Subject Reduction and Minimal Types for Higher Order Subtyping

Adriana Compagnoni

abc@dcs.ed.ac.uk

Department of Computer Science, University of Edinburgh
The King's Buildings, Edinburgh, EH9 3JZ, United Kingdom
Tel: (+44) (131) 650-1000 Fax: (+44) (131) 667-7209

Abstract

We define the typed lambda calculus $F^{\omega}_{\wedge}$, a natural generalization of Girard's system F^{ω} with intersection types and bounded polymorphism. A novel aspect of our presentation is the use of term rewriting techniques to present intersection types, which clearly splits the computational semantics (reduction rules) from the syntax (inference rules) of the system. We establish properties such as Church-Rosser for the reduction relation on types and terms, and Strong Normalization for the reduction on types. We prove that types are preserved by computation (Subject Reduction property), and that the system satisfies the Minimal Types property. On the way to establishing these results, we define algorithms for type inference and subtype checking.

1 Introduction

The formal study of subtyping in programming languages was begun by Reynolds [36] and Cardelli [10], who used a lambda-calculus with subtyping to model the refinement of interfaces in object oriented languages. This led to a considerable body of work, covering an increasing range of object-oriented features by combining subtyping with other type-theoretic constructs, including polymorphic functions [15, 27, 7], records with update and extension operators [10, 14], recursive types [2, 8], and higher-order polymorphism [11, 13, 12, 33].

Type systems with subtyping have also arisen from the study of lambda-calculi with intersection types at the University of Torino [26, 6]. Most of this work has been carried out in the setting of pure lambda-calculi, but it has also been applied to programming language design by Reynolds [37]. Some work has begun on combining intersections with other typing features [34, 17].

The system $F^\omega_\wedge$ (F-omega-meet) was first introduced in [23], where it was shown to be rich enough to provide a typed model of object oriented programming with multiple inheritance. $F^\omega_\wedge$ is an extension of F^ω [30] with bounded quantification and intersection types, which can be seen as a natural generalization of the type disciplines present in the current literature, for example in [27, 34, 35, 18]. Systems including either subtyping or intersection types or both have been widely studied for many years. What follows is not intended to be an exhaustive description, but a framework for the present work.

First-order type disciplines with intersection types have been investigated by the group in Torino [25, 6] and elsewhere (see [16] for background and further references). A second-order λ-calculus with intersection types was studied in [34]. Systems including subtyping were present in [15, 10]. Higher order generalizations of subtyping appear in [9, 24, 33, 8]. $F_\leq$, a second-order λ-calculus with bounded quantification, was studied in [29], and in [34].

Because $F^\omega_\wedge$ has reduction on types, we introduce a conversion rule that includes inter-convertible types in the subtype relation. Therefore, our subtyping relation relates types of a more expressive type system than that presented in [18]. In fact, treating the interaction between interface refinement and encapsulation of objects in object oriented programming has required higher-order generalizations of subtyping: the F-bounded quantification of Canning, Cook, Hill, Olthoff and Mitchell [9] or system $F^\omega_\leq$ [11, 13, 12, 33, 8].

We present a definition of $F^\omega_\wedge$ that differs from the one introduced in [23] in two ways. First, the ill-behaved Castagna and Pierce's quantifier rule has been replaced by Cardelli and Wegner's kernel *Fun* rule. Secondly, we introduce a richer notion of reduction on types, and thereby the four distributivity rules become particular cases of the conversion rule. This new reduction is shown to be confluent and strongly normalizing. The latter simplification was motivated by structural properties of the former presentation.

This new perspective suggests that to study the subtyping relation it is enough to concentrate on types in normal form. Note that the solution cannot be as simple as to restrict the subtyping rules of $F^\omega_\wedge$ to handle only types in normal form and replace conversion by reflexivity. The following is a good example of the problem to be solved. Consider the context $\Gamma \equiv W{:}K, X \leq \Lambda Y{:}K.Y{:}K\to K,\ Z \leq X{:}K\to K$; observe that X and Z are subtypes of the identity on K. Then $\Gamma \vdash X(Z\,W) \leq W$ is not derivable without using conversion, i.e. without performing any β-reduction, even when the conclusion is in normal form. (For a derivation see section 6.1.)

The subtyping rules of $F^\omega_\wedge$ are not syntax directed, in the sense that the form of a derivable subtyping statement does not uniquely determine the last rule of its derivation, i.e. there might be more than one derivation of the same subtyping judgement. To develop a deterministic decision procedure to check subtyping, we need a new presentation of the subtyping relation that provides the foundations for a subtype-checking deterministic algorithm.

We develop a *normal subtyping system*, $NF^\omega_\wedge$, in which only types in normal form are considered. We prove that derivations in $NF^\omega_\wedge$ can be normalized by eliminating transitivity and simplifying reflexivity. This simplification yields an algorithmic presentation, $AlgF^\omega_\wedge$. Moreover, we prove that $AlgF^\omega_\wedge$ is indeed an alternative presentation of the $F^\omega_\wedge$ subtyping relation, that is $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$ (proposition 9.2).

In [38] Steffen and Pierce studied $F^{\omega}_{\leq}$ proving that typing is decidable and that the system satisfies the minimal types property. A central result in the proof of decidability is establishing the decidability of subtyping, a result first proved in [20]. There are several differences between our work and theirs. Our results are for a stronger system which also includes intersection types. A major difference is the choice of the intermediate subtyping system. We define the normal system $NF^{\omega}_{\wedge}$ which provides a generation principle for subtyping, yielding the algorithm $AlgF^{\omega}_{\wedge}$. In [38] the intermediate system, called a reducing system, leads to a much more complicated proof which involves dealing with several notions of reduction and further reformulation of the intermediate system. A generation principle for subtyping is crucial to prove the Subject Reduction property (proposition 12.7), which is not proved in [38].

1.1 Results

- We define the typed lambda calculus $F^{\omega}_{\wedge}$, a natural generalization of Girard's system F^{ω} with intersection types and bounded polymorphism. A novel aspect of our presentation is the use of term rewriting techniques to present intersection types, which clearly splits the computational semantics (reduction rules) from the syntax (inference rules) of the system.

- The reduction rules of $F^{\omega}_{\wedge}$ can be divided into two main groups, reductions on types ($\rightarrow_{\beta\wedge}$) and reductions on terms ($\rightarrow_{\beta\text{fors}}$). Although confluence is not a modular property in general, in our case it is possible to provide a modular proof of it. In section 3, we combine the independent proofs of confluence for reductions on types and confluence for reduction on terms to yield a proof of confluence of the reduction relation in the whole system.

- We prove the strong normalization property of $\rightarrow_{\beta\wedge}$ on well-formed types.

- We define a normalized system $NF^{\omega}_{\wedge}$ equivalent to the original presentation of subtyping, and prove the transitivity elimination and reflexivity simplification properties.

- We define a subtyping algorithm $AlgF^{\omega}_{\wedge}$, and prove that it is equivalent to the original presentation.

- In section 10, we prove that $F^{\omega}_{\wedge}$ satisfies the minimal types property, and we provide an algorithm for computing minimal types.

- We prove that $F^{\omega}_{\wedge}$ satisfies the subject reduction property using the minimal types property.

The original paper [20] defines the system $F^{\omega}_{\wedge}$ and its equivalent normal subtyping system $NF^{\omega}_{\wedge}$. In the current paper we extend this framework to prove Subject Reduction and Minimal Typing.

2 Syntax of $F_\wedge^\omega$

We now present the rules for kinding, subtyping, and typing in $F_\wedge^\omega$. They are organized as proof systems for four interdependent judgement forms:

$\Gamma \vdash \text{ok}$	well-formed context
$\Gamma \vdash T : K$	well-kinded type
$\Gamma \vdash S \leq T$	subtype
$\Gamma \vdash e : T$	well-typed term.

We sometimes use the metavariable Σ to range over statements (right-hand sides of judgements) of any of these four forms.

2.1 Syntactic Categories

The *kinds* of $F_\wedge^\omega$ are those of F^ω: the kind $\star$ of proper types and the kinds $K_1 \rightarrow K_2$ of functions on types (sometimes called type operators).

$$\begin{array}{lcll} \mathbb{K} & ::= & \star & \text{types} \\ & | & \mathbb{K} \rightarrow \mathbb{K} & \text{type operators} \end{array}$$

The language of *types* of $F_\wedge^\omega$ is a straightforward higher-order extension of $F_\leq$, Cardelli and Wegner's second-order calculus of bounded quantification. Like $F_\leq$, it includes type variables (written X), function types ($T \rightarrow T'$), and polymorphic types ($\forall X \leq T{:}K.T'$), in which the bound type variable X ranges over all subtypes of the upper bound T. Moreover, like F^ω, we allow types to be abstracted on types ($\Lambda X{:}K.T$) and applied to argument types ($T\,T'$); in effect, these forms introduce a simply typed λ-calculus at the level of types. Finally, we allow arbitrary finite intersections ($\bigwedge^K [T_1..T_n]$), where all the T_i's are members of the same kind K.

$$\begin{array}{lcll} \mathbb{T} & ::= & X & \text{type variable} \\ & | & \mathbb{T} \rightarrow \mathbb{T} & \text{function type} \\ & | & \forall X \leq \mathbb{T}{:}\mathbb{K}.\mathbb{T} & \text{polymorphic type} \\ & | & \Lambda X{:}\mathbb{K}.\mathbb{T} & \text{operator abstraction} \\ & | & \mathbb{T}\,\mathbb{T} & \text{operator application} \\ & | & \bigwedge^{\mathbb{K}}[\mathbb{T}..\mathbb{T}] & \text{intersection at kind } \mathbb{K} \end{array}$$

We use the abbreviation $\top^K$ for nullary intersections and sometimes $X{:}K$ for $X \leq \top^K{:}K$.

$$\top^K \equiv \bigwedge^K[\,] \qquad X{:}K \equiv X \leq \top^K{:}K$$

We drop the maximal type *Top* of $F_\leq$, since its role is played here by the empty intersection $\top^\star$. For technical convenience, we provide kind annotations on bound variables and intersections so that every type has an "obvious kind," which can be read off directly from its structure and the kind declarations in the context.

The language of terms includes the variables (x), applications ($e\,e$), and functional abstractions ($\lambda x{:}T.e$) of the simply typed λ-calculus, plus the type abstraction ($\lambda X{\leq}T{:}K.e$) and application ($e\,T$) of F^ω. As in $F_\leq$, each type variable is given an upper bound at the point where it is introduced.

Intersection types are introduced by expressions of the form "for($X{\in}T_1..T_n$)e", which can be read as instructions to the type-checker to analyze the expression e separately under the assumptions $X \equiv T_1$, $X \equiv T_2$, $\ldots$, $X \equiv T_n$ and conjoin the results. For example, if $+ : \mathrm{Int}{\to}\mathrm{Int}{\to}\mathrm{Int} \wedge \mathrm{Real}{\to}\mathrm{Real}{\to}\mathrm{Real}$, then we can derive:

$$\mathrm{for}(X{\in}\mathrm{Int},\ \mathrm{Real})\lambda x{:}X.x + x \quad : \quad \mathrm{Int}{\to}\mathrm{Int} \wedge \mathrm{Real}{\to}\mathrm{Real}.$$

$$\begin{array}{rcll} e & ::= & x & \text{variable} \\ & | & \lambda x{:}\mathbb{T}.e & \text{abstraction} \\ & | & e\,e & \text{application} \\ & | & \lambda X{\leq}\mathbb{T}{:}\mathbb{K}.e & \text{type abstraction} \\ & | & e\,\mathbb{T} & \text{type application} \\ & | & \mathrm{for}(X{\in}\mathbb{T}..\mathbb{T})e & \text{alternation} \end{array}$$

The operational semantics of $F^\omega_\wedge$ is given by the following reduction rules on types and terms.

Definition 2.1.1 (*Reduction rules for types*)

1. $(\Lambda X{:}K.T_1)T_2 \to_{\beta\wedge} T_1[X{\leftarrow}T_2]$
2. $S \to \bigwedge^\star[T_1..T_n] \to_{\beta\wedge} \bigwedge^\star[S{\to}T_1 \,..\, S{\to}T_n]$
3. $\forall X{\leq}S{:}K.\bigwedge^\star[T_1..T_n] \to_{\beta\wedge} \bigwedge^\star[\forall X{\leq}S{:}K.T_1 \,..\, \forall X{\leq}S{:}K.T_n]$
4. $\Lambda X{:}K_1.\bigwedge^{K_2}[T_1..T_n] \to_{\beta\wedge} \bigwedge^{K_1\to K_2}[\Lambda X{:}K_1.T_1 \,..\, \Lambda X{:}K_1.T_n]$
5. $(\bigwedge^{K_1\to K_2}[T_1..T_n])\,U \to_{\beta\wedge} \bigwedge^{K_2}[T_1\,U \,..\, T_n\,U]$
6. $\bigwedge^K[T_1 \,..\, \bigwedge^K[S_1..S_n] \,..\, T_m] \to_{\beta\wedge} \bigwedge^K[T_1 \,..\, S_1..S_n \,..\, T_m]$

The first rule is the usual β-reduction rule for types. Rules 2 through 5 express the fact that intersections in positive positions distribute with respect to the other type constructors. Rule 6 states that intersection is an associative operator. In section 5 we consider the reduction defined by rules 1 through 5 as $\to_{\beta\wedge^-}$ and the one defined by 6 as $\to_a$ (a comes from associativity). The left-hand side of each reduction rule is a *redex* and the right-hand side its *reduct*. The relation $\to_{\beta\wedge}$ is extended so as to become a compatible relation with respect to type formation, $\twoheadrightarrow_{\beta\wedge}$ is the transitive and reflexive closure of $\to_{\beta\wedge}$, and $=_{\beta\wedge}$ is the least equivalence relation containing $\rightarrow_{\beta\wedge}$. The capture-avoiding substitution of S for X in T is written $T[X{\leftarrow}S]$. Substitution is written similarly for terms, and is extended point-wise to contexts. The $\beta\wedge$-normal form of a type S is written S^{nf}, and is extended point-wise to contexts.

Definition 2.1.2 (*Reduction rules for terms*)

1. $(\lambda x{:}T_1.e_1)e_2 \rightarrow_{\beta\mathrm{fors}} e_1[x{\leftarrow}e_2]$
2. $(\lambda X{\leq}T_1{:}K_1.e)T \rightarrow_{\beta\mathrm{fors}} e[X{\leftarrow}T]$
3. $(\mathrm{for}(X{\in}T_1..T_n)e_1)e_2 \rightarrow_{\beta\mathrm{fors}} \mathrm{for}(X{\in}T_1..T_n)(e_1\, e_2)$
4. $\mathrm{for}(X{\in}T_1..T_n)e \rightarrow_{\beta\mathrm{fors}} e$, if $X \notin \mathrm{FV}(e)$

Rules 1 and 2 are the β-reductions on terms. Rule 3 says that the for constructor can be pushed to the outermost level. We consider the reduction defined by rules 1 through 3 as $\rightarrow_{\beta\mathrm{for}}$ and the one defined by 4 as $\rightarrow_{\mathrm{s}}$ (s comes from simplification). The left-hand side of each reduction rule is a *redex* and the right-hand side its *reduct*. The relation $\rightarrow_{\beta\mathrm{fors}}$ is extended so as to become a compatible relation with respect to term formation, $\twoheadrightarrow_{\beta\mathrm{fors}}$ is the transitive reflexive closure of $\rightarrow_{\beta\mathrm{fors}}$, and $=_{\beta\mathrm{fors}}$ is the least equivalence relation containing $\rightarrow_{\beta\mathrm{fors}}$.

2.2 Contexts

A *context* Γ is a finite sequence of typing and subtyping assumptions for a set of term and type variables.

The empty context is written $\emptyset$. Term variable bindings have the form $x{:}T$; type variable bindings have the form $X{\leq}T{:}K$, where T is the upper bound of X and K is the kind of T.

Γ	$::=$	$\emptyset$	empty context
	$\mid$	$\Gamma,\ x{:}T$	term variable declaration
	$\mid$	$\Gamma,\ X{\leq}T{:}K$	type variable declaration

When writing nonempty contexts, we omit the initial $\emptyset$. The domain of Γ is written $\mathrm{dom}(\Gamma)$. The functions $\mathrm{FV}(\text{—})$ and $\mathrm{FTV}(\text{—})$ give the sets of free term variables and free type variables of a term, type, or context. Since we are careful to ensure that no variable is bound more than once, we sometimes abuse notation and consider contexts as finite functions: $\Gamma(X)$ yields the bound of X in Γ, where X is implicitly asserted to be in $\mathrm{dom}(\Gamma)$.

Types, terms, contexts, statements, and derivations that differ only in the names of bound variables are considered identical. The underlying idea is that variables are de Bruijn indexes [28].

DEFINITION 2.2.1 (*Closed*)

1. A term e is closed with respect to a context Γ if $\mathrm{FV}(e) \cup \mathrm{FTV}(e) \subseteq \mathrm{dom}(\Gamma)$.

2. A type T is closed with respect to a context Γ if $\mathrm{FTV}(T) \subseteq \mathrm{dom}(\Gamma)$.

3. A typing statement $\Gamma \vdash e : T$ is closed if e and T are closed with respect to Γ.

4. A kinding statement $\Gamma \vdash T : K$ is closed if T is closed with respect to Γ.

5. A subtyping statement $\Gamma \vdash S \leq T$ is closed if S and T are closed with respect to Γ.

We consider only closed typing statements. Observe that in the limit case of the rule T-MEET, when $n = 0$, not having the closure convention would allow nonsensical terms to be typed. On the other hand, the free variable lemma (lemma 4.3) guarantees that kinding statements are closed and the well-kindedness of subtyping (lemma 4.18) ensures that subtyping statements are closed as well.

2.3 Context Formation

The rules for well-formed contexts are the usual ones: a start rule for the empty context and rules allowing a given well-formed context to be extended with either a term variable binding or a type variable binding.

$$\emptyset \vdash \mathrm{ok} \qquad \text{(C-EMPTY)}$$

$$\frac{\Gamma \vdash T : \star \qquad x \notin \mathrm{dom}(\Gamma)}{\Gamma,\, x{:}T \vdash \mathrm{ok}} \qquad \text{(C-VAR)}$$

$$\frac{\Gamma \vdash T : K \qquad X \notin \mathrm{dom}(\Gamma)}{\Gamma,\, X{\leq}T{:}K \vdash \mathrm{ok}} \qquad \text{(C-TVAR)}$$

2.4 Type Formation

For each type constructor, we give a rule specifying how it can be used to build well-formed type expressions. The critical rules are K-OABS and K-OAPP, which form type abstractions and type applications (essentially as in a simply typed λ-calculus).

The well-formedness premise $\Gamma \vdash \mathrm{ok}$ in K-MEET (and in T-MEET below) is required for the case where $n = 0$.

$$\frac{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash \mathrm{ok}}{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash X : K} \qquad \text{(K-TVAR)}$$

$$\frac{\Gamma \vdash T_1 : \star \qquad \Gamma \vdash T_2 : \star}{\Gamma \vdash T_1 {\rightarrow} T_2 : \star} \qquad \text{(K-ARROW)}$$

$$\frac{\Gamma,\, X{\leq}T_1{:}K_1 \vdash T_2 : \star}{\Gamma \vdash \forall X{\leq}T_1{:}K_1.T_2 : \star} \qquad \text{(K-ALL)}$$

$$\frac{\Gamma,\, X{:}K_1 \vdash T_2 : K_2}{\Gamma \vdash \Lambda X{:}K_1.T_2 : K_1 \to K_2} \quad \text{(K-OAbs)}$$

$$\frac{\Gamma \vdash S : K_1 \to K_2 \qquad \Gamma \vdash T : K_1}{\Gamma \vdash S\,T : K_2} \quad \text{(K-OApp)}$$

$$\frac{\Gamma \vdash \text{ok} \qquad \text{for each } i \in \{1..n\}\,,\ \Gamma \vdash T_i : K}{\Gamma \vdash \bigwedge^K [T_1..T_n] : K} \quad \text{(K-Meet)}$$

2.5 Subtyping

The rules defining the subtype relation are a natural extension of familiar calculi of bounded quantification. Aside from some extra well-formedness conditions, the rules S-Trans, S-TVar, and S-Arrow are the same as in the usual, second-order case. Rules S-OAbs and S-OApp extend the subtype relation point-wise to kinds other than $\star$. The rule of type conversion in F^ω, that is, if $\Gamma \vdash e : T$ and $T =_\beta T'$ then $\Gamma \vdash e : T'$, is captured here as the subtyping rule S-Conv, which also gives reflexivity as a special case. The rule S-All is the rule of Cardelli and Wegner's *Fun* language [15] in which the bounds of the quantifiers are equal. Rules S-Meet-G and S-Meet-LB specify that an intersection of a set of types is the set's order-theoretic greatest lower bound.

$$\frac{\Gamma \vdash S : K \qquad \Gamma \vdash T : K \qquad S =_{\beta\wedge} T}{\Gamma \vdash S \leq T} \quad \text{(S-Conv)}$$

$$\frac{\Gamma \vdash S \leq T \qquad \Gamma \vdash T \leq U}{\Gamma \vdash S \leq U} \quad \text{(S-Trans)}$$

$$\frac{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash \text{ok}}{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash X \leq T} \quad \text{(S-TVar)}$$

$$\frac{\Gamma \vdash T_1 \leq S_1 \qquad \Gamma \vdash S_2 \leq T_2 \qquad \Gamma \vdash S_1 \to S_2 : \star}{\Gamma \vdash S_1 \to S_2 \leq T_1 \to T_2} \quad \text{(S-Arrow)}$$

$$\frac{\Gamma,\, X{\leq}U{:}K \vdash S \leq T \qquad \Gamma \vdash \forall X{\leq}U{:}K.S : \star}{\Gamma \vdash \forall X{\leq}U{:}K.S \leq \forall X{\leq}U{:}K.T} \quad \text{(S-All)}$$

$$\frac{\Gamma,\, X{:}K \vdash S \leq T}{\Gamma \vdash \Lambda X{:}K.S \leq \Lambda X{:}K.T} \quad \text{(S-OAbs)}$$

$$\frac{\Gamma \vdash S \leq T \qquad \Gamma \vdash S\,U : K}{\Gamma \vdash S\,U \leq T\,U} \quad \text{(S-OApp)}$$

$$\frac{\text{for each } i \in \{1..n\}\,,\ \Gamma \vdash S \leq T_i \qquad \Gamma \vdash S : K}{\Gamma \vdash S \leq \bigwedge^K [T_1..T_n]} \quad \text{(S-Meet-G)}$$

$$\frac{\Gamma \vdash \bigwedge^K [T_1..T_n] : K}{\Gamma \vdash \bigwedge^K [T_1..T_n] \leq T_i} \quad \text{(S-Meet-LB)}$$

2.6 Term Formation

Except for T-Meet and T-For, the term formation rules are precisely those of the second-order calculus of bounded quantification. T-For provides for type checking under any of a set of alternate assumptions. For each S_i, the type derived for the instance of the body e when X is replaced by S_i is a valid type of the for expression itself. The T-Meet rule can then be used to collect these separate typings into a single intersection. Type-theoretically, T-Meet is the introduction rule for the $\wedge$ constructor; the corresponding elimination rule need not be given explicitly, since it follows from T-Subsumption and S-Meet-LB.

$$\frac{\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \mathrm{ok}}{\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash x : T} \qquad \text{(T-Var)}$$

$$\frac{\Gamma,\, x{:}T_1 \vdash e : T_2}{\Gamma \vdash \lambda x{:}T_1.e : T_1 \to T_2} \qquad \text{(T-Abs)}$$

$$\frac{\Gamma \vdash f : T_1 \to T_2 \qquad \Gamma \vdash a : T_1}{\Gamma \vdash f\, a : T_2} \qquad \text{(T-App)}$$

$$\frac{\Gamma,\, X {\leq} T_1{:}K_1 \vdash e : T_2}{\Gamma \vdash \lambda X {\leq} T_1{:}K_1.e : \forall X {\leq} T_1{:}K_1.T_2} \qquad \text{(T-TAbs)}$$

$$\frac{\Gamma \vdash f : \forall X {\leq} T_1{:}K_1.T_2 \qquad \Gamma \vdash S \leq T_1}{\Gamma \vdash f\, S : T_2[X {\leftarrow} S]} \qquad \text{(T-TApp)}$$

$$\frac{\Gamma \vdash e[X {\leftarrow} S] : T \qquad S : \{S_1..S_n\}}{\Gamma \vdash \mathrm{for}(X {\in} S_1..S_n)e : T} \qquad \text{(T-For)}$$

$$\frac{\Gamma \vdash \mathrm{ok} \qquad \text{for each } i \in \{1..n\}\,,\ \Gamma \vdash e : T_i}{\Gamma \vdash e : \bigwedge^{\star}[T_1..T_n]} \qquad \text{(T-Meet)}$$

$$\frac{\Gamma \vdash e : S \qquad \Gamma \vdash S \leq T}{\Gamma \vdash e : T} \qquad \text{(T-Subsumption)}$$

Most of the rules include premises which have two rather different sorts: *structural premises*, which play an essential role in giving the rule its intended semantic force, and *well-formation premises*, which ensure that the entities named in the rule are of the expected sorts.

We sometimes omit well-formation premises that can be derived from others. For example, in the rule S-Arrow, we drop the premise $\Gamma \vdash T_1 \to T_2 : \star$, since it follows from $\Gamma \vdash S_1 \to S_2 : \star$ using the properties proved in section 4.

2.7 Discussion

An equivalent presentation of intersection types uses binary intersections as in [25]. The intersection of S and T is then written $S \wedge T$, and there is a maximal element at each kind, ω^K. The rules of the system have to be modified according to this alternative notation. In most cases, each of our rules about intersection types has to be replaced by two rules, one for the binary case and another for the maximal element. For example, the reduction rule

$$\forall X {\leq} S{:}K.\bigwedge^{\star}[T_1..T_n] \to_{\beta\wedge} \bigwedge^{\star}[\forall X {\leq} S{:}K.T_1 \;..\; \forall X {\leq} S{:}K.T_n]$$

is replaced by

$$\begin{array}{lll} \forall X{\leq}S{:}K.T_1 \wedge T_2 & \rightarrow_{\beta\wedge} & \forall X{\leq}S{:}K.T_1 \wedge \forall X{\leq}S{:}K.T_2 \quad \text{and} \\ \forall X{\leq}S{:}K.\omega^\star & \rightarrow_{\beta\wedge} & \omega^\star. \end{array}$$

Similar replacement takes place for rules 3 through 5 in definition 2.1.1. The term formation rule K-MEET is replaced by the two following rules.

$$\frac{\Gamma \vdash S : K \qquad \Gamma \vdash T : K}{\Gamma \vdash S \wedge T : K} \tag{K-INT}$$

$$\frac{\Gamma \vdash \text{ok}}{\Gamma \vdash \omega^K : K} \tag{K-MAX}$$

The rule S-MEET-G is replaced by the following two rules.

$$\frac{\Gamma \vdash S \leq T_1 \qquad \Gamma \vdash S \leq T_2}{\Gamma \vdash S \leq T_1 \wedge T_2} \tag{S-INT-G}$$

$$\frac{\Gamma \vdash S : K}{\Gamma \vdash S \leq \omega^K} \tag{S-MAX}$$

In the λ-cube [4], F^ωcorresponds to λ_ω, the system defined by the rules $(\star, \star)$, $(\Box, \star)$, and $(\Box, \Box)$. If K is a kind defined by the grammar $\mathbb{K}$, then

$$\Gamma \vdash_{\lambda_\omega} K : \Box.$$

The rule $(\Box, \Box)$ corresponds to the recursive step in the definition of $\mathbb{K}$; the rule $(\star, \star)$ corresponds to K-ARROW, and K-ALL is the parallel of rule $(\Box, \star)$ enriched with subtyping.

3 Confluence

In this section, we show that the system $F^\omega_\wedge$ is confluent. By that we mean that the reduction $\rightarrow_{\beta\text{fors}} \cup \rightarrow_{\beta\wedge}$ defined by putting together the reduction on terms, $\rightarrow_{\beta\text{fors}}$ (definition 2.1.2), and the reduction on types, $\rightarrow_{\beta\wedge}$ (definition 2.1.1), satisfies the Church-Rosser property. We use the Hindley-Rosen lemma (c.f. 3.3.5 [5]) to establish this result. This factors the proof into two parts:

1. proving that the reductions $\twoheadrightarrow_{\beta\text{fors}}$ and $\twoheadrightarrow_{\beta\wedge}$ commute, and

2. proving that the reductions $\rightarrow_{\beta\text{fors}}$ and $\rightarrow_{\beta\wedge}$ satisfy the Church-Rosser property.

Full details of the proofs of this section as well as intermediate results can be found in [22]. Remember that two binary relations $\rightarrow_1$ and $\rightarrow_2$ commute if given $A \rightarrow_1 B$ and $A \rightarrow_2 C$, there exists D such that $C \rightarrow_1 D$ and $B \rightarrow_2 D$. In order to prove that $\twoheadrightarrow_{\beta\text{fors}}$ and $\twoheadrightarrow_{\beta\wedge}$ commute we use the following lemma.

LEMMA 3.1 (3.3.6 [5]) Let $\rightarrow_1$ and $\rightarrow_2$ be two binary relations on a set X. Suppose that if $A \rightarrow_1 B$ and $A \rightarrow_2 C$, there exists D such that $C \rightarrow_{=1} D$ and $B \twoheadrightarrow_2 D$, where $\rightarrow_{=1}$ is the reflexive closure of $\rightarrow_1$. Hence $\twoheadrightarrow_1$ and $\twoheadrightarrow_2$ commute.

LEMMA 3.2 If $A \rightarrow_{\beta\mathrm{fors}} B$ and $A \rightarrow_{\beta\wedge} C$, there exists D such that $C \rightarrow_{=\beta\mathrm{fors}} D$ and $B \twoheadrightarrow_{\beta\wedge} D$

PROOF: By induction on the structure of E. □

COROLLARY 3.3 $\twoheadrightarrow_{\beta\wedge}$ and $\twoheadrightarrow_{\beta\mathrm{fors}}$ commute.

The Church-Rosser theorem for $\rightarrow_{\beta\wedge}$

We now prove the Church-Rosser property for the reduction defined in 2.1.1. The strategy we use here is similar to the one used in chapter 11 section 1 of [5] to prove the corresponding result for $\rightarrow_\beta$ in the type-free λ-calculus.

In order to prove the Church-Rosser property for $\rightarrow_{\beta\wedge}$ it is sufficient to show the following *strip* lemma.

LEMMA 3.4 (*Strip*) Let S, T_1, and $T_2 \in \mathbb{T}$. If $S \rightarrow_{\beta\wedge} T_1$ and $S \twoheadrightarrow_{\beta\wedge} T_2$, then there exists $T_3 \in \mathbb{T}$ such that $T_1 \twoheadrightarrow_{\beta\wedge} T_3$ and $T_2 \twoheadrightarrow_{\beta\wedge} T_3$.

The idea of the proof is as follows. Let T_1 be the result of replacing the redex R in S by its reduct R'. If we keep track of what happens with R during the reduction $S \twoheadrightarrow_{\beta\wedge} T_2$, then we can find T_3. To be able to trace R we define a new set of terms $\underline{\mathbb{T}}$ where redexes can appear underlined. Consequently, if we underline R in S we only need to reduce all occurrences of the underlined R in T_2 to obtain T_3.

THEOREM 3.5 (*Church-Rosser for* $\rightarrow_{\beta\wedge}$)

If S, T_1, and $T_2 \in \mathbb{T}$ are such that $S \twoheadrightarrow_{\beta\wedge} T_1$ and $S \twoheadrightarrow_{\beta\wedge} T_2$, then there exists $T_3 \in \mathbb{T}$ such that $T_1 \twoheadrightarrow_{\beta\wedge} T_3$ and $T_2 \twoheadrightarrow_{\beta\wedge} T_3$.

PROOF: By induction on the generation of $S \twoheadrightarrow_{\beta\wedge} T_1$. □

The Church-Rosser theorem for $\rightarrow_{\beta\mathrm{fors}}$

Next we prove the Church-Rosser property for the reduction defined in definition 2.1.2.

THEOREM 3.6 (*Church-Rosser for* $\rightarrow_{\beta\mathrm{fors}}$)

Let $e, f_1, f_2 \in \mathbb{E}$. If $e \twoheadrightarrow_{\beta\mathrm{fors}} f_1$ and $e \twoheadrightarrow_{\beta\mathrm{fors}} f_2$, then there exists $f_3 \in \mathbb{E}$ such that $f_1 \twoheadrightarrow_{\beta\mathrm{fors}} f_3$ and $f_2 \twoheadrightarrow_{\beta\mathrm{fors}} f_3$.

The idea of the proof is as follows. We prove that $\rightarrow_{\beta\mathrm{for}}$ and $\rightarrow_{\mathrm{s}}$ are Church-Rosser (theorem 3.7 and lemma 3.8); that $\rightarrow_{\mathrm{s}}$ reduction steps can be postponed (lemma 3.9), and that $\twoheadrightarrow_{\beta\mathrm{for}}$ and $\twoheadrightarrow_{\mathrm{s}}$ commute (lemma 3.10).

Those four results allow us to prove the Church-Rosser theorem for $\rightarrow_{\beta\mathrm{fors}}$. Let e, e_1, $e_2 \in \mathbb{E}$, such that $e \twoheadrightarrow_{\beta\mathrm{fors}} e_1$ and $e \twoheadrightarrow_{\beta\mathrm{fors}} e_2$. Then, by s-postponement, there exist f_1 and f_2; by Church-Rosser for $\rightarrow_{\beta\mathrm{for}}$, there exists f_3; and, by lemma 3.10, there exist f_4 and f_5, and finally, by Church-Rosser for $\rightarrow_{\mathrm{s}}$, there exists e_3 which completes the following diagram.

$$
\begin{tikzcd}
e \arrow[r, two heads, "\beta\mathrm{for}"'] \arrow[d, two heads, "\beta\mathrm{for}"'] & f_1 \arrow[r, two heads, "\mathrm{s}"'] \arrow[d, two heads, dotted, "\beta\mathrm{for}"] & e_1 \arrow[d, two heads, dotted, "\beta\mathrm{for}"] \\
f_2 \arrow[r, two heads, dotted, "\beta\mathrm{for}"'] \arrow[d, two heads, "\mathrm{s}"'] & f_3 \arrow[r, two heads, dotted, "\mathrm{s}"'] \arrow[d, two heads, dotted, "\mathrm{s}"] & f_4 \arrow[d, two heads, dotted, "\mathrm{s}"] \\
e_2 \arrow[r, two heads, dotted, "\beta\mathrm{for}"'] & f_5 \arrow[r, two heads, dotted, "\mathrm{s}"'] & e_3
\end{tikzcd}
$$

The Church-Rosser property for $\rightarrow_{\beta\mathrm{for}}$ follows using the same strategy used to prove theorem 3.5.

THEOREM 3.7 (*Church-Rosser for* $\rightarrow_{\beta\mathrm{for}}$)
If e, f_1, and $f_2 \in \mathbb{E}$ are such that $e \twoheadrightarrow_{\beta\mathrm{for}} f_1$ and $e \twoheadrightarrow_{\beta\mathrm{for}} f_2$, then there exists $f_3 \in \mathbb{E}$ such that $f_1 \twoheadrightarrow_{\beta\mathrm{for}} f_3$ and $f_2 \twoheadrightarrow_{\beta\mathrm{for}} f_3$.

The Church-Rosser property for $\rightarrow_{\mathrm{s}}$ is proved using the Newman's proposition 3.1.25 in [5], by proving that $\rightarrow_{\mathrm{s}}$ is strongly normalizing and weak Church-Rosser.

LEMMA 3.8 (*Church-Rosser for* $\rightarrow_{\mathrm{s}}$) If e, e_1, and $e_2 \in \mathbb{E}$ are such that $e \twoheadrightarrow_{\mathrm{s}} e_1$ and $e \twoheadrightarrow_{\mathrm{s}} e_2$, then there exists e_3 such that $e_1 \twoheadrightarrow_{\mathrm{s}} e_3$ and $e_2 \twoheadrightarrow_{\mathrm{s}} e_3$.

LEMMA 3.9 (*s-postponement*) If $e \rightarrow_{\mathrm{s}} e_1$ and $e_1 \rightarrow_{\beta\mathrm{for}} e_2$, then there exists e_3 such that $e \rightarrow_{\beta\mathrm{for}} e_3$ and $e_3 \twoheadrightarrow_{\mathrm{s}} e_1$.

LEMMA 3.10 If e, e_1, and $e_2 \in \mathbb{E}$ are such that $e \twoheadrightarrow_{\beta\mathrm{for}} e_1$ and $e \twoheadrightarrow_{\mathrm{s}} e_2$ then there exists e_3 such that $e_1 \twoheadrightarrow_{\mathrm{s}} e_3$ and $e_2 \twoheadrightarrow_{\beta\mathrm{for}} e_3$.

Finally, we can state and prove the confluence property for the reduction relation of $F^{\omega}_{\wedge}$.

Confluence of $F^\omega_\wedge$

THEOREM 3.11 (*Church-Rosser for* $\rightarrow_{\beta\text{fors}} \cup \rightarrow_{\beta\wedge}$)
If E, F, and $G \in \mathbb{T} \cup \mathbb{E}$ are such that $E \twoheadrightarrow_{\beta\text{fors}\cup\beta\wedge} F$ and $E \twoheadrightarrow_{\beta\text{fors}\cup\beta\wedge} G$, then there exists $H \in \mathbb{T} \cup \mathbb{E}$ such that $F \twoheadrightarrow_{\beta\text{fors}\cup\beta\wedge} H$ and $G \twoheadrightarrow_{\beta\text{fors}\cup\beta\wedge} H$.

PROOF: By the commutativity of $\twoheadrightarrow_{\beta\text{fors}}$ and $\twoheadrightarrow_{\beta\wedge}$ (corollary 3.3) and the Church-Rosser property of $\rightarrow_{\beta\text{fors}}$ and $\rightarrow_{\beta\wedge}$ (theorems 3.5 and 3.6). □

The Church-Rosser theorem has interesting corollaries that we will use in the sequel.

COROLLARY 3.12 See chapter 3 of [5]. Let R be a reduction satisfying the Church-Rosser property. Then

1. If $T =_R S$, then there exists U such that $T \twoheadrightarrow_R U$ and $S \twoheadrightarrow_R U$.
2. If T is a normal form of S, then $S \twoheadrightarrow_R T$.
3. Each term has at most one R-normal form.

FACT 3.13

1. $\forall X{\leq}S{:}K.T =_{\beta\wedge} \top^\star$ if and only if $T =_{\beta\wedge} \top^\star$.
2. $\Lambda X{:}K.T =_{\beta\wedge} \top^\star$ if and only if $T =_{\beta\wedge} \top^\star$.
3. $S{\rightarrow}T =_{\beta\wedge} \top^\star$ if and only if $T =_{\beta\wedge} \top^\star$.
4. $T\,S =_{\beta\wedge} \top^\star$ if and only if $T =_{\beta\wedge} \top^\star$.

LEMMA 3.14 If $S \twoheadrightarrow_{\beta\wedge} S'$, then $S[X{\leftarrow}U] \twoheadrightarrow_{\beta\wedge} S'[X{\leftarrow}U]$.

4 Structural properties

This section establishes a number of structural properties of $F^\omega_\wedge$. Except where noted, the proofs proceed by structural induction and are straightforward when performed in the order in which they appear.

LEMMA 4.1 If $\Gamma \vdash \Sigma$ and Γ_1 is a prefix of Γ, then $\Gamma_1 \vdash \text{ok}$ as a subderivation. Moreover, except for the case $\Gamma_1 \equiv \Gamma$ and $\Sigma \equiv \text{ok}$, the subderivation is strictly shorter.

LEMMA 4.2 (*Generation for context judgements*)

1. If $\Gamma_1, X{\leq}T{:}K, \Gamma_2 \vdash \text{ok}$, then $\Gamma_1 \vdash T : K$ by a proper subderivation.
2. If $\Gamma_1, x{:}T, \Gamma_2 \vdash \text{ok}$, then $\Gamma_1 \vdash T : \star$ by a proper subderivation.

LEMMA 4.3 (*Free variables*)

1. If $\Gamma \vdash T : K$, then $\mathrm{FTV}(T) \subseteq \mathrm{dom}(\Gamma)$.

2. If $\Gamma \vdash \mathrm{ok}$, then each variable or type variable in $\mathrm{dom}(\Gamma)$ is declared only once.

If one tries to prove Weakening (Corollary 4.6) directly by induction on derivations the induction hypothesis is too weak in the cases for K-All and S-OAbs, for example. This problem occurs in the lambda calculus without subtyping for the abstraction rule, and was identified by McKinna and Pollack for Pure Type Systems. We adapt their idea of *renaming* [32].

Definition 4.4 (*Parallel Substitution*) A *parallel substitution* γ for Γ is an assignment of types to type variables in $\mathrm{dom}(\Gamma)$ and terms to term variables in $\mathrm{dom}(\Gamma)$. A *renaming* for Γ in Δ is a parallel substitution γ from variables to variables such that

- for every $x{:}A$ in Γ, $\gamma(x){:}A[\gamma]$ is in Δ, and
- for every $X{\leq}T{:}K$ in Γ, $\gamma(X){\leq}A[\gamma]{:}K$ is in Δ.

We write $\Sigma[\gamma]$ for the result of performing the substitution γ in the judgement Σ. The renaming $\gamma\{x \mapsto y\}$ maps x to y and behaves like γ elsewhere, similarly for type variables.

Lemma 4.5 (*Renaming*) If $\Delta \vdash \mathrm{ok}$ and γ is a renaming for Γ in Δ then $\Gamma \vdash \Sigma$ implies $\Delta \vdash \Sigma[\gamma]$.

Proof: By induction on the derivation of $\Gamma \vdash \Sigma$. Most cases follow easily using the induction hypothesis or the definition of renaming. We illustrate here the case for K-All, which is representative of the interesting cases.

Let $Z \notin \mathrm{dom}(\Delta)$. Define γ_0 as $\gamma_0 \equiv \gamma\{X \mapsto Z\}$, then γ_0 is a renaming for $\Gamma,\, X{\leq}T_1{:}K_1$ in $\Delta,\, Z{\leq}T_1[\gamma_0]{:}K_1$. By lemmas 4.1 and 4.2(1), there exists a shorter subderivation of $\Gamma \vdash T_1 : K_1$, and by the free variables lemma (lemma 4.3), $X \notin \mathrm{FV}(T_1)$, therefore $T_1[\gamma_0] \equiv T_1[\gamma]$.

We need to show that $\Delta,\, Z{\leq}T_1[\gamma]{:}K_1 \vdash \mathrm{ok}$. By assumption we know that $\Delta \vdash \mathrm{ok}$, by the induction hypothesis, $\Delta \vdash T_1[\gamma] : K_1$. Since we chose Z not to be in $\mathrm{dom}(\Delta)$, by K-TVar, $\Delta,\, Z{\leq}T_1[\gamma]{:}K_1 \vdash \mathrm{ok}$.

We can now apply the induction hypothesis to prove $\Delta,\, Z{\leq}T_1[\gamma]{:}K_1 \vdash T_2[\gamma_0] : \star$. By K-All, $\Delta \vdash \forall Z{\leq}T_1[\gamma]{:}K_1.T_2[\gamma_0] : \star$, and by the definition of substitution $\Delta \vdash (\forall X{\leq}T_1{:}K_1.T_2 : \star)[\gamma]$. □

Weakening now follows as a corollary of renaming taking γ to be the identity substitution.

Corollary 4.6 (*Weakening/Permutation*) Let Γ and Γ' be contexts such that $\Gamma \subseteq \Gamma'$ and $\Gamma' \vdash \mathrm{ok}$. Then $\Gamma \vdash \Sigma$ implies $\Gamma' \vdash \Sigma$.

Proof: Let γ be the identity substitution. Then γ is a renaming for Γ in Δ and $\Sigma[\gamma] \equiv \Sigma$. Then, by Renaming (Proposition 4.5), it follows that $\Delta \vdash \Sigma$. □

LEMMA 4.7 (*Context, kind, and term strengthening*)

1. If $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash \text{ok}$ and $X \notin \text{FTV}(\Gamma_2)$, then $\Gamma_1,\, \Gamma_2 \vdash \text{ok}$.
2. If $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash S : K'$ and $X \notin \text{FTV}(\Gamma_2) \cup \text{FTV}(S)$, then $\Gamma_1,\, \Gamma_2 \vdash S : K'$.
3. If $\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \Sigma$ and $x \notin \text{FV}(\Sigma)$, then $\Gamma_1,\, \Gamma_2 \vdash \Sigma$.

Moreover, the derivations of the conclusions are strictly shorter than the derivation of the premises.

PROOF: Statements 1 and 2 follow by simultaneous induction on the length of derivations, and statement 3 by induction on the derivation of $\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \Sigma$. In all cases lemmas 4.1 and 4.3 are used. □

PROPOSITION 4.8 (*Generation for kinding*)

1. $\Gamma \vdash X : K$ implies $\Gamma \equiv \Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2$ for some Γ_1, T, and Γ_2.
2. $\Gamma \vdash T_1{\to}T_2 : K$ implies $K \equiv \star$ and $\Gamma \vdash T_1, T_2 : \star$.
3. $\Gamma \vdash \forall X{\leq}T_1{:}K_1.T_2 : K$ implies $K \equiv \star$ and $\Gamma,\, X{\leq}T_1{:}K_1 \vdash T_2 : \star$.
4. $\Gamma \vdash \Lambda(X{:}K_1)T_2 : K$ implies $K \equiv K_1{\to}K_2$ and $\Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 : K_2$, for some K_2.
5. $\Gamma \vdash S\,T : K$ implies $\Gamma \vdash S : K'{\to}K$ and $\Gamma \vdash T : K'$, for some K'.
6. $\Gamma \vdash \bigwedge^K[T_1..T_n] : K'$ implies $K \equiv K'$ and $\Gamma \vdash \text{ok}$ and $\Gamma \vdash T_i : K$ for each i.

Moreover, the proofs of the consequents are all strictly shorter than those of the antecedents.

PROOF: In each case the antecedent uniquely determines the last rule of its derivation. The proof follows by inspection of the rules. □

LEMMA 4.9 (*Uniqueness of kinds*) If $\Gamma \vdash T : K$ and $\Gamma \vdash T : K'$, then $K \equiv K'$.

LEMMA 4.10 (*Type substitution*) Let $\Gamma_1 \vdash T : K_U$. Then

1. If $\Gamma_1,\, X{\leq}U{:}K_U,\, \Gamma_2 \vdash S : K_S$, then $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash S[X{\leftarrow}T] : K_S$.
2. If $\Gamma_1,\, X{\leq}U{:}K_U,\, \Gamma_2 \vdash \text{ok}$, then $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash \text{ok}$.

PROOF: By simultaneous induction on derivations of the premises. The proof of part 2 is straightforward using part 1 of the induction hypothesis. We consider the details of the proof of 1. The cases K-ARROW, K-ALL, K-OABS, and K-OAPP follow by straightforward application of part 1 of the induction hypothesis and the corresponding rule, while the case of K-MEET also uses part 2 of the induction hypothesis. We examine the case of K-TVAR, where $S \equiv Y$ for some variable Y. By proposition 4.8(1) $Y{\leq}T_Y{:}K_S : (\Gamma_1,\, X{\leq}U{:}K_U,\, \Gamma_2)$ for some T_Y. There are three cases to consider.

$Y{\le}T_Y{:}K_S \in \Gamma_1$ Then we also have $Y{\le}T_Y{:}K_S \in (\Gamma_1,\, \Gamma_2[X{\leftarrow}T])$. By part 2 of the induction hypothesis, $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash \mathrm{ok}$. Applying K-TVAR, we get $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash Y : K_S$.

$Y{\le}T_Y{:}K_S \equiv X{\le}U{:}K_U$ We know that $\Gamma_1 \vdash T : K_S \equiv K_U$. From the premise of K-TVAR and part 2 of the induction hypothesis, we have $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash \mathrm{ok}$. The result follows by weakening (corollary 4.6).

$Y{\le}T_Y{:}K_S \in \Gamma_2$ Then we have $Y{\le}T_Y[X{\leftarrow}T]{:}K_S \in (\Gamma_1,\, \Gamma_2[X{\leftarrow}T])$. By part 2 of the induction hypothesis, $\Gamma_1,\, \Gamma_2[X{\leftarrow}T] \vdash \mathrm{ok}$, from which the result follows by K-TVAR. □

LEMMA 4.11 (*Subject reduction for kinding judgements*) If $S \twoheadrightarrow_{\beta\wedge} T$ and $\Gamma \vdash S : K$, then $\Gamma \vdash T : K$.

PROOF: In order to prove this result it is enough to prove the following statements by simultaneous induction on the derivation of $\Gamma \vdash S : K$. The rest follows by induction on the definition of $\twoheadrightarrow_{\beta\wedge}$.

1. $\Gamma \vdash \mathrm{ok}$ and $\Gamma \rightarrow_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash \mathrm{ok}$.
2. $\Gamma \vdash S : K$ and $S \rightarrow_{\beta\wedge} T$ implies $\Gamma \vdash T : K$.
3. $\Gamma \vdash S : K$ and $\Gamma \rightarrow_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash S : K$. □

THEOREM 4.12 (*Kind invariance under type conversion*) If $\Gamma \vdash S : K_S$ and $\Gamma \vdash T : K_T$, with $S =_{\beta\wedge} T$, then $K_S \equiv K_T$.

PROOF: By the Church-Rosser theorem 3.5, there exists U such that $S \twoheadrightarrow_{\beta\wedge} U$ and $T \twoheadrightarrow_{\beta\wedge} U$, and the result follows by subject reduction and unicity of kinds. □

LEMMA 4.13 Let $\Gamma \vdash S_j : K$ for each $j \in \{1..m\}$. Then if for every $i \in \{1..n\}$ there exists $j \in \{1..m\}$ such that $\Gamma \vdash S_j \le T_i$, then $\Gamma \vdash \bigwedge^K[S_1..S_m] \le \bigwedge^K[T_1..T_n]$.

A particular case of the previous lemma is the following.

COROLLARY 4.14 Let $\Gamma \vdash S_i : K$ for each $i \in \{1..n\}$. Then $\Gamma \vdash S_i \le T_i$, for every $i \in \{1..n\}$, implies $\Gamma \vdash \bigwedge^K[S_1..S_n] \le \bigwedge^K[T_1..T_n]$.

LEMMA 4.15 Let $\Gamma \vdash S, T : K$. Then $\Gamma \vdash S \le T$ if and only if $\Gamma \vdash S^{\mathrm{nf}} \le T^{\mathrm{nf}}$.

PROOF: We shall consider only one part the other is similar.

$\Rightarrow$) By subject reduction, we have that $\Gamma \vdash S^{\mathrm{nf}} : K$, then, by S-CONV, $\Gamma \vdash S^{\mathrm{nf}} \le S$. By similar reasoning we have $\Gamma \vdash T \le T^{\mathrm{nf}}$. The result follows by applying S-TRANS twice. □

LEMMA 4.16 (*Context modification*) If $\Gamma_1 \vdash U' : K$ and Σ is either ok or $T : K'$, then $\Gamma_1,\ X{\leq}U{:}K,\ \Gamma_2 \vdash \Sigma$ implies $\Gamma_1,\ X{\leq}U'{:}K,\ \Gamma_2 \vdash \Sigma$.

LEMMA 4.17 Let $\Gamma \vdash S_i : K$ for every $i \in \{1..n\}$. If for every j in $\{1..m\}$ there exists i in $\{1..n\}$ such that $\Gamma \vdash S_i \leq T_j$, then $\Gamma \vdash \bigwedge^K[S_1..S_n] \leq \bigwedge^K[T_1..T_m]$.

PROPOSITION 4.18 (*Well-kindedness of subtyping*) If $\Gamma \vdash S \leq T$, then $\Gamma \vdash S : K$ and $\Gamma \vdash T : K$ for some K.

PROOF: By induction on the derivation of $\Gamma \vdash S \leq T$. We show a few representative cases.

S-CONV We are given that $\Gamma \vdash S : K$ and $\Gamma \vdash T : K'$ and $S =_\beta T$. By lemma 4.12, $K \equiv K'$.

S-TVAR We are given that $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash$ ok. $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash X : K$ follows by K-TVAR. Moreover, by lemma 4.2, we have $\Gamma_1 \vdash T : K$, and by weakening (corollary 4.6), $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash T : K$.

S-ARROW We are given $\Gamma \vdash T_1 \leq S_1$ and $\Gamma \vdash S_2 \leq T_2$ and $\Gamma \vdash S_1{\to}S_2 : \star$. By proposition 4.8, $\Gamma \vdash S_1, S_2 : \star$. Further, by the induction hypothesis together with uniqueness of kinds (lemma 4.9), we have $\Gamma \vdash T_1, T_2 : \star$. Finally, the result follows by applying K-ARROW. □

PROPOSITION 4.19 (*Well-kindedness of typing*) If $\Gamma \vdash e : T$, then $\Gamma \vdash T : \star$.

PROOF: By induction on the derivation of $\Gamma \vdash e : T$. We show here a few interesting cases

T-VAR We are given $\Gamma_1,\ x{:}T,\ \Gamma_2 \vdash$ ok. The result follows by generation for context judgements (lemma 4.2) and weakening (corollary 4.6).

T-ABS We are given $\Gamma,\ x{:}T_1 \vdash e : T_2$. By the induction hypothesis, $\Gamma,\ x{:}T_1 \vdash T_2 : \star$. By lemma 4.7, it follows that $\Gamma \vdash T_2 : \star$. Furthermore, by lemmas 4.1 and 4.2, $\Gamma \vdash T_1 : \star$. Hence, K-ARROW yields $\Gamma \vdash T_1{\to}T_2 : \star$.

T-TAPP We know that $\Gamma \vdash f : \forall(X{\leq}T_1{:}K_1)T_2$ and also $\Gamma \vdash S \leq T_1$. By the induction hypothesis, $\Gamma \vdash \forall(X{\leq}T_1{:}K_1)T_2 : \star$ and, by proposition 4.8, $\Gamma,\ X{\leq}T_1{:}K_1 \vdash T_2 : \star$. By lemmas 4.1 and 4.2, there exists a derivation of $\Gamma \vdash T_1 : K_1$. By the well-kindedness of subtyping (proposition 4.18) and uniqueness of kinds (lemma 4.9), we have $\Gamma \vdash S : K_1$. Then, by the type substitution lemma (lemma 4.10), $\Gamma \vdash T_2[X{\leftarrow}S] : \star$.

T-SUB By the induction hypothesis, proposition 4.18 and lemma 4.9. □

5 Strong normalization of $\rightarrow_{\beta\wedge}$

We prove that every type that has a kind in $F^\omega_\wedge$ is strongly normalizing in three steps. We first prove that $\rightarrow_a$ and also $\rightarrow_{\beta\wedge^-}$ are strongly normalizing. Then we prove that both reductions commute, i.e. if $T \rightarrow_a T_1$ and $T_1 \rightarrow_{\beta\wedge^-} T_2$, then there exists S such that $S \rightarrow_a T_2$ and $T \twoheadrightarrow_{\beta\wedge^-}{}^{>0} S$ (in at least one step). Finally, using the previous two steps we prove that $\rightarrow_{\beta\wedge}$ is strongly normalizing.

A type T is called *strongly normalizing* if and only if all reduction sequences starting with T terminate. We write $\mathbb{T}$ for the set of all type expressions and *SN* for the subset of $\mathbb{T}$ of strongly normalizing type expressions. If A and B are subsets of $\mathbb{T}$, then $A \rightarrow B$ denotes the following subset of $\mathbb{T}$

$$A \rightarrow B = \{F \subseteq \mathbb{T} \,|\, \text{for all } a \in A,\ Fa \in B\}.$$

LEMMA 5.1 $\rightarrow_a$ is strongly normalizing.

PROOF: By induction on the number of intersection symbols of the type expression being reduced. □

To prove strong normalization of $\rightarrow_{\beta\wedge^-}$ we use a model-theoretic argument interpreting kinds as sets of normalizing terms, and the soundness of the model gives, as a corollary, the strong normalization property. The interpretation of a kind K, notation $[\![K]\!]$, is defined as follows.

$$\begin{array}{lcl} [\![\star]\!] & = & \mathit{SN} \\ [\![K_1 \rightarrow K_2]\!] & = & [\![K_1]\!] \rightarrow [\![K_2]\!]. \end{array}$$

DEFINITION 5.2 (*Saturated set*) $\boldsymbol{S} \subseteq \mathit{SN}$ is *saturated* if is satisfies the following conditions:

1. If $R_1..R_n \in \mathit{SN}$, then $XR_1..R_n \in \boldsymbol{S}$.
2. If $R_1..R_n, Q \in \mathit{SN}$, then
   (a) if $P[X{\leftarrow}Q]R_1..R_n \in \boldsymbol{S}$, then $(\Lambda X{:}K.P)QR_1..R_n \in \boldsymbol{S}$, for every K and
   (b) if $(\bigwedge^{K_2}[T_1Q, .., T_mQ])R_1, .., R_n \in \boldsymbol{S}$, then $(\bigwedge^{K_1\rightarrow K_2}[T_1, .., T_m])QR_1, .., R_n \in \boldsymbol{S}$, for every K_1.

Intuitively, a set of strongly normalizing type expressions is saturated if it contains all type variables and is closed under expansion of expressions which may have a kind of the form $K_1 \rightarrow K_2$.

LEMMA 5.3

1. *SN* is saturated.

2. If A, B are saturated, then $A \to B$ is saturated.

3. For any kind K, $[\![K]\!]$ is saturated.

DEFINITION 5.4

1. A valuation ρ in $\mathbb{T}$ is a mapping from type variables to types.

2. The interpretation of a type with respect to ρ is

$$[\![T]\!]_\rho = T[X_1 \leftarrow \rho(X_1)..X_n \leftarrow \rho(X_n)],$$

   where $\mathrm{FV}(T) \;=\; \{X_1..X_n\}$.

3. Let ρ be a valuation in $\mathbb{T}$. Then ρ *satisfies* $T : K$, written $\rho \models T : K$, if $[\![T]\!]_\rho : [\![K]\!]$ and ρ *satisfies* $X{\leq}T{:}K$, written $\rho \models X{\leq}T{:}K$, if $\rho(X) : [\![K]\!]$. We say that ρ *satisfies* a context Γ, $\rho \models \Gamma$, if $\rho \models X \leq S{:}K$ for all $X \leq S{:}K : \Gamma$.

4. A context Γ *satisfies* $T : K$, written $\Gamma \models T : K$, if for every ρ such that $\rho \models \Gamma$, it follows that $\rho \models T : K$.

LEMMA 5.5

1. $\top^K \in [\![K]\!]$.

2. If $A_i \in [\![K]\!]$ for each $i \in \{1..n\}$, then $\bigwedge^K[A_1..A_n] \in [\![K]\!]$.

PROOF: We show item 2. Item 1 also follows follows by induction on the structure of K.

$K \equiv \star$ Then, by definition of $[\![K]\!]$, $A_i \in SN$ for each $i \in \{1..n\}$. Since every reduction starting from $\bigwedge^K[A_1..A_n]$ is a reduction consisting only of steps inside the $A_i's$, one has $\bigwedge^K[A_1..A_n] \in SN \equiv [\![K]\!]$.

$K \equiv K_1{\to}K_2$ Let $B \in [\![K_1]\!]$. By the definition of $\to$, $A_iB \in [\![K_2]\!]$, for each $i \in \{1..n\}$. By the induction hypothesis, $\bigwedge^{K_2}[A_1B..A_nB] \in [\![K_2]\!]$. Moreover, $\bigwedge^{K_1\to K_2}[A_1..A_n]B \in [\![K_2]\!]$ by the saturation of $[\![K_2]\!]$, which means that $\bigwedge^{K_1\to K_2}[A_1..A_n] \in [\![K_1{\to}K_2]\!]$. □

PROPOSITION 5.6 (*Soundness*) If $\Gamma \vdash T : K$, then $\Gamma \models T : K$.

PROOF: By induction on the derivation of $\Gamma \vdash T : K$.

We consider the case for K-MEET. The other cases follow by similar reasoning. Let $T \equiv \bigwedge^K[T_1..T_n]$. We have to consider two cases.

$n \not\equiv 0$ We are given $\Gamma \vdash T_i : K$ for each $i \in \{1..n\}$, and, by the induction hypothesis, $\Gamma \models T_i : K$. Let ρ be a valuation such that $\rho \models \Gamma$. Then $[\![T_i]\!]_\rho \in [\![K]\!]$, for each $i \in \{1..n\}$. By lemma 5.5(2), $\bigwedge^K[[\![T_1]\!]_\rho..[\![T_n]\!]_\rho] \in [\![K]\!]$.

$n \equiv 0$ $T \equiv \top^K$. Since $[\![\top^K]\!]_\rho \equiv \top^K$, the result follows by 5.5(1). □

THEOREM 5.7 (*Strong normalization for* $\rightarrow_{\beta\wedge^-}$)
$\Gamma \vdash T : K$ implies that every $(\beta\wedge^-)$-reduction sequence starting from T is finite.

PROOF: By soundness, $\Gamma \models T : K$. Choose ρ_0 such that $\rho_0(X) = X$. Observe that $\rho_0 \models \Gamma$ trivially. Hence $T \equiv [\![T]\!]_{\rho_0} \in [\![K]\!] \subseteq SN$. □

LEMMA 5.8 If $T \rightarrow_a T_1$ and $T_1 \rightarrow_{\beta\wedge^-} T_2$, then there exists S such that $T \twoheadrightarrow_{\beta\wedge^-}^{>0} S$ and $S \rightarrow_a T_2$.

PROOF: By induction on the structure of T. □

COROLLARY 5.9 (*a postponement*) If $T \twoheadrightarrow_a T_1$ and $T_1 \twoheadrightarrow_{\beta\wedge^-} T_2$, then there exists S such that $T \twoheadrightarrow_{\beta\wedge^-}^{>0} S$ and $S \twoheadrightarrow_a T_2$.

PROOF: By induction on the generation of $T \twoheadrightarrow_a T_1$. □

Finally, we can prove strong normalization for $\rightarrow_{\beta\wedge}$.

THEOREM 5.10 (*Strong normalization for* $\rightarrow_{\beta\wedge}$) $\Gamma \vdash T : K$ implies that every $(\beta\wedge)$-reduction sequence starting from T is finite.

PROOF: Let $\Gamma \vdash T : K$. We reason by contradiction. Assume that there is an infinite $\beta\wedge$-reduction sequence starting from T. Then lemma 5.1 and theorem 5.7 imply that there are infinitely many alternations of $\rightarrow_a$ and $\rightarrow_{\beta\wedge^-}$ reduction sequences. By corollary 5.9, we can construct an infinite $(\beta\wedge^-)$-reduction which contradicts theorem 5.7. □

6 Towards a generation principle for subtyping

In this section we start our quest towards a generation principle for the subtyping relation of $F^\omega_\wedge$. First, we develop a *normal subtyping system*, $NF^\omega_\wedge$, in which only types in normal form are considered. We then prove that proofs in $NF^\omega_\wedge$ can be normalized by eliminating transitivity and simplifying reflexivity. This simplification yields an algorithmic presentation, $AlgF^\omega_\wedge$, whose rules are syntax directed. Moreover, we prove that $AlgF^\omega_\wedge$ is indeed an alternative presentation of the $F^\omega_\wedge$ subtyping relation. Formally, $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$, when S and T are well-formed (proposition 9.2).

In the solution for the second order lambda calculus presented in [34], the distributivity rules for intersection types are not considered as rewrite rules. For that reason, new syntactic categories have to be defined (composite and individual canonical types) and an auxiliary mapping (flattening) transforms a type into a canonical type. Our solution does not need either new syntactic categories or elaborate auxiliary mappings, since the role played there by canonical types is performed here by types in normal form.

6.1 Normal Subtyping

An important property of derivation systems is the information that a derivable judgement contains about its proofs. This information is essential to produce results which not only state properties about the subproofs, but also help identify ill formed judgements.

As we mentioned in the introduction, in $F^{\omega}_{\wedge}$ we can prove:

$$W{:}K, X \leq \Lambda Y{:}K.Y{:}K{\to}K,\ Z \leq X{:}K{\to}K \vdash X(Z\,W) \leq W \tag{1}$$

Note that X and Z are subtypes of the identity on K, therefore it makes sense for $X(Z\,W)$ to be a subtype of W. The derivation is as follows: Let $\Gamma \equiv W{:}K,\ X \leq \Lambda Y{:}K.Y{:}K{\to}K,\ Z \leq X{:}K{\to}K$. For the sake of readability we omit kinding judgements.

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash \text{ok}}{\Gamma \vdash X \leq \Lambda Y{:}K.Y}\,\text{S-TVAR}}{\Gamma \vdash X(Z\,W) \leq (\Lambda Y{:}K.Y)Z\,W}\,\text{S-OAPP} \qquad \dfrac{(\Lambda Y{:}K.Y)Z\,W =_{\beta\wedge} Z\,W}{(\Lambda Y{:}K.Y)Z\,W \leq Z\,W}\,\text{S-CONV}}{\Gamma \vdash X(Z\,W) \leq Z\,W}\,\text{S-TRANS}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash \text{ok}}{\Gamma \vdash Z \leq X}\,\text{S-TVAR} \quad \dfrac{\Gamma \vdash \text{ok}}{\Gamma \vdash X \leq \Lambda Y{:}K.Y}\,\text{S-TVAR}}{\Gamma \vdash Z \leq (\Lambda Y{:}K.Y)}\,\text{S-TRANS}}{\Gamma \vdash Z\,W \leq (\Lambda Y{:}K.Y)W}\,\text{S-OAPP} \qquad \dfrac{(\Lambda Y{:}K.Y)W =_{\beta\wedge} W}{(\Lambda Y{:}K.Y)W \leq W}\,\text{S-CONV}}{\Gamma \vdash Z\,W \leq W}\,\text{S-TRANS}$$

$$\dfrac{\Gamma \vdash X(Z\,W) \leq Z\,W \quad \Gamma \vdash Z\,W \leq W}{\Gamma \vdash X(Z\,W) \leq W}\,\text{S-TRANS}$$

This simple example already shows that S-TRANS erases information obtained by S-CONV that is not present in the conclusion any longer. A first step towards an algorithm to check the subtyping relation is to design a set of rules in which the derivable judgements contain all the information about their derivations. To this end we define a set of rules, $NF^{\omega}_{\wedge}$, in which conversion is reduced to a minimum and, as we show in lemma 7.6, transitivity can be eliminated. Both results are proved with a standard cut-elimination argument. This yields a syntax directed subtyping relation, $AlgF^{\omega}_{\wedge}$, which constitutes a decision procedure for the original system.

In the rest of this section, we present the subtyping system $NF^{\omega}_{\wedge}$, which uses the context and type formation rules of $F^{\omega}_{\wedge}$. We define rewriting rules for derivations in $NF^{\omega}_{\wedge}$ (definitions 7.3 and 7.4), and describe a terminating procedure to normalize proofs, which gives, as a consequence, the generation for subtyping (proposition 7.10) and an algorithmic presentation, $AlgF^{\omega}_{\wedge}$ (see section 9).

Finally, in section 9, we show that there is an equivalence between subtyping in $F^{\omega}_{\wedge}$ and subtyping in $AlgF^{\omega}_{\wedge}$.

We now define the *normal subtyping system*, $NF^{\omega}_{\wedge}$. Subtyping statements in $NF^{\omega}_{\wedge}$ are written $\Gamma \vdash_n S \leq T$, and S, T, and all types appearing in Γ are in $\beta\wedge$-normal form.

NOTATION 6.1.1 A, B, and C range over types whose outermost constructor is not an intersection.

REMARK 6.1.2 It is an immediate consequence of the $\beta\wedge$-reduction rules that, if T is in $\beta\wedge$-normal form, then T is either X, $S{\rightarrow}A$, $\forall X{\leq}S{:}K.A$, $\Lambda X{:}K.A$, $A\,S$ where A is not an abstraction, or $\bigwedge^K[A_1..A_n]$. We frequently use this notation as a reminder of the shape of types in normal form.

We now define $lub_\Gamma(S)$. We prove in lemma 8.1 and corollary 8.1.2, that, when defined, it is the smallest type beyond S with respect to Γ.

DEFINITION 6.1.3 (*Least strict Upper Bound*)

$$\begin{array}{rcl} lub_\Gamma(X) & = & \Gamma(X), \\ lub_\Gamma(T\,S) & = & lub_\Gamma(T)\,S. \end{array}$$

DEFINITION 6.1.4 ($\mathrm{NF}^\omega_\wedge$ *subtyping rules*)

$$\frac{\Gamma\vdash S : K}{\Gamma\vdash_n S\leq S} \qquad \text{(NS-REFL)}$$

$$\frac{\Gamma\vdash_n S\leq T \qquad \Gamma\vdash_n T\leq U}{\Gamma\vdash_n S\leq U} \qquad \text{(NS-TRANS)}$$

$$\frac{\Gamma\vdash_n \Gamma(X)\leq A \qquad X\not\equiv A}{\Gamma\vdash_n X\leq A} \qquad \text{(NS-TVAR)}$$

$$\frac{\Gamma\vdash_n T\leq S \qquad \Gamma\vdash_n A\leq B \qquad \Gamma\vdash S{\rightarrow}A : \star}{\Gamma\vdash_n S{\rightarrow}A\leq T{\rightarrow}B} \qquad \text{(NS-ARROW)}$$

$$\frac{\Gamma,\, X{\leq}S{:}K\vdash_n A\leq B \qquad \Gamma\vdash\forall X{\leq}S{:}K.A : \star}{\Gamma\vdash_n \forall X{\leq}S{:}K.A\leq\forall X{\leq}S{:}K.B} \qquad \text{(NS-ALL)}$$

$$\frac{\Gamma,\, X{\leq}\top^K{:}K\vdash_n A\leq B}{\Gamma\vdash_n \Lambda X{:}K.A\leq\Lambda X{:}K.B} \qquad \text{(NS-OABS)}$$

$$\frac{\Gamma\vdash_n (lub_\Gamma(A\,S))^{\mathrm{nf}}\leq B \qquad \Gamma\vdash A\,S : K \qquad A\,S\not\equiv B}{\Gamma\vdash_n T\,S\leq A} \qquad \text{(NS-OAPP)}$$

$$\frac{\forall i\in\{1..m\}\,\Gamma\vdash_n A\leq T_i \qquad \Gamma\vdash A : K}{\Gamma\vdash_n A\leq\bigwedge^K[T_1..T_m]} \qquad \text{(NS-}\forall\text{)}$$

$$\frac{\exists j\in\{1..n\}\,\Gamma\vdash_n S_j\leq A \qquad \forall k\in\{1..n\}\,\Gamma\vdash S_k : K}{\Gamma\vdash_n \bigwedge^K[S_1..S_n]\leq A} \qquad \text{(NS-}\exists\text{)}$$

$$\frac{\forall i\in\{1..m\}\,\exists j\in\{1..n\}\,\Gamma\vdash_n S_j\leq T_i \qquad \forall k\in\{1..n\}\,\Gamma\vdash S_k : K}{\Gamma\vdash_n \bigwedge^K[S_1..S_n]\leq\bigwedge^K[T_1..T_m]} \qquad \text{(NS-}\forall\exists\text{)}$$

As we mentioned in the introduction, an important factor to develop this system was to consider the distributivity rules of the presentation of $F^{\omega}_{\wedge}$ in [23] as reduction rules instead of subtyping rules. This new point of view suggested that an algorithmic system should, to a certain extent, concentrate on normal forms replacing the conversion rule by reflexivity. Consequently, a derivation of a subtyping statement should involve only types in normal form. But enlightened by the simple (counter)example 1 it is not possible to perform all reductions at once. In other words, the system does not satisfy an S-CONV postponement property. Without using S-CONV it is not possible to derive example 1. Hence, the solution is not as simple as replacing S-CONV by NS-REFL.

In general, the interaction between S-TRANS and S-CONV can be analyzed as follows. In S-TRANS the metavariable T of the hypothesis is not present in the conclusion, but this is not a problem by itself (a similar situation appears in the simply typed lambda calculus in its application rule and the system is deterministic). The problem is that in the presence of S-CONV the *vanishing* T can be $\beta\wedge$-convertible to either S or U or to both S and U. What example 1 shows is that S and U may be different normal forms, which means that searching for T is inherently nondeterministic.

We cannot eliminate transitivity completely, we still need it on type variables and on type applications. In $F_{\leq}$ [29] transitivity is eliminated and hidden in a richer variable rule in which deciding whether $\Gamma \vdash X \leq T$ when $T \not\equiv X$ is reduced to deciding whether the bound of X is smaller than or equal to T. The bound of X has the particular property of being the *least strict upper bound* of X. This observation motivated the definition of our NS-OAPP rule, in which we reduce the decision of whether $\Gamma \vdash A\,S \leq B$ when $B \not\equiv A\,S$, to checking if the least strict upper bound of $A\,S$ is smaller than or equal to B (See lemma 8.1 and corollary 8.1.2). The least strict upper bound of $A\,S$, $lub_{\Gamma}(A\,S)$, is obtained from $A\,S$ by replacing its leftmost innermost variable by the corresponding bound in Γ. In our example, $lub_{\Gamma}(X(Z\,W))$ is $(\Lambda Y{:}K.Y)(Z\,W)$. Consequently, $lub_{\Gamma}(A\,S)$ may be other than a normal form. That is the reason we normalize it. The strength of the conversion rule that is not captured by reflexivity is hidden in this normalization step. Since $A\,S$ is a well kinded type, by the free variables lemma (lemma 4.3), $\mathrm{FTV}(A\,S) \subseteq \mathrm{dom}(\Gamma)$. Therefore, $lub_{\Gamma}(A\,S)$ is defined. By lemma 8.1(1), $lub_{\Gamma}(A\,S)$ is well-kinded, and since well-kinded types are strongly normalizing, its normal form exists. The rules S-MEET-LB and S-MEET-G are replaced by NS-$\exists$, NS-$\forall$, and NS-$\forall\exists$. Except for the restriction of types being in normal form NS-ARROW, NS-ALL, and NS-OABS have the same form as S-ARROW, S-ALL, and S-OABS respectively.

7 Structural properties of $NF^{\omega}_{\wedge}$

This section establishes a number of structural properties of $NF^{\omega}_{\wedge}$. The proofs of lemmas 7.1 and 7.2 are similar to those of the corresponding properties for $F^{\omega}_{\wedge}$.

LEMMA 7.1 If $\Gamma \vdash_n S \leq T$ and Γ_1 is a prefix of Γ, then $\Gamma_1 \vdash \mathrm{ok}$ as a subderivation. Moreover, the subderivation is strictly shorter.

Lemma 7.2 (*Weakening/Permutation*) Let Γ and Γ' be contexts such that $\Gamma \subseteq \Gamma'$ and $\Gamma' \vdash \mathrm{ok}$. Then $\Gamma \vdash_n S \leq T$ implies $\Gamma' \vdash_n S \leq T$.

We present rewriting rules on derivations to simplify instances of NS-Refl and NS-Trans. We give a terminating strategy to transform a given derivation into a derivation with occurrences of NS-Refl only applied to type variables or type applications and without occurrences of NS-Trans. To improve readability we omit kinding judgements in the transitivity elimination rules which appear as hypothesis in the redex or in a proper subderivation of the missing ones, as we proved in generation for kinding (proposition 4.8). The derivations of the kinding judgements of each reduct of the reflexivity rules are proper subderivations of the kinding judgements in its redex.

Definition 7.3 (*Reflexivity simplification rules*)

1. $$\boxed{\dfrac{\Gamma \vdash S{\to}A : \star}{\Gamma \vdash_n S{\to}A \leq S{\to}A}\ \textsc{NS-Refl}}$$

$$\Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma \vdash S : \star}{\Gamma \vdash_n S \leq S}\ \textsc{NS-Refl} \qquad \dfrac{\Gamma \vdash A : \star}{\Gamma \vdash_n A \leq A}\ \textsc{NS-Refl}}{\Gamma \vdash_n S{\to}A \leq S{\to}A}\ \textsc{NS-Arrow}}$$

2. $$\boxed{\dfrac{\Gamma \vdash \forall X{\leq}S{:}K.A : \star}{\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq \forall X{\leq}S{:}K.A}\ \textsc{NS-Refl}}$$

$$\Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma,\, X{\leq}S{:}K \vdash A : \star}{\Gamma,\, X{\leq}S{:}K \vdash_n A \leq A}\ \textsc{NS-Refl}}{\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq \forall X{\leq}S{:}K.A}\ \textsc{NS-All}}$$

3. $$\boxed{\dfrac{\Gamma \vdash \Lambda X{:}K.A : K{\to}K'}{\Gamma \vdash_n \Lambda X{:}K.A \leq \Lambda X{:}K.A}\ \textsc{NS-Refl}}$$

$$\Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma,\, X{:}K \vdash A : K'}{\Gamma,\, X{:}K \vdash_n A \leq A}\ \textsc{NS-Refl}}{\Gamma \vdash_n \Lambda X{:}K.A \leq \Lambda X{:}K.A}\ \textsc{NS-OAbs}}$$

4. $$\boxed{\dfrac{\Gamma \vdash \bigwedge^K[A_1..A_n] : K}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq \bigwedge^K[A_1..A_n]}\ \textsc{NS-Refl}}$$

$$\Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma \vdash A_i : K}{\Gamma \vdash_n A_i \leq A_i\, \forall i \in \{1..n\}}\ \textsc{NS-Refl}}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq \bigwedge^K[A_1..A_n]}\ \textsc{NS-}\forall\exists}$$

Definition 7.4 (*Transitivity elimination rules*)

1. $$\boxed{\dfrac{\dfrac{\Gamma \vdash S : K}{\Gamma \vdash_n S \leq S}\,\textsc{NS-Refl} \qquad \Gamma \vdash_n S \leq T}{\Gamma \vdash_n S \leq T}\,\textsc{NS-Trans}} \Rightarrow_T \boxed{\Gamma \vdash_n S \leq T}$$

2. $$\boxed{\dfrac{\Gamma \vdash_n S \leq T \qquad \dfrac{\Gamma \vdash T : K}{\Gamma \vdash_n T \leq T}\,\textsc{NS-Refl}}{\Gamma \vdash_n S \leq T}\,\textsc{NS-Trans}} \Rightarrow_T \boxed{\Gamma \vdash_n S \leq T}$$

3. $$\boxed{\dfrac{\dfrac{\Gamma \vdash_n \Gamma(X) \leq A}{\Gamma \vdash_n X \leq A}\,\textsc{NS-TVar} \qquad \Gamma \vdash_n A \leq B}{\Gamma \vdash_n X \leq B}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma \vdash_n \Gamma(X) \leq A \quad \Gamma \vdash_n A \leq B}{\Gamma \vdash_n \Gamma(X) \leq B}\,\textsc{NS-Trans}}{\Gamma \vdash_n X \leq B}\,\textsc{NS-TVar}}$$

4. $$\boxed{\dfrac{\dfrac{\Gamma \vdash_n T \leq S \quad \Gamma \vdash_n A \leq B}{\Gamma \vdash_n S{\to}A \leq T{\to}B}\,\textsc{NS-Arrow} \qquad \dfrac{\Gamma \vdash_n U \leq T \quad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n T{\to}B \leq U{\to}C}\,\textsc{NS-Arrow}}{\Gamma \vdash_n S{\to}A \leq U{\to}C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma \vdash_n U \leq T \quad \Gamma \vdash_n T \leq S}{\Gamma \vdash_n U \leq S}\,\textsc{NS-Trans} \qquad \dfrac{\Gamma \vdash_n A \leq B \quad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n S{\to}A \leq U{\to}C}\,\textsc{NS-Arrow}}$$

5. $$\boxed{\dfrac{\dfrac{\Gamma, X{\leq}S{:}K \vdash_n A \leq B}{\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq \forall X{\leq}S{:}K.B}\,\textsc{NS-All} \qquad \dfrac{\Gamma, X{\leq}S{:}K \vdash_n B \leq C}{\Gamma \vdash_n \forall X{\leq}S{:}K.B \leq \forall X{\leq}S{:}K.C}\,\textsc{NS-All}}{\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq \forall X{\leq}S{:}K.C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma, X{\leq}S{:}K \vdash_n A \leq B \quad \Gamma, X{\leq}S{:}K \vdash_n B \leq C}{\Gamma, X{\leq}S{:}K \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq \forall X{\leq}U{:}K.C}\,\textsc{NS-All}}$$

6. $$\boxed{\dfrac{\dfrac{\Gamma, X{:}K \vdash_n A \leq B}{\Gamma \vdash_n \Lambda X{:}K.A \leq \Lambda X{:}K.B}\,\textsc{NS-OAbs} \qquad \dfrac{\Gamma, X{:}K \vdash_n B \leq C}{\Gamma \vdash_n \Lambda X{:}K.B \leq \Lambda X{:}K.C}\,\textsc{NS-OAbs}}{\Gamma \vdash_n \Lambda X{:}K.A \leq \Lambda X{:}K.C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma,\, X{:}K \vdash_n A \leq B \quad \Gamma,\, X{:}K \vdash_n B \leq C}{\Gamma,\, X{:}K \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n \Lambda X{:}K.A \leq \Lambda X{:}K.C}\,\textsc{NS-OAbs}}$$

7. $$\boxed{\dfrac{\dfrac{\Gamma \vdash_n \mathit{lub}_\Gamma(A\,S)^{\mathrm{nf}} \leq B}{\Gamma \vdash_n A\,S \leq B}\,\textsc{NS-OApp} \qquad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n A\,S \leq C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma \vdash_n (\mathit{lub}_\Gamma(A\,S))^{\mathrm{nf}} \leq B \quad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n \mathit{lub}_\Gamma(A\,S))^{\mathrm{nf}} \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n A\,S \leq C}\,\textsc{NS-OApp}}$$

8. $$\boxed{\dfrac{\dfrac{\forall i \in \{1..n\}\ \Gamma \vdash_n A \leq A_i}{\Gamma \vdash_n A \leq \bigwedge^K[A_1..A_n]}\,\textsc{NS-}\forall \qquad \dfrac{\exists j \in \{1..n\}\ \Gamma \vdash_n A_j \leq B}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq B}\,\textsc{NS-}\exists}{\Gamma \vdash_n A \leq B}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\exists j \in \{1..n\}\ \Gamma \vdash_n A \leq A_j \quad \Gamma \vdash_n A_j \leq B}{\Gamma \vdash_n A \leq B}\,\textsc{NS-Trans}}$$

9. $$\boxed{\dfrac{\Gamma \vdash_n A \leq B \qquad \dfrac{\forall i \in \{1..n\}\ \Gamma \vdash_n B \leq A_i}{\Gamma \vdash_n B \leq \bigwedge^K[A_1..A_n]}\,\textsc{NS-}\forall}{\Gamma \vdash_n A \leq \bigwedge^K[A_1..A_n]}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\forall i \in \{1..n\}\ \Gamma \vdash_n A \leq B \quad \Gamma \vdash_n B \leq A_i}{\forall i \in \{1..n\}\ \Gamma \vdash_n A \leq A_i}\,\textsc{NS-Trans}}{\Gamma \vdash_n A \leq \bigwedge^K[A_1..A_n]}\,\textsc{NS-}\forall}$$

10. $$\boxed{\dfrac{\dfrac{\exists j \in \{1..n\}\ \Gamma \vdash_n A_j \leq B}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq B}\,\textsc{NS-}\exists \qquad \Gamma \vdash_n B \leq A}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq A}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\exists j \in \{1..n\}\ \Gamma \vdash_n A_j \leq B \quad \Gamma \vdash_n B \leq A}{\exists j \in \{1..n\}\ \Gamma \vdash_n A_j \leq A}\,\textsc{NS-Trans}}{\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq A}\,\textsc{NS-}\exists}$$

11. $$\boxed{\dfrac{\dfrac{\exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le A}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le A}\ \text{NS-}\exists \qquad \dfrac{\forall i \in \{1..n\}\ \Gamma \vdash_n A \le B_i}{\Gamma \vdash_n A \le \bigwedge^K[B_1..B_n]}\ \text{NS-}\forall}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[B_1..B_n]}\ \text{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le A \quad \forall i \in \{1..n\}\ \Gamma \vdash_n A \le B_i}{\forall i \in \{1..n\}\ \exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le B_i}\ \text{NS-Trans}}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[B_1..B_n]}\ \text{NS-}\forall\exists}$$

12. $$\boxed{\dfrac{\dfrac{\forall i \in \{1..n\}\ \exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le B_i}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[B_1..B_n]} \quad \dfrac{\forall k \in \{1..r\}\ \exists i \in \{1..n\}\ \Gamma \vdash_n B_i \le C_k}{\Gamma \vdash_n \bigwedge^K[B_1..B_n] \le \bigwedge^K[C_1..C_r]}\ \text{NS-}\forall\exists}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[C_1..C_r]}\ \text{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\forall k \in \{1..r\}\ \exists i \in \{1..n\}\ \exists j \in \{1..m\} \quad \Gamma \vdash_n A_j \le B_i \qquad \Gamma \vdash_n B_i \le C_k}{\forall k \in \{1..r\}\ \exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le C_k}\ \text{NS-Trans}}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[C_1..C_r]}\ \text{NS-}\forall\exists}$$

13. $$\boxed{\dfrac{\dfrac{\forall i \in \{1..n\}\ \exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le B_i}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le \bigwedge^K[B_1..B_n]}\ \text{NS-}\forall\exists \quad \dfrac{\exists i \in \{1..n\}\ \Gamma \vdash_n B_i \le C}{\Gamma \vdash_n \bigwedge^K[B_1..B_n] \le C}\ \text{NS-}\exists}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le C}\ \text{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\exists j \in \{1..m\}\ \exists i \in \{1..n\}\ \Gamma \vdash_n A_j \le B_i \quad \Gamma \vdash_n B_i \le C}{\exists j \in \{1..m\}\ \Gamma \vdash_n A_j \le C}\ \text{NS-Trans}}{\Gamma \vdash_n \bigwedge^K[A_1..A_m] \le C}\ \text{NS-}\exists}$$

14. $$\boxed{\dfrac{\dfrac{\forall i \in \{1..n\}\ \Gamma \vdash_n A \le B_i}{\Gamma \vdash_n A \le \bigwedge^K[B_1..B_n]}\ \text{NS-}\forall \quad \dfrac{\forall k \in \{1..r\}\ \exists i \in \{1..n\}\ \Gamma \vdash_n B_i \le C_k}{\Gamma \vdash_n \bigwedge^K[B_1..B_n] \le \bigwedge^K[C_1..C_r]}\ \text{NS-}\forall\exists}{\Gamma \vdash_n A \le \bigwedge^K[C_1..C_r]}\ \text{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\forall k \in \{1..r\}\ \exists i \in \{1..n\} \quad \Gamma \vdash_n A \le B_i \qquad \Gamma \vdash_n B_i \le C_k}{\forall k \in \{1..r\}\ \Gamma \vdash_n A \le C_k}\ \text{NS-Trans}}{\Gamma \vdash_n A \le \bigwedge^K[C_1..C_r]}\ \text{NS-}\forall}$$

A derivation of a subtyping statement is in *refl-normal form* if it has no reflexivity redexes and it is in *trans-normal form* if it has no transitivity redexes, and it is in *normal form* if it has neither reflexivity nor transitivity redexes. The elimination of NS-Trans, and the simplification of NS-Refl follow a standard cut-elimination argument.

Lemma 7.5 (*Reflexivity simplification*) Let D be a derivation of a subtyping statement with only one application of NS-Refl. Then D has a refl-normal form.

PROOF: Same argument as in lemma 7.6. □

LEMMA 7.6 (*Transitivity elimination*) Let D be a derivation of a subtyping statement with only one application of NS-TRANS. Then D has a trans-normal form.

PROOF: By induction on the size of D following a case analysis of the last rule of D. If the last rule is not NS-TRANS, then the result follows by the induction hypothesis. Otherwise we consider all possible last rules of the derivations of the premises and note that each possible configuration determines a trans-redex. Finally, observe that each reduction yields either a derivation in normal form or shorter derivations with only one occurrence of NS-TRANS in which case the result follows by the induction hypothesis. □

An immediate corollary of this last result is that transitivity elimination terminates. Given a derivation D of $\Gamma \vdash_n S \leq T$, iterate the previous lemma on all subderivations of D that have only one NS-TRANS application. The number of times the lemma is applied is equal to the number of occurrences of NS-TRANS in D. Furthermore, lemma 7.5 implies that reflexivity simplification terminates. The simplification rules are such that transitivity simplification rules do not create new reflexivity redexes. Therefore, we can reduce all instances of NS-REFL first and then all instances of NS-TRANS, which is a terminating procedure to normalize a derivation. Consequently, we have proved the following corollary.

COROLLARY 7.7 (*Existence of normal derivations*) Given a derivation of $\Gamma \vdash_n S \leq T$. Then there exists a derivation in normal form of $\Gamma \vdash_n S \leq T$.

LEMMA 7.8

1. A derivation in normal form whose last rule is NS-REFL is either a proof of $\Gamma \vdash_n X \leq X$ or $\Gamma \vdash_n A\,T \leq A\,T$.
2. If the last rule of a subtyping derivation D is NS-TRANS, then D is not in normal form.

PROOF:

1. According to the reflexivity elimination rules, any other possible NS-REFL application is a redex.
2. By case analysis of the last rules of the premises of the last rule of D. In each case the result follows either by the induction hypothesis or because the last rule of at least one of the derivations of the premises of D constitutes a redex. □

We can summarize the previous results as follows.

COROLLARY 7.9 If $\Gamma \vdash_n S \leq T$, then there exists a proof of the same judgement with no applications of NS-TRANS and in which NS-REFL is only applied to type variables and type applications.

A consequence of the normalization of proofs is the following generation result.

Proposition 7.10 (*Generation for normal subtyping*)

1. $\Gamma \vdash_n X \leq B$ implies $X \equiv B$ and $\Gamma \vdash X : K$ for some K, or $\Gamma \vdash_n \Gamma(X) \leq B$.
2. $\Gamma \vdash_n S {\to} A \leq B$ implies $B \equiv T {\to} C$, $\Gamma \vdash_n T \leq S$, $\Gamma \vdash_n A \leq C$, and $\Gamma \vdash S {\to} A : \star$.
3. $\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq B$ implies $B \equiv \forall X{\leq}S{:}K.C$, $\Gamma,\, X{\leq}S{:}K \vdash_n A \leq C$, and $\Gamma \vdash \forall X{\leq}S{:}K.A : \star$.
4. $\Gamma \vdash_n \Lambda X{:}K.A \leq B$ implies $B \equiv \Lambda X{:}K.C$ and $\Gamma,\, X{\leq}\top^K{:}K \vdash_n A \leq C$.
5. $\Gamma \vdash_n A\,S \leq B$ implies $B \equiv A\,S$, or $\Gamma \vdash_n (lub_\Gamma(A\,S))^{\mathrm{nf}} \leq B$, and $\Gamma \vdash A\,S : K$.
6. $\Gamma \vdash_n \bigwedge^K[A_1..A_m] \leq B$ implies that there exists $j \in \{1..m\}$ such that $\Gamma \vdash_n A_j \leq B$ and $\forall k \in \{1..m\}\, \Gamma \vdash A_k : K$.
7. $\Gamma \vdash_n A \leq \bigwedge^K[B_1..B_n]$ implies that for each $i \in \{1..n\}$ $\Gamma \vdash_n A \leq B_i$ and $\Gamma \vdash A : K$.
8. $\Gamma \vdash_n \bigwedge^K[A_1..A_m] \leq \bigwedge^K[B_1..B_n]$ implies that for each $i \in \{1..n\}$ there exists $j \in \{1..m\}$ such that $\Gamma \vdash_n A_j \leq B_i$ and $\forall k \in \{1..m\}\, \Gamma \vdash A_k : K$.

Moreover, given a normal proof of any of the antecedents, the proofs of the consequents are proper subderivations.

Proof: In each case, given a proof of the antecedent, there is also a proof in normal form. Due to lemma 7.8(2), such a derivation cannot end with an application of NS-Trans, and, because of lemma 7.8(1), if it ends with NS-Refl, then it is a derivation of a subtyping statement between type variables or type applications. Finally, the result follows by inspection of the other rules. □

Lemma 7.11

1. $\Gamma \vdash_n T \leq \bigwedge^K[A_1..A_n]$ if and only if $\Gamma \vdash_n T \leq A_k$ for each $k \in \{1..n\}$.
2. $\Gamma \vdash_n T \leq \bigwedge^K[A_1..A_n]$ if and only if $\Gamma \vdash_n T \leq \bigwedge^K[A_k]$ for each $k \in \{1..n\}$.
3. Let $\Gamma \vdash \bigwedge^K[A_1..A_n] : K$. Then $\Gamma \vdash_n \bigwedge^K[A_1..A_n] \leq T$ if and only if $\Gamma \vdash_n A_k \leq T$ for some $k \in \{1..n\}$.

Proof: By induction on derivations, using lemma 7.7 and generation. □

8 Equivalence of ordinary and normal subtyping

In this section, we show that a subtyping statement is derivable in $F^\omega_\wedge$ if and only if the corresponding normalized statement is derivable in $NF^\omega_\wedge$. This equivalence is proved in theorem 8.9. As usual, we need some auxiliary properties and definitions, among which we can highlight propositions 8.2 (Soundness) and 8.8 (Completeness).

LEMMA 8.1 Let $lub_\Gamma(S)$ be defined. Then

1. $\Gamma \vdash S : K$ implies $\Gamma \vdash lub_\Gamma(S) : K$.
2. $\Gamma \vdash S \leq lub_\Gamma(S)$.

PROOF: Item 1 follows by induction on derivations, while item 2 follows by induction on the structure of S. □

PROPOSITION 8.2 (*Soundness*) If $\Gamma \vdash_n S \leq T$, then $\Gamma \vdash S \leq T$.

PROOF: By induction on the derivation of $\Gamma \vdash_n S \leq T$. We consider here a few illustrative cases.

NS-TVAR By the induction hypothesis, S-TVAR and S-TRANS.

NS-OAPP By the induction hypothesis, lemma 8.1(2), S-CONV and S-TRANS.

NS-∀∃ We are given that for each k in $\{1..n\}$ $\Gamma \vdash A_k : K$, and for each i in $\{1..m\}$ there is a j in $\{1..n\}$ such that $\Gamma \vdash_n A_j \leq B_i$. By K-MEET, $\Gamma \vdash \bigwedge^K[A_1..A_n] : K$, and, by S-MEET-LB, $\Gamma \vdash \bigwedge^K[A_1..A_n] \leq A_k$ for each k. Hence the result follows by the induction hypothesis, S-TRANS and S-MEET-G. □

Note that the cases for type variable and type application reveal the fact that NS-TVAR and NS-OAPP hide steps of transitivity.

The following lemma says that empty intersections, $\top^K$, are maximal elements of the subtyping order.

LEMMA 8.3

1. $\Gamma \vdash T : K$ implies $\Gamma \vdash_n T \leq \top^K$.
2. $\Gamma \vdash T : K$ implies $\Gamma \vdash T \leq \top^K$.

PROOF: Statement 1 follows by the cases $m = 0$ in NS-∀ and NS-∀∃. Statement 2 is the case $n = 0$ in S-MEET-G. □

LEMMA 8.4

1. $\Gamma \vdash \mathrm{ok}$ implies $\Gamma^{\mathrm{nf}} \vdash \mathrm{ok}$.

2. $\Gamma \vdash T : K$ implies $\Gamma^{\mathrm{nf}} \vdash T : K$.

3. $\Gamma \vdash S \leq T$ implies $\Gamma^{\mathrm{nf}} \vdash S \leq T$.

4. Let $\Gamma_1, \Gamma_2 \vdash \mathrm{ok}$. Then $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash T : K$ implies $\Gamma_1, \Gamma_2 \vdash T : K$.

5. Let $\Gamma_1, \Gamma_2 \vdash \mathrm{ok}$. Then $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash S \leq T$ implies $\Gamma_1, \Gamma_2 \vdash S \leq T$.

6. Let $\Gamma \vdash S, T : K$. Then $\Gamma^{\mathrm{nf}} \vdash S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$ if and only if $\Gamma \vdash S \leq T$.

Proof: Statements 1 and 2 follow by simultaneous induction on the size of derivations using lemma 4.16. Statement 3 follows by induction on the derivation of $\Gamma \vdash S \leq T$ using part 1, part 2, and lemma 4.16. Statement 4 follows by induction on the derivation of $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash T : K$. Item 5 follows by induction on the derivation of $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash S \leq T$, using part 4. Item 6 is a corollary of part 3, part 5 and lemma 4.15. $\square$

In the last lemma, items 1, 2, and 3 show that well formation of contexts, kinding judgements, and subtyping judgements are invariant under normalization of contexts, while items 4 and 5 are the converse of 2 and 3 respectively.

The following lemma states that S-TVar is an admissible rule in $NF^{\omega}_{\wedge}$.

Lemma 8.5 Let Γ be a context in normal form such that $\Gamma \vdash \mathrm{ok}$ and $Y \in \mathrm{dom}(\Gamma)$. Then $\Gamma \vdash_n Y \leq \Gamma(Y)$.

Proof: Let $\Gamma \equiv \Gamma_1, Y \leq T{:}K, \Gamma_2$. By lemma 4.2, $\Gamma_1 \vdash T : K$. If T is not an intersection, then, by NS-Refl and NS-TVar, we have $\Gamma \vdash_n Y \leq T$. If $T \equiv \bigwedge^{K'}[B_1..B_m]$, then by generation for kinding and uniqueness of kinds, $\Gamma \vdash B_i : K$ for each i and $K \equiv K'$. By NS-Refl, $\Gamma \vdash_n B_i \leq B_i$ for each i. Then, by NS-$\exists$ and NS-TVar, it follows that $\Gamma \vdash_n Y \leq B_i$ for each i, and, by NS-$\forall$, $\Gamma \vdash_n Y \leq T$. $\square$

The following lemma shows that the normal subtyping system has the substitution property.

Lemma 8.6 (*Substitution*) If $\Gamma \vdash U : K$ and $\Gamma, X{:}K, \Gamma' \vdash_n S \leq T$, then $\Gamma, (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n (S[X{\leftarrow}U])^{\mathrm{nf}} \leq (T[X{\leftarrow}U])^{\mathrm{nf}}$.

Proof: By induction on the derivation of $\Gamma, X{:}K, \Gamma' \vdash_n S \leq T$. For the sake of clarity, we sometimes leave out kinding judgements and their justifications which follow easily from the structural properties in section 4. We show here a couple of representative cases. Let $\Gamma'' \equiv \Gamma, X{:}K, \Gamma'$.

NS-TVar We are given $\Gamma'' \vdash_n \Gamma''(Y) \leq A$. We have to consider three cases.

1. $Y \equiv X$. By subject reduction, $\Gamma \vdash U^{\mathrm{nf}} : K$, and by lemma 8.3(1), it follows that $\Gamma \vdash_n U^{\mathrm{nf}} \leq \top^K$. By weakening, it follows that $\Gamma, (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n U^{\mathrm{nf}} \leq \top^K$ and, by the induction hypothesis, it follows that $\Gamma, (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n \top^K \leq (A[X{\leftarrow}U])^{\mathrm{nf}}$. Finally, the result follows by NS-Trans.

2. $Y \in \mathrm{dom}(\Gamma)$. By the free variables lemma, $X \notin \mathrm{FV}(\Gamma(Y))$ and $X \not\equiv Y$. By lemmas 4.10, 8.4(1), and 8.5, it follows that $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n Y \leq \Gamma(Y)$, and, by the induction hypothesis, it follows that $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n \Gamma(Y) \leq (A[X{\leftarrow}U])^{\mathrm{nf}}$. Finally, the result follows by NS-Trans.
3. $Y \in \mathrm{dom}(\Gamma')$. By the induction hypothesis, it follows that
$$\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n (\Gamma'(Y)[X{\leftarrow}U])^{\mathrm{nf}} \leq (A[X{\leftarrow}U])^{\mathrm{nf}}.$$
By lemmas 4.10, 8.4(1), and 8.5, $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n Y \leq (\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}})(Y)$. Furthermore, $(\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}})(Y) = (\Gamma'(Y)[X{\leftarrow}U])^{\mathrm{nf}}$. Hence the result follows by NS-Trans.

NS-Arrow We are given that $\Gamma'' \vdash_n T \leq S$ and $\Gamma'' \vdash_n A \leq B$. By the induction hypothesis, $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n (T[X{\leftarrow}U])^{\mathrm{nf}} \leq (S[X{\leftarrow}U])^{\mathrm{nf}}$ and $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n (A[X{\leftarrow}U])^{\mathrm{nf}} \leq (B[X{\leftarrow}U])^{\mathrm{nf}}$. There are four cases to consider, since $(A[X{\leftarrow}U])^{\mathrm{nf}}$ and $(B[X{\leftarrow}U])^{\mathrm{nf}}$ may be intersections or not. We shall consider only two of them to illustrate the proof method.

1. $(A[X{\leftarrow}U])^{\mathrm{nf}}$ and $(B[X{\leftarrow}U])^{\mathrm{nf}}$ are not intersections. Then the result follows by applying NS-Arrow.
2. $(A[X{\leftarrow}U])^{\mathrm{nf}} = \bigwedge^{\star}[C_1..C_n]$ and $(B[X{\leftarrow}U])^{\mathrm{nf}}$ is not an intersection. Then we have that
$$((T{\to}B)[X{\leftarrow}U])^{\mathrm{nf}} = (T[X{\leftarrow}U])^{\mathrm{nf}}{\to}(B[X{\leftarrow}U])^{\mathrm{nf}} \quad \text{and}$$
$$((S{\to}A)[X{\leftarrow}U])^{\mathrm{nf}}$$
$$= \bigwedge^{\star}[(S[X{\leftarrow}U])^{\mathrm{nf}}{\to}C_1..(S[X{\leftarrow}U])^{\mathrm{nf}}{\to}C_n].$$
By lemma 7.10, it follows that for some i $\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n C_i \leq (B[X{\leftarrow}U])^{\mathrm{nf}}$. Applying NS-Arrow,
$\Gamma,\ (\Gamma'[X{\leftarrow}U])^{\mathrm{nf}} \vdash_n (S[X{\leftarrow}U])^{\mathrm{nf}}{\to}C_i \leq (T[X{\leftarrow}U])^{\mathrm{nf}}{\to}(B[X{\leftarrow}U])^{\mathrm{nf}}$. Finally, the result follows by NS-$\exists$. □

This substitution lemma is the key result we use in proving that S-OApp has a corresponding admissible rule in $NF^{\omega}_{\wedge}$.

Lemma 8.7 $\Gamma \vdash S\,U : K$. Then $\Gamma \vdash_n S \leq T$ implies $\Gamma \vdash_n (S\,U)^{\mathrm{nf}} \leq (T\,U)^{\mathrm{nf}}$.

Proof: By induction on the derivation of $\Gamma \vdash_n S \leq T$, assuming a derivation in normal form. The cases for NS-Arrow and NS-All are impossible because of the assumption $\Gamma \vdash S\,U : K$. We show here the interesting cases.

NS-TVar We are given $\Gamma \vdash_n \Gamma(X) \leq A$. By the induction hypothesis, $\Gamma \vdash_n (\Gamma(X)U)^{\mathrm{nf}} \leq (AU)^{\mathrm{nf}}$. We have to consider two cases.

$(AU)^{\mathrm{nf}} \equiv B$ By NS-OApp.

$(AU)^{\mathrm{nf}} \equiv \bigwedge^K[A_1..A_n]$ By lemma 7.11, $\Gamma \vdash_n (\Gamma(X)U)^{\mathrm{nf}} \leq A_k$ for each k in $\{1..n\}$. By NS-OApp, $\Gamma \vdash_n X\,U \leq (A_k)$ for each k, which, by NS-$\forall$, implies $\Gamma \vdash_n X\,U \leq (AU)^{\mathrm{nf}}$.

NS-OAbs We are given $\Gamma, X{:}K \vdash_n A \leq B$. By the substitution lemma 8.6, it follows that $\Gamma \vdash_n (A[X{\leftarrow}U])^{\mathrm{nf}} \leq (B[X{\leftarrow}U])^{\mathrm{nf}}$. On the other hand, we have that $(\Lambda X{:}K.A)U \to_{\beta\wedge} A[X{\leftarrow}U]$ and $(\Lambda X{:}K.B)U \to_{\beta\wedge} B[X{\leftarrow}U]$. Finally, the result follows by the uniqueness of normal forms. $\square$

Proposition 8.8 (*Completeness*) If $\Gamma \vdash S \leq T$, then $\Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

Proof: By induction on the derivation of $\Gamma \vdash S \leq T$, using lemma 8.7 for the case of S-OApp. $\square$

Theorem 8.9 (*Equivalence of ordinary and normal subtyping*) Let $\Gamma \vdash S : K$ and $\Gamma \vdash T : K$. Then $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

Proof: $\Rightarrow$) By completeness (8.8). $\Leftarrow$) By soundness (8.2), it follows that $\Gamma^{\mathrm{nf}} \vdash S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$, and, by lemma 8.4(6), it follows that $\Gamma \vdash S \leq T$. $\square$

8.1 Least strict upper bound

So far we only used that $lub_\Gamma(S)$ is an upper bound of S in the context Γ (See lemma 8.1(2)). We can now give the final motivation of the name we chose, showing that if $lub_\Gamma(S)$ is defined and $T \not=_{\beta\wedge} S$, then $\Gamma \vdash S \leq T$ implies $\Gamma \vdash lub_\Gamma(S) \leq T$. We first show that the corresponding property holds for the normalized system.

Lemma 8.1.1 Let $lub_\Gamma(S)$ be defined. Then

1. If $S \twoheadrightarrow_{\beta\wedge} S'$ and $\Gamma \twoheadrightarrow_{\beta\wedge} \Gamma'$, then $lub_\Gamma(S) \twoheadrightarrow_{\beta\wedge} lub_{\Gamma'}(S')$.
2. If $\Gamma \vdash_n S \leq T$, then $\Gamma \vdash_n lub_\Gamma(S)^{\mathrm{nf}} \leq T$ or $S \equiv T$.

Proof:

1. By induction on the structure of S, observing that if $lub_\Gamma(S)$ is defined, so is $lub_{\Gamma'}(S')$.
2. By induction on the derivation of $\Gamma \vdash_n S \leq T$. It is immediate for the case NS-Refl; for NS-Arrow, NS-All, and NS-OAbs $lub_\Gamma(S)$ is not defined; for the other rules the result follows using the induction hypothesis. $\square$

Corollary 8.1.2 Let $lub_\Gamma(S)$ be defined. Then $\Gamma \vdash S \leq T$ and $T \not=_{\beta\wedge} S$ implies $\Gamma \vdash lub_\Gamma(S) \leq T$.

Proof: By completeness, it follows that $\Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$. By lemma 8.1.1(2), $\Gamma^{\mathrm{nf}} \vdash_n (lub_{\Gamma^{\mathrm{nf}}}(S^{\mathrm{nf}}))^{\mathrm{nf}} \leq T^{\mathrm{nf}}$, because $S^{\mathrm{nf}} \not\equiv T^{\mathrm{nf}}$. By soundness, it follows that $\Gamma^{\mathrm{nf}} \vdash (lub_{\Gamma^{\mathrm{nf}}}(S^{\mathrm{nf}}))^{\mathrm{nf}} \leq T^{\mathrm{nf}}$, which is equivalent to $\Gamma \vdash lub_{\Gamma^{\mathrm{nf}}}(S^{\mathrm{nf}}) \leq T$ by lemma 8.4(6). Finally, (using lemmas 8.1(1) and 4.11, and proposition 4.18 to get the corresponding kinding judgements) it follows that $\Gamma \vdash lub_\Gamma(S) \leq T$ by lemma 8.1.1(1), S-Conv and S-Trans. $\square$

8.2 Example

In this section, we give the derivation in $NF^{\omega}_{\wedge}$ of the example 1 mentioned in the introduction and in section 6.1.

Let $\Gamma \equiv W{:}K,\ X \leq \Lambda Y{:}K.Y{:}K{\to}K,\ Z \leq X{:}K{\to}K$. We present a proof in the normal system of $\Gamma^{nf} \vdash_n X(Z\,W)^{nf} \leq W^{nf}$, Which is the translation of $\Gamma \vdash X(Z\,W) \leq W$. Observe that $\Gamma^{nf} \equiv \Gamma$, $(X(Z\,W))^{nf} \equiv X(Z\,W)$, and $W^{nf} \equiv W$. For the sake of readability we omit kinding judgements. Observe that the derivation in normal form in $NF^{\omega}_{\wedge}$ is substantially shorter than the one in $F^{\omega}_{\wedge}$ shown in section 6.1.

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash W : K}{\Gamma \vdash_n ((\Lambda Y{:}K.Y)W)^{nf} \leq W}\ \text{NS-Refl}}{\Gamma \vdash_n X\,W \leq W}\ \text{NS-OApp}}{\Gamma \vdash_n ((\Lambda Y{:}K.Y)(Z\,W))^{nf} \leq W}\ \text{NS-OApp}}{\Gamma \vdash_n X(Z\,W) \leq W}\ \text{NS-OApp}$$

9 A subtype checking algorithm, $AlgF^{\omega}_{\wedge}$

As it stands, $NF^{\omega}_{\wedge}$ as defined in section 6.1 is not a deterministic algorithm, because its rules are not syntax directed. Fortunately, we are not far away from an algorithmic presentation. In fact, corollary 7.9 is the bridge to the algorithmic presentation of the subtyping relation, $AlgF^{\omega}_{\wedge}$, which states that transitivity steps can be eliminated and reflexivity steps can be simplified. $AlgF^{\omega}_{\wedge}$ is obtained from $NF^{\omega}_{\wedge}$ by removing NS-Trans and restricting NS-Refl to type variables and type applications.

The new reflexivity rules are:

$$\frac{\Gamma \vdash X : K}{\Gamma \vdash_{Alg} X \leq X} \qquad \text{(AlgS-TVarRefl)}$$

$$\frac{\Gamma \vdash T\,S : K}{\Gamma \vdash_{Alg} T\,S \leq T\,S} \qquad \text{(AlgS-OAppRefl)}$$

Lemma 9.1 (*Equivalence of normal and algorithmic subtyping*)
Let $\Gamma \vdash S, T : K$. Then $\Gamma \vdash_n S \leq T$ if and only if $\Gamma \vdash_{Alg} S \leq T$.

Proof: ($\Rightarrow$) By corollary 7.9. ($\Leftarrow$) Immediate. □

We have thereby proved that $AlgF^{\omega}_{\wedge}$ is indeed a sound and complete algorithm to compute $F^{\omega}_{\wedge}$'s subtyping relation.

Proposition 9.2 (*Equivalence of ordinary and algorithmic subtyping*)
Let $\Gamma \vdash S : K$ and $\Gamma \vdash T : K$. Then $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

Proof: By the equivalence of ordinary and normal subtyping (theorem 8.9) and the equivalence of normal and algorithmic subtyping (lemma 9.1). □

10 Type checking and type reconstruction

Given a context Γ, a term e, and a type T, type checking consists of analyzing whether the judgement $\Gamma \vdash e : T$ is derivable from a given set of inference rules. Type checking algorithms for lambda calculi, unless they are formulated using Gentzen's sequent calculus style, involve *guessing* the type of subterms. For example, when e is $e_1\, e_2$, the type of e_2 is not necessarily a subexpression of T, and in order to corroborate or to refute the assertion $\Gamma \vdash e : T$ we need to *infer* a type for e_2.

In this section, we present an algorithm for inferring minimal types in $F^{\omega}_{\wedge}$. Given Γ and e, the type S constructed by the algorithm is a subtype of every T such that $\Gamma \vdash e : T$. In this way, we reduce the problem of whether $\Gamma \vdash e : T$ to that of inferring a type S such that $\Gamma \vdash e : S$ and $\Gamma \vdash S \leq T$. Solving this problem involves not only the typing rules but all the inference rules of $F^{\omega}_{\wedge}$: the rule T-Subsumption depends on a subtyping judgement, the rule T-Var depends on an ok judgement, and the ok judgements depend on kinding judgements. Consequently, type checking uses the full power of the $F^{\omega}_{\wedge}$ system.

As an example, consider type checking the following judgement:

$$\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash f\; a : T_2.$$

The application $f\, a$ can only be formed if f has an arrow type. Using T-Var we can assign type X to f, which means that in order to obtain an arrow type for f we have to *replace* X by its bound, which has the right form. Observe how this *replacement* is performed by T-Subsumption in the following derivation.

$$\dfrac{\dfrac{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash \mathrm{ok}}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash \boxed{f : X}} \quad \dfrac{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash \mathrm{ok}}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash X \leq T_1{\to}T_2}}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash \boxed{f : T_1{\to}T_2}}\ \textsc{T-Sub}$$

$$\dfrac{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash f : T_1{\to}T_2 \quad \dfrac{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash \mathrm{ok}}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash a : T_1}}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}T_1 \vdash f\; a : T_2}\ \textsc{T-App}$$

Note that, in the presence of T-Subsumption, we may actually perform the application when the type of a is a subtype of T_1. Namely, if

$$\dfrac{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, \boxed{a{:}U_1} \vdash a : U_1 \qquad \Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}U_1 \vdash U_1 \leq T_1}{\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}U_1 \vdash \boxed{a : T_1}}\ \textsc{T-Sub}$$

Moreover, we may want to check whether $\Gamma,\, X \leq T_1{\to}T_2,\, f{:}X,\, a{:}U_1 \vdash f\; a : U_2$, where T_2 is a subtype of U_2.

The situation gets more complicated if f has an intersection type. Suppose that

$$\Gamma,\, X \leq T_1{\to}T_2,\, Y \leq S_1{\to}S_2,\, f{:}X \wedge Y \wedge \forall Z{\leq}V_1{:}K.V_2,\, a{:}U_1 \vdash f\; a : U_2,$$

where U_1 is a subtype of T_1 and S_1. An algorithm should not consider the type $\forall Z{\leq}V_1{:}K.V_2$ for f since, in this case, f is applied to a term and not to a type. Then it has to replace X and Y by their bounds, $T_1{\to}T_2$ and $S_1{\to}S_2$. Moreover, given that the type of a, U_1, is a subtype of both S_1 and T_1, it should check whether $S_2 \wedge T_2$ is a subtype of U_2.

Another source of problems in the search for an algorithmic presentation of the typing rules is that types may not be in normal form. Consider the judgement

$$\Gamma,\, X \leq T_1{\to}T_2,\, Z \leq \Lambda Y{:}\star.Y,\, f{:}Z\,X,\, a{:}T_1 \vdash f\,a : T_2, \tag{2}$$

In order to type the application, f should be assigned type $T_1{\to}T_2$. To do that, Z should be replaced by its bound in $Z\,X$. This replacement produces a type which is not in normal form, so $\Lambda Y{:}\star.Y\,X$ has to be normalized to obtain X. Finally, X is replaced by its bound and then the application can be typed.

The main new source of difficulty is the interaction between the need for normalization and the presence of intersection types.

An algorithm to infer types should proceed structurally on the form of the term whose type is to be inferred. This requires us to remove the rules which make our typing rules non-deterministic: we should eliminate T-SUBSUMPTION and T-MEET from the original presentation, and modify the other rules in such a way that we can still type the same set of terms.

We give some preliminary definitions and results before presenting the rules of our new system:

- We define the mapping *flub*, which performs the "replacements" which we motivated with the previous examples.
- We define the function *arrows*, to filter arrow types in order to deal with term application.
- We define the function *alls* to filter polymorphic types to deal with type application.

The function *lub* (definition 6.1.3) is a partial function which is only defined for type variables and type applications. Here, we extend the definition of *lub* to intersection types in such a way that it is defined if the least upper bound is defined for at least one of the types in the intersection.

DEFINITION 10.1 (*Homomorphic extension of lub to intersections, lub**)

$$\begin{array}{rcll} lub^*_\Gamma(X) & = & \Gamma(X), & \\ lub^*_\Gamma(S\,T) & = & lub^*_\Gamma(S)\,T, & \\ lub^*_\Gamma(\bigwedge^K[T_1..T_n]) & = & \bigwedge^K[T'_1..T'_n], & \text{if } \exists i \in \{1..n\} \text{ such that } lub^*_\Gamma(T_i)\!\downarrow, \end{array}$$

where T'_i is $lub^*_\Gamma(T_i)$, if $lub^*_\Gamma(T_i)\!\downarrow$, and T_i otherwise, and $T\!\downarrow$ means T is defined.

LEMMA 10.2 If $lub^*_\Gamma(T)$ is defined, then $\Gamma \vdash T \leq lub^*_\Gamma(T)$.

PROOF: By induction on the complexity of T, using corollary 4.14. $\square$

We define the mapping *flub* which given a type T (and a context Γ) finds the smallest type larger than T (with respect to the subtype relation) having structural information to perform an application.

DEFINITION 10.3 (*Functional Least Upper Bound*) The functional least upper bound of a type T, in a context Γ, $\mathit{flub}_\Gamma(T)$ is defined as follows.

$$\mathit{flub}_\Gamma(T) = \begin{cases} \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(T^{\mathrm{nf}})), & \text{if } \mathit{lub}^*_\Gamma(T^{\mathrm{nf}})\downarrow; \\ T^{\mathrm{nf}}, & \text{otherwise.}^1 \end{cases}$$

The intuition behind the definition of the function *flub* is to find $T_1{\to}T_2$ starting form $Z X$ in the example 2 above. In other words, $\mathit{flub}_\Gamma(Z\,X) = T_1{\to}T_2$. For simplicity we assume $T_1{\to}T_2$ in normal form. Step by step,

$$\begin{aligned} \mathit{flub}_\Gamma(Z\,X) &= \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(Z\,X)) \\ &= \mathit{flub}_\Gamma((\Lambda Y{:}K.Y)\,X) \\ &= \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(((\Lambda Y{:}K.Y)\,X)^{\mathrm{nf}})) \\ &= \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(X)) \\ &= \mathit{flub}_\Gamma(T_1{\to}T_2) \\ &= T_1{\to}T_2. \end{aligned}$$

More generally, *flub* climbs the subtyping hierarchy until it finds an arrow, a quantifier, or an intersection of these two.

LEMMA 10.4 Let $\Gamma \vdash S, T : \star$ and $S =_{\beta\wedge} T$. Then $\mathit{flub}_\Gamma(S) \equiv \mathit{flub}_\Gamma(T)$.

DEFINITION 10.5 (*arrows* and *alls*)

1. $$\begin{aligned} \mathit{arrows}\,(T_1{\to}T_2) &= \{T_1{\to}T_2\}, \\ \mathit{arrows}\,(\bigwedge^\star[T_1..T_n]) &= \cup_{i\in\{1..n\}}\,\mathit{arrows}\,(T_i), \\ \mathit{arrows}\,(T) &= \emptyset, \quad \text{if } T \not\equiv T_1{\to}T_2 \text{ and } T \not\equiv \textstyle\bigwedge^\star[T_1..T_n]. \end{aligned}$$

2. $$\begin{aligned} \mathit{alls}\,(\forall X{\leq}T_1{:}K.T_2) &= \{\forall X{\leq}T_1{:}K.T_2\}, \\ \mathit{alls}\,(\bigwedge^\star[T_1..T_n]) &= \cup_{i\in\{1..n\}}\,\mathit{alls}\,(T_i), \\ \mathit{alls}\,(T) &= \emptyset, \quad \text{if } T \not\equiv \forall X{\leq}T_1{:}K.T_2 \text{ and } T \not\equiv \textstyle\bigwedge^\star[T_1..T_n]. \end{aligned}$$

[1] This step can be optimised in an implementation of the type checking algorithm, allowing us to avoid the normalization of T when T is either an arrow type or a quantified type.

The situation here is significantly more complex than in [34] for $F_\wedge$, an extension of the second order λ-calculus. There it is enough to recursively search for arrows or polymorphic types in the context, because in $F_\wedge$ there is no reduction on types. The information to be searched for is explicit in the context, so the job done here by *flub* is simply an extra case in the definition of *arrows* and *alls*. Namely,

$$\begin{array}{rcl} \mathit{arrows}(X) & = & \mathit{arrows}(\Gamma(X)) \quad \text{and} \\ \mathit{alls}(X) & = & \mathit{alls}(\Gamma(X)). \end{array}$$

Moreover, to prove that *flub* is well-founded is similar for us in complexity to proving termination of subtype checking. The similarity comes from the fact that computing *flub* involves replacing variables by their bounds in a given context and normalizing with respect to $\rightarrow_{\beta\wedge}$. A proof of the well-foundation of *flub* can be found in [19]. In contrast, in [34] it is enough to observe that well-formed contexts cannot contain cycles of variable references.

Notation 10.6 We introduce a new notation for intersection types. The intersection of all types T such that $\phi(T)$ holds is written $\bigwedge^K[T \mid \phi(T)]$. Note that this is an alternative notation to $\bigwedge^K[T_1..T_n]$ such that $\phi(T_i)$ holds if and only if $i \in \{1..n\}$.

We can now define a type inference algorithm for $F^\omega_\wedge$.

Definition 10.7 (*A type inference algorithm*, inf)

$$\frac{\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \mathrm{ok}}{\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash_{\mathrm{inf}} x : T} \qquad (\text{AT-Var})$$

$$\frac{\Gamma,\, x{:}T_1 \vdash_{\mathrm{inf}} e : T_2}{\Gamma \vdash_{\mathrm{inf}} \lambda x{:}T_1.e : T_1 \to T_2} \qquad (\text{AT-Abs})$$

$$\frac{\Gamma \vdash_{\mathrm{inf}} f : T \qquad \Gamma \vdash_{\mathrm{inf}} a : S}{\Gamma \vdash_{\mathrm{inf}} f\, a : \bigwedge^\star[T_i \mid S_i \to T_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T)) \text{ and } \Gamma \vdash S \leq S_i]} \qquad (\text{AT-App})$$

$$\frac{\Gamma,\, X{\leq}T_1{:}K_1 \vdash_{\mathrm{inf}} e : T_2}{\Gamma \vdash_{\mathrm{inf}} \lambda X{\leq}T_1{:}K_1.e : \forall X{\leq}T_1{:}K_1.T_2} \qquad (\text{AT-TAbs})$$

$$\frac{\Gamma \vdash_{\mathrm{inf}} f : T}{\Gamma \vdash_{\mathrm{inf}} f\, S : \bigwedge^\star[T_i[X{\leftarrow}S] \mid \forall X{\leq}S_i{:}K.T_i \in \mathit{alls}(\mathit{flub}_\Gamma(T)) \text{ and } \Gamma \vdash S \leq S_i]} \qquad (\text{AT-TApp})$$

$$\frac{\text{for all } i \in \{1..n\} \quad \Gamma \vdash_{\mathrm{inf}} e[X{\leftarrow}S_i] \in T_i}{\Gamma \vdash_{\mathrm{inf}} \mathrm{for}(X{\in}S_1..S_n)e : \bigwedge^\star[T_1..T_n]} \qquad (\text{AT-For})$$

The algorithmic information of rule AT-App is that in order to find a type for $f\,a$ in Γ, we need to infer a type S for a and a type T for f, and to take the intersection of all the $T_i's$ such that $T_i \to S_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i$.

11 Minimal typing

In this section we show that $F^\omega_\wedge$ satisfies the minimal typing property (theorem 11.11). We first prove that the algorithm *inf* is sound with respect to $F^\omega_\wedge$: if $\Gamma \vdash_{\mathsf{inf}} e : T$, then $\Gamma \vdash e : T$ (proposition 11.4). We then prove that every closed term is typeable using either set of typing rules (lemma 11.8). Finally, we prove that *inf* computes minimal types for $F^\omega_\wedge$ terms (proposition 11.10).

LEMMA 11.1 Let $\Gamma \vdash T : \star$. Then $\Gamma \vdash T \leq \mathit{flub}_\Gamma(T)$.

PROOF: Since *flub* is well-founded, we can proceed by induction on the number of unfolding steps in $\mathit{flub}_\Gamma(T)$. If $\mathit{flub}_\Gamma(T) = T^{\mathrm{nf}}$, the result follows by S-CONV. Otherwise, $\mathit{flub}_\Gamma(T) = \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(T^{\mathrm{nf}}))$. By S-CONV, $\Gamma \vdash T \leq T^{\mathrm{nf}}$. By lemma 10.2, $\Gamma \vdash T^{\mathrm{nf}} \leq \mathit{lub}^*_\Gamma(T^{\mathrm{nf}})$. By the induction hypothesis, $\Gamma \vdash \mathit{lub}^*_\Gamma(T^{\mathrm{nf}}) \leq \mathit{flub}_\Gamma(\mathit{lub}^*_\Gamma(T^{\mathrm{nf}}))$. Finally, by S-TRANS, the result follows. □

LEMMA 11.2 Let $\Gamma \vdash T : \star$. Then

1. $\Gamma \vdash T \leq \bigwedge^\star[S \mid S \in \mathit{arrows}(\mathit{flub}_\Gamma(T))]$.
2. $\Gamma \vdash T \leq \bigwedge^\star[S \mid S \in \mathit{alls}(\mathit{flub}_\Gamma(T))]$.

PROOF: Item 1: Using lemma 11.1, we reduce our problem to proving that
$$\Gamma \vdash T \leq \bigwedge^\star[S \mid S \in \mathit{arrows}(T)],$$
which follows by induction on the structure of T. Item 2 follows similarly. □

LEMMA 11.3

1. If $\Gamma \vdash T \leq T_1 \to T_2$, then $\Gamma \vdash \bigwedge^\star[S \mid S \in \mathit{arrows}(\mathit{flub}_\Gamma(T))] \leq T_1 \to T_2$.
2. If $\Gamma \vdash T \leq \forall X {\leq} T_1{:}K.T_2$, then $\Gamma \vdash \bigwedge^\star[S \mid S \in \mathit{alls}(\mathit{flub}_\Gamma(T))] \leq \forall X {\leq} T_1{:}K.T_2$.

PROOF: We show only case 1 here, case 2 is similar. By induction on the derivation of $\Gamma \vdash T \leq T_1 \to T_2$. The last rule of a derivation of this subtyping statement can only be S-CONV, S-TVAR, S-TRANS, or S-MEET-LB. The first three cases use similar arguments, therefore we consider here only the cases for S-CONV and S-MEET-LB.

S-CONV We are given that $T =_{\beta\wedge} T_1 \to T_2$. By lemma 10.4 and the definition of *flub*, we have that: $\mathit{arrows}(\mathit{flub}_\Gamma(T)) = \mathit{arrows}(\mathit{flub}_\Gamma(T_1 \to T_2)) = \mathit{arrows}((T_1 \to T_2)^{\mathrm{nf}})$

We now have two cases to analyze.

1. If $(T_1 \to T_2)^{\mathrm{nf}} = T_1^{\mathrm{nf}} \to T_2^{\mathrm{nf}}$, then the result follows by S-MEET-LB and S-CONV.
2. Otherwise, let $(T_1 \to T_2)^{\mathrm{nf}} = \bigwedge^\star[T_1^{\mathrm{nf}} \to U_1..T_1^{\mathrm{nf}} \to U_n]$, where $T_2^{\mathrm{nf}} = \bigwedge^\star[U_1..U_n]$. Then, $\mathit{arrows}(\mathit{flub}_\Gamma(T)) = \{T_1^{\mathrm{nf}} \to U_1..T_1^{\mathrm{nf}} \to U_n\}$. Consequently, $\bigwedge^\star[S \mid S \in \mathit{arrows}(\mathit{flub}_\Gamma(T))] = (T_1 \to T_2)^{\mathrm{nf}}$, and the result follows by S-CONV.

S-Meet-LB We are given that $\Gamma \vdash \bigwedge^\star[S_1..T_1{\to}T_2..S_n] \leq T_1{\to}T_2$. By the definition of *flub*, $\mathit{flub}_\Gamma(\bigwedge^\star[S_1..T_1{\to}T_2..S_n]) = \bigwedge^\star[....T_1^{\mathrm{nf}}{\to}A_1..T_1^{\mathrm{nf}}{\to}A_m....]$, where $T_2^{\mathrm{nf}} = \bigwedge^\star[A_1..A_m]$ or $T_2^{\mathrm{nf}} = A_1$. Now, $\mathit{arrows}(\mathit{flub}_\Gamma(\bigwedge^\star[S_1..T_1{\to}T_2..S_n])) \supseteq \{T_1^{\mathrm{nf}}{\to}A_1..T_1^{\mathrm{nf}}{\to}A_m\}$. Then, if $T_2^{\mathrm{nf}} = \bigwedge^\star[A_1..A_m]$, by lemma 4.17; and, if $T_2^{\mathrm{nf}} = A_1$, by S-Meet-LB, we have that

$$\Gamma \vdash \bigwedge^\star[S \mid S \in \mathit{arrows}(\mathit{flub}_\Gamma(\bigwedge^\star[S_1..T_1{\to}T_2..S_n]))] \leq (T_1{\to}T_2)^{\mathrm{nf}}.$$

Finally, the result follows by S-Conv. □

Proposition 11.4 (*Soundness of inf*) If $\Gamma \vdash_{\mathrm{inf}} e : T$, then $\Gamma \vdash e : T$.

Proof: By induction on the derivation of $\Gamma \vdash_{\mathrm{inf}} e : T$. The interesting cases are when the last applied rule is either AT-App and AT-TApp.

AT-App $\Gamma \vdash_{\mathrm{inf}} f\,a : \bigwedge^\star[T_i \mid S_i{\to}T_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i]$ is derived from $\Gamma \vdash_{\mathrm{inf}} f : T$ $\Gamma \vdash_{\mathrm{inf}} a : S$. If $\bigwedge^\star[T_i \mid S_i{\to}T_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i] =_{\beta\wedge} \top^\star$, then the result follows immediately using T-Meet. Otherwise, by the induction hypothesis, we have that $\Gamma \vdash f : T$. By lemma 11.2(1), S-Meet-LB, S-Trans, and T-Subsumption, $\Gamma \vdash f : S_i{\to}T_i$. By the induction hypothesis and T-Subsumption, $\Gamma \vdash a : S_i$. By T-App, $\Gamma \vdash f\,a : T_i$. Finally, by T-Meet,

$$\Gamma \vdash f\,a : \bigwedge^\star[T_i \mid S_i{\to}T_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T)) \text{ and } \Gamma \vdash S \leq S_i].$$

AT-TApp $\Gamma \vdash_{\mathrm{inf}} f\,S : \bigwedge^\star[T_i[X{\leftarrow}S] \mid \forall X{\leq}S_i{:}K.T_i \in \mathit{alls}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i]$ is derived from $\Gamma \vdash_{\mathrm{inf}} f : T$. If $\bigwedge^\star[T_i \mid S_i{\to}T_i \in \mathit{arrows}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i] =_{\beta\wedge} \top^\star$, then the result follows immediately, using T-Meet. Otherwise, assume

$$\mathit{alls}(\mathit{flub}_\Gamma(T)) \equiv \bigwedge^\star[\forall X{\leq}S_1{:}K.T_1..\forall X{\leq}S_n{:}K.T_n].$$

By the induction hypothesis, we have that, $\Gamma \vdash f : T$. By lemma 11.2(2), S-Meet-LB, S-Trans, and T-Subsumption, it follows that

$\Gamma \vdash f : \forall X{\leq}S_i{:}K.T_i$. Since $\Gamma \vdash S \leq S_i$, by T-App, $\Gamma \vdash f\,S : T_i[X{\leftarrow}S]$. Finally, by T-Meet, $\Gamma \vdash f\,S : \bigwedge^\star[T_i[X{\leftarrow}S] \mid \forall X{\leq}S_i{:}K.T_i \in \mathit{alls}(\mathit{flub}_\Gamma(T))$ and $\Gamma \vdash S \leq S_i]$. □

Lemma 11.5 (*Term application*)
If $\Gamma \vdash \bigwedge^\star[S_1{\to}T_1..S_n{\to}T_n] \leq S{\to}T$ and $\Gamma \vdash U \leq S$,
then $\Gamma \vdash \bigwedge^\star[T_j \mid \Gamma \vdash U \leq S_j] \leq T$.

Proof: There are two cases to be considered according to the normal form of $S{\to}T$. The case when $(S{\to}T)^{\mathrm{nf}} \equiv S^{\mathrm{nf}}{\to}T^{\mathrm{nf}}$ is similar to but simpler than the one we consider here. Assume

$$(S{\to}T)^{\mathrm{nf}} \equiv \bigwedge^\star[S^{\mathrm{nf}}{\to}A_1..S^{\mathrm{nf}}{\to}A_m], \text{ where } T^{\mathrm{nf}} \equiv \bigwedge^\star[A_1..A_m].$$

By the equivalence of ordinary and normal subtyping (theorem 8.9),

$$\Gamma^{\mathrm{nf}} \vdash_n \bigwedge^\star[S_1^{\mathrm{nf}}{\to}B_1^1..S_1^{\mathrm{nf}}{\to}B_1^{k_1}..S_n^{\mathrm{nf}}{\to}B_n^1..S_n^{\mathrm{nf}}{\to}B_n^{k_n}] \leq \bigwedge^\star[S^{\mathrm{nf}}{\to}A_1..S^{\mathrm{nf}}{\to}A_m],$$

$$\text{where } T_i^{\mathrm{nf}} = \begin{cases} B_i^1, & \text{if it is not an intersection;} \\ \bigwedge^\star[B_i^1..B_i^{k_i}], & \text{otherwise.} \end{cases}$$

By generation for normal subtyping (proposition 7.10), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ and $l_j \in \{i..k_j\}$ such that $\Gamma^{\mathrm{nf}} \vdash_n S_j^{\mathrm{nf}} \rightarrow B_j^{l_j} \leq S^{\mathrm{nf}} \rightarrow A_i$. Again, by generation for normal subtyping (proposition 7.10), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ and $l_j \in \{i..k_j\}$ such that $\Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq S_j^{\mathrm{nf}}$ and $\Gamma^{\mathrm{nf}} \vdash_n B_j^{l_j} \leq A_i$. By NS-Trans and the equivalence of ordinary and normal subtyping (theorem 8.9), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ and $l_j \in \{i..k_j\}$ such that $\Gamma \vdash U \leq S_j$ and $\Gamma^{\mathrm{nf}} \vdash_n B_j^{l_j} \leq A_i$. By NS-$\exists$, for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ such that $\Gamma \vdash U \leq S_j$ and $\Gamma^{\mathrm{nf}} \vdash_n \bigwedge^\star[B_j^1..B_j^{k_j}] \leq A_i$. By the equivalence of ordinary and normal subtyping (theorem 8.9), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ such that $\Gamma \vdash U \leq S_j$ and $\Gamma \vdash T_j \leq A_i$. By lemma 4.13, S-Conv, and S-Trans, $\Gamma \vdash \bigwedge^\star[T_j \mid \Gamma \vdash U \leq S_j] \leq T$. □

Lemma 11.6 (*Substitution for subtyping*)
If $\Gamma_1 \vdash S_1 \leq T_1$ and $\Gamma_1,\, X{\leq}T_1{:}K_1,\, \Gamma_2 \vdash S_2 \leq T_2$, then $\Gamma_1,\, \Gamma_2[X{\leftarrow}S_1] \vdash S_2[X{\leftarrow}S_1] \leq T_2[X{\leftarrow}S_1]$.

Proof: By straightforward induction on the derivation of $\Gamma_1,\, X{\leq}T_1{:}K_1,\, \Gamma_2 \vdash S_2 \leq T_2$, using weakening (corollary 4.6), the type substitution lemma (lemma 4.10), and lemma 3.14. □

Lemma 11.7 (*Type application*)
If $\Gamma \vdash \bigwedge^\star[\forall X{\leq}S_1{:}K_1.T_1..\forall X{\leq}S_n{:}K_n.T_n] \leq \forall X{\leq}S{:}K.T$ and $\Gamma \vdash U \leq S$,
then $\Gamma \vdash \bigwedge^\star[T_j[X{\leftarrow}U] \mid \Gamma \vdash U \leq S_j] \leq T[X{\leftarrow}U]$.

Proof: There are two cases to be considered according to the normal form of $\forall X{\leq}S{:}K.T$. The case when $(\forall X{\leq}S{:}K.T)^{\mathrm{nf}} \equiv \forall X{\leq}S^{\mathrm{nf}}{:}K.T^{\mathrm{nf}}$ is similar to but simpler than the one we consider here. Assume $(\forall X{\leq}S{:}K.T)^{\mathrm{nf}} \equiv \bigwedge^\star[\forall X{\leq}S^{\mathrm{nf}}{:}K.A_1..\forall X{\leq}S^{\mathrm{nf}}{:}K.A_m]$where $T^{\mathrm{nf}} \equiv \bigwedge^\star[A_1..A_m]$. By the equivalence of ordinary and normal subtyping (theorem 8.9),

$$\begin{aligned}\Gamma^{\mathrm{nf}} \vdash_n\ & \textstyle\bigwedge^\star[\forall X{\leq}S_1^{\mathrm{nf}}{:}K_1.B_1^1..\forall X{\leq}S_1^{\mathrm{nf}}{:}K_1.B_1^{k_1}..\forall X{\leq}S_n^{\mathrm{nf}}{:}K_n.B_n^1..\forall X{\leq}S_n^{\mathrm{nf}}{:}K_n.B_n^{k_n}] \\ & \leq \textstyle\bigwedge^\star[\forall X{\leq}S^{\mathrm{nf}}{:}K.A_1..\forall X{\leq}S^{\mathrm{nf}}{:}K.A_m],\end{aligned}$$

where $T_i^{\mathrm{nf}} = \begin{cases} B_i^1, & \text{if it is not an intersection;} \\ \bigwedge^\star[B_i^1..B_i^{k_i}], & \text{otherwise.}\end{cases}$

By generation for normal subtyping (proposition 7.10), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ and $l_j \in \{i..k_j\}$ such that $\Gamma^{\mathrm{nf}} \vdash_n \forall X{\leq}S_j^{\mathrm{nf}}{:}K_j.B_j^{l_j} \leq \forall X{\leq}S^{\mathrm{nf}}{:}K.A_i$. Again, by generation for normal subtyping (proposition 7.10), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ and $l_j \in \{i..k_j\}$ such that $K \equiv K_j$,$S^{\mathrm{nf}} \equiv S_j^{\mathrm{nf}}$, and $\Gamma^{\mathrm{nf}},\, X{\leq}S_j^{\mathrm{nf}}{:}K \vdash_n B_j^{l_j} \leq A_i$. By NS-$\forall\exists$, for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ such that $\Gamma^{\mathrm{nf}},\, X{\leq}S_j^{\mathrm{nf}}{:}K \vdash_n T_j^{\mathrm{nf}} \leq A_i$. By the equivalence of ordinary and normal subtyping (theorem 8.9), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ such that $\Gamma,\, X{\leq}S_j{:}K \vdash T_j \leq A_i$. Furthermore, by S-Conv and S-Trans, $\Gamma \vdash U \leq S_j$. Then, by the substitution lemma for subtyping (lemma 11.6), for each $i \in \{1..m\}$ there exist $j \in \{1..n\}$ such that $\Gamma \vdash T_j[X{\leftarrow}U] \leq A_i[X{\leftarrow}U]$. By NS-$\forall\exists$, $\Gamma \vdash \bigwedge^\star[T_j[X{\leftarrow}U] \mid \Gamma \vdash U \leq S_j] \leq \bigwedge^\star[A_1[X{\leftarrow}U]..A_m[X{\leftarrow}U]]$. By the definition of substitution, $\Gamma \vdash \bigwedge^{\star}[T_j[X\leftarrow U] \mid \Gamma \vdash U \leq S_j] \leq T^{\mathrm{nf}}[X\leftarrow U]$. Finally, by lemma 3.14, S-CONV, and S-TRANS, $\Gamma \vdash \bigwedge^{\star}[T_j[X\leftarrow U] \mid \Gamma \vdash U \leq S_j] \leq T[X\leftarrow U]$, □

Usually, the next step to prove the accuracy of an algorithm, *inf* in our case, would be to prove a completeness result: if the term e has type T with respect to the context Γ in the the typing system $F^{\omega}_{\wedge}$ then the algorithm *inf* finds a type T' for e in Γ. In the present situation this result is not strong enough, since every closed term is typeable in both systems. One easily proves that

LEMMA 11.8

1. If e is closed in Γ, then there exists T such that $\Gamma \vdash e : T$.

2. If e is closed in Γ, then there exists T such that $\Gamma \vdash_{\mathrm{inf}} e : T$.

We use the fact that *inf* is deterministic, which means that the rules are invertible, to prove that *inf* finds a minimal type.

PROPOSITION 11.9 (*Generation for inf*) The form of a derivable typing statement uniquely determines the last applied rule.

PROPOSITION 11.10 (*inf computes minimal types*)
If $\Gamma \vdash e : T$ and $\Gamma \vdash_{\mathrm{inf}} e : T'$, then $\Gamma \vdash T' \leq T$.

PROOF: By induction on the derivation of $\Gamma \vdash e : T$. We illustrate the proof technique showing the case for T-APP. We are given that $e \equiv f\ a$, $\Gamma \vdash f : V\rightarrow T$, and $\Gamma \vdash a : V$. By generation for *inf* (proposition 11.9), $\Gamma \vdash_{\mathrm{inf}} f : U$, $\Gamma \vdash_{\mathrm{inf}} a : S$, and $T' \equiv \bigwedge^{\star}[T_i \mid S_i\rightarrow T_i \in \mathit{arrows}(\mathit{flub}_{\Gamma}(U))$ and $\Gamma \vdash S \leq S_i]$. By the induction hypothesis, $\Gamma \vdash U \leq V\rightarrow T$, and $\Gamma \vdash S \leq V$. By lemma 11.3(1), $\Gamma \vdash \bigwedge^{\star}[S_i\rightarrow T_i \mid S_i\rightarrow T_i \in \mathit{arrows}(\mathit{flub}_{\Gamma}(U))] \leq V\rightarrow T$. Finally, by the term application lemma (lemma 11.5), it follows that $\Gamma \vdash \bigwedge^{\star}[T_i \mid \Gamma \vdash S \leq S_i] \leq T$, where $S_i\rightarrow T_i \in \mathit{arrows}(\mathit{flub}_{\Gamma}(U))$. In other words, $\Gamma \vdash \bigwedge^{\star}[T_i \mid S_i\rightarrow T_i \in \mathit{arrows}(\mathit{flub}_{\Gamma}(U))] \leq T$.
□

Finally, we have proved the following result.

THEOREM 11.11 (*Minimal typing for $F^{\omega}_{\wedge}$*) Given a term e and a context Γ, there exists T such that for every T', if $\Gamma \vdash e : T'$, then $\Gamma \vdash T \leq T'$.

12 Subject reduction

The $F^{\omega}_{\wedge}$ system is layered in three syntactic categories: kinds, types, and terms. Since terms do not appear in either types or kinds, reductions in type expressions can be studied independently from the reductions of terms. In section 2, we proved that reduction on types preserves kinding properties: the sub-language of types and kinds satisfies the subject reduction property (lemma 4.11):

$$\text{if} \quad \Gamma \vdash S : K \quad \text{and} \quad S \twoheadrightarrow_{\beta\wedge} T, \quad \text{then} \quad \Gamma \vdash T : K.$$

In this section, we show the subject reduction property for typing judgements (proposition 12.7):

$$\text{if} \quad \Gamma \vdash e : T \quad \text{and} \quad e \twoheadrightarrow_{\beta\text{for}} e', \quad \text{then} \quad \Gamma \vdash e' : T.$$

In other words, reductions on terms are also safe.

LEMMA 12.1 If $Y \notin \mathrm{FV}(S)$, then

1. $e[Y \leftarrow T][X \leftarrow S] \equiv e[X \leftarrow S][Y \leftarrow T[X \leftarrow S]]$
2. $U[Y \leftarrow T][X \leftarrow S] \equiv U[X \leftarrow S][Y \leftarrow T[X \leftarrow S]]$

PROOF: By induction on the structure of e and U respectively. □

LEMMA 12.2 (*Substitution for typing*)

1. If $\Gamma_1 \vdash e_1 : S_1$ and $\Gamma_1,\, x{:}S_1,\, \Gamma_2 \vdash e_2 : S_2$, then $\Gamma_1,\, \Gamma_2 \vdash e_2[x \leftarrow e_1] : S_2$.
2. If $\Gamma_1 \vdash S \leq S_1$ and $\Gamma_1,\, X \leq S_1{:}K_1,\, \Gamma_2 \vdash e_2 : S_2$, then $\Gamma_1,\, \Gamma_2[X \leftarrow S] \vdash e_2[X \leftarrow S] : S_2[X \leftarrow S]$.

PROOF:

1. By induction on the derivation of $\Gamma_1,\, x{:}S_1,\, \Gamma_2 \vdash e_2 : S_2$.
2. By induction on the derivation of $\Gamma_1,\, X \leq S_1{:}K_1,\, \Gamma_2 \vdash e_2 : S_2$, using the type substitution lemma (lemma 4.10) in the T-VAR and T-MEET cases; the substitution lemma for subtyping (lemma 11.6) and lemma 12.1 in the case for T-TAPP; lemma 12.1 in the T-FOR case, and the substitution lemma for subtyping (lemma 11.6) in the T-SUBSUMPTION case. □

LEMMA 12.3 $\Gamma \vdash \top^\star \leq T$ if and only if $T =_{\beta\wedge} \top^\star$ and $\Gamma \vdash T : \star$.

PROOF: If $T =_{\beta\wedge} \top^\star$, then the result follows by S-CONV. Otherwise, if $\Gamma \vdash \top^\star \leq T$, by the well-kindedness of subtyping (proposition 4.18), T-MEET, and uniqueness of kinds (lemma 4.9), $\Gamma \vdash T : \star$. By the equivalence of ordinary and algorithmic subtyping (proposition 9.2), $\Gamma^{\mathrm{nf}} \vdash_{\mathit{Alg}} \top^\star \leq T^{\mathrm{nf}}$, which can only be derived using ALGS-∀∃ where T^{nf} is the empty intersection. □

Given $\Gamma \vdash S \leq T$, generation for normal subtyping (proposition 7.10) and the equivalence of ordinary and normal subtyping (theorem 8.9) provide subtyping information about the normal forms of S and T. We can also show that subtyping is structural for arrow types, quantified types and type operators, without reducing the terms in the subtyping relation to normal form. An implementation of a subtyping algorithm for $F^\omega_\wedge$ could take advantage of this fact by delaying normalizing steps, which might result in having to consider fewer recursive calls or calls with smaller arguments.

LEMMA 12.4 (*Generation for ordinary subtyping*)

1. $\Gamma \vdash T_1 \to T_2 \leq S_1 \to S_2$ and $S_2 \neq_{\beta\wedge} \top^\star$ if and only if $\Gamma \vdash S_1 \leq T_1$ and $\Gamma \vdash T_2 \leq S_2$.

2. $\Gamma \vdash \forall X{\leq}T_1{:}K_T.T_2 \leq \forall X{\leq}S_1{:}K_S.S_2$ and $S_2 \neq_{\beta\wedge} \top^\star$ if and only if $K_S \equiv K_T$, $T_1 =_{\beta\wedge} S_1$, and $\Gamma, X{\leq}T_1{:}K_T \vdash T_2 \leq S_2$.

3. $\Gamma \vdash \Lambda X{:}K_T.T_2 \leq \Lambda X{:}K_S.S_2$ and $S_2 \neq_{\beta\wedge} \top^\star$ if and only if $\Gamma, X{:}K_S \vdash T_2 \leq S_2$ and $K_T \equiv K_S$.

PROOF: The three statements are proved using a similar argument. We consider here the proof of part 2. If $K_S \equiv K_T$, $T_1 =_{\beta\wedge} S_1$, and $\Gamma, X{\leq}T_1{:}K_T \vdash T_2 \leq S_2$, then, by S-ALL and S-CONV, $\Gamma \vdash \forall X{\leq}T_1{:}K_T.T_2 \leq \forall X{\leq}S_1{:}K_S.S_2$. Conversely, let $\Gamma \vdash \forall X{\leq}T_1{:}K_T.T_2 \leq \forall X{\leq}S_1{:}K_S.S_2$ and $S_2 \neq_{\beta\wedge} \top^\star$. Lemma 12.3 implies that $T_2^{\mathrm{nf}} \neq_{\beta\wedge} \top^\star$. Then we have to consider four cases according to whether S_2^{nf} and T_2^{nf} are intersection types or not. We illustrate the proof argument considering just one case. Let

$$(\forall X{\leq}T_1{:}K_T.T_2)^{\mathrm{nf}} \equiv \forall X{\leq}{T_1}^{\mathrm{nf}}{:}K_T.T_2^{\mathrm{nf}}, \quad \text{and}$$
$$(\forall X{\leq}S_1{:}K_S.S_2)^{\mathrm{nf}} \equiv \bigwedge^\star[\forall X{\leq}{S_1}^{\mathrm{nf}}{:}K_S.A_1..\forall X{\leq}{S_1}^{\mathrm{nf}}{:}K_S.A_n],$$

where $S_2^{\mathrm{nf}} \equiv \bigwedge^\star[A_1..A_n]$. By the equivalence of ordinary and normal subtyping (theorem 8.9) and generation for normal subtyping (proposition 7.10), for each $i \in \{1..n\}$ $\Gamma^{\mathrm{nf}} \vdash_n \forall X{\leq}{T_1}^{\mathrm{nf}}{:}K_T.T_2^{\mathrm{nf}} \leq \forall X{\leq}{S_1}^{\mathrm{nf}}{:}K_S.A_i$ and, again generation for normal subtyping (proposition 7.10) implies that $\Gamma^{\mathrm{nf}}, X{\leq}T_1^{\mathrm{nf}}{:}K_T \vdash_n T_2^{\mathrm{nf}} \leq A_i$, and $T_1^{\mathrm{nf}} \equiv S_1^{\mathrm{nf}}$. By NS-$\forall$, $\Gamma^{\mathrm{nf}}, X{\leq}T_1^{\mathrm{nf}}{:}K_T \vdash_n T_2^{\mathrm{nf}} \leq S_2^{\mathrm{nf}}$ and $\Gamma, X{\leq}T_1{:}K_T \vdash T_2 \leq S_2$, by the equivalence of ordinary and normal subtyping (theorem 8.9). □

LEMMA 12.5 (*Generation for typing*)

1. If $\Gamma \vdash \lambda x{:}S_1.e : S$, then there exists S_2 such that $\Gamma, x{:}S_1 \vdash e : S_2$ and $\Gamma \vdash S_1 \to S_2 \leq S$.

2. If $\Gamma \vdash \lambda X{\leq}S_1{:}K_1.e : S$, then there exists S_2 such that $\Gamma, X{\leq}S_1{:}K_1 \vdash e : S_2$ and $\Gamma \vdash \forall X{\leq}S_1{:}K_1.S_2 \leq S$.

3. If $\Gamma \vdash \mathrm{for}(X{\in}\{U_1..U_n\})e : T$, then there exist $T_1..T_n$ such that, for each i in $\{1..n\}$, $\Gamma \vdash e[X{\leftarrow}U_i] : T_i$ and $\Gamma \vdash \bigwedge^\star[T_1..T_n] \leq T$.

PROOF: Each statement is proved by induction on the derivation of the typing statement in the antecedent. We exhibit here the proof of part 3. We proceed by case analysis on the last rule of the derivation of $\Gamma \vdash \mathrm{for}(X{\in}\{U_1..U_n\})e : T$.

T-FOR We are given that $\Gamma \vdash e[X{\leftarrow}U] : T$ for some $U \in \{U_1..U_n\}$. Since every closed term has a type, we have that, for each i in $\{1..n\}$, $\Gamma \vdash e[X{\leftarrow}U_i] : T_i$, and, by S-MEET-LB, $\Gamma \vdash \bigwedge^\star[T_1..T..T_n] \leq T$.

T-Meet Let $T \equiv \bigwedge^\star[S_1..S_k]$. We are given that, $\Gamma \vdash \mathrm{ok}$ and $\Gamma \vdash \mathrm{for}(X{\in}\{U_1..U_n\})e : S_j$, for each j in $\{1..k\}$. By the induction hypothesis, for each $j \in \{1..k\}$ and each $i \in \{1..n\}$, there exist T_{j_i} such that $\Gamma \vdash e[X{\leftarrow}U_i] : T_{j_i}$, and $\Gamma \vdash \bigwedge^\star[T_{j_1}..T_{j_n}] \leq S_j$, and, by the minimal type property (theorem 11.11), there exist $T_1..T_n$ such that $\Gamma \vdash e[X{\leftarrow}U_i] : T_i$, and $\Gamma \vdash T_i \leq T_{j_i}$, by lemma 4.17, it follows that $\Gamma \vdash \bigwedge^\star[T_1..T_n] \leq \bigwedge^\star[T_{j_1}..T_{j_n}]$, and by S-Trans, $\Gamma \vdash \bigwedge^\star[T_1..T_n] \leq S_j$. Finally, by S-Meet-G, it follows that $\Gamma \vdash \bigwedge^\star[T_1..T_n] \leq \bigwedge^\star[S_1..S_k]$.

T-Sub We are given that $\Gamma \vdash \mathrm{for}(X{\in}\{U_1..U_n\})e : S$, and $\Gamma \vdash S \leq T$. The result follows by the induction hypothesis and S-Trans. □

Since terms cannot occur in types, subject reduction for terms does not need to consider reductions in contexts.

Proposition 12.6 (*One step subject reduction for typing judgements*)
If $\Gamma \vdash e : T$ and $e \rightarrow_{\beta\mathrm{for}} e'$, then $\Gamma \vdash e' : T$.

Proof: Since every term has type $\top^\star$, the interesting case is when $T \neq_{\beta\wedge} \top^\star$. This proposition follows by induction on the derivation of $\Gamma \vdash e : T$. We consider the cases where e is a redex; the other cases follow by straightforward application of the induction hypothesis.

T-App There are two possibilities for e to be a redex.

1. $e \equiv (\lambda x{:}S_1.e_1)\, e_2$, $e' \equiv e_1[x{\leftarrow}e_2]$, and $T \equiv T_2$. We are given that $\Gamma \vdash \lambda x{:}S_1.e_1 : T_1{\rightarrow}T_2$ and $\Gamma \vdash e_2 : T_1$. By the generation lemma for typing (lemma 12.5), there exists S_2 such that, $\Gamma,\, x{:}S_1 \vdash e_1 : S_2$ and $\Gamma \vdash S_1{\rightarrow}S_2 \leq T_1{\rightarrow}T_2$. Since $T_2 \neq_{\beta\wedge} \top^\star$, by the generation lemma for ordinary subtyping (lemma 12.4), $\Gamma \vdash T_1 \leq S_1$ and $\Gamma \vdash S_2 \leq T_2$. Then, by T-Subsumption, it follows that $\Gamma,\, x{:}S_1 \vdash e_1 : T_2$ and $\Gamma \vdash e_2 : S_1$. Finally, by the substitution lemma for typing (lemma 12.2(1)), $\Gamma \vdash e_1[x{\leftarrow}e_2] : T_2$.
2. $e \equiv (\mathrm{for}(X{\in}U_1..U_n)e_2)\, e_1$, $e' \equiv \mathrm{for}(X{\in}U_1..U_n)(e_2\, e_1)$, and $T \equiv T_2$. We are given that $\Gamma \vdash \mathrm{for}(X{\in}U_1..U_n)e_2 : T_1{\rightarrow}T_2$ and $\Gamma \vdash e_1 : T_1$. By the generation lemma for typing (lemma 12.5), there exist $V_1..V_n$ such that $\Gamma \vdash e_2[X{\leftarrow}U_i] : V_i$ for each $i \in \{1..n\}$, and $\Gamma \vdash \bigwedge^\star[V_1..V_n] \leq T_1{\rightarrow}T_2$.

   We write $V_i^{\mathrm{nf}} \equiv A_{i_1}$, if it is not an intersection,
   $V_i^{\mathrm{nf}} \equiv \bigwedge^\star[A_{i_1}..A_{i_{k_i}}]$, otherwise.

   Note that $\bigwedge^\star[V_1..V_n]^{\mathrm{nf}} \equiv \bigwedge^\star[A_{1_1}..A_{1_{k_1}}..A_{n_1}..A_{n_{k_n}}]$. By the equivalence of ordinary and normal subtyping (theorem 8.9), $\Gamma^{\mathrm{nf}} \vdash_n \bigwedge^\star[A_{1_1}..A_{1_{k_1}}..A_{n_1}..A_{n_{k_n}}] \leq (T_1{\rightarrow}T_2)^{\mathrm{nf}}$. There are two cases to consider according to the form of $(T_1{\rightarrow}T_2)^{\mathrm{nf}}$. If $(T_1{\rightarrow}T_2)^{\mathrm{nf}}$ is $T_1^{\mathrm{nf}}{\rightarrow}T_2^{\mathrm{nf}}$ or $\bigwedge^\star[T_1^{\mathrm{nf}}{\rightarrow}B_1..T_1^{\mathrm{nf}}{\rightarrow}B_r]$ where $T_2^{\mathrm{nf}} \equiv \bigwedge^\star[B_1..B_r]$. We show here the latter; the former is simpler.

   By generation for normal subtyping (proposition 7.10), for every $s \in \{1..r\}$ there exist $l \in \{1..n\}$ and $j \in \{1..k_l\}$ such that $\Gamma^{\mathrm{nf}} \vdash_n A_{l_j} \leq T_1^{\mathrm{nf}}{\rightarrow}B_s$, and, by NS-∃

or NS-Refl, $\Gamma^{\mathrm{nf}} \vdash_n V_l^{\mathrm{nf}} \leq A_{l_j}$, by NS-Trans, for every $s \in \{1..r\}$ there exists $l \in \{1..n\}$ such that $\Gamma^{\mathrm{nf}} \vdash_n V_l^{\mathrm{nf}} \leq T_1^{\mathrm{nf}} \rightarrow B_s$, and, by the equivalence of ordinary and normal subtyping (theorem 8.9), $\Gamma \vdash V_l \leq T_1 \rightarrow B_s$. By T-Subsumption, for every $s \in \{1..r\}$ there exists $l \in \{1..n\}$ $\Gamma \vdash e_2[X \leftarrow S_l] : T_1 \rightarrow B_s$. By T-App, for every $s \in \{1..r\}$ there exists $l \in \{1..n\}$ $\Gamma \vdash (e_2[X \leftarrow S_l])\, e_1 : B_s$, and since X is not a free variable of e_1 we have that, for every $s \in \{1..r\}$ there exists $l \in \{1..n\}$ $\Gamma \vdash e_2\, e_1[X \leftarrow S_l] : B_s$. Applying T-For, we get that for every $s \in \{1..r\}$ $\Gamma \vdash \mathrm{for}(X \in U_1..U_n) e_2\, e_1 : B_s$, by T-Meet, $\Gamma \vdash \mathrm{for}(X \in U_1..U_n) e_2\, e_1 : T_2^{\mathrm{nf}}$. Finally, by S-Conv and T-Subsumption, $\Gamma \vdash \mathrm{for}(X \in U_1..U_n) e_2\, e_1 : T_2$.

T-TApp There are two possibilities for e to be a redex. We show only one case here. The case when $e \equiv (\mathrm{for}(X \in U_1..U_n) e_2)\, S$ follows a similar argument to the one used for the case $e \equiv (\mathrm{for}(X \in U_1..U_n) e_2)\, e_1$ in T-App.

If $e \equiv (\lambda X \leq S_1 {:} K_S . e_2)\, S$, $e' \equiv e_2[X \leftarrow S]$, and $T \equiv T_2[X \leftarrow S]$, we have that $\Gamma \vdash \lambda X \leq S_1 {:} K_S . e_1 : \forall X \leq T_1 {:} K_T . T_2$ and $\Gamma \vdash S \leq T_1$. By the generation lemma for typing (lemma 12.5), there exists S_2 such that $\Gamma,\ X \leq S_1 {:} K_S \vdash e_2 : S_2$ and $\Gamma \vdash \forall X \leq S_1 {:} K_S . S_2 \leq \forall X \leq T_1 {:} K_T . T_2$. Since $T_2[X \leftarrow S] \neq_{\beta\wedge} \top^\star$, lemma 3.14 implies that $T_2 \neq_{\beta\wedge} \top^\star$. Then, by the generation lemma for ordinary subtyping (lemma 12.4), $\Gamma,\ X \leq S_1 {:} K_S \vdash S_2 \leq T_2$, $S_1 =_{\beta\wedge} T_1$, and $K_S \equiv K_T$. By T-Subsumption, $\Gamma,\ X \leq S_1 {:} K_S \vdash e_2 : T_2$, and, by S-Trans and S-Conv, $\Gamma \vdash S \leq S_1$. Finally, by the substitution lemma for typing (lemma 12.2(2)), $\Gamma \vdash e_2[X \leftarrow S] : T_2[X \leftarrow S]$.

T-For Let $e \equiv \mathrm{for}(X \in U_1..U_n) e_1$, where $X \notin \mathrm{FTV}(e_1)$ and $e' \equiv e_1$. We are given that $\Gamma \vdash e_1[X \leftarrow U] : T$, with $U \in \{U_1..U_n\}$. Since $e_1 \equiv e_1[X \leftarrow U]$, the result holds. □

We now have all the results needed in order to prove that reduction on terms preserves typing. The following proposition, the subject reduction property for $F_\wedge^\omega$ terms, is a consequence of the previous one.

Proposition 12.7 (*Subject reduction for typing judgements*)
If $\Gamma \vdash e : T$ and $e \twoheadrightarrow_{\beta\mathrm{for}} e'$, then $\Gamma \vdash e' : T$.

Proof: By induction on the derivation of $e \twoheadrightarrow_{\beta\mathrm{for}} e'$, using proposition 12.6. □

13 Conclusions

We defined the typed lambda calculus $F_\wedge^\omega$, a natural generalization of Girard's system F^ω with intersection types and bounded polymorphism. A novel aspect of our presentation is the use of term rewriting techniques to present intersection types, which clearly splits the computational semantics (reduction rules) from the syntax (inference rules) of the system. We prove the Church-Rosser property of the reduction relation for types and terms in $F_\wedge^\omega$ and the strong normalization result for the reduction on types.

The normalized subtyping system, $NF_\wedge^\omega$, is a significant technical contribution which is the key to finding a generation principle for the subtyping relation in $F_\wedge^\omega$ and an algorithm,

$AlgF^{\omega}_{\wedge}$, which is shown to be equivalent to the original presentation. A generation principle is the key result needed to prove the Subject Reduction property for $F^{\omega}_{\wedge}$, which we established in section 12.

We also presented a type inference algorithm and prove that it computes Minimal Types for $F^{\omega}_{\wedge}$.

The work presented here has been extracted from the Ph.D thesis of the author [22], and a short version of sections 6 to 9 has been published in CSL'94 [21]. The techniques developed here have been applied to study the combination of dependent types and subtyping in [3]. Martín Abadi and Luca Cardelli used these techniques in their book "A Theory of Objects" [1], demonstrating that our method extends naturally to a higher-order system with recursive types and object constructors.

14 Acknowledgements

I want to express my gratitude to Mariangiola Dezani-Ciancaglini for her scientific and moral support which made the present work possible. I am also grateful for discussions with Henk Barendregt, Healfdene Goguen, James McKinna, Ugo de'Liguoro, and Benjamin Pierce.

This research was supported by the Dutch organization for scientific research, NWO-SION project *Typed lambda calculus*, and by a European Community TMR grant.

References

[1] M. Abadi and L. Cardelli. *A Theory of Objects*. Springer-Verlag, 1996.

[2] R. M. Amadio and L. Cardelli. Subtyping recursive types. *ACM Transactions on Programming Languages and Systems*, 15(4):575–631, 1993. A preliminary version appeared in POPL '91 (pp. 104–118), and as DEC Systems Research Center Research Report number 62, August 1990.

[3] D. Aspinall and A. Compagnoni. Subtyping dependent types. In *Eleventh Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA.*, July 27-30 1996. Preliminary version July 1995.

[4] H. Barendregt. Lambda calculi with types. In T. S. E. M. S. Abramsky, Dov M. Gabbay, editor, *Handbook of Logic in Computer Science*, volume 2, pages 117–309. Clarendon Press. Oxford, 1992.

[5] H. P. Barendregt. *The Lambda Calculus*. North Holland, revised edition, 1984.

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

[7] V. Breazu-Tannen, T. Coquand, C. Gunter, and A. Scedrov. Inheritance as implicit coercion. *Information and Computation*, 93:172–221, 1991.

[8] K. B. Bruce and J. Mitchell. PER models of subtyping, recursive types and higher-order polymorphism. In *Proceedings of the Nineteenth ACM Symposium on Principles of Programming Languages*, Albequerque, NM, January 1992.

[9] P. Canning, W. Cook, W. Hill, W. Olthoff, and J. Mitchell. F-bounded quantification for object-oriented programming. In *Fourth International Conference on Functional Programming Languages and Computer Architecture*, pages 273–280, September 1989.

[10] L. Cardelli. A semantics of multiple inheritance. *Information and Computation*, 76:138–164, 1988. Preliminary version in *Semantics of Data Types*, Kahn, MacQueen, and Plotkin, eds., Springer-Verlag LNCS 173, 1984.

[11] L. Cardelli. Types for data-oriented languages. In *First Conference on Extending Database Technology*, volume 303 of *Lecture Notes in Computer Science*. Springer-Verlag, May 1988.

[12] L. Cardelli. Notes about $\mathrm{F}^{\omega}_{<:}$. Unpublished manuscript, October 1990.

[13] L. Cardelli. Typeful programming. In E. J. Neuhold and M. Paul, editors, *Formal Description of Programming Concepts*. Springer-Verlag, 1991. An earlier version appeared as DEC Systems Research Center Research Report #45, February 1989.

[14] L. Cardelli and J. Mitchell. Operations on records. *Mathematical Structures in Computer Science*, 1:3–48, 1991. Also in [31], and available as DEC Systems Research Center Research Report #48, August, 1989, and in the proceedings of MFPS '89, Springer LNCS volume 442.

[15] L. Cardelli and P. Wegner. On understanding types, data abstraction, and polymorphism. *Computing Surveys*, 17(4), December 1985.

[16] F. Cardone and M. Coppo. Two extensions of Curry's type inference system. In P. Odifreddi, editor, *Logic and Computer Science*, number 31 in APIC Studies in Data Processing, pages 19–76. Academic Press, 1990.

[17] F. Cardone, M. Dezani-Ciancaglini, and U. de' Liguoro. Combining type disciplines, 1993. Manuscript.

[18] G. Castagna and B. Pierce. Decidable bounded quantification. In *Proceedings of Twenty-First Annual ACM Symposium on Principles of Programming Languages, Portland, OR*. ACM, January 1994.

[19] A. Compagnoni. Decidable higher order subtyping. Technical report, University of Edinburgh, LFCS, August 1997.

[20] A. B. Compagnoni. Subtyping in $F^{\omega}_{\wedge}$ is decidable. Technical Report ECS-LFCS-94-281, LFCS, University of Edinburgh, January 1994.

[21] A. B. Compagnoni. Decidability of higher-order subtyping with intersection types. In *Proceedings of the Annual Conference of the European Association for Computer Science Logic, CSL'94, Kazimierz, Poland*, number 933 in Lecture Notes in Computer Science. Springer-Verlag, June 1995. Preliminary version available as University of Edinburgh technical report ECS-LFCS-94-281, under the title "Subtyping in $F^{\omega}_{\wedge}$ is decidable", January 1994.

[22] A. B. Compagnoni. *Higher-Order Subtyping with Intersection Types.* PhD thesis, University of Nijmegen, The Netherlands, January 1995. ISBN 90-9007860-6.

[23] A. B. Compagnoni and B. C. Pierce. Higher-order intersection types and multiple inheritance. *Mathematical Structures in Computer Science*, 6:469–501, 1996. Preliminary version available under the title *Multiple Inheritance via Intersection Types* as University of Edinburgh technical report ECS-LFCS-93-275 and Catholic University Nijmegen computer science technical report 93-18, Aug. 1993.

[24] W. R. Cook, W. L. Hill, and P. S. Canning. Inheritance is not subtyping. In *Seventeenth Annual ACM Symposium on Principles of Programming Languages*, pages 125–135, San Francisco, CA, January 1990. Also in [31].

[25] M. Coppo and M. Dezani-Ciancaglini. A new type-assignment for λ-terms. *Archiv. Math. Logik*, 19:139–156, 1978.

[26] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre-Dame Journal of Formal Logic*, 21(4):685–693, October 1980.

[27] P.-L. Curien and G. Ghelli. Coherence of subsumption: Minimum typing and type-checking in $F_{\leq}$. *Mathematical Structures in Computer Science*, 2:55–91, 1992.

[28] N. G. de Bruijn. Lambda-calculus notation with nameless dummies: a tool for automatic formula manipulation with application to the Church-Rosser theorem. *Indag. Math.*, 34(5):381–392, 1972.

[29] G. Ghelli. *Proof Theoretic Studies about a Minimal Type System Integrating Inclusion and Parametric Polymorphism.* PhD thesis, Università di Pisa, March 1990. Technical report TD–6/90, Dipartimento di Informatica, Università di Pisa.

[30] J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures de l'arithmétique d'ordre supérieur.* PhD thesis, Université Paris VII, 1972.

[31] C. A. Gunter and J. C. Mitchell. *Theoretical Aspects of Object-Oriented Programming: Types, Semantics, and Language Design.* The MIT Press, 1994.

[32] J. McKinna and R. Pollack. Pure type systems formalized. In M. Bezem and J. F. Groote, editors, *Proceedings of the International Conference on Typed Lambda Calculi and Applications*, pages 289–305. Springer–Verlag, LNCS 664, Mar. 1993.

[33] J. C. Mitchell. Toward a typed foundation for method specialization and inheritance. In *Proceedings of the 17th ACM Symposium on Principles of Programming Languages*, pages 109–124, January 1990. Also in [31].

[34] B. C. Pierce. *Programming with Intersection Types and Bounded Polymorphism.* PhD thesis, Carnegie Mellon University, December 1991. Available as School of Computer Science technical report CMU-CS-91-205.

[35] B. C. Pierce and D. N. Turner. Simple type-theoretic foundations for object-oriented programming. *Journal of Functional Programming*, 4(2):207–247, Apr. 1994. A preliminary

version appeared in Principles of Programming Languages, 1993, and as University of Edinburgh technical report ECS-LFCS-92-225, under the title "Object-Oriented Programming Without Recursive Types".

[36] J. Reynolds. Using category theory to design implicit conversions and generic operators. In N. D. Jones, editor, *Proceedings of the Aarhus Workshop on Semantics-Directed Compiler Generation*, number 94 in Lecture Notes in Computer Science. Springer-Verlag, January 1980. Also in [31].

[37] J. C. Reynolds. Preliminary design of the programming language Forsythe. Technical Report CMU-CS-88-159, Carnegie Mellon University, June 1988.

[38] M. Steffen and B. Pierce. Higher-order subtyping. In *IFIP Working Conference on Programming Concepts, Methods and Calculi (PROCOMET)*, June 1994. An earlier version appeared as University of Edinburgh technical report ECS-LFCS-94-280 and Universität Erlangen-Nürnberg Interner Bericht IMMD7-01/94, February 1994.