# A Semantics for Static Type Inference

Gordon Plotkin
Department of Computer Science
University of Edinburgh
Edinburgh, EH9 3JZ, SCOTLAND
**gdp@lfcs.edinburgh.ac.uk**

**Abstract**

Curry's system for F-deducibility is the basis for static type inference algorithms for programming languages such as ML. If a natural "preservation of types by conversion" rule is added to Curry's system, it becomes undecidable, but complete relative to a variety of model classes. We show completeness for Curry's system itself, relative to an extended notion of model that validates reduction but not conversion. Two proofs are given: one uses a term model and the other a model built from type expressions. Extensions to systems with polymorphic or intersection types are also considered.

## 1 Introduction

Curry's system for F-deducibility is one of the simplest systems for type inference for the untyped λ-calculus [3,10,11,18]. It is a decidable system that can be considered as the basis of static type inference algorithms for languages such as ML [27]. These algorithms are static, in that the type of a term is calculated without any reductions being carried out. To this end, the calculation is syntax-directed, in that the types of terms are found as a function of the types of their immediate subterms.

Curry's system has, however, a certain defect from a semantical point of view: it is not complete with respect to any immediately obvious semantics. Hindley remarks in [18] that one would expect that, in any decent semantics, interconvertible terms would have the same interpretation. A certain *equality rule* is then valid, that if two terms are interconvertible then they have the same types. Adding this rule yields a proper extension of Curry's system. Hindley and others [4,18,19] proved that completeness indeed holds for the extended system with respect to any of various classes of models; these differ only in how type expressions are interpreted (the simple semantics, the F-semantics and the quotient-set semantics). Unfortunately, this extended system does not seem to support any static type inference (or type checking) algorithms; furthermore, type inference is not total recursive (and type checking is undecidable).

A. Meyer asked whether, nonetheless, a semantic analysis of Curry's system is possible; clearly such an analysis necessitates the use of a wider class of models than those considered by Hindley. We present here one such analysis based on the notion of a *model of reduction*. This terminology was first introduced in [22], but what is called a model of reduction here is called a model of expansion there; the reason for such a – *prima facie* confusing – switch is discussed below.

Models of reduction arise if one considers an analysis of the equality rule into two rules, a *reduction* rule and an *expansion* rule. The reduction rule states that if a term has a type and the term reduces to another then the latter has the same type; the expansion rule states that if a term reduces to another and the latter has a type then so does the former. According to the Subject Reduction Theorem, the first rule is an admissible[1] rule for F-deducibility. However the second rule is not; indeed, as is shown below, adding it produces a system as powerful as that obtained by adding the equality rule.

This analysis of the equality rule into two is carried out formally in section 2. We expect a notion of model that verifies the reduction rule, but not necessarily the expansion rule. To this end, in section 3, models of reduction are taken to be partially ordered and model reduction (by an inequality) but not necessarily conversion (by an equality). Types are modelled as sets that are upper-closed in the ordering; then the reduction rule is indeed modelled, but not necessarily the expansion rule. The idea of modelling reduction by a partial order is, *per se*, hardly new. In the context of combinatory logic, Meyer, Bunder and Powers [26] considered an *asymmetric* combinatory logic, with such axioms as $Kxy \leq x$ and $Sxyz \leq xz(yz)$. For the $\lambda$-calculus the idea was largely anticipated by Girard in [16]. In the categorical literature, order-enriched categories [23] and the more general 2-categories [32,33] have been considered. In the context of universal algebra, Meseguer [25] considered categorical, preorder and partial order interpretations of a "rewriting logic" for conditional rewriting modulo a set of equations; he proves soundness and completeness results relative to all three classes of interpretations.

In section 4, following an idea of Mitchell [28], we consider a generalisation of the simple notion of type interpretation, for the

---

1 We may informally understand a rule to be *admissible* in a formal system if, for any instantiation of the schematic variables that occur in the rule, the conclusion of the rule is provable in the formal system provided its premises are.

system of simple functional types, and give two completeness proofs. One, after the manner of Hindley, uses a term model. The other, after the manner of Barendregt, Coppo and Dezani-Ciancaglini, uses a model built from type expressions; it is very simple, being nothing but the powerset of the set of type expressions. The model of the λ-calculus in [30] was obtained as a modification of this structure. The same structure was known at about the same time to R.K. Meyer who dubbed it the "Fool's Model"; it is a model of asymmetric combinatory logic [5,26]. The simple semantics and the F-semantics are also considered.

In section 5 completeness results for the more complex polymorphic type discipline are obtained. There are results for both the general notion of type interpretation and also for a version of the simple notion. Again both term model and "type-expression model" proofs of completeness are given. Here only certain sets of type expressions are used, being those satisfying a certain infinitary condition; consequently, application is not continuous. Finally, in section 6, we consider the intersection type discipline of Coppo *et al* [6,9]. Here completeness has already been obtained for the simple semantics [4,17] and the F-semantics [12]. We obtain completeness for two more general classes of type interpretations, where there is a pleasing correspondence between the type-expression models and the set-theoretic models of Plotkin and Engeler [30, 14].

It would be interesting to have a formal definition of a notion of a static type inference system. Such a system provides the basis for static type inference (or type checking) algorithms. It is desirable that the system admits a reduction rule. That implies that the types of terms after reduction can be forecast; for programming languages it follows that type-checked programs will not suffer type errors. On the other hand, the expansion rule should be neither present, nor derivable[2], in such systems. Systems where an expansion rule is present (or derivable) explicitly allow *dynamic* type inference, in the sense that a term can have any type attributable to one of its reducts, that is to type a term one can "look ahead" in its computation.

One would expect that all the systems considered in this paper are static (with the exception of the extensions of Curry's system by expansion or equality rules). Formal evidence to this effect can be provided by theorems such as the Subject Construction Theorem for simple types (see Theorem 9B1 in [10] and Theorem 14D2 in [11]), Mitchell's characterisation of his pure typing system for the polymorphic type discipline [28] and Lemma 2.8 for the intersection type discipline of Barendregt, Coppo, and Dezani [3]. These show that the systems are syntax-directed, in that the types inferred for a term are a function of the types inferred for their immediate subterms. Note that there is, unfortunately, no guarantee for such systems that type inference is total recursive - let alone efficient, nor that type checking is (efficiently)

---

2 We may informally understand a rule to be *derivable* in a formal system if there is a proof of the conclusion from the premises, without instantiating the schematic variables occurring in the rule.

decidable. For example, the equality rule is admissible in the systems for intersection types, and therefore typability of closed terms is undecidable. In the case of the polymorphic discipline, the decision problem of typability is an outstanding open problem [15].

It would also be of interest to extend the semantic analysis to other type disciplines. The combination of the polymorphic type discipline with the intersection type discipline of Coppo *et al* is of immediate interest, and one can also consider, for example, existential and union types and recursive types [1,6,7,24]. In a different direction, it would be interesting to extend the language to include other features such as, perhaps, a recursion combinator. Continuing in this way, it would be particularly interesting to see a treatment of a programming language equipped with an operational semantics (rather than notions of reduction and conversion) and a fixed denotational semantics (rather than a range of models). While the first difference may not be very significant, having a fixed model may well be. Moreover, the fascinating issue is raised of refining the techniques of the denotational semantics of programming languages to produce models sensitive to reduction.

Several technical questions are raised below: what the proper notion of quotient-set semantics may be, what should be the categorical notion of a type interpretation, whether there is a continuous type-expression model of reduction of universal types that would support a proof of completeness, and whether the simple containment relation on polymorphic type expressions is decidable. The problems of finding complete systems for models of $\beta$-reduction and F-semantics in the cases of either polymorphic types or intersection types are also left unsolved.

## 2 Curry's System for F-deducibility

Let us begin with syntactical and notational matters. Generally speaking, the notation and terminology of Hindley in [18] will be followed, but with a few differences and additions. We write "type expressions" rather than "type schemes". Type assignment statements are written in the form $M{:}\beta$ rather than $\beta M$. The letters $\Gamma$, $\Delta$ are used to vary over finite bases. When writing $\Gamma, x{:}\alpha$ for $\Gamma \cup \{x{:}\alpha\}$, it is assumed that x is a subject of no type assignment statement in $\Gamma$. Throughout, we do not distinguish $\alpha$-equivalent lambda expressions. On occasion we use vector notation, writing, for example, $\vec{x}$ for lists of variables. We will make free use of evident conventions; for example if $\vec{M}$ is the list $M_1,\ldots,M_n$ and $\vec{x}$ is the list $x_1,\ldots,x_n$ then we write the multiple substitution $[M_1/x_1, \ldots , M_n/x_n]N$ as $[\vec{M}/\vec{x}]N$. It proves convenient to adopt the convention that such multiple substitutions are allowed even when two of the variables in $\vec{x}$ are identical; in that case, the term in $\vec{M}$ that is to be substituted is taken to be the one corresponding to the rightmost occurrence of the variable. With this convention the formula

$$[\vec{M}/\vec{x}][\vec{N}/\vec{y}]L =_\alpha [\vec{M}/\vec{x}, [\vec{M}/\vec{x}]\vec{N}/\vec{y}]L$$

holds unrestrictedly.

There is a convenient sequent-style presentation of F-deducibility for the $\lambda$-calculus [28]. This uses sequents of the form $\Gamma \vdash M{:}\alpha$; there are three rules, and we call this system of rules $\lambda\!\to$ (in [3] Barendregt calls it $\lambda\!\to\!-\mathrm{Curry}$, and uses the notation $\lambda\!\to\!-\mathrm{Church}$ to refer to the simply typed $\lambda$-calculus).

(Var) $\qquad \Gamma \vdash x{:}\alpha \quad$ (if $x{:}\alpha$ is in $\Gamma$)

$$(\to\mathrm{I}) \qquad \frac{\Gamma, x{:}\alpha \vdash M{:}\beta}{\Gamma \vdash \lambda x.M{:}\alpha\to\beta}$$

$$(\to\mathrm{E}) \qquad \frac{\Gamma \vdash M{:}\alpha\to\beta \;,\; \Gamma \vdash N{:}\alpha}{\Gamma \vdash MN{:}\beta}$$

Then *F-deducibility* is deducibility in the system $\lambda\!\to$; that is, the relation $\Gamma \vdash_F M{:}\alpha$ is defined to hold iff $\Gamma \vdash M{:}\alpha$ is provable using these rules.

The equality rule discussed in the Introduction depends on a notion of conversion. In the case of $\beta$-conversion the rule is:

$$(\mathrm{EQ}_\beta) \qquad \frac{\Gamma \vdash M{:}\alpha \;,\; M =_\beta N}{\Gamma \vdash N{:}\alpha}$$

Now, by the Church-Rosser Theorem, we can split this rule up into two, a *β-reduction rule* ($\twoheadrightarrow_\beta$), and a *β–expansion rule* ($\twoheadleftarrow_\beta$):

$$(\twoheadrightarrow_\beta) \qquad \frac{\Gamma \vdash M{:}\alpha \,,\, M \twoheadrightarrow_\beta N}{\Gamma \vdash N{:}\alpha}$$

$$(\twoheadleftarrow_\beta) \qquad \frac{\Gamma \vdash N{:}\alpha \,,\, M \twoheadrightarrow_\beta N}{\Gamma \vdash M{:}\alpha}$$

The Subject Reduction Theorem can be read as stating that ($\twoheadrightarrow_\beta$) is an admissible rule of the system $\lambda\!\rightarrow$; it is therefore not surprising that the whole strength of ($EQ_\beta$) rests with ($\twoheadleftarrow_\beta$).

**Proposition 1** ($EQ_\beta$) is an admissible rule of the system $\lambda\!\rightarrow$ plus ($\twoheadleftarrow_\beta$)
**Proof** We have to show that if $M =_\beta N$ and $\Gamma \vdash M{:}\alpha$ is provable in the system $\lambda\!\rightarrow$ plus ($\twoheadleftarrow_\beta$) so is $\Gamma \vdash N{:}\alpha$. Under these assumptions $\Gamma \vdash M{:}\alpha$ is provable in the system $\lambda\!\rightarrow$ plus ($EQ_\beta$). Therefore by the Equality Postponement Theorem [18] there is a term M' with $\Gamma \vdash_F M'{:}\alpha$ and $M' =_\beta M$. So $M' =_\beta N$, and therefore by the Church-Rosser Theorem there is a term N' with $M' \twoheadrightarrow_\beta N' \twoheadleftarrow_\beta N$. So, applying the Subject Reduction Theorem we get $\Gamma \vdash_F N'{:}\alpha$ and, finally by ($\twoheadleftarrow_\beta$) we get that $\Gamma \vdash N{:}\alpha$ is provable in the extended system. ■

There are two other admissible rules for the system $\lambda\!\rightarrow$ that will prove useful:

$$(\text{Weakening}) \qquad \frac{\Gamma \vdash M{:}\beta}{\Gamma \,,\, x{:}\alpha \vdash M{:}\beta}$$

$$(\text{Strengthening}) \qquad \frac{\Gamma \,,\, x{:}\alpha \vdash M{:}\beta}{\Gamma \vdash M{:}\beta} \quad \text{(provided x is not free in M)}$$

The Subject Reduction Theorem also holds for βη-reduction, and we can introduce rules ($EQ_{\beta\eta}$), ($\twoheadrightarrow_{\beta\eta}$) and ($\twoheadleftarrow_{\beta\eta}$) like those above. The Equality Postponement Theorem holds for βη-conversion, and so we find that ($EQ_{\beta\eta}$) is an admissible rule of the system $\lambda\!\rightarrow$ plus ($\twoheadleftarrow_{\beta\eta}$).

## 3 Models of Reduction

From the analysis of the equality rule ($EQ_\beta$) it seems reasonable to model ($\twoheadrightarrow_\beta$), but not ($\twoheadleftarrow_\beta$). The rule ($\twoheadrightarrow_\beta$) might be read as stating that if type-theoretic information holds also of M, and if $M \twoheadrightarrow_\beta N$, then the information holds of N. So, if we follow Scott and model information by a partial order, we would want the denotation of M to be less than that of N. We will therefore work with an ordered variant of the notion of a syntactical λ-model [2, §5]. To this end, following an idea of Mitchell [29], we first define interpretations, and only then consider models.

An *ordered lambda interpretation* of the λ-calculus is a triple:

$$\mathcal{P} = \langle P, \cdot, [\![\ \cdot\ ]\!](\cdot) \rangle$$

where P is a partial order, and $\cdot$ is a binary monotone operation over P of *application*, and $[\![\ \cdot\ ]\!](\cdot)$ is a mapping from (α–equivalence classes of) expressions and environments to P such that the following four conditions hold:

(i) $[\![ x ]\!](\rho) = \rho(x)$

(ii) $[\![ MN ]\!](\rho) = [\![ M ]\!](\rho) \cdot [\![ N ]\!](\rho)$

(iii) if $[\![ M ]\!](\rho(x := a)) \leq [\![ N ]\!](\rho(x := a))$, for all a in P, then $[\![ \lambda x.M ]\!](\rho) \leq [\![ \lambda x.N ]\!](\rho)$

(iv) if $\rho \restriction FV(M) = \rho' \restriction FV(M)$ then $[\![ M ]\!](\rho) = [\![ M ]\!](\rho')$

The lambda interpretations of Mitchell in [29] correspond to the case where P is discretely ordered. We may now define a *(syntactical) model of β-reduction* to be an ordered lambda interpretation in which:

(v) $[\![ \lambda x.M ]\!](\rho) \cdot a \leq [\![ M ]\!](\rho(x := a))$

holds. If we took equality in (v) we would obtain an ordered version of the usual notion of a syntactical λ-model [2]. Models of β-reduction are models of β-expansion in the sense of [22]; it may be that the authors of that paper took the view that β-reduction should correspond to a reduction in the order. Other authors take reduction, or rewriting, as increasing the order [16,23,25,32,33].

It is worth discussing further why reduction should be viewed as increasing information. There is the type-theoretic argument given above: by the Subject Reduction Theorem, reduction increases the number of types statically determinable. If, for example, one models types by subsets the denotation of a term (in a model) is in the intersection of the denotations of all the statically determinable type expressions. Reduction *decreases* this subset, thereby *increasing* information, as there is less uncertainty as to which element the term denotes. There is also a related argument that the point of computation is to gather information, namely what the result of the computation is. Reduction increases information about the form of the computation of the result, and may give information about the form of the result itself.

In working with interpretations there is an important principle of substitution to the effect that substitution commutes with interpretation:

$$\text{(Sub)} \qquad [\![ [N/x]M ]\!](\rho) = [\![ M ]\!](\rho(x := [\![ N ]\!](\rho)))$$

**Lemma 1** (Sub) holds in any ordered lambda interpretation.
**Proof** Let $\mathcal{P}$ be an ordered lambda interpretation. We consider first the case that x does not occur free in N, and proceed by induction on the structure of M. The cases where M is a variable or an application present no difficulty, so suppose that M has the form $\lambda y.L$ where we may assume without loss of generality that y does not occur free in N.

Let $\rho' = \rho(x := [\![N]\!](\rho))$. Then we have that:

$$[\![[N/x]M]\!](\rho) = [\![\lambda y.[N/x]L]\!](\rho)$$
$$= [\![\lambda y.[N/x]L]\!](\rho')$$

for x cannot occur free in $\lambda y.[N/x]L$ as it is not free in N. Therefore by (iii) it is enough to prove that

$$[\![[N/x]L]\!](\rho'(y := a)) = [\![L]\!](\rho'(y := a))$$

for all a in P. For this we calculate as follows:

$$[\![[N/x]L]\!](\rho'(y := a)) = [\![L]\!](\rho'(y := a)(x := [\![N]\!](\rho'(y := a))))$$

(by induction hypothesis applied to L)

$$= [\![L]\!](\rho'(y := a)(x := [\![N]\!](\rho)))$$

(by (iv), since neither x nor y occur free in N)

$$= [\![L]\!](\rho'(y := a))$$

In case x does occur free in N, we can proceed as in [2, § 5.3.3]. Let z be a variable which is not free in N or M and then we may calculate that:

$$[\![[N/x]M]\!](\rho) = [\![[N/z][z/x]M]\!](\rho)$$
$$= [\![[z/x]M]\!](\rho(z := [\![N]\!](\rho)))$$

(by the above, since z does not occur free in N)

$$= [\![M]\!](\rho(z := [\![N]\!](\rho))(x := [\![z]\!](\rho(z := [\![N]\!](\rho)))))$$

(by the above, since x does not occur free in z)

$$= [\![M]\!](\rho(z := [\![N]\!](\rho))(x := [\![N]\!](\rho))) \text{ (by (i))}$$
$$= [\![M]\!](\rho(x := [\![N]\!](\rho)))$$

(by (iv) since z does not occur free in M). ■

This improves the proof of (Sub) in [2] where the equality version of (v) is also assumed. The possibility of this lemma was suggested to the author by work of Jacobs, Margaria and Zacchi [21]. There is a multiple substitution consequence of (Sub):

$$[\![ [\vec{N}/\vec{x}]M ]\!](\rho) = [\![ M ]\!](\rho(\vec{x} := [\![ \vec{N} ]\!](\rho)))$$

Here a rightmost convention is adopted to resolve conflicts in the interpretation of $\rho(\vec{x} := \vec{a})$, in case the same variable occurs more than once in $\vec{x}$.

Environments can be ordered by the pointwise ordering, where $\rho \le \rho'$ iff for all variables x, it is the case that $\rho(x) \le \rho'(x)$. Then one has a further useful property of ordered lambda interpretations:

**Lemma 2** In every ordered lambda interpretation $[\![ N ]\!](\rho)$ is monotone in $\rho$.

**Proof** Let N be a term, and let $\rho, \rho'$ be an ordered pair of environments. We have to show that $[\![ N ]\!](\rho) \leq [\![ N ]\!](\rho')$. To this end, we proceed by structural induction on N. The cases where N is a variable or an application are easy, so let us consider the case where it is an abstraction of the form $\lambda x.M$. Now let $\vec{y}$ be a non-repeating list of the free variables of $\lambda x.M$, and let $\vec{z}$ be another non-repeating list of variables of the same length as $\vec{y}$, not containing x and having no variable in common with $\vec{y}$. Let $\rho''$ be $\rho'(\vec{z} := \rho(\vec{y}))$.

Then for any element, a, of the interpretation:

$$[\![ [\vec{z}/\vec{y}]M ]\!](\rho''(x := a)) = [\![ M ]\!](\rho''(x := a)(\vec{y} := [\![ \vec{z} ]\!](\rho''(x := a))))$$

(by the multiple substitution version of (Sub))

$$= [\![ M ]\!](\rho''(x := a)(\vec{y} := \rho(\vec{y})))$$

$$= [\![ M ]\!](\rho(x := a)) \text{ (by (iv))}$$

$$\leq [\![ M ]\!](\rho'(x := a))$$

(by induction hypothesis)

$$= [\![ M ]\!](\rho''(x := a)) \text{ (by (iv))}$$

So by (iii) we get that $[\![ \lambda x.[\vec{z}/\vec{y}]M ]\!](\rho'') \leq [\![ \lambda x.M ]\!](\rho'')$, and so:

$$[\![ \lambda x.M ]\!](\rho) = [\![ \lambda x.M ]\!](\rho'')(\vec{y} := \rho(\vec{y})) \text{ (by (iv))}$$

$$= [\![ \lambda x.M ]\!](\rho'')(\vec{y} := [\![ \vec{z} ]\!](\rho''))$$

$$= [\![ [\vec{z}/\vec{y}]\lambda x.M ]\!](\rho'')$$

(by the multiple substitution version of (Sub))

$$\leq [\![ \lambda x.M ]\!](\rho'')$$

(by what we have previously proved)

$$= [\![ \lambda x.M ]\!](\rho') \text{ (by (iv))}. \blacksquare$$

Models of β-reduction do indeed model β-reduction. It is useful to have available a formal system to axiomatizes β-reduction. The system has judgements of the form $M \leq N$ and the following axioms and rules:

$$M \leq M$$

$$\frac{M \leq N\ ,\ N \leq L}{M \leq L}$$

$$\frac{M \leq M',\ N \leq N'}{MN \leq M'N'}$$

$$\frac{M \leq N}{\lambda x.M \leq \lambda x.N}$$

$$(\lambda x.M)N \leq [N/x]M$$

Clearly $M \leq N$ is provable in this system iff $M \twoheadrightarrow_\beta N$.

**Proposition 2** If $M \twoheadrightarrow_\beta N$ then $[\![ M ]\!] \leq [\![ N ]\!]$.

**Proof** The proof is by induction on the size of the proof in the formal system. All cases are evident except for the last where one calculates that:

$$\begin{aligned}
[\![ (\lambda x.M)N ]\!](\rho) &= [\![ \lambda x.M ]\!](\rho)\cdot [\![ N ]\!](\rho) && \text{by (ii)}\\
&\leq [\![ M ]\!](\rho(x := [\![ N ]\!](\rho))) && \text{by (v)}\\
&= [\![ [N/x]M ]\!](\rho) && \text{by (Sub).} \blacksquare
\end{aligned}$$

The class of models of β-reduction is not only sound for β-reduction, but is also complete for it, as a term model argument shows. This model,

$$\mathcal{R}_\beta = \langle R, \cdot, [\![ \cdot ]\!](\cdot) \rangle$$

is constructed from equivalence classes of terms. Let

$$[M] = \{ N \mid M \twoheadrightarrow_\beta N \twoheadrightarrow_\beta M \}$$

and then take

$$R = \{ [M] \mid M \text{ a } \lambda\text{-term} \}$$

and partially order it by putting:

$$[M] \leq [N] \text{ iff } M \twoheadrightarrow \beta\ N$$

Finally, define application by:

$$[M]\cdot[N] = [MN]$$

and the interpretation function by:

$$[\![ M ]\!](\rho) = [[\vec{L}/\vec{z}]M],$$

where $\vec{z}$ is a list of the free variables of M, and where $[\vec{L}]$ is $\rho(\vec{z})$.

It is clear that this is a good definition, that application is monotone, and that conditions (i),(ii),(iv) all hold. Let us verify the others. For condition (iii), suppose that $[\![ M ]\!](\rho(x := a)) \leq [\![ N ]\!](\rho(x := a))$, for all a in R. Now let $\vec{z}$ be a list of the free variables in λx.M or λx.N, and set $\rho(\vec{z})$ equal to $[\vec{L}]$. Let y be a variable not free in λx.M or λx.N or any term in $\vec{L}$. Then taking a to be [y] we get, by the assumption, that:

$$[[\vec{L}/\vec{z}, y/x]M] \leq [[\vec{L}/\vec{z}, y/x]N]$$

Therefore:

$$\begin{aligned}
[\![ \lambda x.M ]\!](\rho) &= [[\vec{L}/\vec{z}]\lambda x.M]\\
&= [\lambda y.[\vec{L}/\vec{z}, y/x]M]\\
&\leq [\lambda y.[\vec{L}/\vec{z}, y/x]N] \text{ (by the above)}\\
&= [\![ \lambda x.N ]\!](\rho)
\end{aligned}$$

Finally for (v) we calculate:

$$[\![ \lambda x.M ]\!](\rho)\cdot[N] = [(\lambda x.[\vec{L}/\vec{z}]M)N]$$

(where $\vec{z}$ is a list of the free variables of λx.M, and where $[\vec{L}]$ is $\rho(\vec{z})$)

$$\leq [[\vec{L}/\vec{z}, N/x]M]$$

(we can assume without loss of generality that x is not a free variable of any term in $\vec{L}$)

$$= [\![ M ]\!](\rho(x := [N]))$$

There is a particularly useful environment $\rho_0$ defined by $\rho_0(x) = [x]$. It has the property that for any term M, $[\![ M ]\!](\rho_0) = [M]$.

**Completeness Theorem 1** If $[\![ M ]\!] \leq [\![ N ]\!]$ holds in all models of β-reduction then $M \twoheadrightarrow_\beta N$.

**Proof** Use the term model and take the denotations of M and N in the environment $\rho_0$. ■

Turning to βη-reduction, define a *(syntactical) model of βη-reduction* to be a model of β-reduction such that

(vi) $[\![ \lambda x.Mx ]\!](\rho) \leq [\![ M ]\!](\rho)$ (x not free in M)

Now βη-reduction can be axiomatized by adding the scheme

$\lambda x.Mx \leq M$ (x not free in M)

to the above rules, and one sees that syntactical models of βη-reduction do indeed model βη-reduction. There is a term model $\mathcal{R}_{\beta\eta}$ defined analogously to $\mathcal{R}_{\beta}$, using βη-reduction rather than β-reduction. This term model can be used to yield the expected completeness result for βη-reduction and syntactical models of βη-reduction.

An interesting question is how the present work fits in with the categorical viewpoint. In a cartesian closed category, a model of the λ-calculus is provided by an object which has the space of its endomorphisms as a retract [2]. There is an analogous version of the models of reduction considered here. A category is *order-enriched* if each hom-set is equipped with a partial order structure so that composition is monotonic; it is a *cartesian closed* order-enriched category, if it is order-enriched and cartesian closed and the natural isomorphisms associated to the products and exponentials are monotonic. Then a *categorical model of β-reduction* is provided by an object X in such a category and morphisms $F: X \rightarrow X^X$, and $G: X^X \rightarrow X$ such that $F \circ G \leq id$; a *categorical model of βη-reduction* is provided if, in addition, $G \circ F \leq id$. It is intended to publish the details of the correspondence between the two notions of models of β-reduction elsewhere.

The ordered categorical notion was considered in the case of the category of complete partial orders and continuous functions in [22], and for the full subcategory of qualitative domains in [16]; 2-categorical notions were considered by Seely in [33]. Using the setting of cartesian closed order enriched categories, one can describe a systematic way to construct models of β-reduction (or of βη-reduction): given a model X,F,G of β-conversion (or βη-conversion) in such a category, decrease F or G, for example by pre- or post-composing with a projection. Again, in the category of complete partial orders (for example) consider the $D_n$ used in the construction of the $D_\infty$ model. There $D_n$ is a projection of $D_{n+1}$, which is ${D_n}^{D_n}$, and so is a model of βη-reduction (see [2] for more details of such model constructions). Finally, in [16], Girard showed how infinite models of βη-reduction can be approximated by finite ones.

## 4 Type Interpretations

We model types as subsets of the domains of models of reduction. Fixing a model of β-reduction $\mathcal{P}$ as above, say that a *type interpretation* is a pair

$$\mathcal{T}y = \langle \mathrm{Ty}, \rightarrow \rangle$$

where Ty is a collection of upper closed subsets of P, and $\rightarrow$ is a binary "arrow" function over Ty such that:

(Arrow 1) For any X,Y in Ty, and a in $X \rightarrow Y$, $(a \cdot X)$ is a subset of Y.

(Arrow 2) For any X,Y in Ty, if $[\![ M ]\!](\rho(x := a)) \in Y$ whenever $a \in X$ then $[\![ \lambda x.M ]\!](\rho) \in X \rightarrow Y$.

Define operations on upper-closed subsets of P by:

$$(X \rightarrow_S Y) \;=\; \{ a \mid (a \cdot X) \subset Y \}$$

$$(X \rightarrow_F Y) \;=\; \{ [\![\lambda x.M]\!](\rho) \mid \forall a \in X. [\![M]\!](\rho(x := a)) \in Y \} \uparrow$$

(Here $a \cdot X$ is $\{a \cdot b \mid b \in X\}$, and $Z\uparrow$ is $\{b \mid \exists a \in Z. a \leq b\}$, the upper closure of Z.) Then the above conditions can be re-phrased as:

$$(X \rightarrow_F Y) \subset (X \rightarrow Y) \subset (X \rightarrow_S Y)$$

We say that an interpretation is *simple* (respectively is an *F-interpretation*) if the arrow operation is $\rightarrow_S$ (respectively $\rightarrow_F$).

**Lemma 3** Let $\mathcal{P}$ be a syntactical model of β-reduction; let Ty be a collection of upper closed subsets of P; and let $\rightarrow$ be a binary function over Ty. For any a in P set $\varepsilon(a) = [\![\lambda y.xy]\!](\rho(x := a))$ (where ρ is any environment). Then for all X,Y in Ty:

1. $\varepsilon(X \rightarrow_S Y) \subset (X \rightarrow_F Y)$
2. If $\mathcal{P}$ is a syntactical λ-model then $X \rightarrow_F Y = (X \rightarrow_S Y) \cap F$, where F is $\varepsilon(P)$.
3. If $\mathcal{P}$ is a syntactical model of βη-reduction, and $\langle \mathrm{Ty}, \rightarrow \rangle$ is a type interpretation then $(X \rightarrow_F Y) = (X \rightarrow_S Y)$

**Proof** 1. Suppose that a is in $(X \rightarrow_S Y)$. Then for any b in X, $a \cdot b$ is in Y. But $a \cdot b$ is $[\![xy]\!](\rho(x := a)(y := b))$, and so $\varepsilon(a)$ is in $(X \rightarrow_F Y)$.

2. In one direction, suppose that a is in $(X \rightarrow_F Y)$ and so has the form $[\![\lambda x.M]\!](\rho)$, where $[\![M]\!](\rho(x := b))$ is in Y whenever b is in X. So for any b in X, $a \cdot b$ is in Y, as it is equal to $[\![M]\!](\rho(x := b))$ since $\mathcal{P}$ is a model of β-conversion. Further,

$$a = [\![\lambda x.M]\!](\rho) = [\![\lambda x.(\lambda x.M)x]\!](\rho) = [\![\lambda x.zx]\!](\rho(z := [\![\lambda x.M]\!](\rho)))$$

(the last by Lemma 1) and so a is also in F. In the other direction, if a is in F, then $\varepsilon(a) = a$ and so, by Part 1, if a is in $(X \rightarrow_S Y)$ it is in $(X \rightarrow_F Y)$.

3. In one direction, if a is in $(X \rightarrow_S Y)$, then $\varepsilon(a)$ is in $(X \rightarrow_F Y)$, by Part 1. But $\varepsilon(a) \leq a$ as $\mathcal{P}$ is a syntactical model of βη-reduction; therefore as $(X \rightarrow_F Y)$ is upper-closed, a is in it. The other direction is part of the assumption that $\mathcal{T}y$ is a type interpretation. ■

In case $\mathcal{P}$ is a syntactical λ-model the two arrow conditions are particular cases of those of Mitchell [28]. He did not consider types as necessarily being sets. Rather, in his "inference model" of polymorphic types, a set of types was given together with an arrow function operating on the types, and an assignment of sets to them. Here an account of types entirely in terms of sets is preferred; while this makes no difference as regards simple types, it does for both polymorphic and intersection types. Previously to Mitchell's work, other authors had considered two particular interpretations, the simple semantics and the F-semantics. The simple semantics consists of all subsets of P with $X \rightarrow_S Y$ as the arrow operation; the F-semantics has instead $X \rightarrow_F Y$ as the arrow operation; that these are available follows from Lemma 3.2.

Unfortunately these semantics do not seem to be available for models of β-reduction in general. The problem is that it does not seem to be true in general that $X \rightarrow_F Y$ is a subset of $X \rightarrow_S Y$, and that is needed for the proof of soundness below. So it is possible that taking all upper closed subsets for Ty and taking $\rightarrow_S$, say, as the arrow operation will not yield a type interpretation. We will obtain a completeness result for both simple interpretations and F-interpretations in which not all subsets of P are in Ty; here Lemma 3.3 will prove useful as it shows that for syntactical models of βη-reduction, all notions of type interpretation coincide.

It is interesting to contrast this situation with, on the one hand, the case of completeness of (EQβ), and, on the other hand, with that of completeness of polymorphic typing with (EQβ). In the former case, completeness holds for interpretations where all subsets are allowed as types; in the latter, while there are interpretations with all sets allowed as types, the class of such models fails to be complete (for trivial reasons).

In previous work (see [18,28] in particular) an extended notion of type was considered, where types were taken as equivalence relations over subsets or, which amounts to the same thing, as partial equivalence relations (the symmetric transitive relations). To formulate such a notion here, one would expect a relation between the order on the models of β-reduction and the partial equivalence relations (just as we ask that, as sets, types be upper closed). For lack of such a condition we do not pursue this notion further. Finally, a categorical setting has been claimed for models of reduction. One wonders what the corresponding notion of type would be.

Given a type interpretation, a valuation of the type variables, or type environment, is a function, V, assigning elements of Ty to type variables. Any such can be extended to all type expressions by putting $V(\alpha \rightarrow \beta) = V(\alpha) \rightarrow V(\beta)$. Then a statement $M{:}\alpha$ is *satisfied* by $\rho, V$ if $[\![ M ]\!](\rho)$ is in $V(\alpha)$. We write $\Gamma \models M{:}\alpha$ to mean that whenever any $\rho, V$ satisfy every statement in Γ, then they also satisfy $M{:}\alpha$.

**Soundness Theorem 2** If $\Gamma \vdash_F M{:}\alpha$ then $\Gamma \models M{:}\alpha$.

**Proof** The proof is by induction on the size of the proof of $\Gamma \vdash M{:}\alpha$. Choose $\rho, V$ and assume they satisfy every statement in Γ. The case where x is a variable is trivial, and the case where it is an application is handled by the first arrow condition. If M has the form $\lambda x.N$ then $\alpha$ has have the form $\beta \to \beta'$, and there is a shorter proof of $\Gamma, x{:}\beta \vdash N{:}\beta'$. Now we can apply the second arrow condition. Choose a an element of $V(\beta)$. Then $\rho(x := a), V$ satisfy every statement in $\Gamma, x{:}\beta$ and so by the induction hypothesis, they satisfy the statement $N{:}\beta'$, which is to say that $[\![ N ]\!](\rho(x := a))$ is in $V(\beta')$. So by the second arrow condition we can conclude that $[\![ \lambda x.N ]\!](\rho)$ is in $V(\beta) \to V(\beta')$, which is just that $\rho, V$ satisfy the statement $M{:}\alpha$, as required. ■

The requirement that types be upper closed implies that the following rule is sound for type interpretations:

$$\text{(LEQ)} \quad \frac{\Gamma \vdash M{:}\alpha \;,\; M \leq N}{\Gamma \vdash N{:}\alpha}$$

(where $M \leq N$ is to be interpreted as $[\![M]\!] \leq [\![N]\!]$ in a model of $\beta$-reduction.) This rule can be seen as our replacement for the equality rule

$$\text{(EQ)} \quad \frac{\Gamma \vdash M{:}\alpha \;,\; M{=}N}{\Gamma \vdash N{:}\alpha}$$

which we do not want for type inference systems. The strength of the rule (LEQ) depends on the strength of the formal system for judgements of the form $M \leq N$; for example, with the above system for $\beta$-reduction, the Subject Reduction Theorem for $\beta$-reduction is equivalent to the admissibility of the rule. Since the proof of soundness does not depend on the assumption that types are upper closed, we could drop the assumption, enabling us to consider types as equivalence relations along the lines of Mitchell [28]. However that seems against the spirit of our approach, one is anyhow left with the problem of finding a notion of types as equivalence relations that validates (LEQ), and, finally, the work below on simple type interpretations for the polymorphic type discipline does make use of the assumption of upper closure.

**The term model** We are going to give two proofs of completeness. For the first we begin by describing a type interpretation for the term model of $\beta$-reduction $\mathcal{R}_\beta$ given above, relative to a given finite basis $\Gamma$, and term A. Let $\mathcal{B}$ be a basis extending $\Gamma$, such that the only type statements in $\mathcal{B}$ with subject a free variable of A are those in $\Gamma$, and such that for each type expression $\alpha$ there are infinitely many statements in $\mathcal{B}$ of the form $x{:}\alpha$. Write $\mathcal{B} \vdash_F M{:}\alpha$ to mean that $\Gamma' \vdash_F M{:}\alpha$, for some finite subset $\Gamma'$ of $\mathcal{B}$. Define:

$$X_\alpha = \{ [M] \mid \mathcal{B} \vdash_F M{:}\alpha \}$$

(by the Subject Reduction Theorem this is a good definition),
and then put:

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression} \}$$

(Note that $X_\alpha = X_{\alpha'}$ implies that $\alpha = \alpha'$. For, take a variable x with $x{:}\alpha$ in $\mathcal{B}$; then [x] is in $X_\alpha$; then [x] is in $X_{\alpha'}$ and so $\mathcal{B} \vdash_F x{:}\alpha'$ and so $x{:}\alpha'$ is in $\mathcal{B}$

and so $\alpha = \alpha'$, as required.) Now one can unambiguously define the arrow operation by:

$$X_\alpha \to X_\beta = X_{\alpha\to\beta}$$

which concludes our description of the type interpretation.

To see that the first arrow condition holds, suppose that [M] is in $X_{\alpha\to\beta}$ and [N] is in $X_\alpha$. Then $\mathcal{B} \vdash_F M{:}\alpha\to\beta$ and also $\mathcal{B} \vdash_F N{:}\alpha$. It then follows that $\Gamma' \vdash_F M{:}\alpha\to\beta$ and $\Gamma'' \vdash_F N{:}\alpha$, for suitable $\Gamma'$, $\Gamma''$. By the Weakening rule, we have $\Gamma' \cup \Gamma'' \vdash_F M{:}\alpha\to\beta$ and $\Gamma' \cup \Gamma'' \vdash_F N{:}\alpha$. Hence by the $(\to E)$ rule $\Gamma' \cup \Gamma'' \vdash_F MN{:}\beta$, showing that [MN] is in $X_\beta$, as required.

To see that the second arrow condition holds, suppose $[\![M]\!](\rho(x := a))$ is in $X_\beta$ whenever a is in $X_\alpha$. Suppose that $\vec{z}$ is a list of the free variables of M other than x, and $[\vec{L}]$ is $\rho(\vec{z})$. Choose $y{:}\alpha$ in $\mathcal{B}$ where y is not free in M or any term in $\vec{L}$, and put a = [y]. It follows that $\mathcal{B} \vdash_F [\vec{L}/\vec{z}, y/x]M{:}\beta$. Therefore it follows by $(\to I)$ that $\mathcal{B} \vdash_F \lambda y.\ [\vec{L}/\vec{z}, y/x]M{:}\alpha\to\beta$, and we have that $\mathcal{B} \vdash_F [\vec{L}/\vec{z}]\lambda y.[y/x]M{:}\alpha\to\beta$. So as y is not free in M we get that $[\![\lambda x.M]\!](\rho) \in X_\alpha \to X_\beta$ as required.

A particular valuation $V_0$ of the type variables will be useful. It is defined by: $V_0(t) = X_t$. Clearly, $V_0(\alpha) = X_\alpha$, for any type expression $\alpha$.

**Completeness Theorem 3** If $\Gamma \models M{:}\alpha$ holds in every model of $\beta$-reduction and every type interpretation, then $\Gamma \vdash_F M{:}\alpha$.

**Proof** Suppose that in every model of $\beta$-reduction and type interpretation, $\Gamma \models M{:}\alpha$. Choose the model of $\beta$-reduction to be the term model and choose the type interpretation as above, with this $\Gamma$ and with A taken to be M. Then $\rho_0, V_0$ satisfies every statement in $\Gamma$ and so it also satisfies $M{:}\alpha$, and so we have [M] in $X_\alpha$. Thus $\Gamma' \vdash_F M{:}\alpha$ for some $\Gamma' \subset \mathcal{B}$. By applying the Strengthening rule followed, if needed, by Weakening we finally obtain that $\Gamma \vdash_F M{:}\alpha$ as required. ■

The type interpretation used for the completeness proof is simple. To prove this one just has to show that $X_\alpha \to_S X_\beta$ is a subset of $X_\alpha \to X_\beta$. To this end, suppose that for all a in $X_\alpha$ that $[M]\cdot a$ is in $X_\beta$. Let x be a variable such that the statement $x{:}\alpha$ is in $\mathcal{B}$, and put a = [x]. Then we get that $\mathcal{B} \vdash_F Mx{:}\beta$. Now it may easily be proved that if $\Gamma \vdash_F NL{:}\beta$, then there is an $\alpha'$ such that $\Gamma \vdash_F N{:}\alpha'\to\beta$ and $\Gamma \vdash_F L{:}\alpha'$. Applying this to the above case we see that $\Gamma \vdash_F x{:}\alpha'$, and so the statement $x{:}\alpha'$ is in $\Gamma$, and so as $x{:}\alpha$ is too, $\alpha' = \alpha$, and so it is true that $\Gamma \vdash_F M{:}\alpha\to\beta$, as required.

**$\beta\eta$-reduction** By using $\mathcal{R}_{\beta\eta}$ instead of $\mathcal{R}_\beta$, and working with the analogous simple type interpretation one proves a somewhat stronger completeness theorem, where the models are of $\beta\eta$-reduction, and (by Lemma 3.3) the type interpretations are simple, or F-interpretations (or even both!).

**Completeness Theorem 4** If $\Gamma \models M{:}\alpha$ holds in every syntactical model of $\beta\eta$-reduction and every simple type interpretation (or F-interpretation), then $\Gamma \vdash_F M{:}\alpha$.

**A type-expression model** The collection of type expressions actually forms a model of $\beta\eta$-reduction which can be used to show the completeness of the system $\lambda\rightarrow$ with respect to both simple interpretations and F-interpretations. Take P to be $\mathcal{P}$(Type) partially ordered by subset, where Type is the set of all type expressions.

Application is defined by:

$$a\cdot b = \{ \beta \mid \text{for some } \alpha \text{ in } b, \alpha\rightarrow\beta \text{ is in } a \}$$

And $[\![ \cdot ]\!](\cdot)$ is defined by the inductive clauses:

$$[\![ x ]\!](\rho) = \rho(x)$$

$$[\![ MN ]\!](\rho) = [\![ M ]\!](\rho)\cdot [\![ N ]\!](\rho)$$

$$[\![ \lambda x.M ]\!](\rho)=\{ \alpha\rightarrow\beta \mid \beta \in [\![ M ]\!](\rho(x := \alpha)) \}$$

Here and below we confuse $\{\alpha\}$ with $\alpha$.

This defines an ordered lambda interpretation. The only not completely obvious point is that the interpretation respects $\alpha$-equivalence; we omit the verification. For condition (v), calculate that:

$$\begin{aligned} [\![ \lambda x.M ]\!](\rho)\cdot a &= \{ \alpha\rightarrow\beta \mid \beta \in [\![ M ]\!](\rho(x := \alpha)) \}\cdot a \\ &= \{ \beta \mid \text{for some } \alpha \text{ in } a, \beta \in [\![ M ]\!](\rho(x := \alpha)) \} \\ &\subset [\![ M ]\!](\rho(x := a)) \end{aligned}$$

(by the monotonicity of $[\![ M ]\!](\rho)$ in $\rho$). To see that the type-expression model is a syntactical model of $\beta\eta$-reduction, let us calculate that for $\lambda x.Mx$ with x not free in M:

$$\begin{aligned} [\![ \lambda x.Mx ]\!](\rho) &= \{ \alpha\rightarrow\beta \mid \beta \in [\![ Mx ]\!](\rho(x := \alpha)) \} \\ &= \{ \alpha\rightarrow\beta \mid \exists\, \alpha' \in \{\alpha\}.\ \alpha'\rightarrow\beta \in [\![M]\!](\rho(x := \alpha)) \} \\ &= \{ \alpha\rightarrow\beta \mid \alpha\rightarrow\beta \in [\![M]\!](\rho(x := \alpha)) \} \\ &\subset [\![M]\!](\rho) \text{ (as x is not free in M)} \end{aligned}$$

According to a claim made above this interpretation can be viewed as a structure $X^X \rightarrow X \rightarrow X^X$ in some cartesian closed order-enriched category. It is more pleasing, however, to do this directly in a naturally available category, and here we take the category of complete partial orders and continuous maps. The partial order P is such a complete partial order - indeed it is an algebraic complete lattice. Application is continuous and yields F: $P\rightarrow P^P$ as F(a)(b) = a.b; the continuous function G: $P^P \rightarrow P$ is defined by:

$$G(f) = \{ \alpha\rightarrow\beta \mid \beta \in f(\alpha) \}$$

and one can show that:

$$[\![ \lambda x.M ]\!](\rho)=G(\lambda\alpha{:}P.\ [\![ M ]\!](\rho(x := \alpha)))$$

(making use of the typed $\lambda$-calculus).

For the type interpretation define:

$$X_\alpha = \{ a \mid \alpha \in a \}$$

and then put:

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression} \}$$

Noting that $X_\alpha$ determines $\alpha$ since $\{\alpha\}$ is the only singleton in $X_\alpha$, we can define the arrow operation by:

$$X_\alpha\rightarrow X_\beta = X_{\alpha\rightarrow\beta}$$

We have to check that the arrow conditions hold. For the first, suppose that a is in $X_{\alpha\to\beta}$ and b is in $X_\alpha$. Then $\alpha\to\beta$ is in a, $\alpha$ is in b and so $\beta$ is in $a\cdot b$ and we see that a.b is in $X_\beta$. For the second suppose that whenever $a \in X_\alpha$ then $[\![ M ]\!](\rho(x := a)) \in X_\beta$. Then $\beta \in [\![ M ]\!](\rho(x := \alpha))$ and so $\alpha\to\beta \in [\![ \lambda x.M ]\!](\rho)$ showing that $[\![ \lambda x.M ]\!](\rho) \in X_{\alpha\to\beta}$ as required. Note that by Lemma 3.3 this is both a simple type interpretation and an F-interpretation.

To show completeness we will need an appropriate environment and type environment. For the environment, for any finite basis $\Gamma$ define an environment by:

$$\hat{\Gamma}(x) = \{ \alpha \mid x{:}\alpha \in \Gamma \}$$

The next lemma is – roughly speaking – to the effect that the denotation of a term is the set of types that can be inferred of it. This is characteristic of type expression models.

**Lemma 4** Let M be a term. Then $[\![M]\!](\hat{\Gamma}) = \{ \alpha \mid \Gamma \vdash_F M{:}\alpha \}$

**Proof** The proof is by induction on M. There are three cases. First, if M is a variable x, then:

$$[\![M]\!](\hat{\Gamma}) = \{ \alpha \mid x{:}\alpha \in \Gamma \}$$
$$= \{ \alpha \mid \Gamma \vdash_F x{:}\alpha \}$$

Second, if M is an application NL, then

$$[\![NL]\!](\hat{\Gamma}) = [\![N]\!](\hat{\Gamma}) \cdot [\![L]\!](\hat{\Gamma})$$
$$= \{ \gamma \mid \Gamma \vdash_F N{:}\gamma \}\cdot\{ \alpha \mid \Gamma \vdash_F L{:}\alpha \}$$

(by the induction hypothesis)

$$= \{ \alpha\to\beta \mid \Gamma \vdash_F N{:}\alpha\to\beta \}\cdot\{ \alpha \mid \Gamma \vdash_F L{:}\alpha \}$$

(by the definition of application)

$$= \{ \beta \mid \Gamma \vdash_F NL{:}\beta \}$$

Third, if M is an abstraction $\lambda x.N$, then

$$[\![\lambda x.N]\!](\hat{\Gamma}) = \{ \alpha\to\beta \mid \beta \in [\![N]\!](\hat{\Gamma}(x := \alpha)) \}$$
$$= \{ \alpha\to\beta \mid \Gamma, x{:}\alpha \vdash_F N{:}\beta \}$$

(by induction hypothesis since $\hat{\Gamma}(x := \alpha)$ is $(\Gamma,x{:}\alpha)$^)

$$= \{ \alpha\to\beta \mid \Gamma \vdash_F \lambda x.N{:}\alpha\to\beta \}$$
$$= \{ \gamma \mid \Gamma \vdash_F \lambda x.N{:}\gamma \}. \blacksquare$$

This lemma corresponds, roughly, to part 2 of the Key Theorem in [26]. To make this remark precise requires developing the combinatory logic version of the present work; this is left as a (very pleasant!) exercise for the reader.

Now the type environment is defined by $V_1(t) = X_t$, and one has that $V_1(\alpha) = X_\alpha$. For the second proof of Completeness Theorem 4, assume that $\Gamma \models M{:}\alpha$. Note that $\hat{\Gamma}$, $V_1$ satisfies every type statement $x{:}\alpha$ in $\Gamma$ as $\{\alpha\} \in X_\alpha$. So $\hat{\Gamma}$, $V_1$ satisfies $M{:}\alpha$, which means that $[\![M]\!](\hat{\Gamma})$ is in $X_\alpha$, and so $\Gamma \vdash_F M{:}\alpha$ by Lemma 4.

Since this proof of the Completeness Theorem makes no use of the Subject Reduction Theorem, and since rule $(\twoheadrightarrow_{\beta\eta})$ is sound for any model of $\beta\eta$-reduction and type interpretation, we now have a semantic proof of the Subject Reduction Theorem for $\beta\eta$-reduction. In fact a direct proof for the system $\lambda\!\rightarrow$ is very simple, but perhaps the technique would have applications in a more complex setting.

## 5 Polymorphic Type Inference

A system for polymorphic type inference is obtained by adding universal quantification $\forall t.\alpha$ to type expressions (see [3,13,15,16,28,34,36] for information on this system and further references). We do not distinguish $\alpha$-equivalent type expressions – those which differ only in the names of bound variables. There are also evident notions of $FTV(\alpha)$, the free type variables of $\alpha$, substitution $[\sigma/t]\alpha$ and multiple substitution $[\vec{\sigma}/\vec{t}]\alpha$. A natural *basic system* for inferring polymorphic types was given by Mitchell [28]. It is obtained by adding to $\lambda\rightarrow$ the following two rules:

$$(\forall I) \qquad \frac{\Gamma \vdash M:\alpha}{\Gamma \vdash M:\forall t.\alpha}$$

(provided that t does not appear free in any type in $\Gamma$)

$$(\forall E) \qquad \frac{\Gamma \vdash M:\forall t.\alpha}{\Gamma \vdash M:[\beta/t]\alpha}$$

The relation of *generic instantiation*, $\alpha \subset_g \beta$, is defined to hold iff $\alpha$ and $\beta$ have the respective forms $\forall \vec{s}.\delta$ and $\forall \vec{r}.[\vec{\sigma}/\vec{s}]\delta$, where no variable in $\vec{r}$ is free in $\forall \vec{s}.\delta$. In [28] it is shown that $\alpha \subset_g \beta$ holds iff $x:\alpha \vdash x:\beta$ is provable, and that generic instantiation is decidable. The following rule is admissible (as is easy to show using the two rules for universal quantification):

$$(\text{SubTypes}_g) \qquad \frac{\Gamma \vdash M:\alpha \,,\ \alpha \subset_g \beta}{\Gamma \vdash M:\beta}$$

The Subject Reduction Theorem for $\beta$-reduction and Weakening and Strengthening continue to hold. However the Subject Reduction Theorem does not hold for $\beta\eta$-reduction [28]; for example, the sequent $y:\forall t.\alpha\rightarrow\beta \vdash \lambda x.yx:\alpha\rightarrow\forall t.\beta$, is provable, but the sequent $y:\forall t.\alpha\rightarrow\beta \vdash y:\alpha\rightarrow\forall t.\beta$ is not, as $\alpha\rightarrow\forall t.\beta$ is not a generic instance of $\forall t.\alpha\rightarrow\beta$.

One might well expect that the basic system would play a role analogous to that of $\lambda\rightarrow$ but a new phenomenon enters into play, which concerns an equality rule, at the level of type expressions rather than expressions:

$$(\text{EQTypes}) \qquad \frac{\Gamma \vdash M:\alpha \,,\ \alpha=\beta}{\Gamma \vdash M:\beta}$$

In the case of $\lambda\rightarrow$ there were no non-trivial valid equalities between type expressions; here there are. If types are to be interpreted as sets, and universal quantification is to be interpreted as an intersection, then - for example - $\forall t.\alpha$ and $\alpha$ will receive the same interpretation (where t does not occur in $\alpha$). Such quantifications are termed *vacuous*. We consider an extension of the basic system with the rule (EQTypes) together with the following rules and axioms for the equality of types:

$$\alpha = \alpha$$

$$\frac{\alpha = \beta}{\beta = \alpha}$$

$$\frac{\alpha = \beta\ ,\ \beta = \gamma}{\alpha = \gamma}$$

$$\frac{\alpha = \beta\ ,\ \alpha' = \beta'}{\alpha \to \alpha' = \beta \to \beta'}$$

$$\frac{\alpha = \beta}{\forall t.\alpha\ =\ \forall t.\beta}$$

$$\forall t.\alpha\ =\ \alpha\ (t \notin\ FTV(\alpha))$$

$$\forall s.\forall t.\alpha = \forall t.\forall s.\alpha$$

Let us call this system "$\lambda{\to}_{\forall}$" and for the notion of *F∀-deducibility*, the relation $\Gamma \vdash_{F\forall} M{:}\alpha$ is taken to be that $\Gamma \vdash M{:}\alpha$ is provable using this system of rules. Let us also define $\alpha =_{F\forall} \beta$ to hold just when $\alpha = \beta$ is provable. The system yields different typings than the basic system; for example, if $t \notin FTV(\alpha)$, then $\lambda x.x$ has type $((\forall t.\alpha) \to \beta) \to (\alpha \to \beta)$ in this system, but not in the basic system. Note that it is the fourth of the rules, the congruence rule for arrow, that makes the difference; if omitted, the resulting system is equivalent to the basic system.

Both Weakening and Strengthening hold for $\lambda{\to}_{\forall}$ by straightforward inductive proofs. To show that the Subject Reduction Theorem for $\beta$-reduction holds we relate to the basic system of Mitchell. For any type expression $\alpha$, let $\alpha^*$ be the type expression obtained from $\alpha$ by first removing all vacuous quantifications, and then permuting iterated quantifications so that in every subterm of the form $\forall s.\forall t.\beta$, the first occurrence of s in $\beta$ is to the left of the first occurrence of t in $\beta$. This provides a normal form for the above system for proving equality of type expressions in that first, for any type expression $\alpha$, $\alpha =_{F\forall} \alpha^*$ and second for any type expressions $\alpha$ and $\beta$, it holds that $\alpha =_{F\forall} \beta$ iff $\alpha^*$ and $\beta^*$ are $\alpha$-equivalent. In the following proposition the $(\cdot)^*$ transformation is applied also to bases; this is taken to mean that it is applied to the type expressions occurring in the bases.

**Proposition 3** The sequent $\Gamma \vdash M{:}\alpha$ is provable in $\lambda_{\rightarrow\forall}$ iff it is provable in the basic system that $\Gamma^* \vdash M{:}\alpha^*$.
**Proof** The implication from left to right is proved by a straightforward induction which we omit. The converse follows from the fact that provability in $\lambda_{\rightarrow\forall}$ is invariant under substitution of provably equal type expressions, whether in the assumptions (use (EQTypes) "at the top of the proof"), or in the conclusion (apply (EQTypes) directly). ■

The relation of *modified generic instantiation*, $\alpha \subset_{mg} \beta$, is defined to hold iff $\alpha^* \subset_g \beta^*$. From the above remarks on generic instantiation and Proposition 3, one sees: that $\alpha \subset_{mg} \beta$ holds iff $x{:}\alpha \vdash_{F\forall} x{:}\beta$; that modified generic instantiation is decidable and that in the system $\lambda_{\rightarrow\forall}$ the following rule is admissible:

$$(\text{SubTypes}_{mg}) \qquad \frac{\Gamma \vdash M{:}\alpha\ ,\ \alpha \subset_{mg} \beta}{\Gamma \vdash M{:}\beta}$$

Similarly, one sees that the Subject Reduction Theorem for β-reduction holds for the system, but that Subject Reduction does not hold for βη-reduction. Finally if $\alpha \subset_{mg} \beta \subset_{mg} \alpha$, then $\alpha$ and $\beta$ are α-equivalent; this fails for generic instantiation - for example, take $\forall t.\alpha$ and $\alpha$, with t not free in $\alpha$.

A *type interpretation* is now a triple

$$\mathcal{Ty} = \langle Ty, \rightarrow, [\![\cdot]\!](\cdot) \rangle$$

where $\langle Ty, \rightarrow \rangle$ is an interpretation as above and $[\![\cdot]\!](\cdot)$ is a mapping from type expressions and valuations of the type variables to types such that:

(i) $[\![ t ]\!](V) = V(t)$
(ii) $[\![ \alpha\rightarrow\beta ]\!](V) = [\![ \alpha ]\!](V) \rightarrow [\![ \beta ]\!](V)$
(iii) $[\![ \forall t.\alpha ]\!](V) = \bigcap \{ [\![\alpha]\!](V(t := X)) \mid X \in Ty \}$

Note that we have to take these as conditions rather than as an inductive definition, for in the third case it is needed that the intersection is indeed in Ty. This definition can be viewed as a special case of the definition in [28] where types represent sets of elements of the model rather than necessarily being such sets. We omit the verification of this claim, which would involve embedding $\mathcal{Ty}$ in a model of the typed λ-calculus. (See [13, 35] for other applications of Mitchell's idea of inference models.)

**Proposition 4** Type interpretations have the following properties:
(iv) $[\![ \forall t.\alpha ]\!](V) = [\![ \forall t'.[t'/t]\alpha ]\!](V)$ ($t'$ is not free in $\forall t.\alpha$)
(v) if $V \restriction FTV(\alpha) = V' \restriction FTV(\alpha)$ then $[\![ \alpha ]\!](V) = [\![ \alpha ]\!](V')$
(vi) if for all X in Ty, $[\![ \alpha ]\!](V(t := X)) = [\![ \beta ]\!](V(t := X))$ then $[\![ \forall t.\alpha ]\!](V) = [\![ \forall t.\beta ]\!](V)$
(Sub) $[\![ [\beta/t]\alpha ]\!](V) = [\![ \alpha ]\!](V(t := [\![ \beta ]\!](V) ))$

**Proof** The proof is straightforward, and is omitted. ■

As before, a statement M:$\alpha$ is *satisfied* by $\rho$,V if $[\![ M ]\!](\rho)$ is an element of $[\![ \alpha ]\!](V)$, and what $\Gamma \models$ M:$\alpha$ means is defined accordingly.
**Soundness Theorem 5** If $\Gamma \vdash_{F\forall}$ M:$\alpha$ then $\Gamma \models$ M:$\alpha$.
**Proof** The proof is by induction on the size of the proof that $\Gamma \vdash$ M:$\alpha$. Choose $\rho$,V and assume they satisfy every statement in $\Gamma$. The rules of system $\lambda\!\to$ are dealt with as in the proof of Theorem 2. In the case of ($\forall$I), $\alpha$ has the form $\forall$t.$\beta$, and there is a shorter proof of $\Gamma \vdash$ M:$\beta$, where t does not occur free in any type in $\Gamma$. Therefore by (v), for any type X it is the case that $\rho$,V(t := X) satisfy every statement in $\Gamma$ and so satisfy M:$\beta$. Thus $[\![ M ]\!](\rho)$ is in $[\![\beta]\!](V(t := X))$; so by (iii), $\rho$,V satisfy M:$\alpha$.

In the case of ($\forall$E), we have that for some $\beta$,$\sigma$ the type expression $\alpha$ has the form $[\sigma/t]\beta$ and there is a shorter proof of $\Gamma \vdash$ M:$\forall$t.$\beta$. So by induction hypothesis, $[\![ M ]\!](\rho) \in [\![ \forall t.\beta ]\!](V)$. But

$$[\![\forall t.\beta]\!](V) \subset [\![\beta]\!](V(t := [\![ \sigma ]\!](V))) \quad \text{(by (iii))}$$
$$= [\![ [\sigma/t]\beta ]\!](V) \text{ (by (Sub))}$$

showing that $\rho$,V satisfy M:$[\sigma/t]\beta$. Finally the soundness of the rule (EQTypes) follows from the easily proved fact that, if $\alpha =_{F\forall} \beta$ then it follows that $[\![ \alpha ]\!](V) = [\![ \beta ]\!](V)$. ■

**The term model** A completeness proof can be based on the term model of $\beta$-reduction $\mathcal{R}_\beta$. For a type interpretation, given any finite basis $\Gamma$ and term A define a basis $\mathcal{B}$ as above, and with the analogous understanding of $\mathcal{B} \vdash_{F\forall}$ M:$\alpha$. Define:

$$X_\alpha = \{ [M] \mid \mathcal{B} \vdash_{F\forall} M{:}\alpha \}$$

as before, again making use of the Subject Reduction Theorem and take:

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression } \}$$

To see that $X_\alpha$ determines $\alpha$, suppose that $X_\alpha \subset X_\beta$. Taking a statement of the form x:$\alpha$ in $\mathcal{B}$, we get that $\mathcal{B} \vdash_{F\forall}$ x:$\beta$, and hence, by Strengthening, that x:$\alpha \vdash_{F\forall}$ x:$\beta$, and so $\alpha \subset_{mg} \beta$. Conversely, if $\alpha \subset_{mg} \beta$ then $X_\alpha \subset X_\beta$. For if $\mathcal{B} \vdash_{F\forall}$ M:$\alpha$ then $\mathcal{B} \vdash_{F\forall}$ M:$\beta$ by (SubTypes$_{mg}$). Therefore $X_\alpha \subset X_\beta$ holds iff $\alpha \subset_{mg} \beta$, and it follows that $X_\alpha = X_\beta$ iff $\alpha$ and $\beta$ are $\alpha$-equivalent.

Now the arrow operation can be defined by:

$$X_\alpha \to X_\beta = X_{\alpha\to\beta}$$

The arrow conditions hold, with the same proof as before, making use of Weakening. It remains to define $[\![\alpha]\!](V)$, and we put:

$$[\![\alpha]\!](V) = X_{[\vec{\sigma}/\vec{s}]\alpha}$$

where $\vec{s}$ is a list of the free type variables of $\alpha$ and $X_{\vec{\sigma}} = V(\vec{s})$. We have to verify conditions (i),(ii) and (iii). The first is trivial. For the second, we may calculate that:

$$[\![ \alpha\to\beta ]\!](V) = X_{[\vec{\sigma}/\vec{s}](\alpha\to\beta)}$$

(where $\vec{s}$ is a list of the free type variables of $\alpha\to\beta$ and $X_{\vec{\sigma}} = V(\vec{s})$)

$$= (X_{[\vec{\sigma}/\vec{s}]\alpha}) \to (X_{[\vec{\sigma}/\vec{s}]\beta})$$
$$= [\![ \alpha ]\!](V) \to [\![ \beta ]\!](V)$$

The verification of the third condition is split into two inclusions. In one direction,

$$[\![ \forall t.\alpha ]\!](V) = X_{[\vec{\sigma}/\vec{s}]\forall t.\alpha}$$

(where $\vec{s}$ is a list of the free type variables of $\forall t.\alpha$ and $X_{\vec{\sigma}} = V(\vec{s})$, and where we may assume that t differs from all the variables in $\vec{s}$ and does not appear free in any type expression in $\vec{\sigma}$)

$$= \{ [M] \mid \mathcal{B} \vdash_{F\forall} M\colon \forall t.[\vec{\sigma}/\vec{s}]\alpha \}$$
$$\subset \{ [M] \mid \mathcal{B} \vdash_{F\forall} M\colon [\beta/t][\vec{\sigma}/\vec{s}]\alpha \} \text{ by } (\forall E)$$
$$= \{ [M] \mid \mathcal{B} \vdash_{F\forall} M\colon [\beta/t, \vec{\sigma}/\vec{s}]\alpha \}$$
$$= X_{[\beta/t,\vec{\sigma}/\vec{s}]\alpha}$$
$$= [\![ \alpha ]\!](V(t := [\![ \beta ]\!](V)))$$

In the other direction, suppose [M] is in $[\![\alpha]\!](V(t := X))$ for all types X. Let $\vec{s}$ be a list of the free type variables of $\forall t.\alpha$, and suppose $X_{\vec{\sigma}} = V(\vec{s})$. Choose $t'$ different from all the variables in $\vec{s}$, and not free either in $\alpha$, or in any type expression in $\vec{\sigma}$, or in any $\beta$ where $x{:}\beta$ is a statement in $\mathcal{B}$ whose subject x is free in M. Take X to be $X_{t'}$. Then

$$\Gamma' \vdash_{F\forall} M\colon[t'/t, \vec{\sigma}/\vec{s}]\alpha$$

for some $\Gamma'$ a subset of $\mathcal{B}$. Applying Strengthening as necessary, we can assume that the subject of any statement in $\Gamma'$ is a free variable of M. Hence by $(\forall I)$

$$\Gamma' \vdash_{F\forall} M\colon \forall t'.[t'/t, \vec{\sigma}/\vec{s}]\alpha.$$

But $\forall t'.[t'/t, \vec{\sigma}/\vec{s}]\alpha$ is $\alpha$-equivalent to $[\vec{\sigma}/\vec{s}]\forall t.\alpha$, and so M is an element of $[\![ \forall t.\alpha ]\!](V)$, as required.

Finally, with $\rho_0$ and $V_0$ defined as before, one obtains:

**Completeness Theorem 6** If $\Gamma \models M{:}\alpha$ holds in every model of $\beta$-reduction and every type interpretation, then $\Gamma \vdash_{F\forall} M{:}\alpha$.

**Proof** The proof is just like the term model one for the system $\lambda\rightarrow$. ■

**A type-expression model** There is also a completeness proof *via* a type-expression model of $\beta$-reduction. Say that a *$\forall$-filter* over the set of polymorphic type expressions is a subset F such that:

(1) if $\alpha \in F$ and $\alpha \subset_{mg} \beta$ then $\beta \in F$

(2) if $[\beta/t]\alpha \in F$ for all type expressions $\beta$, then $\forall t.\alpha \in F$

Here the universal quantifications are treated as the (infinite) conjunction of their substitution instances; this idea appears already in a filter definition discussed in [21].

For any set F of type expressions let $F\uparrow_\forall$ be the least $\forall$-filter containing F. For singletons, we have that $\{\alpha\}\uparrow_\forall = \{\beta \mid \alpha \subset_{mg} \beta\}$. Clearly this is the least set satisfying (1); to see that it satisfies (2), suppose that $\alpha \subset_{mg} [\gamma/t]\beta$ for all type expressions $\gamma$. Then, in particular $\alpha \subset_{mg} [s/t]\beta$, where s is a type variable not occurring free in $\alpha$ or $\forall t.\beta$, and so $\alpha \subset_{mg} \forall s.[s/t]\beta$ (as is easily shown) which is $\alpha$-equivalent to $\forall t.\beta$. Finally, sets which have the form $\{ \alpha \mid \Gamma \vdash_{F\forall} M{:}\alpha \}$ are $\forall$-filters. That (1) holds follows by using (SubTypes$_{mg}$). For (2), suppose that $[\beta/t]\alpha$ is in $\{ \alpha \mid \Gamma \vdash_{F\forall} M{:}\alpha \}$ for all type expressions $\beta$, take $\beta$ to be a type variable not appearing free in any type expression in $\Gamma$ or in $\forall t.\alpha$, and apply the rule ($\forall$I).

Now for the model of $\beta$-reduction we take P to be the set of $\forall$-filters, ordered by subset. As such, it is a complete lattice where meet is intersection. Application is defined by:

$$a{\cdot}b = \{\ \beta \mid \text{for some } \alpha \text{ in } b,\ \alpha\to\beta \text{ is in } a\ \}{\uparrow_\forall}$$

And $[\![\ \cdot\ ]\!](\cdot)$ is defined by the inductive clauses

$$[\![\ x\ ]\!](\rho) = \rho(x)$$

$$[\![\ MN\ ]\!](\rho) = [\![\ M\ ]\!](\rho)\cdot [\![\ N\ ]\!](\rho)$$

$$[\![\ \lambda x.M\ ]\!](\rho)=\{\ \alpha\to\beta \mid \beta\in [\![\ M\ ]\!](\rho(x := \alpha{\uparrow_\forall}))\ \}{\uparrow_\forall}$$

As before, this defines an ordered lambda interpretation with the only not completely obvious point being that the interpretation respects $\alpha$-equivalence; we again omit the verification.

The clause for abstraction can be made more explicit:

**Lemma 5**

$[\![\ \lambda x.M\ ]\!](\rho) = \{\forall\vec{s}.(\alpha\to\beta) \mid \text{for all } \vec{\sigma}, [\vec{\sigma}/\vec{s}]\beta\in [\![\ M\ ]\!](\rho(x := [\vec{\sigma}/\vec{s}]\alpha{\uparrow_\forall}))\ \}$

**Proof** To see this holds, note first that, on the one hand, the right hand side is clearly contained in the left hand side, as that satisfies (2), and on the other, it contains $\{\ \alpha\to\beta \mid \beta\in [\![\ M\ ]\!](\rho(x := \alpha{\uparrow_\forall}))\ \}$. It therefore suffices to show that the right hand side is a $\forall$-filter.

To see that (1) is satisfied, suppose $\vec{s},\alpha,\beta$, are such that for all $\vec{\sigma}$, $[\vec{\sigma}/\vec{s}]\beta$ is in $[\![\ M\ ]\!](\rho(x := [\vec{\sigma}/\vec{s}]\alpha{\uparrow_\forall}))$; suppose too that we have $\vec{\tau}$, of the same length as $\vec{s}$, and $\vec{r}$ with no variable free in $\forall\vec{s}.\alpha\to\beta$. We wish to show that $\forall\vec{r}.[\vec{\tau}/\vec{s}](\alpha\to\beta)$ is in the right hand side. To this end we choose any $\vec{\sigma}$ of the same length as $\vec{r}$ and have to show that $[\vec{\sigma}/\vec{r}]\,[\vec{\tau}/\vec{s}]\beta$ is an element of $[\![\ M\ ]\!](\rho(x := [\vec{\sigma}/\vec{r}]\ [\vec{\tau}/\vec{s}](\alpha{\uparrow_\forall})))$, which is an immediate consequence of the first supposition.

To see that (2) is satisfied suppose that for all $\beta$, $[\beta/t]\alpha$ is in the right hand side. Let $\alpha$ have the form $\forall\vec{s}.(\gamma\to\delta)$. If t occurs in $\vec{s}$, then $\alpha$ is in the right hand side (since then $\alpha = [\beta/t]\alpha$ for any $\beta$) and so $\forall t.\alpha$ is in the right hand side, as (1) is satisfied. If t does not occur in $\vec{s}$, then we take an arbitrary $\beta$ and an arbitrary $\vec{\sigma}$ of the same length as $\vec{s}$ and prove that $[\beta/t,\vec{\sigma}/\vec{s}]\delta\in [\![M]\!](\rho(x := [\beta/t,\vec{\sigma}/\vec{s}]\gamma{\uparrow_\forall})$. To this end, by assumption the right hand side contains $[\beta/t]\alpha$. But this is $\alpha$-equivalent to the type expression $\forall\vec{s}'.[\beta/t,\vec{s}'/\vec{s}](\gamma\to\delta)$, where $\vec{s}'$ is a non-repeating list of variables, all distinct from t, and not occurring free in $\beta$ or $(\gamma\to\delta)$. Therefore $[\vec{\sigma}/\vec{s}'][\beta/t,\vec{s}'/\vec{s}]\delta$ is in $[\![\ M\ ]\!](\rho(x := [\vec{\sigma}/\vec{s}'][\beta/t,\vec{s}'/\vec{s}]\gamma{\uparrow_\forall})$. But then it follows that the type expression $[\vec{\sigma}/\vec{s}'][\beta/t,\vec{s}'/\vec{s}]\delta$ is $\alpha$-equivalent to $[\vec{\sigma}/\vec{s}',\beta/t,\vec{\sigma}/\vec{s}]\delta =_\alpha [\beta/t,\vec{\sigma}/\vec{s}]\delta$, and similarly for $\gamma$. ■

One now sees that the set of functional type expressions in $[\![\ \lambda x.M\ ]\!](\rho)$ is the set $\{\ \alpha\to\beta \mid \beta\in [\![\ M\ ]\!](\rho(x := \alpha{\uparrow_\forall}))\ \}$, and so for condition (v) we have that:

$$\begin{aligned}[\![\ \lambda x.M\ ]\!](\rho)\cdot a &= \{\ \alpha\to\beta \mid \beta\in [\![\ M\ ]\!](\rho(x := \alpha{\uparrow_\forall}))\ \}\cdot a\\ &= \{\ \beta \mid \text{for some } \alpha \text{ in } a,\ \beta\in [\![\ M\ ]\!](\rho(x := \alpha{\uparrow_\forall})\ \}{\uparrow_\forall}\\ &\subset [\![\ M\ ]\!](\rho(x := a))\end{aligned}$$

using the monotonicity of $[\![\ M\ ]\!](\rho)$ in $\rho$.

The partial order P is an algebraic complete lattice. Unfortunately, however, application is not continuous. For example, set

$$a = \{(\forall t.t\to t)\to s\}{\uparrow_\forall}$$

and for all finite sets of type expressions, X, put

$$b_X = \{\alpha\to\alpha \mid \alpha\in X\}{\uparrow_\forall}$$

Then a and $b_X$ are $\forall$-filters, the set $\{b_X\}$ is directed with least upper bound $(\forall t.t\to t){\uparrow_\forall}$, but $a.b_X$ is $\varnothing$ and $a.(\forall t.t\to t){\uparrow_\forall}$ is $s{\uparrow_\forall}$. On the other hand, application is continuous in a weaker sense. Say that a subset of a partial order is *countably directed* if it contains an upper bound of any of its countable subsets and say that a function of complete partial orders is *countably continuous* if it preserves least upper bounds of countably directed sets. The category of complete partial orders and countably continuous functions can be shown to be a cartesian closed order-enriched category. Now assuming that there are countably many type variables, there are countably many type expressions. One can then show that the least upper bound of a countably directed subset of P is the union of that set, and it follows that application is countably continuous. One can define F and G by:

$$F(a)(b) = a.b$$

$$G(f) = \{\, \alpha\to\beta \mid \beta\in f(\alpha{\uparrow_\forall}) \,\}{\uparrow_\forall}$$

and one has that:

$$[\![ \lambda x.M ]\!](\rho)=G(\lambda\alpha{:}P.\ [\![ M ]\!](\rho(x := \alpha{\uparrow_\forall})))$$

The question arises whether there is a type expression model in the category of complete partial orders and continuous functions which can be made the basis of a completeness proof; it is conjectured that there is none, although, admittedly, the conjecture is not precise.

For the type interpretation define:

$$X_\alpha = \{\, a \mid \alpha\in a \,\}$$

and then put

$$Ty = \{\, X_\alpha \mid \alpha \text{ a type expression} \,\}$$

Noting that $X_\alpha \subset X_\beta$ iff $\alpha \subset_{mg} \beta$ we see that $X_\alpha$ determines $\alpha$ up to $\alpha$-equivalence and so we can define the arrow operation by

$$X_\alpha\to X_\beta = X_{\alpha\to\beta}$$

The two arrow conditions are straightforwardly verified. Finally we can define the interpretation of type expressions by the conditions (i),(ii),(iii) above, as the intersection of any collection of $\forall$-filters is itself a $\forall$-filter.

One can show that

$$X_{\forall t.\alpha} = \bigcap \{X_{[\beta/t]\alpha} \mid \beta \text{ a type expression} \}$$

and it is just here that good use is made of the infinitary condition in the definition of $\forall$-filters. The inclusion from left to right is proved using condition (1): if $a \in X_{\forall t.\alpha}$ then $\forall t.\alpha \in a$ and so $[\beta/t]\alpha \in a$ (for any $\beta$), and so $a \in X_{[\beta/t]\alpha}$ (for any $\beta$). The inclusion from right to left is proved using condition (2): if $[\beta/t]\alpha \in a$ (for any $\beta$) then $\forall t.\alpha \in a$. Using this equation, one can prove by induction that for any type expression $\alpha$ and type environment V,

$[\![\, \alpha \,]\!](V) = X_{[\vec{\sigma}/\vec{s}]\alpha}$

where $\vec{s}$ is a list of the free type variables in $\alpha$ and $X_{\vec{\sigma}}$ is $V(\vec{s})$. In the proof of completeness the type environment $V_1$ is defined by:

$V_1(t) = X_t$

Then by the above remark, $V_1(\alpha) = X_\alpha$ for any type expression $\alpha$.

As above, for any finite basis $\Gamma$ we may define an environment by:

$\hat{\Gamma}(x) = \{\, \alpha{\uparrow_\forall} \mid x{:}\alpha \in \Gamma \,\}$

**Lemma 6** Let M be a term. Then $[\![M]\!](\hat{\Gamma}) = \{\, \alpha \mid \Gamma \vdash_{F\forall} M{:}\alpha \,\}$

**Proof** The proof is by induction on M. There are three cases. The first case where M is a variable is easy and is omitted. Second, suppose M is an application NL. Then in one direction,

$$[\![NL]\!](\hat{\Gamma}) = [\![N]\!](\hat{\Gamma}) \cdot [\![L]\!](\hat{\Gamma})$$
$$= \{\, \gamma \mid \Gamma \vdash_{F\forall} N{:}\gamma \,\} \cdot \{\, \alpha \mid \Gamma \vdash_{F\forall} L{:}\alpha \,\}$$

(by the induction hypothesis)

$$= \{\, \alpha\to\beta \mid \Gamma \vdash_{F\forall} N{:}\alpha\to\beta \,\} \cdot \{\, \alpha \mid \Gamma \vdash_{F\forall} L{:}\alpha \,\}$$

(by the definition of application)

$$= \{\, \beta \mid \Gamma \vdash_{F\forall} N{:}\alpha\to\beta \text{ and } \Gamma \vdash_{F\forall} L{:}\alpha\}{\uparrow_\forall}$$
$$\subset \{\, \beta \mid \Gamma \vdash_{F\forall} NL{:}\beta \,\}$$

(using the rule $(\to E)$ and the fact that $\{\beta \mid \Gamma \vdash_{F\forall} NL{:}\beta\}$ is a $\forall$-filter).

Conversely suppose that $\Gamma \vdash_{F\forall} NL{:}\delta$. Then $\Gamma^* \vdash NL{:}\delta^*$ is provable in the basic system, by Proposition 3, and so by the “characterisation of pure typing” in [28] there are types $\alpha$, $\beta$ and a sequence $\vec{s}$ of type variables such that $\Gamma^* \vdash N{:}\ \forall \vec{s}.\alpha\to\beta$ and $\Gamma^* \vdash L{:}\ \forall \vec{s}.\alpha$ are provable in the basic system, and $\forall \vec{s}.\beta \subset_g \delta^*$. (Mitchell’s pure typing system is the same as the basic system considered here). Let $\vec{\sigma}$ be any sequence of type expressions of the same length as $\vec{s}$. Since $\Gamma^* \vdash N{:}\ \forall \vec{s}.\alpha\to\beta$ is provable in the basic system, it follows that $\Gamma \vdash N{:}\ [\vec{\sigma}/\vec{s}](\alpha\to\beta)$ is provable in the basic system, and so (by the induction hypothesis) that $[\vec{\sigma}/\vec{s}](\alpha\to\beta)$ is in $[\![N]\!](\hat{\Gamma})$; similarly $[\vec{\sigma}/\vec{s}](\alpha)$ is in $[\![L]\!](\hat{\Gamma})$. Therefore $[\vec{\sigma}/\vec{s}]\beta$ is in $[\![NL]\!](\hat{\Gamma})$, and so as this is a $\forall$-filter and as $\vec{\sigma}$ is arbitrary we see that $\forall \vec{s}.\beta$ and hence $\delta$ is in $[\![NL]\!](\hat{\Gamma})$.

Third, if M is an abstraction $\lambda x.N$, then in one direction,

$$[\![\lambda x.N]\!](\hat{\Gamma}) = \{\, \alpha\to\beta \mid \beta \in [\![N]\!](\hat{\Gamma}(x := \alpha{\uparrow_\forall}))\}{\uparrow_\forall}$$
$$= \{\, \alpha\to\beta \mid \Gamma, x{:}\alpha \vdash_{F\forall} N{:}\beta \,\}{\uparrow_\forall}$$

(by induction hypothesis)

$$\subset \{\, \delta \mid \Gamma \vdash_{F\forall} \lambda x.N{:}\delta \,\}$$

(using the rule $(\to I)$ and the fact that $\{\, \delta \mid \Gamma \vdash_{F\forall} \lambda x.N{:}\delta \,\}$ is a $\forall$-filter)

Conversely, suppose that $\Gamma \vdash_{F\forall} \lambda x.N{:}\delta$. Then $\Gamma^* \vdash \lambda x.N{:}\delta^*$ is provable in the basic system, by Proposition 3, and so by the characterisation of pure typing in [28], there are type expressions $\alpha$, $\beta$ and a sequence $\vec{s}$ of type variables, none of which appears free in any type expression in $\Gamma^*$, such that $\Gamma^*, x{:}\alpha \vdash N{:}\beta$ is provable in the basic system, and it holds that $\forall \vec{s}.(\alpha\to\beta) \subset_g \delta$. Let $\vec{\sigma}$ be any sequence, of the same length as $\vec{s}$, of type expressions. Since $\Gamma^*, x{:}\alpha \vdash N{:}\beta$ is provable in the basic system, and no type variable in $\vec{s}$ appears free in any type expression in $\Gamma^*$ it follows that $\Gamma, x{:}[\vec{\sigma}/\vec{s}]\alpha \vdash N{:}[\vec{\sigma}/\vec{s}]\beta$ is provable in the basic system. So by the induction hypothesis $[\vec{\sigma}/\vec{s}]\beta$ is in $[\![N]\!](\hat{\Gamma}(x := [\vec{\sigma}/\vec{s}]\alpha{\uparrow_\forall}))$ and so

$[\vec{\sigma}/\vec{s}](\alpha\to\beta) \in [\![ \lambda x.N ]\!](\hat{\Gamma})$, and so as this is a $\forall$-filter and as $\vec{\sigma}$ is arbitrary, $\forall\vec{s}.\alpha\to\beta$ and hence $\delta$ is in $[\![ \lambda x.N ]\!](\hat{\Gamma})$. ■

With this lemma, the proof of completeness now follows very much as it did for the system $\lambda\to$.

**The simple semantics.** Simple type interpretations validate the following rule, introduced by Mitchell:

$$\text{(simple)}\quad \frac{\Gamma \vdash \lambda x.Mx{:}\alpha\to\beta}{\Gamma \vdash M{:}\alpha\to\beta} \qquad (x \notin FV(M))$$

To see this is the case, suppose we have a simple interpretation which satisfies $\Gamma \vdash \lambda x.Mx{:}\alpha\to\beta$, where x is not free in M, and we also have environments $\rho$,V satisfying every statement in $\Gamma$. Take a in $V(\alpha)$ to show that $[\![M]\!](\rho)\cdot a$ is in $V(\beta)$. By the assumptions, $[\![\lambda x.Mx]\!](\rho)\cdot a \in V(\beta)$. But $V(\beta)$ is upper closed and $[\![\lambda x.Mx]\!](\rho)\cdot a \leq [\![M]\!](\rho)\cdot a$, yielding the desired conclusion.

The above counter-example to Subject Reduction for $\beta\eta$-reduction also shows that (simple) is not a derived rule of the polymorphic typing system $\lambda\to\forall$. So, as simple type interpretations validate (simple) they cannot be complete for this polymorphic typing system. Instead, they are complete for the basic system for polymorphic types extended with (simple). Let us call this extended system "$\lambda\to\forall_S$", and write $\mathcal{B} \vdash_S M{:}\alpha$ to mean $\Gamma \vdash M{:}\alpha$ is provable in it, for some finite $\Gamma \subset \mathcal{B}$. It is easily seen that both Weakening and Strengthening hold for this system. The analogue to (modified) generic instantiation is *simple containment* $\alpha \subset_S \beta$, which is defined to be the least relation between type expressions which satisfies the following axioms and rules:

(gen) $\qquad \forall\vec{s}.\alpha \subset \forall\vec{r}.[\vec{\sigma}/\vec{s}]\alpha$

(no variable in $\vec{r}$ is free in $\forall\vec{s}.\alpha$)

(dist) $\qquad \forall\vec{s}.(\alpha\to\beta) \subset \forall\vec{s}.\alpha\to\forall\vec{s}.\beta$

(arrow) $\qquad$ if $\alpha' \subset \alpha$ and $\beta \subset \beta'$ then $(\alpha\to\beta) \subset (\alpha'\to\beta')$

(trans) $\qquad$ if $\alpha \subset \beta$ and $\beta \subset \gamma$ then $\alpha \subset \gamma$

(congruence) $\qquad$ if $\alpha \subset \beta$ then $\forall s.\alpha \subset \forall s.\beta$

The simple containment $\alpha \subset_S \beta$ holds iff $x{:}\alpha \vdash_S x{:}\beta$ (see [28]); it is an open problem whether simple containment is decidable. According to Theorem 16 of Mitchell [28], the system $\lambda\to\forall_S$ is equivalent to the system $\lambda\to\forall$ extended by the rule:

$$(\text{SubTypes}_S)\quad \frac{\Gamma \vdash M{:}\alpha\ ,\ \alpha \subset_S \beta}{\Gamma \vdash M{:}\beta}$$

Since modified generic instantiation is a sub-relation of simple containment, it follows that the system $\lambda\to\forall_S$ is stronger than $\lambda\to\forall$.

Subject Reduction for $\beta\eta$-reduction holds for the system $\lambda{\rightarrow}\forall_S$. This is an immediate consequence of:

**Proposition 5** $\Gamma \vdash_S M{:}\alpha$ iff there is a $\lambda$-term N such that both $N \twoheadrightarrow_{\beta\eta} M$ and also $\Gamma \vdash_{F\forall} N{:}\alpha$.

**Proof** The implication from left to right is a straightforward induction on the size of proof of $\Gamma \vdash_S M{:}\alpha$, and is remarked on p228 of [28]. Mitchell's Lemma 16 shows that the converse holds for the basic system for polymorphic types extended by ($\text{SubTypes}_S$) and that is equivalent to $\lambda{\rightarrow}\forall_S$, as remarked above. ■

**The term model** With this result a proof of completeness can be based on the term model $\mathcal{R}_{\beta\eta}$, following the above lines. For a type interpretation, given any finite basis $\Gamma$ and term A, define a basis $\mathcal{B}$ as above, and with the analogous understanding of $\mathcal{B} \vdash_S M{:}\alpha$. Define:

$$X_\alpha = \{ [M] \mid \mathcal{B} \vdash_S M{:}\alpha \}$$

making use of Subject Reduction for $\beta\eta$-reduction, and take

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression} \}$$

Here $X_\alpha$ does not quite determine $\alpha$; we get instead that $X_\alpha \subset X_\beta$ holds iff $\alpha \subset_S \beta$; the implication from left to right is proved as for the system $\lambda{\rightarrow}\forall$; the other direction follows from an application of the rule ($\text{SubTypes}_S$). It follows that $X_\alpha = X_\beta$ iff $\alpha \equiv_S \beta$ (the equivalence associated with simple containment).

Now, using the (arrow) rule, $\rightarrow$ can be defined by:

$$X_\alpha \rightarrow X_\beta = X_{\alpha\rightarrow\beta}$$

as before. The arrow conditions hold, and are proved as before. For the interpretation of type expressions, put:

$$[\![\alpha]\!](V) = X_{[\vec{\sigma}/\vec{s}]\alpha}$$

where $\vec{s}$ is a list of the free type variables of $\alpha$ and $X_{\vec{\sigma}} = V(\vec{s})$. The verification of the three conditions proceeds as before. It remains to see that this semantics is indeed simple. One direction is the first arrow condition. For the other suppose that whenever [N] is in $X_\alpha$ then [MN] is in $X_\beta$. Choose $x{:}\alpha$ in $\mathcal{B}$ with x not free in M. Then [Mx] is in $X_\beta$, which is to say that $\mathcal{B} \vdash_S Mx{:}\beta$. So by ($\rightarrow$I), $\mathcal{B} \vdash_S \lambda x.Mx{:}\alpha\rightarrow\beta$, and so by (simple) we have that $\mathcal{B} \vdash_S M{:}\alpha\rightarrow\beta$ as required. Now one chooses $\rho_0, V_0$ as before, and obtains:

**Completeness Theorem 7** If $\Gamma \models M{:}\alpha$ holds in every syntactical model of $\beta\eta$-reduction and every simple type interpretation, then $\Gamma \vdash_S M{:}\alpha$.

**A type expression model** Again there is a completeness proof *via* a type expression model of reduction. Say now that a *simple $\forall$-filter* over the set of polymorphic type expressions is a subset F such that:

($1_S$) if $\alpha \in F$ and $\alpha \subset_S \beta$ then $\beta \in F$

($2_S$) if $[\beta/t]\alpha \in F$ for all type expressions $\beta$, then $\forall t.\alpha \in F$

As before, $\{\alpha\}{\uparrow}_{\forall_S} = \{\beta \mid \alpha \subset_S \beta \}$ is a simple $\forall$-filter (where for any set F of type expressions, $F{\uparrow}_{\forall_S}$ is the least simple $\forall$-filter containing F). The proof is as above for $\{\alpha\}{\uparrow}_\forall$; one now needs that if $\alpha \subset_S \beta$ and t is not free $\alpha$ in then it follows that $\alpha \subset_S \forall t.\beta$. But this holds as by (gen) $\alpha \subset_S \forall t.\alpha$, and by (congruence) $\forall t.\alpha \subset_S \forall t.\beta$. Sets of the form $\{ \alpha \mid \Gamma \vdash_S M{:}\alpha \}$ are simple $\forall$-filters, and this is proved as in the above case. The model of reduction is the set of simple $\forall$-filters, partially ordered by subset, application is defined by:

$a \cdot b = \{ \beta \mid \text{for some } \alpha \text{ in } b, \alpha \to \beta \text{ is in } a \} \uparrow_{\forall_S}$

And $[\![ \cdot ]\!](\cdot)$ is defined by the inductive clauses

$[\![ x ]\!](\rho) = \rho(x)$

$[\![ MN ]\!](\rho) = [\![ M ]\!](\rho) \cdot [\![ N ]\!](\rho)$

$[\![ \lambda x.M ]\!](\rho) = \{ \alpha \to \beta \mid \beta \in [\![ M ]\!](\rho(x := \alpha\uparrow_{\forall_S})) \} \uparrow_{\forall_S}$

As before this defines a model of $\beta$-reduction; for condition (v) one needs:

**Lemma 8**

$[\![ \lambda x.M ]\!](\rho) = \{ \forall \vec{s}.(\alpha \to \beta) \mid \text{for all } \vec{\sigma}, [\vec{\sigma}/\vec{s}]\beta \in [\![ M ]\!](\rho(x := [\vec{\sigma}/\vec{s}]\alpha\uparrow_{\forall_S})) \}$

**Proof** First, that (1) and ($2_S$) are satisfied is shown as in the proof of Lemma 5. Next that ($1_S$) is satisfied is shown by induction on the number of axioms and rules used to show that $\alpha \subset_S \beta$. The proof divides into cases according to the last rule or axiom used to show $\alpha \subset_S \beta$. The case (gen) is just that (1) is satisfied, which has already been proved. In case (dist) let us assume that $\forall \vec{s}.(\alpha \to \beta)$ is in the right hand side and show that $\forall \vec{s}.\alpha \to \forall \vec{s}.\beta$ is too. To this end, it is enough to show that $[\vec{\sigma}/\vec{s}]\beta \in [\![ M ]\!](\rho(x := \forall \vec{s}.\alpha\uparrow_{\forall_S}))$, for arbitrary $\vec{\sigma}$ of the same length as $\vec{s}$. But since $\forall \vec{s}.(\alpha \to \beta)$ is in the right hand side, it follows that $[\vec{\sigma}/\vec{s}]\beta$ is an element of $[\![ M ]\!](\rho(x := [\vec{\sigma}/\vec{s}]\alpha\uparrow_{\forall_S}))$, and the conclusion follows using the monotonicity of $[\![ \cdot ]\!](\cdot)$ in its second argument and the simple containment $\forall \vec{s}.\alpha \subset_S [\vec{\sigma}/\vec{s}]\alpha$.

The case (arrow) is rather similar, and so is omitted. The case (trans) is immediate using the inductive hypothesis twice. Finally we come to the case (congruence). Here we need the fact - easily proved by induction - that if $\alpha \subset_S \beta$, then $[\vec{\sigma}/\vec{s}]\alpha \subset_S [\vec{\sigma}/\vec{s}]\beta$ can be proved using the same number of rules. So suppose that $\alpha \subset_S \beta$, and $\forall s.\alpha$ is in the right hand side. We have to show that so is $\forall s.\beta$. Choose a type expression $\sigma$. Since $\forall s.\alpha$ is in the right hand side so is $[\sigma/s]\alpha$. Now as $\alpha \subset_S \beta$, and $[\sigma/s]\alpha \subset_S [\sigma/s]\beta$ can be proved using the same number of rules, and so we can apply the inductive hypothesis to see that $[\sigma/s]\beta$ is in the right hand side. But $\sigma$ as was arbitrary, it follows from the satisfaction of (2) that $\forall s.\beta$ is in the right hand side. ■

The partial order of $\forall$-filters is a complete lattice, but (as is shown by a variant of the corresponding counterexample given above) application is again not continuous. As before, the model of reduction can be located in the category of complete partial orders and countably continuous functions. In the definition of the type interpretation one has $X_\alpha \subset X_\beta$ iff $\alpha \subset_S \beta$, and so $X_\alpha = X_\beta$ iff $\alpha \equiv_S \beta$; as before, this difference causes no problems. The arrow conditions hold; one proves as above that

$X_{\forall t.\alpha} = \bigcap \{ X_{[\beta/t]\alpha} \mid \beta \text{ a type expression} \}$

and then for any type expression $\alpha$ and type environment V,

$[\![ \alpha ]\!](V) = X_{[\vec{\sigma}/\vec{s}]\alpha}$

where $\vec{s}$ is a list of the free type variables in $\alpha$ and $X_{\vec{\sigma}}$ is $V(\vec{s})$.

One has also now to show that the type interpretation is simple. To this end, note first that one direction is the first arrow condition. For the other suppose that whenever b is in $X_\alpha$ then a.b is in $X_\beta$. Then taking b to be $\alpha\!\uparrow_{\forall S}$ we have that $\beta \in (a.\alpha\!\uparrow_{\forall S})$, and wish to show that $\alpha\to\beta \in a$, as then $a \in X_{\alpha\to\beta}$ as desired. To this end we prove by induction on the inductive construction of $\{\beta \mid \text{for some } \delta \text{ in } \alpha\!\uparrow_{\forall S}, \delta\to\beta \in a\}\!\uparrow_{\forall S}$ (which is $(a.\alpha\!\uparrow_{\forall S})$) that for any $\beta$, if $\beta$ is in $\{\beta \mid \text{for some } \delta \text{ in } \alpha\!\uparrow_{\forall S}, \delta\to\beta \in a\}\!\uparrow_{\forall S}$ then $\alpha\to\beta \in a$. The first case is where for some $\delta$ in $\alpha\!\uparrow_{\forall S}$, $\delta\to\beta \in a$. Here $\alpha \subset_S \delta$, and so by (arrow) $(\delta\to\beta) \subset_S (\alpha\to\beta)$, and hence $\alpha\to\beta \in a$. In the second case there is a $\delta \subset_S \beta$, with $\alpha\to\delta \in a$, and we get that $\alpha\to\beta \in a$, again using (arrow). Finally suppose that for all $\delta$, $\alpha\to[\delta/t]\gamma$ is in a, and $\beta$ has the form $\forall t.\gamma$. One can suppose without loss of generality that t is not free in $\alpha$, and so for all $\delta$, $[\delta/t](\alpha\to\gamma)$ is in a and we get that $\forall t.(\alpha\to\gamma)$ is in a, and so by (dist) and (arrow) $\alpha\to\forall t.\gamma$ is in a, concluding the inductive proof.

The proof of the analogue of Lemma 6 proceeds just as before, but without the need for an analogue of Proposition 3, using instead Mitchell's characterisation of pure typing for the simple type discipline, and completeness follows. Finally, let us show that – as may be expected – the $\forall$-filter model of $\beta$-reduction obtained is even a syntactical model of $\beta\eta$-reduction. Suppose x is not free in M. Then one calculates that:

$$[\![\lambda x.Mx]\!](\rho) = \{\forall \vec{s}.(\alpha\to\beta) \mid \text{for all } \vec{\sigma}, [\vec{\sigma}/\vec{s}]\beta \in [\![Mx]\!](\rho(x{:=}[\vec{\sigma}/\vec{s}]\alpha\!\uparrow_{\forall S}))\}$$

(by Lemma 8)

$$= \{\forall \vec{s}.(\alpha\to\beta) \mid \text{for all } \vec{\sigma}, [\vec{\sigma}/\vec{s}]\beta \in [\![M]\!](\rho).[\vec{\sigma}/\vec{s}]\alpha\!\uparrow_{\forall S}\}$$

(as x is not free in M)

$$\subset \{\forall \vec{s}.(\alpha\to\beta) \mid \text{for all } \vec{\sigma}, [\vec{\sigma}/\vec{s}]\alpha\to[\vec{\sigma}/\vec{s}]\beta \in [\![M]\!](\rho)\}$$

(by the same reasoning as used to show the type interpretation simple)

$$\subset [\![M]\!](\rho) \text{ (as } [\![M]\!](\rho) \text{ satisfies condition } (2_S))$$

## 6. The Intersection Type Discipline

Intersection types were independently introduced by Coppo and Dezani-Ciancaglini, by Sallé and by Pottinger (see [3,6,9,20,36] for references and further discussion). The surprising feature of such systems is that even though ($EQ_\beta$) is not derivable, it is admissible - provided empty intersections are allowed. We may then use ordinary models for completeness proofs. We are back in the world considered first by Barendregt, Coppo and Dezani-Ciancaglini and by Hindley, with the caveat that only systems sound for the wider class of models of β-reduction are to be considered. There is nonetheless some interest in pursuing such systems. First the availability of non-simple type interpretations allows completeness proofs for other, perhaps simpler, type inference systems. Secondly these systems are very closely connected to the models of the λ-calculus given by Plotkin [30] and Engeler [14].

A natural basic system with intersection types is obtained by adding to the system $\lambda\!\to$ binary and empty intersections of types: $\alpha \cap \beta$ and $\omega$. The rules we take are

$$(\cap I) \qquad \frac{\Gamma \vdash M{:}\alpha, \ \ \Gamma \vdash M{:}\beta}{\Gamma \vdash M{:}\alpha\cap\beta}$$

$$(\cap E) \qquad 1.\ \frac{\Gamma \vdash M{:}\alpha\cap\beta}{\Gamma \vdash M{:}\alpha}$$

$$2.\ \frac{\Gamma \vdash M{:}\alpha\cap\beta}{\Gamma \vdash M{:}\beta}$$

$$(\omega I) \qquad \Gamma \vdash M{:}\omega$$

This system has not been considered in the literature (see [9] for a closely related system) as it is not complete for simple type interpretations. Completeness for our more general type interpretations requires the addition of an extra rule, as in the case of polymorphism. As before, there is a possibility of non-trivial equalities between type expressions; for example if $\cap$ is interpreted as intersection then $((\alpha \cap \beta) \to \gamma)$ and $((\beta \cap \alpha) \to \gamma)$ have equal interpretations.

We consider the extension of the basic system for intersection types with the rule (EQTypes), and the following rules and axioms for the equality of types:

$$\alpha = \alpha$$

$$\frac{\alpha = \beta}{\beta = \alpha}$$

$$\frac{\alpha = \beta\ ,\ \beta = \gamma}{\alpha = \gamma}$$

$$\frac{\alpha = \beta\ ,\ \alpha' = \beta'}{\alpha \to \alpha' = \beta \to \beta'}$$

$$\frac{\alpha = \beta\ ,\ \alpha' = \beta'}{\alpha \cap \alpha' = \beta \cap \beta'}$$

$$((\alpha \cap \beta) \cap \gamma) = (\alpha \cap (\beta \cap \gamma))$$
$$(\alpha \cap \beta) = (\beta \cap \alpha)$$
$$(\alpha \cap \alpha) = \alpha$$
$$(\alpha \cap \omega) = \alpha$$

Essentially this system was previously considered in [6,20]. Let us call it "$\lambda{\to}{\cap}\omega$" and for the notion of *$F{\cap}\omega$-deducibility*, the relation $\Gamma \vdash_{F\cap\omega} M{:}\alpha$, is taken to be that $\Gamma \vdash M{:}\alpha$ is provable using this system of rules. Let us also define $\alpha =_{F\cap\omega} \beta$ to hold just when $\alpha = \beta$ is provable. The system yields different typings than the basic system; for example $\lambda x.x$ has type $((\alpha \cap \beta) \to \gamma) \to ((\beta \cap \alpha) \to \gamma)$ in this system, but not in the basic system. Note that it is again the congruence rule for arrow, that makes the difference; if omitted, the resulting system is equivalent to the basic system.

Once one has an intersection type available, one can define a formal subtype relation by letting $\alpha \subset \beta$ abbreviate $\alpha = \alpha\cap\beta$; we write $\alpha \subset_{F\cap\omega} \beta$ to mean that $\alpha \subset \beta$ is provable in $\lambda{\to}{\cap}\omega$. The following rule is derivable in $\lambda{\to}{\cap}\omega$:

$$\text{(SubTypes)} \qquad \frac{\Gamma \vdash M{:}\alpha\ ,\ \alpha \subset \beta}{\Gamma \vdash M{:}\beta}$$

It follows, in particular that if $\alpha \subset_{F\cap\omega} \beta$ then $x{:}\alpha \vdash_{F\cap\omega} x{:}\beta$; the converse also holds, the proof being a straightforward induction. Both Weakening and Strengthening hold for $\lambda{\to}{\cap}\omega$, by straightforward inductive proofs. Further, $(EQ_\beta)$ is admissible, as shown in [6].

The system is closely relate to the model of the untyped $\lambda\beta$-calculus given in [30]. Let T be the smallest set containing all type variables and closed under the rule that, if u and v are finite subsets of T, then the pair <u,v> (to be written as $u \to v$) is in T. In [30] a model of the untyped $\lambda$-calculus was given consisting, up to isomorphism, of all subsets of T, but with the (unnecessary) assumption that there is just one type variable; more details are given below. Here T will provide us with a way to define normal forms of type expressions. To each type expression $\alpha$ we associate a finite subset $\alpha^\sigma$ of T by a recursive definition:

$$\begin{array}{lcl} t^\sigma & = & \{t\} \\ (\alpha \to \beta)^\sigma & = & \{\alpha^\sigma \to \beta^\sigma\} \\ \omega^\sigma & = & \varnothing \\ (\alpha \cap \beta)^\sigma & = & (\alpha^\sigma \cup \beta^\sigma) \end{array}$$

Now to each element $\alpha$ of T we associate a type expression $\alpha^\tau$, and to each finite subset u of T we associate a type expression $u^\tau$. This is done by a simultaneous recursive definition, in which a fixed linear ordering of T is assumed available:

$$\begin{array}{lcl} t^\tau & = & t \\ (u \to v)^\tau & = & u^\tau \to v^\tau \\ \{\alpha_1,\ldots,\alpha_{n-1},\alpha_n\}^\tau & = & (\alpha_1{}^\tau \cap .... (\alpha_{n-1}{}^\tau \cap \alpha_n{}^\tau) \ldots ) \end{array}$$

where in the last equation the right hand side is understood to be $\omega$ if n=0, $\alpha_1{}^\tau$ if n=1, and otherwise, the $\alpha_i$ are taken in the assumed ordering. One can show, by straightforward inductive proofs, first that if $\alpha =_{F\cap\omega} \beta$ then $\alpha^\sigma$ and $\beta^\sigma$ are identical, and second $\alpha =_{F\cap\omega} \alpha^{\sigma\tau}$ for any type expression $\alpha$, and third that $\alpha^{\tau\sigma} = \alpha$ and $u^{\tau\sigma} = u$ for any $\alpha$ in T, and any finite subset u of T. It follows that we have a normal form for type expressions, in that first $\alpha =_{F\cap\omega} \beta$ iff $\alpha^{\sigma\tau}$ and $\beta^{\sigma\tau}$ are identical, and second that for any type expression $\alpha$, $\alpha =_{F\cap\omega} \alpha^{\sigma\tau}$. It also follows that $\alpha =_{F\cap\omega} \beta$ is equivalent to the identity of $\alpha^\sigma$ and $\beta^\sigma$. Now derivability in $\lambda{\to}\cap\omega$ can be characterised in terms of the basic system. In the following proposition the $(\cdot)^\sigma$ and $(\cdot)^\tau$ transformations are applied also to bases; this is intended to mean that the transformations are applied to type expressions, or elements or finite subsets of T, as appropriate.

**Proposition 6** The sequent $\Gamma \vdash M{:}\alpha$ is provable in $\lambda{\to}\cap\omega$ iff it is provable in the basic system that $\Gamma^{\sigma\tau} \vdash M{:}\alpha^{\sigma\tau}$.
**Proof** The proof is like that of Proposition 3. ■

Finally, we can better understand subtyping. First we have that:

$$\begin{array}{lll} \alpha \subset_{F\cap\omega} \beta & \text{iff} & \alpha =_{F\cap\omega} \alpha\cap\beta \\ & \text{iff} & \alpha^\sigma = \alpha^\sigma \cup \beta^\sigma \\ & \text{iff} & \alpha^\sigma \supset \beta^\sigma \end{array}$$

Now every type expression is a formal intersection of type variables and function type expressions (taking $\omega$ to be an empty such intersection). It then follows that $\alpha \subset_{F\cap\omega} \beta$ iff every such type variable or function type expression in $\beta$ is provably equal to one such in $\alpha$.

*Type interpretations* $\langle Ty,\to\rangle$ are as for the system $\lambda{\to}$ except that Ty must be closed under finite intersections. Valuations are extended to type expressions as before for functional types; for the intersection type expressions we put $V(\alpha \cap \beta) = V(\alpha)\cap V(\beta)$ and $V(\omega) = P$ (using the above notation). With the evident definition of satisfaction, the soundness theorem is then straightforwardly proved.

In [35] van Bakel considers a strict type assignment system and proves a completeness theorem for it, using a variation of Mitchell's inference model of polymorphic types. The provable sequents of his system are those of the basic system, except that the type expressions occurring are restricted to be Hindley's "Normal type schemes" [17].

**The term model** For our first proof of completeness, we use the term model of β-reduction $\mathcal{C}_\beta = \langle C, \cdot, [\![\cdot]\!](\cdot) \rangle$ whose elements are equivalence classes of (open) terms under β–conversion, with the trivial partial order, and whose definition is otherwise exactly as that of $\mathcal{R}_\beta$ (dropping the partial order, this is, of course the usual term model of open terms under β–conversion [2]). For the type interpretation, given any finite basis Γ and term A define a basis $\mathcal{B}$ as above, and with the analogous understanding of $\mathcal{B} \vdash_{F\cap\omega} M{:}\alpha$. Define:

$$X_\alpha = \{ [M] \mid \mathcal{B} \vdash_{F\cap\omega} M{:}\alpha \}$$

as before, again making use of the admissibility of ($EQ_\beta$) and take

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression} \}$$

To see that $X_\alpha$ essentially determines α, suppose that $X_\alpha \subset X_\beta$. Taking a statement of the form $x{:}\alpha$ in $\mathcal{B}$, we get that $\mathcal{B} \vdash_{F\cap\omega} x{:}\beta$, and hence, by Strengthening, it follows that $x{:}\alpha \vdash_{F\cap\omega} x{:}\beta$, and so that $\alpha \subset_{F\cap\omega} \beta$. Therefore if $X_\alpha = X_\beta$ then $\alpha =_{F\cap\omega} \beta$, and in view of the rule (EQTypes) the converse also holds. Therefore, → can be defined by:

$$X_\alpha \rightarrow X_\beta = X_{\alpha\rightarrow\beta}$$

and the arrow conditions are verified as before. Since it is easily verified that:

$$X_\omega = C$$
$$X_{\alpha\cap\beta} = X_\alpha \cap X_\beta$$

Ty is closed under finite intersections, and so we have a proper type interpretation. Completeness can now be proved along the same lines as before.

**A type expression model** The other approach to completeness is to use a filter model along the lines of Barendregt *et al* [4]. Here the filter model turns out to be isomorphic to the set-theoretic model of Plotkin.

A *filter* is a set F of type expressions such that:

1. if $\alpha \subset_{F\cap\omega} \beta$, and $\alpha \in F$, then $\beta \in F$
2. $\omega \in F$
3. if $\alpha, \beta \in F$, then $\alpha \cap \beta \in F$

Every filter can be mapped to a subset of T by:

$$F^\sigma = \bigcup\{\alpha^\sigma \mid \alpha \in F\}$$

and every subset of T can be mapped to a filter by:

$$a^\tau = \{\alpha \mid \alpha^\sigma \subset a\}$$

Using the properties given above of the $(\cdot)^\sigma$ and $(\cdot)^\tau$ transformations, it is easy to see that these two mappings are inverses; they are also clearly monotonic. The relation between filter models and set-theoretic models has also been considered in [8], where it is shown that the filter model of [4] can be embedded in Engeler's model; use is made there of a function similar to the $(\cdot)^\sigma$ transformation.

Since the filter and set-theoretic models considered here are isomorphic, we at liberty to work with either, and choose the simpler model $\mathcal{P}(T)$ of all subsets of T. This forms a complete lattice under the subset ordering. The model can be presented as a syntactical model of β-reduction. Application is defined by:

$$a \cdot b = \bigcup\{ v \mid \text{for some } u \text{ a finite subset of } b,\ u \to v \text{ is in } a \}$$

and $[\![ \cdot ]\!](\cdot)$ is defined by the inductive clauses

$$[\![ x ]\!](\rho) = \rho(x)$$

$$[\![ MN ]\!](\rho) = [\![ M ]\!](\rho) \cdot [\![ N ]\!](\rho)$$

$$[\![ \lambda x.M ]\!](\rho) = \{ u \to v \mid v \subset [\![ M ]\!](\rho(x := u)) \}$$

The model can be placed in the category of complete partial orders and continuous functions with F defined as before, and:

$$G(f) = \{ u \to v \mid v \subset f(u) \}$$

For the type interpretation define:

$$X_\alpha = \{ a \mid \alpha^\sigma \subset a \}$$

and then put

$$Ty = \{ X_\alpha \mid \alpha \text{ a type expression} \}$$

Note that $X_\alpha \subset X_\beta$ holds iff $\beta^\sigma \subset \alpha^\sigma$ holds, iff $\alpha \subset_{F\cap\omega} \beta$ holds. Thus $X_\alpha$ determines $\alpha$ up to provable equality and we can define the arrow operation by

$$X_\alpha \to X_\beta = X_{\alpha \to \beta}$$

The arrow conditions are easy to verify, and as we also have the equations

$$X_\omega = \mathcal{P}(T)$$

$$X_{\alpha\cap\beta} = X_\alpha \cap X_\beta$$

we have a type interpretation.

It is necessary now to relate type inference to denotation in the model, and to this end we introduce a subsidiary type inference system where we take finite subsets and elements of T as the "type expressions", adapting the usual notions of type assignment statements, and bases in the evident way except that we only consider bases where the type assignment statements have the form x:u, where u is a finite subset of T. The system has the following rules:

$$\frac{\Delta \vdash M:\alpha \quad (\alpha \in u)}{\Delta \vdash M:u}$$

$$\Delta, x:u \vdash x:\alpha \quad (\alpha \in u)$$

$$\frac{\Delta\ ,\ x{:}u \vdash M{:}v}{\Delta \vdash \lambda x.M{:}u \to v}$$

$$\frac{\Delta \vdash M{:}u \to v\ \ ,\ \ \Delta \vdash N{:}u}{\Delta \vdash MN{:}\alpha}\quad (\alpha \in v)$$

We write $\Delta \vdash_T M{:}\delta$ for derivability of sequents in this system (where $\delta$ is an element or finite subset of T). The system can be thought of as providing normal forms for proofs in $\lambda{\to}\cap\omega$, although no theorem to that effect will be proved. However we do consider the relationship as far as provable sequents is concerned.

**Proposition 7** 1. $\Gamma \vdash_{F\cap\omega} M{:}\alpha$ iff $\Gamma^{\sigma} \vdash_T M{:}\alpha^{\sigma}$.
2. $\Delta \vdash_T M{:}\delta$ iff $\Delta^{\tau} \vdash_{F\cap\omega} M{:}\delta^{\tau}$

($\delta$ an element or finite subset of T)

**Proof** First one shows the implications from left to right by straightforward inductive arguments. For the converse to 1, if $\Gamma^{\sigma} \vdash_T M{:}\alpha^{\sigma}$, then $\Gamma^{\sigma\tau} \vdash_{F\cap\omega} M{:}\alpha^{\sigma\tau}$, by 2. So $\Gamma \vdash_{F\cap\omega} M{:}\alpha$ since $\lambda{\to}\cap\omega$-provability is invariant under substitution of provably equal type expressions. The converse to 2 is even simpler, making use of the facts that $u^{\tau\sigma} = u$, for u a finite subset of T, and $\alpha^{\tau\sigma} = \{\alpha\}$, for $\alpha$ in T. ■

We can now proceed with the proof of completeness *via* the set theoretic model. For any finite T-basis $\Delta$ define an environment by:

$$\hat{\Delta}(x) = \bigcup\{\ u\ \mid\ x{:}u \in \Delta\ \}$$

**Lemma 9** Let M be a term. Then $[\![M]\!](\hat{\Delta}) = \{\ \alpha\ \mid\ \Delta \vdash_T M{:}\alpha\ \}$
**Proof** The proof is much like that of Lemma 4. ■

Now a type environment is defined by $V_1(t)=X_t$, and one has that $V_1(\alpha)=X_\alpha$. For completeness, assume $\Gamma \models M{:}\alpha$ holds. Note that it holds that $(\Gamma^{\sigma})^{\wedge}$, $V_1$ satisfy every type statement $x{:}\alpha$ in $\Gamma$, as $(\Gamma^{\sigma})^{\wedge}(x)$ is $\alpha^{\sigma}$, and $V_1(\alpha)$ is $X_\alpha$. So $(\Gamma^{\sigma})^{\wedge}$, $V_1$ satisfies $M{:}\alpha$, which means that $[\![M]\!]((\Gamma^{\sigma})^{\wedge})$ is in $X_\alpha$, that is that $\alpha^{\sigma} \subset [\![M]\!]((\Gamma^{\sigma})^{\wedge})$. So $\Gamma^{\sigma} \vdash_T M{:}\alpha^{\sigma}$ by Lemma 9, and so by Proposition 7, $\Gamma \vdash_{F\cap\omega} M{:}\alpha$, concluding the proof. As $\mathcal{P}(T)$ is actually a model of the $\lambda\beta$-calculus, the completeness result gives a semantic proof of the admissibility of $(EQ_\beta)$ for by $\lambda{\to}\cap\omega$.

**Other systems of intersection types** Let us conclude with some remarks on Engeler's model [14] and also on the case of simple type interpretations, as considered by Barendregt *et al* [4], and by Hindley [17]. Instead of T, Engeler considers the least set B containing a given set A (which we take here to be the set of all type variables) and such that if u is a finite subset of B and $\beta$ is an element of B then the pair $<u,\beta>$ (to be written as $u \to \beta$) is in B. Then $\mathcal{P}(B)$ is a model of the $\lambda\beta$-calculus. Application is defined by:

$$a \cdot b = \{\ \beta \mid \text{for some } u \text{ a finite subset of } b,\ u \to \beta \text{ is in } a\ \}$$

and $[\![\ \cdot\ ]\!](\cdot)$ is defined by the inductive clauses:

$$[\![\ x\ ]\!](\rho) = \rho(x)$$

$$[\![ MN ]\!](\rho) = [\![ M ]\!](\rho) \cdot [\![ N ]\!](\rho)$$

$$[\![ \lambda x.M ]\!](\rho) = \{ u \to \beta \mid \beta \in [\![ M ]\!](\rho(x := u)) \}$$

Again, the model can be placed in the category of complete partial orders and continuous functions with F defined as before, and now:

$$G(f) = \{ u \to \beta \mid \beta \in f(u) \}$$

For the corresponding type system, one adds these axioms to $\lambda\!\to\!\cap\omega$:

$$(\alpha \to \omega) = \omega$$

$$(\alpha \to (\beta \cap \gamma)) = (\alpha \to \beta) \cap (\alpha \to \gamma)$$

The development of the theory can proceed along the lines developed above for $\lambda\!\to\!\cap\omega$, with evident minor variations. For example when defining the transformations one puts:

$$(\alpha \to \beta)^\sigma \;=\; \{\alpha^\sigma \to \gamma \mid \gamma \in \beta^\sigma\}$$

$$(u \to \beta)^\tau \;=\; u^\tau \to \beta^\tau$$

One considers type interpretations which satisfy the above extra two type equations, and completeness can be proved by using either a term model or Engeler's model. For the latter, the modified type inference system is:

$$\frac{\Delta \vdash M:\alpha \quad (\alpha \in u)}{\Delta \vdash M:u}$$

$$\Delta , x:u \vdash x:\alpha \quad (\alpha \in u)$$

$$\frac{\Delta , x:u \vdash M:\beta}{\Delta \vdash \lambda x.M:u \to \beta}$$

$$\frac{\Delta \vdash M:u \to \beta \;,\; \Delta \vdash N:u}{\Delta \vdash MN:\beta}$$

The system of Barendregt *et al* [4] can be equivalently formulated as $\lambda\!\to\!\cap\omega$ plus the two rules above plus the conditional rule

$$\frac{\alpha' \subset \alpha}{(\alpha \to \beta) \subset (\alpha' \to \beta)}$$

Hindley gave a completeness proof *via* term models in [17] and Barendregt *et al* used a filter model in [4]. This filter model is isomorphic to yet another set theoretic model, consisting of the up-closed subsets of a certain partial order $\leq_B$ defined on Engeler's set B. This partial order is defined to be the least partial order $\leq$ over B such that:

If

for every $\beta$ in v there is an $\alpha$ in u such that $\alpha \leq \beta$

and also $\alpha \leq \beta$

then $u \to \alpha \leq v \to \beta$.

Using the results in [17] on normal type expressions, one can show that $\alpha \leq \beta$ in the sense of [17] iff $\alpha^{\sigma} \leq_B \beta^{\sigma}$, and also that for any $\alpha$ in B, $\alpha^{\tau\sigma} \equiv_B \alpha$, where $\equiv_B$ is the equivalence associated to $\leq_B$. With this one can show that the filter model of Barendregt *et al*, is isomorphic to the collection of up-closed subsets of B, with application defined as for Engeler's model and with $[\![ \cdot ]\!](\cdot)$ defined by the similar inductive clauses:

$$[\![ x ]\!](\rho) = \rho(x)$$

$$[\![ MN ]\!](\rho) = [\![ M ]\!](\rho) \cdot [\![ N ]\!](\rho)$$

$$[\![ \lambda x.M ]\!](\rho) = \{ u \to \beta \mid \beta \in [\![ M ]\!](\rho(x := u\uparrow)) \}$$

(where $u\uparrow$ is the least up-closed subset containing u). The model can be placed in the category of complete partial orders and continuous functions: take F as before, and

$$G(f) = \{ u \to \beta \mid \beta \in f(u\uparrow) \}$$

It may well also be possible to treat this type discipline directly, along the lines followed for the previous two systems.

## Acknowledgements

Part of this work was done while the author was visiting the MIT Laboratory for Computer Science in 1981 and 1982. He is grateful to A. Meyer for supporting the visit, with the aid of an NSF grant. He also thanks H. Yokouchi who pointed out both that completeness of F-interpretations for Curry's theory of functionality could be proved by using the term model formed from $\beta\eta$-reduction, and also that it was necessary to modify Mitchell's type assignment system to obtain completeness for the type interpretations considered here - correcting an error in [31]. Finally, thanks are due to M. Abadi, M. Dezani-Ciancaglini, R. Hindley, B. Jacobs, M. Coppo, I. Margaria, R. K. Meyer, J. Mitchell, S. van Bakel and M. Zacchi for useful discussions.

## References

[1] Barbanera, F. and Dezani-Ciancaglini, M. Intersection and union types, in: Theoretical Aspects of Computer Software, eds. Ito, T. and Meyer, A.R. *Lecture Notes in Computer Science* **526** pp. 651-674. Berlin: Springer Verlag (1991).

[2] Barendregt, H.P. *The lambda calculus: its syntax and semantics.* Amsterdam: North-Holland (1984).

[3] Barendregt, H.P. Lambda calculi with types. To appear in *Handbook of Logic in Computer Science, Volume II*, eds. Abramsky, S., Gabbay, D.M. and Maibaum, T.S.E. Oxford: Oxford University Press.

[4] Barendregt, H.P., Coppo M., and Dezani-Ciancaglini, M. A filter lambda model and the completeness of type assignment. *J. Symbolic Logic* **48**/4, pp. 931-940 (1983).

[5] Bunder, M.W. and Meyer, R.K. A result for combinators, BCK logics and BCK algebras. *Logique et Analyse* **109**, pp. 33-40. (1985)

[6] Cardone, F. and Coppo, M. Two extensions of Curry's type inference system, in: *Logic and Computer Science*, ed. Odifreddi, P. APIC Series in Data Processing **31**. London: Academic Press (1990)

[7] Cardone, F. and Coppo, M. Type inference with recursive types: syntax and semantics. *Information and Computation* **92**(1) pp. 48-80 (1991)

[8] Coppo, M., Dezani-Ciancaglini, M., Honsell F. and Longo, G. Extended type structures and filter lambda models, in: *Logic Colloquium '82*, eds. Lolli, G., Longo, G. and Marcja, A. pp. 241-262. Amsterdam: North Holland (1983)

[9] Coppo, M., Dezani-Ciancaglini, M. and Venneri, B. Functional characters of solvable terms. *Z. Math. Log. Grund. Math.* **27**(1) pp. 45-58 (1981)

[10] Curry, H.B. and Feys, R. *Combinatory logic, Vol. I*, Studies in Logic and the Foundations of Mathematics. Amsterdam: North Holland. (1958)

[11] Curry, H.B., Hindley, J.R. and Seldin, J.P. *Combinatory logic, Vol. II.* Studies in Logic and the Foundations of Mathematics. Amsterdam: North Holland. (1972)

[12] Dezani-Ciancaglini, M. and Margaria, I. A characterisation of F-complete type assignments. *Theor. Comp. Sci.* **45** pp. 121-157 (1986)

[13] Dezani-Ciancaglini, M. and Margaria, I. Polymorphic types, fixed-point combinators and continuous lambda models, in: *Formal Description of Programming Concepts III*, ed. Wirsing, M. pp. 425-448. Amsterdam: North Holland (1987)

[14] Engeler, E. Algebras and combinators, *Algebra Universalis* **13**, no. 3, pp. 389-392. (1981)

[15] Giannini, P., Ronchi della Rocca, S. and Honsell, F. Type Inference, some results, some problems. To appear in *Fundamentae Informaticae*

[16] Girard, J.Y. The system F of variable types, fifteen years later. *Theor. Comp. Sci.* **22**, pp. 1-17 (1983)

[17] Hindley, R. The simple semantics for Coppo-Dezani-Sallé types. *Lecture Notes in Computer Science* **137** pp. 212-226. Berlin: Springer Verlag (1982)

[18] Hindley, R. The completeness theorem for typing lambda terms. *Theor. Comp. Sci.* **22**, pp. 1-17 (1983).

[19] Hindley, R. Curry's type rules are complete with respect to the F-semantics too. *Theor. Comput. Sci.* **22**, pp. 127-133 (1983).

[20] Hindley, R. Types with intersection, an introduction. To appear in *Formal Aspects of Computing.*

[21] Jacobs, B., Margaria, I. and Zacchi, M. Expansion and conversion models in the lambda calculus from filters with polymorphic types. Unpublished manuscript.

[22] Jacobs, B., Margaria, I. and Zacchi, M. Filter models with polymorphic types. *Theor. Comp. Sci.* **95** pp. 143-158 (1992)

[23] Jay, C. B. Modelling reduction in confluent categories, in: Proceedings of the Durham Symposium on Applications of Categories in Computer Science, eds. Fourman, M.P., Johnstone, P.T. and Pitts, A.M. *LMS Lecture Notes Series* **177** pp. 143-162. Cambridge: Cambridge University Press (1992)

[24] MacQueen, D., Plotkin, G. and Sethi, R. An ideal model for recursive polymorphic types. *Information and Control* **71** pp. 95-130 (1986)

[25] Meseguer, J. Conditional rewriting logic as a unified model of concurrency. *Theor. Comp. Sci.* **96** pp. 73-155 (1992)

[26] Meyer, R.K., Bunder, M.A. and Powers, L. Implementing the Fool's model of combinatory logic. To appear in special issue, *Journal of Automated Reasoning*.

[27] Milner, R. A theory of type polymorphism in programming. *J. Comput. System Sci.* **17**, pp. 348-375 (1985).

[28] Mitchell, J.C. Polymorphic type inference and containment. *Information and Computation* **76**, pp. 211-249 (1988).

[29] Mitchell, J.C. A type-inference approach to reduction properties and semantics of polymorphic expressions (Summary) in *Logical Foundations of Functional Programming*, ed. Huet, G. pp. 195-212. Reading: Addison-Wesley (1990).

[30] Plotkin, G.D. A set-theoretical definition of application. Research Memorandum MIP-R-95, 32 pp. Dept. of Machine Intelligence and Perception, University of Edinburgh (1972).

[31] Plotkin, G.D. A semantics for type checking, in: Theoretical Aspects of Computer Software, eds. Ito, T. and Meyer, A.R. *Lecture Notes in Computer Science* **526** pp. 1-17. Berlin: Springer Verlag (1991).

[32] Rydeheard, D.E. and Stell, J.G. Foundations of equational deduction: a categorical treatment of equational proofs and unification algorithms, in: Category theory and Computer Science, eds. Pitt, A. *et al.* *Lecture Notes in Computer Science* **283** pp. 114-139. Berlin: Springer Verlag (1987).

[33] Seely, R.A.G. Modelling computations a 2-categorical framework, in: *Proceedings of the Second Annual IEEE Symposium on Logic in Computer Science*, pp. 65-71. Los Alamitos: Computer Society Press of the IEEE (1987).

[34] Tiuryn, J. Type inference problems: a survey, in: Proceedings, Mathematical Foundations of Computer Science, ed. Rovan, B. *Lecture Notes in Computer Science* **452** pp. 105-120. Berlin: Springer Verlag (1990).

[35] van Bakel, S. Complete restrictions of the intersection type discipline. To appear in *Theor. Comp. Sci.*

[36] Yokouchi, H. Embedding second order type system into intersection type system. To appear in *Information and Computation*.