$\rightarrow$	(set) function space or interpretations	23
$\Rightarrow$	function sorts	23
Nat	set of natural numbers	24
V	set of ordinary variables	24
x, y, x	ordinary variables	24
$\vec{x}, \vec{y}, \vec{z}$	lists of ordinary variables	24
FV	family of sets of function variables	24
FV_m	set of function variables with arity $m \in Nat$	24
f	function variable	24
BT	Binding Terms	24
T	set of ordinary terms	24

FT	family of sets of function terms	24
FT_m	set of function terms with arity $m \in Nat$	24
ε	empty list	25
t_j	ordinary terms with subscript j	25
$\vec{t}(j)$	jth element in term list $\vec{t}$	25
$\|\vec{t}\|$	length of term list $\vec{t}$	25
$\vec{ft}$	list of function terms	25
ft_i	function term with subscript i	25
(β)	β-conversion axiom in Lambda Calculus	26
app	application operator in Lambda Calculus	26
$\simeq$	(meta) equality or Indistinguishability	26
(λ-**let**)	axiom of the equivalence among λ, **app** and **let** operators	26
($\exists$-*elim*)	existential quantifier elimination rule	26
$\hookrightarrow$	quasi-dependent implication	27
True	constant representing value *true*	27
$\frac{\bullet}{\bullet}$	inference rule	27
$\bullet$	arbitrary object	27
t, u, v	(ordinary) terms	27
if b **then** t **else** u	**if**-statement	27
$b \supset (t, u)$	**if**-statement	27
(?-**then**-τ-elim)	rule for eliminating τ in **if**-statements prefixed by an input	27
$\alpha?x.t$	**in**-$\alpha(\langle x : t \rangle)$ in CCS	27
$\mapsto$	dependent implication	27
$\mathcal{F}_m(A)$	subset of function space $A^m \rightarrow A$	29
m	natural number	29
0	zero in natural numbers	29
$\mathcal{F}$	pair like $< A', \{\mathcal{F}_m(A) \| m \in Nat\} >$ where $A' \subseteq A$	29
$\subseteq$	inclusion or sub-set	29
$C_{m,a}$	m-ary constant function mapping everything to a	29
$\vec{a}$	list of elements	29

$\pi_{m,i}$	m-ary ith-projection function	29
$\odot$	(extensional) composition functional	29
$g(\vec{h})$	$g\odot < \vec{h} >$ and $g\odot < \vec{h} > (\vec{a}) = g(h_1(\vec{a}), h_2(\vec{a}), ..., h_{\lvert\vec{h}\rvert}(\vec{a}))$	29
$\circ$	(intensional) composition functional	30
σ	Binding Operator	30
$\Sigma^{BO}_{<\vec{m},n>}$	set of Binding Operators with arity $<\vec{m},n>$	30
$\mathcal{A}_\sigma$	interpretation functional for BO σ in eBA $\mathbf{A}$	30
A^n	product $\underbrace{A \times A \times ... \times A}_{n \text{ times}}$ where $n \in Nat$	30
$\mathcal{F}_{\vec{m}}$	$\mathcal{F}_{m_1} \times \mathcal{F}_{m_2} \times ... \times \mathcal{F}_{m_\ell}$	30
k	natural number	30
$\vec{a^i}$	list with superscript i	30
$\mathbf{A}$	extensional Binding Algebra (or eBA)	30
$\mathcal{A}_\sigma$	functional for the interpretation of σ in $\mathbf{A}$	30
$\sigma^{\mathbf{A}}$	functional for the interpretation of σ in $\mathbf{A}$	30
σ	either a BO, first order operation, or its interpretation	30
$\vec{\rho}$	family of environments (or variable assignments)	31
ρ	(ordinary) environment	31
ρ_k	function environment with arity $k \in Nat$	31
$<\rho,\vec{\varphi}>$	pair of an ordinary environment and a family of function environments	31
$\vec{\varphi}$	family of function environments	31
φ_k	function environment with arity $k \in Nat$	31
$<\rho,\varphi>$	simplified version of $<\rho,\vec{\varphi}>$	31
$\rho[a/x]$	single-updated environment from environment ρ	31
$\rho[\vec{a}/\vec{x}]$	multiple-updated environment from environment ρ	31
$\lvert\vec{a}\rvert$	length of list $\vec{a}$	32
$\lvert\vec{x}\rvert$	length of variable list $\vec{x}$	32
$\mathcal{A}$	interpretation associated with $\mathbf{A}$	32
$\mathcal{A}[\![\bullet]\!](\rho,\varphi)$	interpretation in $\mathbf{A}$ with environment $<\rho,\varphi>$	32
$\vec{t}$	term list	32
$\mathcal{A}[\![\vec{t}]\!](\rho,\varphi)$	list of interpretations in $\mathbf{A}$ associated with term list $\vec{t}$	32

$\mathcal{A}[\![\vec{ft}]\!](\rho,\varphi)$	list of interpretations in $\mathbf{A}$ associated with function term list $\vec{ft}$	32
$\|\vec{ft}\|$	length of function term list $\vec{ft}$	32
p, q	binding terms, i.e. either ordinary terms or function terms	32
$p \simeq q$	Binding Equation	32
$\models_{eBA}$	(extensional) satisfaction in eBAs	32
$\mathbf{A} \models_{eBA} p \simeq q$	(extensional) satisfaction associated with eBA $\mathbf{A}$	32
$\models_{eBA} p \simeq q$	(extensional) satisfaction (independent of particular $\mathbf{A}$)	33
γ	set of either (first order) equations or Binding Equations	33
Δ	either a (first order) equation or a Binding Equation	33
$\gamma \mapsto \Delta$	either a (first order) dependent equation or a dependent Binding Equation	33
$\gamma \hookrightarrow \Delta$	either a (first order) quasi-dependent equation or a quasi-dependent Binding Equation	33
$\mathbf{A} \models^{d}_{eBA} \gamma \mapsto \Delta$	(extensional) dependent satisfaction associated with eBA $\mathbf{A}$	33
$\mathbf{A} \models^{q}_{eBA} \gamma \hookrightarrow \Delta$	(extensional) quasi-dependent satisfaction associated with eBA $\mathbf{A}$	33
M	set of indices	34
Ω	uBE, i.e. the form of $\{\gamma^{(m)} \hookrightarrow \Delta^{(m)} \mid m \in M\} \mapsto (\gamma \hookrightarrow \Delta)$	34
$\mathbf{A} \models \Omega$	(extensional) universal satisfaction associated with $\mathbf{A}$	34
$\mathbf{T}$	either (first order) term algebra or term eBA	34
$\mathcal{F}^{\mathbf{T}}$	explicitly closed family associated with term eBA $\mathbf{T}$	34
$\zeta : \mathbf{A} \to \mathbf{A}'$	eBH from eBA $\mathbf{A}$ to another eBA $\mathbf{A}'$	34
$\bullet_{[p]}$	object in term eBA $\mathbf{T}$ associated with binding term p	35
ζ	extensional Binding Homomorphism (or eBH)	35
$Ker(\zeta)$	binding kernel of eBH ζ	35
∇_ζ	binding core of eBH ζ	35
$\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$	intersection of kernels of eBHs	35
$\mathbf{T}/\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$	quotient term eBA over kernel $\bigcap_{\zeta:\mathbf{T}\to\mathbf{A}} Ker(\zeta)$	35
ν_ζ	natural eBH associated with kernel of ζ	35
$\mathbf{A}/Ker(\zeta)$	quotient eBA over kernel of eBH ζ	35

$\hat{\zeta}$	extensional Binding Homomorphism associated with eBH ζ and $\zeta = \hat{\zeta} \odot \nu_\zeta$	35
$\mathbf{A} \dot{\models}_{eBA} p \simeq q$	admissible satisfaction associated with $\mathbf{A}$	37
$\mathbf{B}$	intensional Binding Algebra (or iBA)	37
$\vec{B}$	family of (intensional) carriers	37
$pr_{k,i}$	object of k-ary ith-projection in intensional carriers of iBAs	37
$\circ_{m,k}$	object of the compositional in intensional carriers of iBAs	37
$\sigma_k^{\mathbf{B}}$	intensional functional in iBA $\mathbf{B}$ associated with BO σ and index k	37
$(B_k)^m$	product $\underbrace{B_k \times B_k \times ... \times B_k}_{m \text{ times}}$ and $m \in Nat$	37
$\mathring{A}$	(hidden) carrier for intensional carriers of iBAs	37
$\mathring{A}^k$	product $\underbrace{\mathring{A} \times \mathring{A} \times ... \times \mathring{A}}_{k \text{ times}}$ and $k \in Nat$	37
$g\circ_{m,k} < \vec{h} >$	readable version of $\circ(g, \vec{h})$	38
$\circ_{m,k}(g, \vec{h})$	intensional composition of g and $\vec{h}$ with $m = \lvert\vec{h}\rvert$	38
$g(\vec{h})$	more readable version of $\circ_{m,k}(g, \vec{h})$ with hidden $m, k \in Nat$ and $\lvert\vec{h}\rvert = m$	38
$\vec{g}(\vec{h})$	list of $g_i(\vec{h})$ where $g_i = \vec{g}(i)$ for $1 \leq i \leq n$	38
$Pr_{i,j}^k$	list of $pr_{k,i}, pr_{k,i+1}, ..., pr_{k,j}$ and $i \leq j \leq k$	38
$\vec{\psi}$	family of function environments associated with iBAs	39
ψ_m	function environment with arity $m \in Nat$ associated with iBAs	39
$\{\vec{x}\}$	set of all variables appearing in list $\vec{x}$	39
$\mathcal{B}_{\vec{x}}$	(intensional) interpretation in iBA $\mathbf{B}$	39
$\mathcal{B}_{\vec{x}}[\![t]\!]_\psi$	interpretation of term t in iBA $\mathbf{B}$	39
$\mathcal{B}_{\vec{x}}[\![ft]\!]_\psi$	interpretation of function term ft in iBA $\mathbf{B}$	39
$\vec{\imath}$	identity substitution	39
$\vec{\imath}[\vec{y} := \vec{z}]$	updated identity substitution	39
$\vec{\bullet}\lceil_k$	prefix list obtained from first k elements of list $\vec{\bullet}$	39
$\vec{x}(j)$	jth element in ordinary variable list $\vec{x}$	39
$Free(t)$	set of free ordinary variables and free function variables in term t	39
$Free(ft)$	set of free ordinary variables and free function variables in function term ft	39

$\mathbf{B} \models_{iBA} p \simeq q$	(intensional) satisfaction of iBA $\mathbf{B}$	40
$\vdash_{iBA}$	(intensional) Equational Calculus (or Logic) for iBAs	40
$\vdash_{EQ}$	(first order) Equational Calculus (or Logic) for many-sorted first order algebras	40
$\mathbf{B} \models^{d}_{iBA} \gamma \mapsto \Delta$	(intensional) dependent satisfaction related to iBA $\mathbf{B}$	41
$\mathbf{B} \models^{q}_{iBA} \gamma \hookrightarrow \Delta$	(intensional) quasi-dependent satisfaction related to iBA $\mathbf{B}$	41
$\mathbf{B} \models^{u}_{iBA} \Omega$	(intensional) universal satisfaction of related to iBA $\mathbf{B}$	41
$\dot{\vdash}_{eBA}$	admissible Equational Calculus (or Logic) for eBAs	42
Γ	set of axioms (or premises) for derivations of various calculi	42
$p \simeq q$	Binding Equation	42
$p, q \in T$	ordinary terms of binding terms	42
$p, q \in FT_k$	function terms with arity $k \in Nat$ of binding terms	42
$\vec{\varrho}$	family of (substitution) functions	42
$\emptyset$	empty set	43
$\Gamma \vdash_{iBA} \Delta$	a (intensional) judgement that Δ is derivable from Γ	43
(α)	α-conversion axiomatic schema	43
(ξ)	(intensional) inference rule for adding binding primitives	43
(ξ^{-1})	(intensional) inference rule for dropping off binding primitives	43
$t[\vec{y} := \vec{x}]$	term obtained by applying substitution $\vec{t}[\vec{y} := \vec{x}]$ to term t	43
$p\vec{\varrho}$	binding term obtained by applying substitution $\vec{\rho}$ to it	43
$\dot{\vdash}^{d}_{eBA}$	admissible dependent equational calculus	44
$\vdash^{d}_{iBA}$	intensional dependent equational calculus	44
$\dot{\vdash}^{q}_{eBA}$	admissible quasi-dependent equational calculus	44
$\vdash^{q}_{iBA}$	intensional quasi-dependent equational calculus	44
$\dot{\vdash}^{u}_{eBA}$	admissible universal equational calculus	44
$\vdash^{u}_{iBA}$	intensional universal equational calculus	44
CL	set of axioms associated with Combinatory Logic	46
λ^*	counterpart of λ-abstraction operator in Combinatory Logic	46
$\lambda^* x.u$	term obtained by applying λ^* to bind x in u	46

$\mathbf{I}$	identity combinator in Combintory Logic	46
$\not\vdash$	impossible (syntactic) derivation	46
$\mathbf{S}$	S-combinator in Combinatory Logic	46
$\mathbf{K}$	K-combinator in Combinatory Logic	46
A_β	set of Curry's axioms making Combinatory Logic equivalent to Lambda Calculus	46
$Alg(\Sigma^{BO})$	class of (first order) eBAs	47
$eBAlg(\Sigma^{BO})$	class of eBAs	48
$\eta_{\Sigma^{BO}}$	embedding functor from category $Alg(\Sigma^{BO})$ to category $eBAlg(\Sigma)$	48
Σ^{VBTO}	signature for Variable Binding Term Operators	48
Σ	sum operator in Calculus of Constructions	48
Π	product operator in Calculus of Constructions	48
$\vec{V}$	list of variables in V (with ω length)	49
α	ordinal (number)	49
ω	least limit ordinal	49
c_k	kth cylindrifier in Cylindric Algebra	49
$\exists_k^{cyl}$	BO associated with kth cylindrifier in Cylindric Algebra	49
$d_{k,j}$	(k,j)-diagonal element in Cylindric Algebra	49
$\circ^{cyl}$	BO for the composition functional associated with Cylindric Algebra	50
$\exists^{mon}$	BO associated with Monadic Algebra	50
$\exists_1$	the usual existential quantifier with arity 1	50
$\exists_{\vec{x}}^{pol}$	BO associated with Polyadic Algebra	50
$\exists_k$	the usual k-ary existential quantifier	50
$A^{(\sigma,\tau)}$	function space from $A^{(\sigma)}$ to $A^{(\tau)}$ associated with Plotkin's type hierarchy	51
$A^{(\sigma)}$	(function) space associated with Plotkin's type hierarchy	51
$A^{(\tau)}$	(function) space associated with Plotkin's type hierarchy	51
$\mathcal{F}_m^{\mathbf{A}}$	set of m-ary functions associated with eBA $\mathbf{A}$	51
$\mathcal{F}_m^{\mathbf{A}'}$	set of m-ary functions associated with eBA $\mathbf{A}'$	51
$\mathcal{F}^{\mathbf{A}}$	explicitly closed family associated with eBA $\mathbf{A}$	51
$\zeta(\mathcal{F}^{\mathbf{A}})$	image of eBH ζ over $\mathcal{F}^{\mathbf{A}}$	51

$\mathcal{F}^{\mathbf{A}'}$	explicitly closed family associated eBA $\mathbf{A}'$	51
$\mathcal{C}$	2-category	55
$Ob(\mathcal{C})$	class of objects in 2-cateogory $\mathcal{C}$	55
$\mathcal{C}(a,b)$	class of homos from a to b in 2-category $\mathcal{C}$	55
A^m	cartesian product of m copies of A in 2-category $\mathcal{C}$	55
I	set of sorts for many-sorted first order algebras	62
Σ	signature of many-sorted first order algebras	62
$\Sigma_{\kappa,i}$	first order signature with rank (or arity) $< \kappa, i >$	62
$< \kappa, i >$	rank for operations in first order algebras	62
ε	empty list	62
κ	list of sorts in I	62
i	sort in first order algebras	62
$\vec{\chi}$	family of sets of variables for first order algebras	62
χ_i	set of variables in $\vec{\chi}$ associated with sort i	62
$\|\chi_i\|$	cardinal of χ_i	62
$\vec{X}$	sub-set of variables in χ	62
$T(X)$	family of sets of first order terms associated with Σ and X	62
$\mathbf{D}$	many-sorted first order algebra	63
$\vec{D}$	family of carriers associated with $\mathbf{D}$	63
$\mathcal{D}$	interpretation associated with $\mathbf{D}$	63
σ	first order operator (or operation)	63
$\Sigma_{\kappa \to i}$	set of first order operators with rank $\kappa \to i$	63
$\mathcal{D}(\sigma)$	interpretation of first order operator σ in $\mathbf{D}$	63
$\mathbf{D}_\sigma$	interpretation of σ in $\mathbf{D}$	63
$\sigma^{\mathbf{D}}$	interpretation of σ in $\mathbf{D}$	63
$\mathbf{T}(X)$	first order term algebra associated with Σ and X	63
$\mathcal{T}$	interpretation associated with first order term algebra	63
$\vec{\rho}$	variable assignment or environment	63
$\mathcal{D}[\![\bullet]\!](\vec{\rho})$	interpretation in $\mathbf{D}$	63
X_i	set of variables in X with sort i	63
$t \simeq_X u$	first order equation indexed by variable X	64

$\models_{EQ}$	equational satisfaction	64
$\mathbf{D} \models_{EQ} t \simeq_X u$	equational satisfaction of $\mathbf{D}$ related to index X	64
$(\mathbf{D}, \vec{\rho}) \models t \simeq_X u$	satisfaction of $\mathbf{D}$ related to environment $\vec{\rho}$ and variable index X	64
$\gamma_X \mapsto \Delta_X$	dependent equation indexed by variable X	64
γ_X	set of equations indexed by variable X	64
Δ_X	equation indexed by variable X	64
$\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_X$	dependent satisfaction of $\mathbf{D}$ related to index X	64
$\gamma_X \hookrightarrow \Delta_X$	quasi-dependent equation indexed by variable X	64
$\mathbf{D} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$	quasi-dependent satisfaction of $\mathbf{D}$ related to index X	64
Ω_X, Ω	$\{\gamma_X^{(m)} \hookrightarrow \Delta_X^{(m)} \mid m \in M\} \mapsto (\gamma_X \hookrightarrow \Delta_X)$	64
$\mathbf{D} \models_{uEQ} \Omega$	universal satisfaction of $\mathbf{D}$	64
$< X, t, t' >$	version of $t \simeq_X t'$	66
$T(X)_i$	set of terms associated with sort i	66
$\forall X.t \simeq t'$	universally quantified (first order) equation	66
True	constant with value *true*	66
False	constant with value *false*	66
and	logical connective *and*	66
or	logical connective *or*	66
fool	operator (or predicate) *fool*	66
Γ	set of first order equations	67
$\mathbf{D} \models \Gamma$	$\mathbf{D} \models t \simeq_X t'$ for every $t \simeq_X t' \in \Gamma$	67
$Alg_{\Sigma,\Gamma}$	class of (first order) algebras satisfying Γ	67
$\mathcal{K}$	class of first order algebras	67
$\mathcal{K} \models t \simeq_X t'$	$\mathbf{D} \models t \simeq_X t'$ for each $\mathbf{D} \in \mathcal{K}$	67
$\Gamma \models t \simeq_X t'$	$\mathbf{D} \models \Gamma$ implies $\mathbf{D} \models t \simeq_X t'$ for every $\mathbf{D}$	67
$\iota : \mathbf{D} \to \mathbf{D}'$	Σ-homomorphism from $\mathbf{D}$ to $\mathbf{D}'$	68
$\vec{\rho}^{\#}$	Σ-homomorphism extended from ρ	68
η_X	embedding map from X to $\mathbf{T}(X)$	68
ι	(first order) Σ-homomorphism	68
$\preceq$	indicating sub-Σ-algebra	68
$\mathbf{D} \preceq \mathbf{D}'$	$\mathbf{D}$ is a sub-Σ-algebra of $\mathbf{D}'$	69

$\mathcal{D}'\lceil_{\vec{D}}$	interpretation obtained by projecting (limiting) $\mathcal{D}'$ onto $\vec{D}$	69
$\mathcal{D}'\lceil_{\mathbf{D}}$	interpretation obtained by projecting (limiting) $\mathcal{D}'$ onto $\mathbf{D}$	69
$\bigcap\mathcal{K}$	intersection of class $\mathcal{K}$ of sub-Σ-algebras	69
$[d]$	$\{d' \in D \mid \iota(d) = \iota(d')\}$	69
$\vec{\theta}$, $\vec{\theta}'$	Σ-congruence	69
Φ	class of Σ-congruences	69
$\bigcap\Phi$	intersection of class Φ of Σ-congruences	69
$d/\vec{\theta}$	$\{d' \in D_i \mid d\,\vec{\theta}_i\,d'\}$	70
$D_i/\vec{\theta}$	$\{d/\vec{\theta} \mid d \in D_i\}$	70
$\vec{D}/\vec{\theta}$	family of $D_i/\vec{\theta}$ for each $i \in I$	70
$[d]_{\vec{\theta}}$	$d/\vec{\theta}$	70
$[d]$	$d/\vec{\theta}$	70
$\mathbf{D}/\vec{\theta}$	quotient Σ-algebra of $\mathbf{D}$ over $\vec{\theta}$	70
$\nu_{\vec{\theta}}$	natural Σ-homomorphism associated with $\vec{\theta}$	70
$\vec{\theta}'/\vec{\theta}$	congruence obtained from $\vec{\theta}$ and $\vec{\theta}'$	70
ι	Σ-homomorphism	70
$\nu_{\vec{\theta}'}$	natural homomorphism associated with $\vec{\theta}'$	70
$\nu_{(\vec{\theta}/\vec{\theta}')}$	natural homomorphism associated with $\vec{\theta}/\vec{\theta}'$	70
$ker(\iota)$	kernel of homomorphism ι	70
$ker_i(\iota)$	kernel of homomorphism ι with sort i	70
$\nu_{ker(\iota)}$	natural homomorphism associated with kernel $ker(\iota)$	70
ν_ι	simple version of $\nu_{ker(\iota)}$	70
$\hat{\iota}$	homomorphism associated with ι and $\iota = \hat{\iota} \circ \nu_\iota$	70
$G(\vec{D})$	carriers of the sub-algebra generated from $\vec{D}$	71
$\mathbf{G}(\vec{D})$	sub-algebra generated from $\vec{D}$	71
gen	generating function	71
gen^*	least closure function of generating function gen	71
$\mathbf{T}(X)$	$\mathbf{T}(X)$ with hidden Σ assumed	71
$Alg_{\Sigma,\emptyset}$, Alg	class of all first order algebras	72
$\hat{\iota}$	homomorphism associated with ι and $\iota = \hat{\iota} \circ \nu_{\vec{\theta}}$	72
$ker_X(\iota)$	kernel of ι associated with X	72

$ker(\iota)$	kernel of homomorphism ι	72
$Ker_X(\mathbf{D})$	$\bigcap_{\iota:\mathbf{T}\to\mathbf{D}} ker_X(\iota)$	73
$\tilde{\theta}$	family of collections of congruences (like $\vec{\theta}^{\mathbf{D}}$)	75
$\vec{\theta}^{\mathbf{D}}$	congruence associated with $\mathbf{D}$	75
$\tilde{\mathbf{T}}(\vec{\chi})$	$\{\mathbf{T}(X) \mid X \subseteq \vec{\chi}\}$	75
$\tilde{K}(\mathbf{D})$	$\{Ker_X(\mathbf{D}) \mid X \subseteq \vec{\chi}\}$	75
$\tilde{K}(\mathcal{K})$	$\bigcap_{\mathbf{D}\in\mathcal{K}} \tilde{K}(\mathbf{D})$	75
$\tilde{\mathbf{T}}(\vec{\chi})/\tilde{K}(\mathcal{K})$	$\{\mathbf{T}(X)/\bigcap_{\mathbf{D}\in\mathcal{K}} Ker_X(\mathbf{D}) \mid X \subseteq \vec{\chi}\}$	76
$\tilde{\bullet}$	family of things indexed by different variable sets	76
$\vec{\bullet}$	family of things indexed by different sorts	76
Γ'_X	set of equations sharing a same variable index X	76
$dom(\iota_2)$	domain of ι_2	77
$\iota_2 \uplus \iota_1$	homomorphism ι_1 modified by homomorphism ι_2	77
$\tilde{\theta}$	family of congruences	78
$\mathcal{K}^T$	class of first order algebras	78
Γ	set of equations	80
$\tilde{\Gamma}$	family of sets of first order equations	81
$\sqcup$	least fixed point operator	81
$\sqcup\tilde{\mathcal{M}}(\tilde{\Gamma})$	least fixed point of $\tilde{\mathcal{M}}$ containing Γ	81
$\bigcap\mathcal{R}$	$r \in \bigcap\mathcal{R}$ iff $r \in R$ for each $R \in \mathcal{R}$	81
$\tilde{\phi}$	family of functions from equations to pairs of terms	81
$\tilde{\phi}^{-1}$	reversed version of $\tilde{\phi}$	81
$\tilde{C}_X(\mathcal{K})_Y$	consequence relation associated with term algebra index X and equation index Y	83
Y	family of sets of variables associated with sorts	83
$\tilde{C}_{X,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)}$	consequence family projecting to $\mathbf{T}_\Sigma(X) \times \mathbf{T}_\Sigma(X)$	83
$\tilde{C}_{Y,Y}\lceil_{\mathbf{T}_\Sigma(X)\times\mathbf{T}_\Sigma(X)}$	consequence family projecting to $\mathbf{T}_\Sigma(X) \times \mathbf{T}_\Sigma(X)$	83
$\overset{\exists\iota}{\subseteq}$	inclusion depending on the existence of ι	84
$\vec{Z}$	family of sets of variables associated with sorts	85
$\mathbf{T}_\Sigma(X_j)_i$	set of terms sorted by i and associated with variable X_j 85	
$\mathbf{T}_\Sigma(Y)_i$	family of sets of terms associated with variable Y_i	85

$\mathbf{D} \models_{dEQ} \gamma_X \mapsto \Delta_Y$	satisfaction for dependent equations associated with **D**	86
$\mathcal{K} \models_{dEQ} \gamma_X \mapsto \Delta_Y$	satisfaction for dependent equations associated class $\mathcal{K}$	86
$\mathbf{D} \models_{dEQ} \Gamma$	satisfaction for sets of dependent equations associated with **D**	87
$\Gamma \models_{dEQ} \gamma_X \mapsto \Delta_Y$	satisfaction of dependent equations associated with Γ	87
γ_X	set of equations indexed by X	87
Δ_Y	equation indexed by Y	87
$Eq(\Gamma)$	equations from Γ	87
$\vdash^{-}_{dEQ}$	restricted calculus for dependent equations	88
$\vdash^{-}_{dEQ}$	restricted calculus for dependent equations	88
$\sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma)$	least fixed point of $\tilde{\mathcal{M}}^{-}_{dEQ}$ containing Γ	89
$\phi(\tilde{\mathcal{M}}^{-}_{dEQ}(\Gamma))\lceil_{\simeq}$	$\{<t,t'> \mid \emptyset \mapsto t \simeq_X t' \in \sqcup\tilde{\mathcal{M}}^{-}_{dEQ}(\tilde{\Gamma})\}$	90
$\Sigma_{i,bool}$	first order signature for Example 2.3.5	90
A_{bool}	truth values	90
$true$	value $true$	90
$false$	value $false$	90
true	function mapping everything to value $true$	90
$\mathbf{D} \not\models x \simeq_{\{x,y\}} y$	$x \simeq_{\{x,y\}} y$ is unsound in **D**	90
$n_{\mathbf{true}}(t)$	function counting the number of **true** appearances in terms	90
$\vdash_{dEQ}$	calculus for dependent equations	90
$\emptyset \hookrightarrow \Delta_X$	pure equation in the form of quasi-dependent equations	93
Δ_X	one version of $\emptyset \hookrightarrow \Delta_X$	93
$\mathbf{D} \models \Gamma$	satisfaction of quasi-dependent equations associated with **D**	93
Γ	set of quasi-dependent equations	93
$\mathcal{K} \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$	satisfaction of quasi-dependent equations associated with class $\mathcal{K}$	93
$\Gamma \models_{qEQ} \gamma_X \hookrightarrow \Delta_X$	satisfaction of a quasi-dependent equations associated with Γ	93
$q\text{-}d$	syntactic translation from quasi-dependent equations to dependent equations	93
$q\text{-}d[\![\Gamma]\!]$	image of translation q-d on Γ	94
$\vdash^{d}_{qEQ}$	restricted calculus for quasi-dependent equations	96

$(\omega\text{-}d\text{-}q)$	over-strong rule for the deduction of quasi-dependent equations	97
δ	substitution for skolemization technique	98
$\mathbf{D}''$	many-sorted first order algebra	98
$\vdash_{qEQ}$	calculus for quasi-dependent equations	98
$\mathbf{D} \models \Omega$	satisfaction of a universal equation associated with $\mathbf{D}$	99
$\mathcal{K} \models_{uEQ} \Omega$	satisfaction of a universal equation associated with a class $\mathcal{K}$	99
$\mathbf{D} \models_{uEQ} \Gamma$	satisfaction of universal equations associated with $\mathbf{D}$	99
$\mathcal{K} \models_{uEQ} \Gamma$	satisfaction of universal equations associated with class $\mathcal{K}$	99
(u-id)	identity rule in $\vdash_{uEQ}$	101
(u-rfl)	reflective rule in $\vdash_{uEQ}$	101
(u-sym)	symmetric rule in $\vdash_{uEQ}$	101
(u-trs)	transitive rule in $\vdash_{uEQ}$	101
(u-cmp)	composition rule in $\vdash_{uEQ}$	101
(u-sub)	substitution rule in $\vdash_{uEQ}$	101
(u-d-ctr)	contraction rule for dependent equations in $\vdash_{uEQ}$	101
(u-q-ctr)	contraction rule for quasi-dependent equations in $\vdash_{uEQ}$	102
(u-d-intr)	introduction rule of dependent equations in $\vdash_{uEQ}$	102
(u-q-d)	transformation rule from quasi-dependent equations to dependent equations in $\vdash_{uEQ}$	102
(u-skm)	skolemization rule in $\vdash_{uEQ}$	102
(u-q-u)	introduction rule of universal equations in $\vdash_{uEQ}$	102
$\mathbf{0}$	syntax name for zero of natural numbers	104
$< Nat, +_{Nat}, \mathbf{0} >$	group structure of natural natural numbers	104
$(\forall X.\gamma) \mapsto (\forall X.t \simeq u)$	universally-quantified dependent equation	105
$\forall X.(\gamma \hookrightarrow t \simeq u)$	universally-quantified quasi-dependent equation	105
$Congr_{S,\Sigma}$	complete lattice of first order congruences	106
θ_Γ	functional from $Congr_{S,\Sigma} \rightarrow Congr_{S,\Sigma}$	106
$\equiv_1$	first order congruence	106
$\theta_\Gamma(\equiv_1)$	first order congruence	106
$\equiv_2$	first order congruence	106

$\equiv_\Gamma$	least fixpoint of θ_Γ	106
$[\Gamma]$	least closure of Γ in (2.6.3.*)	106
$\equiv_{[\Gamma]}$	least fixed point of (2.6.3.ii.a) and (2.6.3.ii.b)	107
$(\omega\text{-}d\text{-}q)$	invalid ω-rule for deductions of quasi-dependent equations	107
$\vdash^+_{qEQ}$	unsound deduction map for quasi-dependent equations	107
$(q\text{-}skm)$	skolemization rule in $\vdash_{qEQ}$	108
$\dot{\rightarrow}$	admissible eBHs	111
$\vec{\mathbf{e}}\lfloor_k$	list obtained from list $\vec{\mathbf{e}}$ by droffing off the first k elements	114
$\mathbf{A}' \prec \mathbf{A}$	$\mathbf{A}'$ is a sub-eBA of $\mathbf{A}$	117
$\mathbf{A}' \preceq \mathbf{A}$	$\mathbf{A}'$ is a perfect sub-eBA of $\mathbf{A}$	117
$curry_{k,m}(g)$	currying	118
$\mathbf{A'}_1 \bigcap \mathbf{A'}_2$	intersection	119
$\bigcap \mathcal{K}$	intersection of class $\mathcal{K}$	120
$[\mathbf{X}]$	sub-eBA generated from $\mathbf{X}$	120
$Sub(\mathbf{X})$	sub-eBA generated from $\mathbf{X}$	120
$\mathbf{ext}(\mathbf{X})$	extension of $\mathbf{X}$	120
$ext(X)$	extension of X	120
$ext_k(X)$	extension of X associated with arity k	120
$\mathbf{ext}^{\jmath+1}(\mathbf{X})$	$\jmath+1$ times of extensions of $\mathbf{X}$	121
$\mathbf{ext}^0(\mathbf{X})$	zero times of extensions of $\mathbf{X}$	121
$\mathbf{A}'\lceil_{A'}$	projection of $\mathbf{A}'$ to A'	122
$\mathcal{F}(A)^{\mathbf{A}'}\lceil_{A'}$	limiting (projecting) domains of functions in $\mathcal{F}(A)^{\mathbf{A}'}$ to products of multiple copies of A'	122
$\zeta(\mathcal{G})$	image of $\mathcal{G}$ under eBH ζ	123
$\dot{\approx}$	equivalence relation in building term eBA $\mathbf{T}$	125
$(\dot{\approx}\text{-}\alpha)$	α-conversion rule in $\mathcal{M}_{\dot{\approx}}$	125
$(\dot{\approx}\text{-}\xi^{-1}\text{-rule})$	anti-ξ-rule in $\mathcal{M}_{\dot{\approx}}$	125
$(\dot{\approx}\text{-}\dot{\xi}\text{-rule})$	ξ-rule in $\mathcal{M}_{\dot{\approx}}$	126
$(\dot{\approx}\text{-cmp-1})$	composition rule for function variables in $\mathcal{M}_{\dot{\approx}}$	126
$(\dot{\approx}\text{-cmp-2})$	composition rule for Binding Operators in $\mathcal{M}_{\dot{\approx}}$	126

$[t]$	representative of the set of terms associated with t through $\dot{\approx}$	127
$[ft]$	representative of the set of function terms associated with ft through $\dot{\approx}$	127
$[T]$	set of representatives of terms through $\dot{\approx}$	127
$ft(\vec{t})$	$ft\|\|_{\|\vec{x}\|}[\vec{x} := \vec{t}]$	127
$\bullet_{[t]}$	object in term eBA $\mathbf{T}$ associated with t	127
$\bullet_{[ft]}$	function in term eBA $\mathbf{T}$ associated with ft	127
$\odot$	composition functional	128
$curry_{k,m_i}$	currying	128
$\mathcal{F}([T])$	family of function sets with domains on products of multiple copies of $[T]$	129
$\mathbf{T}$	term eBA	129
$\vec{\Im}$	syntactic extension function over binding terms	129
$\Im^0(\vec{X})$	zero times of extensions $\Im$ on $\vec{X}$	130
$\Im^{j+1}(\vec{X})$	j times of extensions $\Im$ on $\vec{X}$	130
ext	semantic extension **ext**	131
$\zeta'(\mathbf{X})$	image of $\mathbf{X}$ under ζ'	135
$[\zeta'(\mathbf{X})]$	generated sub-eBA from image $\zeta'(\mathbf{X})$	135
a/ϑ	representative of a set of objects associated with a through ϑ	137
g/ϑ	function representative of a collection of functions associated with g through ϑ	137
$(g/\vartheta)(\vec{a}/\vartheta)$	applying function g/ϑ associated with g and ϑ to object list $\vec{a}/\vartheta$ associated with $\vec{a}$ and ϑ	137
$(g/\vartheta)\odot < \vec{h}/\vartheta >$	compostition of function g/ϑ and function list $\vec{h}/\vartheta$	137
$(\sigma^{\mathbf{A}}/\vartheta)\odot < curry_{k,\vec{m}}(\vec{g}/\vartheta), (\vec{h}/\vartheta) >$	$(\sigma^{\mathbf{A}}\odot < curry_{k,\vec{m}}(\vec{g}), \vec{h} >)/\vartheta$	137
$\mathcal{F}/\vartheta$	$< A/\vartheta, \mathcal{F}_m/\vartheta_{m\in Nat} >$	137
$(C_{k,a}/\vartheta)(\vec{a}/\vartheta)$	$C_{k,a}(\vec{a})/\vartheta$	138
$C_{k,a/\vartheta}$	$C_{k,a}/\vartheta \in \mathcal{F}_k/\vartheta$	138
$(\pi_{k,i}/\vartheta)(\vec{a}/\vartheta)$	$\pi_{k,i}(\vec{a})/\vartheta$	138
$\pi_{k,i}/\vartheta$	$(\pi_{k,i}/\vartheta)(\vec{a}/\vartheta) == a_i/\vartheta$ for every $\vec{a}/\vartheta$	138
$\mathbf{A}/\vartheta$	quotient eBA	138
$\mathcal{A}_\sigma^{(\mathbf{A}/\vartheta)}$	interpretation of σ in quotient eBA $\mathbf{A}/\vartheta$	138

ρ/ϑ	$\rho/\vartheta(x) = \rho(x)/\vartheta$ for $x \in V$	138
$(\mathcal{A}/\vartheta)[\![p]\!]_{<\rho/\vartheta,\varphi/\vartheta>}$	interpretation in quotient eBA $\mathbf{A}/\vartheta$	138
ν_ϑ	natural BH associated with eBC ϑ	139
$\vec{\nabla}_\zeta$	core of eBH ζ	141
$\hat{\zeta}$	eBH associated with ζ such that $\zeta = \hat{\zeta} \odot \nu_\zeta$	147
$\hat{\zeta}^{-1}$	reversed map of $\hat{\zeta}$	148
$\dot{\vartheta}$	admissible eBC	149
$\dot{\vartheta}_{\mathbf{A}}$	admissible eBC on term eBA $\mathbf{T}$ associated with eBA $\mathbf{A}$	149
$\dot{\vartheta}_{\mathcal{K}}$	admissible eBC on term eBA $\mathbf{T}$ associated with class $\mathcal{K}$	149
$\mathbf{A}\dot{\models}_{eBA} p \simeq q$	admissible satisfaction of a BE (or admissible BE) associated with $\mathbf{A}$	151
$\mathcal{K}\dot{\models}_{eBA} p \simeq q$	admissible satisfaction of a BE associated with class $\mathcal{K}$	151
$\mathcal{K}\dot{\models}_{eBA} \Gamma$	satisfaction of admissible BEs associated with class $\mathcal{K}$	151
Γ	set of admissible BEs	151
$Adm_{\Sigma BO}(\Gamma)$	class of eBAs admissibly satisfying Γ	151
$\Gamma\dot{\models}_{eBA} p \simeq q$	$\mathbf{A}\dot{\models}\Gamma$ implies $\mathbf{A}\dot{\models} p \simeq q$ for each eBA $\mathbf{A}$	151
$\dot{I}_{\mathcal{K}}(\mathbf{X})$	set of BEs admissibly satisfied by class $\mathcal{K}$	151
$\dot{\vartheta}_{\mathcal{K}}$	admissible eBC associated with class $\mathcal{K}$	152
$\check{\zeta}$	eBH associated with ζ such that $\zeta = \check{\zeta} \odot \nu_\vartheta$	153
$\mathcal{T}$	interpretation associated with term eBA $\mathbf{T}$	155
$\vec{E}$	set of pairs of binding terms	155
$\vartheta_A(\vec{E})$	least admissibly-invariant eBC containing $\vec{E}$ on term eBA $\mathbf{T}$	155
ϕ	function taking representatives of equivalent binding terms under $\dot{\approx}$ and returning elements in term eBA	156
$\dot{\vartheta}_{Adm(\Gamma)}$	admissible eBC associated with class $Mod(\Gamma)$	156
(adm-ξ)	admissible ξ rule different from ξ rule	158
$\prod_{k\in Ind} \mathbf{A}_k$	product of $\{A_k \mid k \in Ind\}$	160
$\sum_{k\in Ind} A_k$	sub-direct product of $\{A_k \mid k \in Ind\}$	160
Diag	diagonal eBC	160
diag	diagonal eBC on a quotient eBA	160
$\mathbf{I}$	Isomorphism operator over classes of eBAs	162
$\mathbf{S}_p$	operator of perfect sub-eBAs over classes of eBAs	162

$\mathbf{P}$	(direct) product operator over classes of eBAs	162
$\cong$	is isomorphic to	162
γ	set of Binding Equations	167
Γ	set of dependent Binding Equations	167
$\mathbf{A}\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$	admissible dependent satisfaction of eBA $\mathbf{A}$	167
$\mathcal{A}[\![p \simeq q]\!]_{<\rho,\varphi>}$	$\mathcal{A}[\![p]\!]_{<\rho,\varphi>} = \mathcal{A}[\![q]\!]_{<\rho,\varphi>}$ for admissible $<\rho,\varphi>$	167
$\mathbf{A}\dot{\models}^{d}_{eBA}\Gamma$	$\mathbf{A}\dot{\models}\gamma \mapsto \Delta$ for each $\gamma \mapsto \Delta \in \Gamma$	168
$\mathcal{K}\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$	$\mathbf{A}\dot{\models}\gamma \mapsto \Delta$ for every $\mathbf{A} \in \mathcal{K}$	168
$\mathcal{K}\dot{\models}^{d}_{eBA}\Gamma$	$\mathbf{A}\dot{\models}\Gamma$ for each $\mathbf{A} \in \mathcal{K}$	168
$\Gamma\dot{\models}^{d}_{eBA}\gamma \mapsto \Delta$	for every eBA $\mathbf{A}$, $\mathbf{A} \models \Gamma$ implies $\mathbf{A} \models \gamma \mapsto \Delta$	168
$Adm_{\Sigma Bo}(\Gamma)$	for every eBA $\mathbf{A}$ and $\mathbf{A}\dot{\models}\Gamma$	168
$\dot{\vartheta}_{\mathbf{A}}$	admissible binding kernel associated with $\mathbf{A}$ and with respect to Γ	168
(d-ξ)	(ξ) rule in form of dBEs	169
(d-ξ^{-1})	anti (ξ) rule in form of dBEs	169
$(\mathbf{A}, <\rho,\varphi>)\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$	$\mathcal{A}[\gamma]_{<\rho,\varphi>}$ implies $\mathcal{A}[\![\Delta]\!]_{<\rho,\varphi>}$ for every admissible $<\vec{\rho},\varphi>$	170
$\mathbf{A}\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$	$(\mathbf{A}, <\rho,\varphi>)\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$ for every admissible $<\rho,\varphi>$	170
$\mathbf{A}\dot{\models}^{q}_{eBA}\Gamma$	$\mathbf{A}\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$ for each $\gamma \hookrightarrow \Delta \in \Gamma$	170
$\mathcal{K}\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$	$\mathbf{A}\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$ for every $\mathbf{A} \in \mathcal{K}$	170
$\mathcal{K}\dot{\models}^{q}_{eBA}\Gamma$	$\mathbf{A}\dot{\models}\Gamma$ for each $\mathbf{A} \in \mathcal{K}$	170
$\Gamma\dot{\models}^{q}_{eBA}\gamma \hookrightarrow \Delta$	for every eBA $\mathbf{A}$, $\mathbf{A}\dot{\models}\Gamma$ implies $\mathbf{A}\dot{\models}\gamma \mapsto \Delta$	170
(q-ξ^{-1})	anti (ξ) rule in form of qBEs	171
$\mathbf{A}\dot{\models}^{u}_{eBA}\Omega$	admissible satisfaction of a uBE associated with $\mathbf{A}$	172
$\mathcal{K}\dot{\models}^{u}_{eBA}\Omega$	natural extension of $\mathbf{A}\dot{\models}^{u}_{eBA}\Omega$ to a class $\mathcal{K}$ of eBAs	172
$\mathbf{A}\dot{\models}^{u}_{eBA}\Gamma$	$\mathbf{A}\dot{\models}\Omega$ for each $\Omega \in \Gamma$	172
$\mathcal{K}\dot{\models}^{u}_{eBA}\Gamma$	natural extension of $\mathbf{A}\dot{\models}\Gamma$ to a collection Γ of admissible uBEs	172
$tr_{\mathcal{X}}[\![\bullet]\!]$	syntactic translation from binding terms to terms of b-clones	176
$\underline{tr}[\![\bullet]\!]$	syntactic translation from terms of b-clones to binding terms	176
$\mathbf{B}$	iBA	177
$\mathbf{B}''$	iBA	178

$Pr^m_{1,m}$	string of $pr_{m,i}, pr_{m,i+1}, ..., pr_{m,j}$ $(i \leq i \leq j \leq m)$	179
$\odot$	extensional composition functional	179
Π_i	ith projection functional	179
$Free$	function which takes binding terms and returns free variables and free function variables in them	182
$\|\|_m$	function taking function terms and returning terms without binding primitives	182
$\|_m$	function taking function terms and returning string of binding variables	182
$\vec{t}\,\vec{\varrho}$	term list of $t_1\vec{\varrho}, t_2\vec{\varrho}, ..., t_m\vec{\varrho}$	183
$\vec{ft}\vec{\varrho}$	function term list of $ft_1\vec{\varrho}, ft_2\vec{\varrho}, ..., ft_n\vec{\varrho}$	183
$rename$	predicate on substitutions	183
$BT^{(j)}$	family of sets of binding terms with depths at most to j	184
$FT^{(j)}$	family of sets of function terms with depths at most to j	184
ν	variable or function variable	185
k_ν	least depth of (function) term $\vec{\varrho}[\nu]$	185
n_ν	arity associated with (function) variable ν	185
$\vec{\varrho} \asymp \vec{\varrho'}$	equivalence between substitutions	185
LHS	Left Hand Side of ...	187
RHS	Right Hand Side of ...	187
B_k	carrier with index k in iBA **B**	191
Σ	first order signature (of b-clones)	192
$\underline{f}$	(ordinary) variables of b-clones	192
$\underline{pr}_{k,i}$	element of b-clones with sort k	192
$\underline{o}_{m,k}$	operator of b-clones	192
$\underline{\sigma}_k$	operator of b-clones	192
$_(\sigma, k)$	bijection from pairs of Binding Operators and natural numbers to operators of b-clones	192
$T(\vec{X})$	set of terms of b-clones with variable set $\vec{x}$	192
T_k	set of terms of b-clones with sort k	192
X_k	variables of b-clones with sort k	192
D	b-clone (a kind of first order algbera)	194
$\mathcal{D}$	interpretation associated with (b-clone) **D**	194

ρ_k	variable assignment associated with sort k	194
$tr_{\vec{x}}$	syntactic translation from binding terms to terms of b-clones	194
$tr_{\vec{x}}[\![\vec{t}]\!]$	string of $tr_{\vec{x}}[\![t_1]\!], tr_{\vec{x}}[\![t_2]\!], ..., tr_{\vec{x}}[\![t_{\|\vec{t}\|}]\!]$	195
$tr_{\vec{x}}[\![\vec{ft}]\!]$	string of $tr_{\vec{x}}[\![ft_1]\!], tr_{\vec{x}}[\![ft_2]\!], ..., tr_{\vec{x}}[\![ft_{\|\vec{ft}\|}]\!]$	195
$\underline{tr}$	syntactic translation from terms of b-clones to binding terms	196
$\Gamma \vdash_{iBA} \Delta$	Δ is derivable from Γ in $\vdash_{iBA}$	204
$\underline{\Gamma} \cup Ax_b \vdash_{EQ} \underline{\Delta}$	$\underline{\Delta}$ is derivable from $\underline{\Gamma}$ in $\vdash_{EQ}$ with Ax_b	207
$\vdash_\alpha p \simeq q$	$p \simeq q$ is derivable in $\vdash_{iBA}$	207
Σ^{BO}	binding signature	233
Σ^{VBTO}	signature of variable binding term operators	233
Ax^{log}	set of axiomatic schemas for First Order Logic	258
Σ^{λ}	signature of Lambda Calculus	270
app	application operator in Lambda Calculus	270
λ	λ-abstraction operator in Lambda Calculus	270
T^{λ}	set of ordinary λ-terms	270
FT_1^{λ}	set of function λ-terms with arity 1	270
$\vdash_{\lambda}$	Lambda Calculus	270
Σ^{c}	signature of Combinatory Logic	270
K	**K** combinator in Combinatory Logic	270
S	**S** combinator in Combinatory Logic	270
ap	application operator in Combinatory Logic	270
T^{c}	set of ordinary terms of Combinatory Logic	270
Cl	equational theory of Combinatory Logic	270
$\vdash_{c}$	derivation sign of Combinatory Logic	270
(K)	**K**-axiom in Combinatory Logic	270
(S)	**S**-axiom in Combinatory Logic	270
$\tilde{x}, \tilde{y}$	variables over $\vec{\chi}$	270
-	bijective map between variables in Lambda Calculus and Combinatory Logic	271
λ-C	syntactic translation from λ-terms to CL-terms	271
C-λ	syntactic translation from CL-terms to λ-terms	271

T^{λ}	set of terms of Lambda Calculus	271
T^{c}	set of terms of Combinatory Logic	271
$\mathbf{K}_{\lambda}$	**K**-combinator in Lambda Calculus	271
$\mathbf{S}_{\lambda}$	**S**-combinator in Lambda Calculus	271
ab	λ-abstraction operator in Combinatory Logic	272
I	identity combinator in Combinatory Logic	272
$\vdash_{c-}$	weaken derivation sign of Combinatory Logic	273
CL_{λ_a}	equational theory of λ-algebras	275
$CL(A_a)$	another version of CL_{λ_a}	275
A_a	set of axioms or schemas for λ-algebras	275
$[\,]$	context symbol	279
$Cont^{c}$	set of contexts for Combinatory Logic	279
$\mathbf{ab}(\langle \underline{x} : C[\,] \rangle)$	**ab** over function contexts	279
$\mathbf{ab}(\langle \underline{x} : C \rangle)[\,]$	$\mathbf{ab}(\langle \underline{x} : C[\,] \rangle)$	280
$\mathcal{C}$	Cartesian Closed Category	288
$u * v$	composition	289
$u; v$	sequential composition	289
T	terminal object	289
$u \times v$	cartesian product	289
$t \cdot t'$	application	289
$\pi_u^{u \times v}$	first projection	289
$\pi_v^{u \times v}$	second projection	289
$< u, v >$	pairing	289
v^u	exponent	289
ev	evaluation	289
U	reflexive	290
$\sim^{hp}$	observational equivalence generated by $\sqsubseteq$ in [69]	293
CCS	Calculus of Communicating Systems	294
Σ^{ccs}	signature for pure finite CCS	294
$E_{\mathbf{eq}}$	simple boolean language with one 2-ary predicate **eq**	295
Σ^{bool}	signature of a simple boolean language	295

Σ^{ccs}	signature for finite CCS	295
Σ^{msgs}	signature of communication messages	295
$\alpha!e$	ouput action in CCS	296
$\alpha?x$	input action in CCS	296
$\mathbf{Nat}_\perp$	set of syntactic names for natural numbers	296
depth	depth function on CCS terms	296
$T_a(V_m \cup FV_{m \Rightarrow a})$	set of CCS terms with variables V_m and function variables $FV_{m \Rightarrow a}$	296
T_a	short for $T_a(V_m \cup FV_{m \Rightarrow a})$	296
Eval	collection of all possible evaluations in CCS	296
$t\ [\![\bullet]\!]^{src} \xrightarrow{\Xi} [\![\bullet]\!]^{tgt}\ u$	form of transitions in CCS	296
$\sim_\omega$	observation up to ω level with an evaluation	299
$\sim^1$	(first) strong bisimulation	299
$\sim^2$	(second) strong bisimulation	299
$\sim_c^2$	strong bisimulation	301
$\sim_c$	strong bisimulation independent of contexts	301
$Eq_{\sim_c}$	set of axiomatic schemas for strong bisimulation	301
Subterm	function taking CCS terms and returning sets of their sub-terms	302
$t \notin IF$	t is not a if-statement	305
$t \in IF$	t is a if-statement	305
$path([\![\bullet]\!], t)$	path of t under evaluation $[\![\bullet]\!]$	308
$paths(t)$	set of paths of t	308
$subterm(p, t)$	sub-term of t under path p	308
$premise(p, t)$	premises along path p on t	308
$b \leftrightarrow b'$	b and b' are semantically equivalent with each other	310
Eq_τ^{ccs}	set of equational axiomatic schemas associated with strong bisimulation	312
$\approx$	weak bisimulation	314
$\approx_c$	weak bisimulation independent of contexts	314
$\approx^*$	weak equivalence	315
$\Longrightarrow$	weak transition	315

$Eq_{\tau-1}^{ccs}$	set of extra axiomatic schemas besides Eq_τ for weak bisimulation	320
$tr_{k,\vec{x}}^{ext}$	extended translation of $tr_{\vec{x}}$	342
$\underline{tr}^{ext}$	extended translation of $\underline{tr}$	343

Index of Names and Organizations

Aczel, 2, 7, 17, 18, 28, 38

Barendregt, 46, 269
Barwise, 52
Beeson, 17
Benthem, 52
Bergstra, 45
Birkhoff, 2, 18, 35, 65, 159, 250
Blishen, 7
Buchberger, 7
Bunder, 46
Burstall, 6, 56

Cao, 7
Chinese Academy of Sciences, 6
Church, 53
CSLI, 7
Curry, 46, 271

Dalen, 17
Dam, 7

Ehrig, 104
Ershov, 46

Farres-Casals, 7
Feferman, 17
Fenstad, 17, 52
Fourman, 7

Frege, 2, 28
Friedman, 3, 19, 28, 38, 51, 234, 250

Gödel, 45
Givant, 46
Goguen, 56, 104

Halmos, 49
Hayashi, 267
Henkin, 27, 49, 104
Hennessy, 293
Hilbert, 45
Hindley, 46
Hinman, 45
Hu, 257
Huet, 53

Institution, 56

Jones, 7

Kechris, 2, 17, 30
Kleene, 6, 17, 23, 45
Klop, 46
Kolaitis, 17
Kolmogorov, 21
Koymans, 54, 290
Kreisel, 45

LFCS, 7

Lu, 257

Maclane, 54
Mahr, 104
Matijasevic, 45
Meseguer, 104
Milner, 7, 314
Mitchell, 7
Moggi, 7
Moller, 7
Monk, 49
Moschovakis, 2, 17, 30

Paczkowski, 7
Paulin-Mohring, 46
Peano, 53
Petri, 6, 296
Plotkin, 3, 6, 17, 51, 52, 56, 293, 296
Presburger, 53

Robinson, 53

Sannella, 56
Sato, 267
Scott, 17
Seldin, 46
Shapiro, 18, 53
Statman, 28, 51
Strong, 45

Tait, 45
Tarlecki, 56
Tarski, 46, 49, 53
Taylor, 53
Tofte, 7
Troelstra, 17
Tucker, 45

Volken, 286, 287

Walker, 7, 315
Wang, 46
Westerstahl, 52

Zhou, 6

Index of Abbreviations

BE, 27
BO, 18, 21, 48

CC, 48
CCC, 288
CCS, 19, 21
CL, 46, 269
CSLI, 7
CSP, 21

dBE, 33
dEQ, 64

eBA, 18, 29
eBC, 35, 42, 136
eBH, 34
EQ, 64

FBO, 18, 46, 52
FOL, 260

iBA, 19, 37

LFCS, 7
LOTOS, 21
LR, 51

ML, 21

qBE, 33
qEQ, 64

SND, 261

uBE, 34

VBTO, 20, 48
VSLI, 6

Index of Definitions

$(Ab'_{\mathbf{K}})$, 276

$(Ab'_{\mathbf{S}})$, 276

(Ab_f), 276

$(Ab_{\mathbf{I}})$, 278

$(Ab_{\mathbf{K}})$, 279

$(Ab_{\mathbf{S}})$, 279, 280

(Ab_η), 282

(Ab_ξ), 280

(Ab_f), 278

$(\mathbf{KS}'')$, 278

$(\mathbf{K}'')$, 278

$(\mathbf{K}')$, 278

$(\mathbf{K_{ab}})$, 279

$(\mathbf{KS}')$, 278

$(\mathbf{KS_{ab}})$, 279, 280

$(\mathbf{SK_{ab}})$, 278

$(\mathbf{S}'')$, 278

$(\mathbf{S}')$, 278

$(\mathbf{S_{ab}})$, 279

A'_a, 280

$CL(A_a)$, 275

Σ-algebra, 63

Σ-congruence, 69

Σ-equation, 66

Σ-isomorphism, 70

β-schema, 270

$\cong$, 162

λ-algebra, 275

λ-family, 286

λ-model, 282

$\vdash$-free, 305

$cl_\alpha(\Gamma)$, 339

ext_c-axiom, 282

ext_c-schema, 282

ext_λ, 282

K-axiom, 270

S-axiom, 270

(!-+-τ-elim), 321

(!-τ-elim), 320

(!-forward), 305

(+-τ-+-elim), 321

(+-τ-elim), 320

(+-ass), 303

(+-cmm), 303

(+-forward), 305

(+-rdc), 303

(?-**else**-τ-elim), 321

(?-**then**-τ-elim), 321

(?-τ-+-elim), 321

(?-τ-elim), 320

(?-forward), 305

(Ax_b-assoc), 206

(Ax_b-left-proj), 206
(Ax_b-right-proj), 206
(Ax_b-unif), 206
(Nil-+-elim), 303
(Nil-|-elim), 304
(α), 43
(β), 270
(β-schema), 270
(**else**-cut), 304
(**else**-forward), 304
(**if-else**), 304
(**if-then**), 304
(**if**-both), 304
(**if**-forward), 305
(**then**-cut), 304
(**then**-forward), 304
(η), 282
($\exists$-elim), 259
($\exists$-intr), 259
($\forall$-elim), 259
($\forall$-intr), 259
(|-ass), 303
(|-cmm), 303
(|-forward), 305
(|-to-+), 304
($\neg$-elim), 258
(ν-cmp), 139
(ν-cons), 139
(ν-proj), 139
($\supset$-elim), 258
($\supset$-intr), 258
(τ-elim), 320
(τ-forward), 305
($\vee$-elim), 258
($\wedge$-intr), 258
(ξ), 43
(ξ^{-1}), 43
(ext_c-axiom), 282
(ext_c-schema), 282
(FOL-$\exists$-elim), 261
(FOL-$\exists$-intr), 261
(FOL-$\forall$-elim), 261
(FOL-$\forall$-intr), 261
(FOL-$\neg$-elim), 260
(FOL-$\supset$-elim), 260
(FOL-$\supset$-intr), 260
(FOL-$\vee$-elim), 260
(FOL-$\wedge$-intr), 260
(FOL-cut), 260
(FOL-id), 260
(FOL-left-$\vee$-intr), 260
(FOL-left-$\wedge$-elim), 260
(FOL-right-$\vee$-intr), 261
(FOL-right-$\wedge$-elim), 260
(FOL-wkn), 260
(adm-ξ), 158
(b-cln-assoc), 193
(b-cln-left-proj), 193
(b-cln-right-proj), 193
(b-cln-unif), 193
(cmp-1), 43
(cmp-2), 44
(crs-sub), 80
(d-ξ), 169

(d-ξ^{-1}), 169
(d-MP), 89
(d-both-sub), 91
(d-cmp), 89, 92
(d-ctr), 89
(d-cut), 89
(d-id), 88
(d-intr), 91
(d-post-sub), 91
(d-rfl), 88
(d-sub), 89
(d-sym'), 89
(d-sym), 92
(d-trs'), 89
(d-trs), 92
(d-wkn), 89
(eBA-cmp), 29
(eBA-cons), 29
(eBA-proj), 29
(eBA-unif), 30
(eBC-cmp-1), 136
(eBC-cmp-2), 136
(eBC-ext), 136
(eq-MP), 80
(eq-cmp), 80
(eq-ctr), 80
(eq-cut), 80
(eq-id), 80
(eq-rfl), 80
(eq-sym), 80
(eq-trs), 80
(eq-wkn), 81
(gen-func-sub), 205
(gen-ord-sub), 205
(gen-pure-func-sub), 205
(iBA-assoc), 38
(iBA-left-proj), 38
(iBA-right-proj), 38
(iBA-unif), 38
(left-$\vee$-intr), 259
(left-$\wedge$-elim), 258
(m-Ax_b-assoc), 252
(m-Ax_b-left-proj), 252
(m-Ax_b-right-proj), 252
(m-Ax_b-unif), 252
(ord-sub), 205
(pure-func-sub), 204
(q-ξ^{-1}), 171
(q-cmp), 96
(q-ctr), 97
(q-cut), 97
(q-d), 97
(q-id), 96
(q-rfl), 96
(q-skm), 98
(q-sub), 97
(q-sym), 96
(q-trs), 96
(q-wkn), 97
(right-$\vee$-intr), 259
(right-$\wedge$-elim), 258
2-category, 55

($\tilde{\approx}$-α), 125

($\dot{\approx}$-ξ), 126
($\dot{\approx}$-ξ^{-1}), 125
admissible calculus $\dot{\vdash}_{eBA}$, 157
calculus $\dot{\vdash}^{d}_{eBA}$, 169
calculus $\dot{\vdash}^{q}_{eBA}$, 171
substitution functions $\vec{\varrho}$, 181

admissible, 151
admissible eBC, 149
admissible eBH, 145
admissible free generator, 146
admissible substitution, 156
admissibly free, 145
admissibly indistinguishable, 37
admissibly invariant, 154
axiomx Ax_b, 206

b-clone, 193
b-clone's term, 192
binding clone, 193
binding core of ζ, 141
binding kernel, 141
binding pre-algebra, 37
binding terms, 24

calculus $\dot{\vdash}^{u}_{eBA}$, 173
calculus $\vdash_{\lambda}$, 270
calculus $\vdash_{dEQ}$, 90
calculus $\vdash^{-}_{dEQ}$, 88
calculus $\vdash_{iBA}$, 204
calculus $\vdash^{d}_{iBA}$, 237
calculus $\vdash^{q}_{iBA}$, 241
calculus $\vdash^{u}_{iBA}$, 245
calculus $\vdash^{d-}_{iBA}$, 236
calculus $\vdash^{u-}_{iBA}$, 243
calculus $\vdash_{qEQ}$, 98
calculus $\vdash^{d}_{qEQ}$, 96
calculus $\vdash_{snd}$, 260
calculus $\vdash_{uEQ}$, 101
concise semi-nf, 309
conclusion, 87
consequence family, 83
context, 300
correspondence $\frown$, 192
cross-fully invariant, 78

direct product, 160
double eBC, 139

eBA, 30
eBC, 136
eBH, 118
eBH-cmp, 118
eBH-cons, 118
eBH-func, 118
eBH-proj, 118
eBH-unif, 118
environment, 31
equation, 66
extended $\underline{tr}^{ext}$, 343
extended $tr^{ext}_{k,\vec{x}}$, 342
extensional Σ^{BO}-algebra, 30
extensional Binding Algebra, 30
extensional Binding Congruence, 136
extensional Binding Homomorphism, 118
extensional iBA, 179

first bisimulation $\sim^1$, 299
first weak observational equivalence, 316
formula, 258
free Σ-algebra, 71
free generator, 134
free variable, 182
freely-generated, 134
fully invariant, 75
func-sub, 43
function terms, 24
functional $\mathcal{O}_\tau$, 298

generated sub-eBA, 120

iBA, 38
identity context, 300
indistinguishable, 32
intensional Binding Algebra, 38
intensional satisfaction $\models_{iBA}$, 201
interpretation, 32

logical, 51
Logical Relation, 162

model, 67, 99

natural homomorphism, 70
normal form, 306

observational equivalence $\sim_\omega$, 299
operational semantics, 296
ordinary terms, 24

perfect sub-eBA, 117
post-condition, 87
pre-condition, 87
premise, 87
pseudo semi-closed, 277

quotient Σ-algebra, 70
quotient eBA, 138

refined semi-nf, 310
relative substitution $\asymp^\Gamma$, 211

satisfaction $\models_{dEQ}$, 86
second bisimulation $\sim^2$, 299
second weak observational equivalence, 316
semi-closed, 301
semi-Normal form, 321
standard semi-Nf, 328
strong bisimulation $\sim_c^2$, 301
sub-Σ-algebra, 68
sub-direct embedding, 160
sub-direct product, 160
sub-eBA, 116

term eBA, 129
third bisimulation, 311
third weak bisimulation, 322
translation $\underline{tr}$, 196

uniform over $< A', \mathcal{F} >$, 30
universal mapping property, 134

valuation, 31
Variable Binding Term Operator, 48
variable index eliminable, 83
variable index free, 83

Index of Subjects

(Σ, Γ)-algebra, 67

Σ-algebra, 67, 194

Σ-equation, 90

Σ-homomorphism, 67

Σ-isomorphism, 70

Σ-term, 196

Σ^{BO}-algebra, 112

α-convertible, 39, 200

β-algebra, 54, 288, 290, 292

β-axiom, 345

β-conversion, 26, 44, 54, 55, 277

β-schema, 345

λ-abstraction, 2, 38, 46, 272, 273, 287

λ-algebra, 269, 274, 275, 282

λ-definability, 51

λ-definable, 51

λ-definable function, 56

λ-definable functional, 17

λ-expression, 21

λ-family, 2, 287

λ-model, 269, 282, 283, 287

λ-term, 269

$\lambda\beta\eta$-calculus, 286

$\vdash$-free, 321, 322

ω-level, 311

ω-rule, 294

$\mathcal{P}\omega$ model, 17

($\dot{\approx}$-ext), 127

(η-axiom), 282

(ξ) rule, 238, 239

(d-ξ), 171

(ord-sub) rule, 205

(q-ξ), 171

if-statement, 293

let-expression, 21

1-1 correspondence, 328

2-category, 54

admissible ξ rule, 158

syntactic expressive power, 197

abstract algebra, 54

abstraction, 21

adding, 107, 203

Adjunction, 22, 266

Admissibility, 20, 36, 42, 147, 153, 157, 162

admissible, 2, 37, 51

admissible BE, 37, 42, 152, 155

admissible Binding Equation, 37

Admissible Completeness, 42, 112

admissible condition, 19

admissible dependent implication, 167

admissible eBC, 152

admissible eBH, 152
admissible environment, 37
admissible equality, 112, 152, 338
admissible interpretation, 37
admissible model, 159
admissible quasi-dependent implication, 167
admissible substitution, 170
admissible uBE, 167, 172
admissible universal BE, 172
admissible universal equality, 173
admissible variety, 3, 162
admissibly free, 149
admissibly invariant, 113
algebra, 63
algebraic, 286
algebraic characterization, 337
algebraic function, 286
applied first order language, 53
arbitrary finite binding, 22
arbitrary finite Binding Operator, 22
arity, 24, 50, 62, 270, 343
associative, 289
asynchronous communication, 7
axiom, 27, 339
Axiom of Choice, 154
axiomatic scheme, 339
axiomatization, 308

b-clone, 40–42, 177, 192, 194, 342
BE, 204, 242
bijection, 271, 290
bijective, 70, 140
binding, 2, 17, 18, 42, 46, 47,.52, 109, 308, 309
Binding Algebra, 19, 28, 265
Binding Congruence, 42, 112
binding core, 35, 140
binding core of an eBH, 35
Binding Equation, 2, 3, 18, 27, 204
Binding Homomorphism, 112, 337
binding kernel, 35, 140
binding kernel of eBH, 42
Binding Operator, 2, 18, 21, 48
binding over sorts, 254
binding pre-algebra, 38, 40, 177
binding primitive, 21, 23, 180, 254, 257
binding signature, 249, 253
binding substitution, 153
binding term, 23, 24, 34, 342
binding variable, 182, 188
Birkhoff's approach, 2, 19, 28, 41, 105, 112, 144, 150, 153, 250, 345
Birkhoff's method, 65, 147
Birkhoff's Theorem, 65, 73, 74, 88, 96, 100, 152, 168, 170, 173
Birkhoff's variety, 159
bisimulation, 298
BO, 45, 48
Boolean Algebra, 50, 261, 266
boolean expression, 268
border line, 233
bound, 46, 49, 182, 188, 301
bound variable, 39, 46, 222, 276

boundness, 130
breakdown, 292, 327, 337, 343

calculus $\vdash_{eBA}$, 3, 339
calculus $\vdash_{iBA}$, 3, 339
Calculus of Communicating Systems, 3, 19
Calculus of Constructions, 46, 48
call-by-name, 323
carrier, 34
Cartesian Closed Category, 54, 288
CAT-enriched category, 55
categorical presentation, 22
categoricity, 53
Category Theory, 54, 104, 292
causality, 294
CC, 48
CCC, 290, 292
CCS, 27, 44, 293, 294, 330
Church-Rosser property, 18, 26
circular argument, 344
CL, 270
clarification, 103
clone, 104
closed binding, 22, 46
closure, 339
co-limit, 104
coding, 180
combinator, 272
Combinatory Algebra, 274
Combinatory Logic, 3, 19, 44, 46, 269, 285
Communication Mechanisms, 6
communication message, 295
commutativity, 35, 42, 112, 147, 337
complete, 3, 43, 44, 83, 91, 99, 103, 106, 109, 238, 257, 261, 262
complete lattice, 106
completeness, 42, 65, 83, 90, 97, 104, 159, 169, 172, 174, 177, 203, 229, 232, 233, 238, 241, 242, 245, 261, 327, 328, 342
composition, 31, 34, 136, 291
compositionality, 69
computation, 3
concentration of a suitable class of functionals, 31
concise, 309, 311, 322
concurrency, 2
conditional equation, 64, 103
congruence, 65, 69, 301
consistency, 180
constant, 31, 34, 48, 192, 249
constructor, 253
context, 279
contraction, 327
core, 150
counter-example, 90, 95, 107, 171
counterpart, 42, 46, 79, 129, 198, 272–274, 286
cross-fully invariant, 79, 81, 105
currying, 290
cut rule, 43
cylinder, 49

Cylindric Algebra, 20, 22, 49

data-dependency, 3, 19, 27, 44, 293, 315, 330
data-dependent, 308
dBE, 44, 242
dBE derivabiltiy, 237
dead-lock-free, 6
decoding, 180
definability, 114
definable, 286
definable element, 285
denotational semantics, 56
dependent BE, 41
dependent Binding Equation, 33
dependent equation, 64, 94, 104
dependent implication, 86, 234, 264
dependent Indistinguishability, 33, 86
depth, 310
depth function, 296
derivability, 196, 207, 218, 262, 264, 277, 280
derivable, 204, 264, 279, 280
derivable relation, 260
derivation closure, 281
derivation preservation, 203
derivations, 277
diagonal eBC, 160
diagonal relation, 160
diminished second order language, 18, 53
Diophantine, 45
Diophantine equation, 19, 45
direct product, 3, 338
directly reachable, 324
Discretive Category, 55
Domain Theory, 6
double quotient algebra, 70
dropping off, 114, 202

eBA, 2, 30, 47, 54
eBC, 112
eBH, 112
eliminable, 328
embedding, 48, 290
empty carrier, 66, 83, 84, 104
empty sort, 66, 81, 104, 262
empty-sort, 265
enough point, 51
environment, 63, 68, 151, 194, 289
equality, 28, 64, 150, 181, 194, 196, 203, 229, 343
equality **eq**, 265
equalizer, 55
equation, 64
equation with variable index, 65
Equational calculus, 41
equational characterization, 312
equational implication, 64, 103
Equational Logic, 3, 28
equational logics, 66, 103
equational reasoning, 50
equationalization, 44
equationally definable, 105, 276

equivalence, 69
equivalent, 185
equivalent substitution, 185
evaluation, 32, 290, 311
evaluation environment, 296
existence of homomorphisms, 66
existential quantifier, 20, 49, 53, 299
explicit closedness, 29–31, 122, 128, 138
explicitly closed, 29, 30, 34, 287
exponent, 289
expressible, 334
expressive power, 45, 50, 197, 295
extended translation, 343
extension, 28
extensional, 3, 28
extensional approach, 41, 51
extensional Binding Algebra, 18, 29, 30
extensional Binding Congruence, 35
extensional Binding Homomorphism, 34
extensional binding isomorphism, 140
Extensionality, 35, 36, 42, 117, 122, 177, 282

FBO, 46, 52, 112
finite, 2, 55, 62
finite Binding Operator, 23
finite limit, 55
finite order, 17
finite type, 17
finiteness, 300
first order, 2, 18, 40, 53
first order algebra, 2, 31, 40–42, 47, 62, 116, 122, 150
first order congruence, 136
first order language, 103
First Order Logic, 3, 19, 44, 48–50, 53, 109, 257, 330
first order subalgebra, 117
first order term algebra, 63
fixed point, 81
flow chart, 45
Flowchart Computability, 19
flowchart computable, 46
FOL, 260
formula, 267
formula-as-type, 267
Framework for Binding Operators, 2, 18, 52, 53, 112, 249, 337
Framework for BOs, 24, 26, 28, 48, 269, 293
Free Algebra, 22
free eBA, 134
free function variable, 182
free quotient term algebra, 73
free variable, 39, 157, 182, 186–188, 195, 196, 301
freeness, 135
Frege Structure, 2, 17, 28, 29
fresh, 98
fresh constant, 98
Friedman's approach, 29, 37, 41, 234, 250, 345
Full Abstraction, 20, 56

full type hierarchy, 51
fully invariant, 78, 81, 105
fully invariant congruence, 75
function, 249
function environment, 39
function term, 24
function variable, 24
functional, 30, 34, 42, 106, 249
functional $\mathcal{O}$, 315
functional completeness, 6
functional programming, 26, 103
functional substitution, 31, 271, 342
functionality, 269, 273
functionals, 28
functor, 48
functorial, 288

generatable, 134
generated Σ-algebra, 71
generated algebra, 71
generated eBA, 140
generated sub-Σ-algebra, 71
generated term eBA, 133
generator, 71, 124
geometry, 49
guidance, 6

handshaking, 7
Henkin's method, 261, 265
higher order, 18, 23, 24, 53, 253
higher sort, 253
higher type, 2, 17, 18, 23, 24, 45, 53, 54, 253
higher-sorted, 24, 253
higher-sorted FBO, 253
Hilbert 10th problem, 45
homo, 55
homomorphism, 65

iBA, 3, 41, 42, 54, 193, 194, 261, 289
identifiable, 317
identity context, 279
identity function, 81
identity preservation, 197
Identity rule, 43
identity substitution, 39, 107, 157, 158, 207
image, 140
incomplete, 22, 90
inconsistency, 22
independence, 6, 225
independency, 309
independent, 289
indexed operation, 40
Indistinguishability, 33, 64, 265
inductive definable relation, 17
inductive reasoning, 328
inference rule, 34, 264, 338
infinite, 299
infinite binding, 22
infinite sum, 293
infinite sum term, 299
influence, 311
initial object, 134
injective, 68, 70, 291

injectiveness, 147
Institution, 20
institution-independency, 57
intension, 28
intensional, 3, 28
intensional approach, 41, 51
intensional Binding Algebra, 37
inter-substitutable, 286
internalization, 265
interpretation, 23, 30–32, 63, 194
intersection, 69, 119, 148, 160
Intuitionistic Logic, 46, 267
intuitionistic type theory, 267

Jumelage meeting, 6
justify, 84, 122

kernel, 65, 150
kernel of homomorphisms, 70
Kolmogorov Complexity, 21
Koymans' isomorphism, 290

Lambda Calculus, 2, 17, 19, 26, 44, 46, 56, 269, 285, 345
larger variable index, 84
least fixed point, 81, 89
Liar paradox, 52
limit, 55
linear Lambda Calculus, 56, 270
LISP language, 267
logic, 2
logic programming, 103, 267
Logical Design, 6
logical eBH, 51
logical reasoning, 50
Logical Relation, 3, 19, 20, 51

many-sorted, 24, 42, 62, 78, 249
many-sorted b-clone, 251
many-sorted binding term, 249
many-sorted first order algebra, 62
Martin-Löf, 267
mathematical logic, 259
maximal fixed point, 299, 316
Milner, 3
minimal fixed point, 81
mistake, 259
ML, 26, 57
model, 31, 78, 83, 172, 242
model-theoretic, 41, 266
Modus Ponens, 43
Monad, 22
Monadic Algebra, 50
monotonic, 6, 81, 130, 299, 316
monotonicity, 121, 124
morphism, 51

natural eBH, 139
Natural Language, 20, 52
necessary and sufficient, 2, 86, 105, 147
necessity, 276
nested **if**-statements, 311
nested **if**-statements, 308
noetherian, 306
non-admissible, 159
non-empty, 265

non-emptyness, 109
non-existence, 98, 109, 285
non-existence of interpretations, 67
non-first-order object, 53
non-free, 199, 224
non-free variable, 113, 186, 202
non-reachable, 309
non-strong-normalizability, 322
Number Theory, 28, 45

object, 28, 55
observability, 294
observational equivalence, 293, 298
operation, 192
operational semantics, 296
ordinary term, 24
ordinary variable, 24

pair, 289
parallel computation, 3
partial Logical Relation, 51
partial order, 81
path, 308, 332
Peano Arithmetic, 53
perfect sub-eBA, 37, 42, 112
perfectness, 161
permutation, 107, 202
Petri Net, 6, 296
Polyadic Algebra, 20, 49, 50
polymorphic typed, 253, 254
polymorphism, 254
polynomial, 28
power of binding, 45
predicate, 265
Presburger Arithmetic, 53
preservation, 41, 161
Prestructure, 3, 38
priority, 294
product, 55, 159, 182, 289, 290
program, 267
programming, 2, 44
projection, 31, 34, 41, 48, 50, 289–291
Projective Algebra, 50
proof, 267
proof power, 19, 55, 275, 277, 339, 345
proof-theoretic, 266
proposition, 267
Propositional Logic, 50
Protocols, 6
pseudo semi-closedness, 277
pure equation, 106

qBE, 170
qBE derivability, 240, 241
quantification, 104, 266
quantifier, 49, 52
quasi-dependent BE, 41
quasi-dependent Binding Equation, 33, 338
quasi-dependent equation, 64, 93, 100, 104
quasi-dependent implication, 234, 264
quasi-dependent Indistinguishability, 33
quasi-equation, 65, 103
quotient algebra, 65, 69, 288

quotient Binding Algebra, 112
quotient eBA, 35, 112, 140
quotient term algebra, 288

r.e. relation, 17
rank, 62
reachability, 311
reachable, 309
Realizability, 46
Recursion Theory, 17, 45
recursive equation, 45
recursive function, 17, 19, 45, 56
recursively enumerable predicate, 45
redex, 306
reduction system, 306
redundancy, 233
redundant, 233
refined, 310, 311, 322
refined semi-Nf, 322
refined semi-nf, 311
Reflectivity rule, 43
reflexive object, 290
Relation Algebra, 50
relationship between eBA and iBA, 177
relative derivability, 216
remedy, 2, 19, 36, 41, 153, 167
renaming, 183
retraction pair, 290
Robinson, 56
Robinson Arithmetic, 53
role, 26, 33, 72, 84, 122, 186, 224, 277, 309, 324, 327, 342
role of binding, 40, 180
role of variable indices, 67

same, 186
satisfaction, 76, 78, 79, 95, 109, 201
satisfaction $\models_{eBA}$, 2, 338
satisfaction $\models_{iBA}$, 338
satisfaction of equations, 66
schema, 27
second order, 2, 18, 40, 53
Second Order Logic, 22
second order Logical Relation, 51
Self-independent, 6
self-independent, 296
semantic, 23, 218
semantic composition, 201
semantic equality, 197
semantic preservation, 201, 229
semantic substitution, 74, 113, 126
semantics, 28
semi-closed, 301, 307, 339
semi-Nf, 321
semi-nf, 307
semi-normal form, 306
semi-normalizability, 307
sequent natural deduction, 260
set inclusion, 81
Set Theory, 17, 46
set-theoretic, 41
signature, 2, 18, 22, 62, 251, 262
signature Σ, 62
signature Σ^c, 270

signature Σ^{λ}, 270
signature Σ^{BO}, 23
signature Σ^{ccs}, 294
signature Σ^{log}, 257
significant, 262
silent action, 314
simple algebra, 56
simple typed, 253
simultaneous substitution, 39, 181
single-sorted, 23, 62, 78, 249
Situation Theory, 52
size of algebra, 84
skolemization, 98, 108
smaller index, 84
sound, 3, 43, 83, 91, 99, 103, 106, 109, 126, 238, 261, 262
soundness, 42, 65, 83, 90, 97, 104, 159, 169, 172, 174, 177, 202, 203, 229, 239, 261, 264, 320, 342
source evaluation, 296
specification, 6
standard, 328
standard interpretation, 261
standard model, 31, 56
strong bisimulation, 44, 294, 311, 312, 314, 327, 330
strong normalizable, 306
strong normalization, 306
strong transition, 324, 327
strongly bisimulated, 301
structural induction, 114
structural operational semantics, 56
sub-direct embedding, 160, 161
sub-direct product, 160
sub-eBA, 37, 42, 112
sub-extensional Binding Algebra, 112, 116
subalgebra, 42, 68
substitution, 46, 75, 94, 154, 182, 183, 185, 187, 207, 277, 342, 343
substitution exchange, 273
substitution rule, 42, 91, 328, 343, 344
subterm, 332
suitable class of functionals, 2, 30
surjective, 69, 70, 139, 291
Symmetricity rule, 43
syntactic, 23, 218
syntactic preservation, 229
syntactic substitution, 74, 126, 201
syntactic sugar, 308
syntactical λ-model, 287
syntactical applicative structure, 287

target evaluation, 296
target terms, 324
Tarskian doctrine, 22
term algebra, 71, 72
term Binding Algebra, 266
term eBA, 34, 112, 125
term rewriting, 103
terminal object, 289
termination, 333
ternary logic, 6
textual substitution, 301

transition system, 296, 315
Transitivity rule, 43
translation, 41, 42, 194, 271
translation λ-C, 274
translation $\underline{tr}$, 196
translation C-λ, 274
translation $tr_{\vec{x}}$, 195
true, 267
type, 267
type-2 functional, 23, 45
type-2 operator, 18, 23
type-free, 23
typed lambda calculi, 6
typed presentation, 23

uBE, 44, 242, 257
uBE derivability, 245
unary, 2
Uncertainty, 6
uniform, 30, 34, 287
uniform over $\mathcal{F}$, 30
uniformity, 30, 32, 123, 128, 136, 138
unique, 29, 68
uniqueness, 72
Universal Algebra, 2, 17, 18, 22
universal algebra, 71
universal BE, 41
universal Binding Equation, 34, 338
universal equality, 234
universal equation, 64, 99
universal Horn clause, 65, 103
universal quantification, 66
universal quantifier, 66
universally quantified, 105
unobservable change, 317
unsound, 97

validation, 109
valuation, 113
value-passing, 293
Variable Binding Term Operator, 20, 48, 52
variable index eliminable, 85
variable index free, 86, 104
variable indices, 76
variable permutation, 218
variable sort, 254
variable-indexed equation, 66
variety, 159
VLSI circuit, 6

weak bisimulation, 7, 44, 294, 314, 327, 330
weak transition, 324, 327
Weakening rule, 43
weakly bisimulated, 327
well-definedness, 32, 40, 113, 115, 118, 127, 136, 156, 177, 181, 185, 188, 191, 195, 196, 272–274