Subtyping in $F^{\omega}_{\wedge}$ is decidable

Adriana B. Compagnoni
LFCS University of Edinburgh
Technical Report ECS-LFCS-94-281 *

January 1994

Abstract

Combining intersection types with higher-order subtyping yields a typed model of object-oriented programming with multiple inheritance [CP93]. The target calculus, $F^{\omega}_{\wedge}$, a natural generalization of Girard's system F^{ω} with intersection types and bounded polymorphism, is of independent interest. We prove that subtyping in $F^{\omega}_{\wedge}$ is decidable, which yields as a corollary the decidability of subtyping in $F^{\omega}_{\leq}$, its intersection free fragment, because the $F^{\omega}_{\wedge}$ subtyping system is a conservative extension of that of $F^{\omega}_{\leq}$. The calculus presented in [CP94] does not have reductions on types whereas subtyping in $F^{\omega}_{\wedge}$ is closed under $\beta\wedge$-conversion on types. The presence of $\beta\wedge$-conversion in $F^{\omega}_{\wedge}$ subtyping system drastically increases the complexity of the problem to be solved.

Moreover, we establish basic structural properties of $F^{\omega}_{\wedge}$ and we prove that the type assignment system is sound using a model based on partial equivalence relations.

1 Introduction

The system $F^{\omega}_{\wedge}$ was first introduced in [CP93] where it was shown to be be rich enough to provide a typed model of object oriented programming with multiple inheritance. $F^{\omega}_{\wedge}$ is an extension of F^{ω} [Gir72] 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 [CG92, Pie91, PT93, CP94]. 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 [CDC78, BCD83] and elsewhere (See [CC90] for background and further references). A second-order λ-calculus with intersection types was studied in [Pie91]. Systems including subtyping were present in [CW85, Car88]. Higher order generalizations of subtyping appear in [CCH^{+}89, CHC90, Mit90a, BM92]. $F_{\leq}$, a second-order λ-calculus with

*LFCS, University of Edinburgh, King's Buildings, Edinburgh EH9 3JZ, U.K. Electronic mail: abc@dcs.ed.ac.uk. On leave from Catholic University, Nijmegen. The Netherlands.

bounded quantification, was studied in [Ghe90], and in [Pie91] it was proved that subtyping in $F_{\leq}$ was undecidable and that undecidability was caused by the subtyping rule for bounded quantification. The picture changed when G.Castagna [CP94] proposed an alternative rule, S-ALL, which preserves decidability and which is still rich enough to solve the problems that motivated the definition of $F_{\leq}$ and of Cardelli and Mitchell's system $F^{\omega}_{\leq}$. Consequently, the decidability of subtyping in systems including this new rule became an open problem. In [CP94] the decidability of subtyping was proved for an extension of system F with bounded polymorphism, where all bounds appearing in S-ALL have the ground kind $\star$, a main limitation of that system. Allowing bounds of functional kind forces us to introduce a rule S-CONV to have invariance of subtyping under $\beta\wedge$-conversion of types. This makes the present subtyping relation much more expressive than that one presented in [CP94]. 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 [CCH+89] or Cardelli and Mitchell's system $F^{\omega}_{\leq}$ [Car, Mit90a, BM92]–.

In this paper we give a positive answer to the decidability of subtyping in the presence of $\beta\wedge$-convertible types. We prove that subtyping in $F^{\omega}_{\wedge}$ is decidable, which *a fortiori* gives the decidability of subtyping for the $F^{\omega}_{\leq}$ fragment because the former is a conservative extension of the latter – namely, each subtyping statement derivable in $F^{\omega}_{\wedge}$ containing no intersections other than the empty ones is also derivable in $F^{\omega}_{\leq}$.

We present a definition of $F^{\omega}_{\wedge}$ equivalent to the one introduced in [CP93] in which the four distributivity rules become particular cases of S-CONV and there is, instead, a richer notion of reduction on types which we show is confluent and strongly normalizing. This simplification was motivated by structural properties from the former presentation. Furthermore, this new presentation provides a new view of the system that is the key to proving the decidability of subtyping.

This new perspective suggests that to prove the decidability of subtyping 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 S-CONV by reflexivity. The following is a good example of the problem to be solved. Consider $\Gamma \equiv W{:}K, X \leq \text{Fun}\,(Y{:}K)\,Y{:}K\rightarrow K,\ Z \leq X{:}K\rightarrow K$. Then $\Gamma \vdash X(Z\,W) \leq W$, which is not derivable without using S-CONV, i.e. without performing any β-reduction.

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.

The solution is divided in two main steps. First, we develop a *normal subtyping system*, $NF^{\omega}_{\wedge}$, in which only types in normal form are considered and such that all subtyping statements in normal form derivable in $F^{\omega}_{\wedge}$ are also derivable in $NF^{\omega}_{\wedge}$. We prove that proofs in $NF^{\omega}_{\wedge}$can be normalized eliminating transitivity and simplifying reflexivity what gives rise to a generation lemma for normal subtyping (proposition 3.4.4). Moreover, we prove that $NF^{\omega}_{\wedge}$ is indeed an alternative presentation of the $F^{\omega}_{\wedge}$ subtyping relation, namely, the following result holds (theorem 3.5.8). $\Gamma \vdash S \leq T \iff \Gamma^{nf} \vdash_n S^{nf} \leq T^{nf}$. (This syntax is standard and, moreover, is fully explained in sections 2 and 3.)

The second and last step towards the decidability of subtyping in $F^{\omega}_{\wedge}$ is to prove the decidability for $NF^{\omega}_{\wedge}$. For that we provide a terminating algorithm, *check*, –see section 4– that is sound and complete with respect to $NF^{\omega}_{\wedge}$. Namely, $\Gamma \vdash_n S \leq T \iff \underline{check}(\Gamma \vdash S \leq T) = true$, which concludes the proof of decidability.

To check subtyping for type applications the algorithm reduces the checking of $\Gamma \vdash_n A\,F \leq B$ to check if $\Gamma \vdash_n \mathrm{LUB}(A\,F)^{nf} \leq B$. Where LUB substitutes the leftmost innermost variable of $A\,F$ by its bound in Γ. Such replacement may produce a term that is not in normal form, in which case we normalize it afterwards. The main problem here is that the complexity of the types to be examined in the recursive call does not decrease. This means that the proof of termination of the algorithm is not immediate. In particular, the proof of termination presented in [CP94] cannot be modified to serve our purposes because of the interaction between $\beta\wedge$-reduction and the substitution of type variables by its bounds in our system. See section 4.

In section 2, we give the definition of $F^{\omega}_{\wedge}$ and prove syntactical properties of it, as for example the Church-Rosser theorem for the $\beta\wedge$-reduction and the Strong Normalization property for the language of types and kinds. In section 3, we present the normal subtyping system $NF^{\omega}_{\wedge}$ and prove the equivalence between normal and ordinary subtyping. In section 4, we prove termination of the algorithm *check*, which concludes the proof of decidability. In section 5, we summarize the extension of our results to an alternative calculus with Cardelli and Wegner's *Fun* quantifier rule [CW85].

For the sake of richness of the presentation, in section 6 we give a model based on partial equivalence relations and we prove soundness of typing and subtyping.

2 Syntax

In this section we present the syntax of $F^{\omega}_{\wedge}$ we prove structural properties and the confluence and strong normalization properties for the reduction on types.

2.1 The system

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

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

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

Most of the rules include premises of 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. In an algorithmic presentation of the system (on which an implementation might be based), the well-formation premises would be replaced by the meta-theoretic observation that "recursive calls" in the premises of all the rules preserve the well-formedness of the "arguments" named in the conclusion.

In the interest of brevity, we omit well-formation premises that can be derived from others. For example, in the rule S-Arrow, we drop the premise $\Gamma \vdash T_1 \rightarrow T_2 \in \star$, since it follows from $\Gamma \vdash S_1 \rightarrow S_2 \in \star$ using the properties proved in section 2.3.

2.1.1 Notation

The *kinds* of $F^\omega_\wedge$ 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).

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

The language of *types* of $F^\omega_\wedge$ 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 ($\mathrm{All}\,(X \leq T{:}K)\,T'$), 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 ($\mathrm{Fun}\,(X{:}K)\,T$) 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.

$$
\begin{array}{llll}
\mathbb{T} & ::= & X & \text{type variable} \\
 & & \mathbb{T} \rightarrow \mathbb{T} & \text{function type} \\
 & & \mathrm{All}\,(X \leq \mathbb{T}{:}\mathbb{K})\,\mathbb{T} & \text{polymorphic type} \\
 & & \mathrm{Fun}\,(X{:}\mathbb{K})\,\mathbb{T} & \text{operator abstraction} \\
 & & \mathbb{T}\,\mathbb{T} & \text{operator application} \\
 & & \bigwedge^{\mathbb{K}}[\mathbb{T}..\mathbb{T}] & \text{intersection at kind } \mathbb{K}
\end{array}
$$

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

$$
\begin{array}{lll}
\top^K & \triangleq & \bigwedge^K[\,] \\
X{:}K & \triangleq & X \leq \top^K{:}K
\end{array}
$$

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. Observe that $\mathrm{Fun}\,(X{:}K)\,T$ can be read as short for $\mathrm{Fun}\,(X \leq \top^K{:}K)\,T$

The language of terms includes the variables (x), functional abstractions ($\mathrm{fun}\,(x{:}T)\,e$), and applications ($e\,e$) of the simply typed λ-calculus, plus the type abstraction ($\mathrm{fun}\,(X \leq T{:}K)\,e$) and application ($e\,T$) of F^ω, where, 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 "$\mathrm{for}\,(X \in T_1..T_n)\,e$," which can be read as instructions to the typechecker to analyze the expression e separately under the assumptions $X \equiv T_1$, $X \equiv T_2$, $\ldots$, etc. and conjoin the results. For example, if Int$\leq$Real and $+ \in$ Int$\to$Int$\to$Int $\wedge$ Real$\to$Real$\to$Real, then we can derive

$$\text{for}\,(X \in \text{Int, Real})\;\; \text{fun}\,(x{:}X)\;\; x + x \quad \in \quad \text{Int}{\to}\text{Int} \wedge \text{Real}{\to}\text{Real}.$$

e	$::=$	x	variable
		$\text{fun}\,(x{:}\mathbb{T})\,e$	abstraction
		$e\,e$	application
		$\text{fun}\,(X{\leq}\mathbb{T}{:}\mathbb{K})\,e$	type abstraction
		$e\,\mathbb{T}$	type application
		$\text{for}\,(X \in \mathbb{T}..\mathbb{T})\,e$	alternation

2.1.2 Reduction rules

1. $(\text{Fun}\,(X{:}K)\,T_1)T_2 \to_{\beta\wedge} T_1[X{\leftarrow}T_2]$
2. $S \to \bigwedge^{\star}[T_1..T_n] \to_{\beta\wedge} \bigwedge^{\star}[S{\to}T_1 \;..\; S{\to}T_n]$
3. $\text{All}\,(X{\leq}S{:}K)\,\bigwedge^{\star}[T_1..T_n] \to_{\beta\wedge} \bigwedge^{\star}[\text{All}\,(X{\leq}S{:}K)\,T_1 \;..\; \text{All}\,(X{\leq}S{:}K)\,T_n]$
4. $\text{Fun}\,(X{:}K_1)\,\bigwedge^{K_2}[T_1..T_n] \to_{\beta\wedge} \bigwedge^{K_1\to K_2}[\text{Fun}\,(X{:}K_1)\,T_1 \;..\; \text{Fun}\,(X{:}K_1)\,T_n]$
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 left-hand side of each reduction rule is a *redex* and the right-hand side its *reduct*. $\to_{\beta\wedge}$ is extended as to become a compatible relation with respect to type formation, $\twoheadrightarrow_{\beta\wedge}$ is the transitive reflexive closure of $\to_{\beta\wedge}$, and $=_{\beta\wedge}$ is the transitive reflexive symmetric closure of $\to_{\beta\wedge}$. In section 4.1 we will 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 capture avoiding substitution of S for X in T is written $T[X{\leftarrow}S]$. Substitution is written similarly for terms, and is extended pointwise to contexts. The $\beta\wedge$-normal form of a type S is written S^{nf}, and is extended pointwise to contexts.

2.1.3 Contexts

A *context* Γ is a finite sequence of typing and/or subtyping assumptions for a set of variables and/or type variables. The empty context is written $\emptyset$.

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

When writing nonempty contexts, we omit the initial $\emptyset$.

The domain of Γ is written dom(Γ). The functions FV(—) and FTV(—) give the sets of free 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 [dB72]).

In the interest of brevity, we omit well-formation premises that can be derived from others.

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.

$$\emptyset \vdash \mathrm{ok} \tag{C-Empty}$$

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

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

2.1.4 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).

Note that we extend the context with $\top^{K_1}$ in K-All instead of T_1, which is not the case in T-All (C.f. S-All below.)

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

$$\frac{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash \mathrm{ok}}{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash X \in K} \tag{K-TVar}$$

$$\frac{\Gamma \vdash T_1 \in \star \qquad \Gamma \vdash T_2 \in \star}{\Gamma \vdash T_1 {\rightarrow} T_2 \in \star} \tag{K-Arrow}$$

$$\frac{\Gamma \vdash T_1 \in K_1 \qquad \Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star}{\Gamma \vdash \mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2 \in \star} \tag{K-All}$$

$$\frac{\Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in K_2}{\Gamma \vdash \mathrm{Fun}\,(X{:}K_1)\,T_2 \in K_1 {\rightarrow} K_2} \tag{K-OAbs}$$

$$\frac{\Gamma \vdash S \in K_1 {\rightarrow} K_2 \qquad \Gamma \vdash T \in K_1}{\Gamma \vdash S\,T \in K_2} \tag{K-OApp}$$

$$\frac{\Gamma \vdash \mathrm{ok} \qquad \text{for each } i,\ \Gamma \vdash T_i \in K}{\Gamma \vdash \bigwedge^K [T_1..T_n] \in K} \tag{K-Meet}$$

2.1.5 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, S-ARROW, and S-ALL are the same as in the usual, second-order case. Rules S-OABS and S-OAPP extend the subtype relation pointwise to kinds other than $\star$.

F^ω's rule of type conversion (if $\Gamma \vdash e \in T$ and $T =_\beta T'$ then $\Gamma \vdash e \in T'$) is captured here as the subtyping rule S-CONV, which also gives reflexivity as a special case.

The rule S-ALL compares the bodies of two quantifiers assuming a trivial upper bound for the variable X instead of the usual assumption $X{\leq}T_1{:}K_1$ [CG92].

Rules S-MEET-G and S-MEET-LB specify that an intersection of a set of types is their order-theoretic greatest lower bound.

$$\frac{\Gamma \vdash S \in K \qquad \Gamma \vdash T \in K \qquad S =_{\beta\wedge} T}{\Gamma \vdash S \leq T} \tag{S-Conv}$$

$$\frac{\Gamma \vdash S \leq T \qquad \Gamma \vdash T \leq U}{\Gamma \vdash S \leq U} \tag{S-Trans}$$

$$\frac{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash \mathrm{ok}}{\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash X \leq T} \tag{S-TVar}$$

$$\frac{\Gamma \vdash T_1 \leq S_1 \qquad \Gamma \vdash S_2 \leq T_2 \qquad \Gamma \vdash S_1 \rightarrow S_2 \in \star}{\Gamma \vdash S_1 \rightarrow S_2 \leq T_1 \rightarrow T_2} \tag{S-Arrow}$$

$$\frac{\Gamma \vdash T_1 \leq S_1 \qquad \Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash S_2 \leq T_2 \qquad \Gamma \vdash \mathrm{All}\,(X{\leq}S_1{:}K_1)\,S_2 \in \star}{\Gamma \vdash \mathrm{All}\,(X{\leq}S_1{:}K_1)\,S_2 \leq \mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2} \tag{S-All}$$

$$\frac{\Gamma,\, X{\leq}\top^{K}{:}K \vdash S \leq T}{\Gamma \vdash \mathrm{Fun}\,(X{:}K)\,S \leq \mathrm{Fun}\,(X{:}K)\,T} \tag{S-OAbs}$$

$$\frac{\Gamma \vdash S \leq T \qquad \Gamma \vdash S\,U \in K}{\Gamma \vdash S\,U \leq T\,U} \tag{S-OApp}$$

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

$$\frac{\Gamma \vdash \bigwedge^K[T_1..T_n] \in K}{\Gamma \vdash \bigwedge^K[T_1..T_n] \leq T_i} \tag{S-Meet-LB}$$

2.1.6 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 typechecking 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.

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

$$\frac{\Gamma,\, x{:}T_1 \vdash e \in T_2}{\Gamma \vdash \text{fun}\,(x{:}T_1)\, e \in T_1 {\rightarrow} T_2} \qquad \text{(T-Abs)}$$

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

$$\frac{\Gamma,\, X {\leq} T_1{:}K_1 \vdash e \in T_2}{\Gamma \vdash \text{fun}\,(X {\leq} T_1{:}K_1)\, e \in \text{All}\,(X {\leq} T_1{:}K_1)\, T_2} \qquad \text{(T-TAbs)}$$

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

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

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

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

2.2 The Church-Rosser theorem

In this section we prove the Church-Rosser property for the reduction defined in 2.1.2. The strategy we use here is similar to the one used in chapter 11 section 1 of [Bar84] 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 –. Given S, T_1, T_2 in $\mathbb{T}$ such that $S \rightarrow_{\beta\wedge} T_1$ and $S \twoheadrightarrow_{\beta\wedge} T_2$, then there exists T_3 such that $T_1 \twoheadrightarrow_{\beta\wedge} T_3$ and $T_2 \twoheadrightarrow_{\beta\wedge} T_3$. Graphically:

$$\begin{tikzcd}
S \arrow[r, two heads, "\beta\wedge"'] \arrow[d, "\beta\wedge"'] & T_2 \arrow[d, two heads, "\beta\wedge"] \\
T_1 \arrow[r, two heads, "\beta\wedge"'] & T_3
\end{tikzcd}$$

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 one 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. Then if we underline R in S we only need to reduce all occurrences of the underlined R in T_2 to obtain T_3.

Definition 2.2.1 (Underlining) *1. $\underline{\mathbb{T}}$ is the set of terms defined by the following abstract syntax.*

$$
\begin{array}{rcl}
\underline{\mathbb{T}} & ::= & X \mid \underline{\mathbb{T}}\to\underline{\mathbb{T}} \\
& \mid & All\,(X\leq\underline{\mathbb{T}}{:}\mathbb{K})\;\underline{\mathbb{T}} \mid Fun\,(X{:}\mathbb{K})\;\underline{\mathbb{T}} \\
& \mid & \underline{\mathbb{T}}\underline{\mathbb{T}} \mid \bigwedge^{\mathbb{K}}[\underline{\mathbb{T}}..\underline{\mathbb{T}}] \\
& \mid & (Fun\,(X{:}\mathbb{K})\;\underline{\mathbb{T}})\;\underline{\mathbb{T}} \mid \bigwedge^{\mathbb{K}}[\underline{\mathbb{T}}\;..\;\underline{\bigwedge^{\mathbb{K}}[\underline{\mathbb{T}}\;..\;\underline{\mathbb{T}}]}\;..\;\underline{\mathbb{T}}] \\
& \mid & \underline{\underline{\mathbb{T}}\to\bigwedge^{\star}[\underline{\mathbb{T}}\;..\;\underline{\mathbb{T}}]} \mid \underline{All\,(X\leq\underline{\mathbb{T}}{:}\mathbb{K})\;\bigwedge^{\star}[\underline{\mathbb{T}}\;..\;\underline{\mathbb{T}}]} \\
& \mid & \underline{Fun\,(X{:}\mathbb{K})\;\bigwedge^{\mathbb{K}}[\underline{\mathbb{T}}\;..\;\underline{\mathbb{T}}]} \mid \underline{\bigwedge^{\mathbb{K}\to\mathbb{K}}[\underline{\mathbb{T}}\;..\;\underline{\mathbb{T}}]\;\underline{\mathbb{T}}}
\end{array}
$$

Observe that only redexes are underlined.

2. *Underlined (one step) reduction $\to_{\underline{\beta\wedge}}$ is defined starting with the rewriting rules*

   (a) $(Fun\,(X{:}K)\,T_1)\,T_2 \to_{\underline{\beta\wedge}} T_1[X\leftarrow T_2]$
   (b) $S \to \bigwedge^{\star}[T_1..T_n] \to_{\underline{\beta\wedge}} \bigwedge^{\star}[S\to T_1\;..\;S\to T_n]$
   (c) $All\,(X\leq S{:}K)\,\bigwedge^{\star}[T_1..T_n] \to_{\underline{\beta\wedge}} \bigwedge^{\star}[All\,(X\leq S{:}K)\,T_1\;..\;All\,(X\leq S{:}K)\,T_n]$
   (d) $Fun\,(X{:}K_1)\,\bigwedge^{K_2}[T_1..T_n] \to_{\underline{\beta\wedge}} \bigwedge^{K_1\to K_2}[Fun\,(X{:}K_1)\,T_1\;..\;Fun\,(X{:}K_1)\,T_n]$
   (e) $\bigwedge^{K_1\to K_2}[T_1..T_n]\,U \to_{\underline{\beta\wedge}} \bigwedge^{K_2}[T_1\,U\;..\;T_n\,U]$
   (f) $\bigwedge^{K}[T_1\;..\;\bigwedge^{K}[S_1..S_n]\;..\;T_m] \to_{\underline{\beta\wedge}} \bigwedge^{K}[T_1\;..\;S_1..S_n\;..\;T_m]$
   (g) $\underline{(Fun\,(X{:}K)\,T_1)\,T_2} \to_{\underline{\beta\wedge}} T_1[X\leftarrow T_2]$
   (h) $\bigwedge^{K}[T_1\;..\;\underline{\bigwedge^{K}[S_1..S_n]}\;..\;T_m] \to_{\underline{\beta\wedge}} \bigwedge^{K}[T_1\;..\;S_1..S_n\;..\;T_m]$
   (i) $\underline{S \to \bigwedge^{\star}[T_1..T_n]} \to_{\underline{\beta\wedge}} \bigwedge^{\star}[S\to T_1\;..\;S\to T_n]$
   (j) $\underline{All\,(X\leq S{:}K)\,\bigwedge^{\star}[T_1..T_n]} \to_{\underline{\beta\wedge}} \bigwedge^{\star}[All\,(X\leq S{:}K)\,T_1\;..\;All\,(X\leq S{:}K)\,T_n]$
   (k) $\underline{Fun\,(X{:}K_1)\,\bigwedge^{K_2}[T_1..T_n]} \to_{\underline{\beta\wedge}} \bigwedge^{K_1\to K_2}[Fun\,(X{:}K_1)\,T_1\;..\;Fun\,(X{:}K_1)\,T_n]$
   (l) $\underline{\bigwedge^{K_1\to K_2}[T_1..T_n]\,U} \to_{\underline{\beta\wedge}} \bigwedge^{K_2}[T_1\,U\;..\;T_n\,U]$.

   $\to_{\underline{\beta\wedge}}$ is extended as to become a compatible relation with respect to $\underline{\mathbb{T}}$, and $\twoheadrightarrow_{\underline{\beta\wedge}}$ is the transitive reflexive closure of $\to_{\underline{\beta\wedge}}$.

3. *If $T \in \underline{\mathbb{T}}$, then $|T| \in \mathbb{T}$ is obtained from T by erasing all underlinings.*

4. *The capture avoiding substitution for underlined terms is written as usual, $T[X\leftarrow S]$.*

Definition 2.2.2 *The map $\varphi : \underline{\mathbb{T}} \to \mathbb{T}$ is defined inductively as follows.*

1. $\varphi(X) \equiv X$;

2. $\varphi(T_1 \to \mathbb{T}_2) \equiv \varphi(T_1) \to \varphi(\mathbb{T}_2)$;

3. $\varphi(All\,(X\leq T_1{:}K)\,T_2) \equiv All\,(X\leq\varphi(T_1)\,{:}K)\varphi(T_2)$;

4. $\varphi(Fun\,(X{:}K)\,T) \equiv Fun\,(X{:}K)\,\varphi(T)$;

5. $\varphi(T_1\,T_2) \equiv \varphi(T_1)\,\varphi(T_2)$;

6. $\varphi(\bigwedge^K[T_1..T_n]) \equiv \bigwedge^K[\varphi(T_1)\;..\;\varphi(T_n)]$;

7. $\varphi(\underline{(Fun\,(X{:}K)\,T_1)\,T_2}) \equiv \varphi(T_1)[X\leftarrow\varphi(T_2)]$;

8. $\varphi(\underline{S \to \bigwedge^\star[T_1..T_n]}) \equiv \bigwedge^\star[\varphi(S \to T_1)\;..\;\varphi(S \to T_n)]$;

9. $\varphi(\underline{All\,(X{\leq}S{:}K_1)\,\bigwedge^\star[T_1..T_n]}) \equiv \bigwedge^\star[\varphi(All\,(X{\leq}S{:}K_1)\,T_1)\;..\;\varphi(All\,(X{\leq}S{:}K_1)\,T_n)]$;

10. $\varphi(\underline{Fun\,(X{:}K_1)\,\bigwedge^{K_2}[T_1..T_n]}) \equiv \bigwedge^{K_1\to K_2}[\varphi(Fun\,(X{:}K_1)\,T_1)\;..\;\varphi(Fun\,(X{:}K_1)\,T_n)]$;

11. $\varphi(\underline{\bigwedge^{K_1\to K_2}[T_1..T_n]\,S}) \equiv \bigwedge^{K_2}[\varphi(T_1\,S)\;..\;\varphi(T_n\,S)]$;

12. $\varphi(\bigwedge^K[T_1\;..\;\underline{\bigwedge^K[S_1..S_n]}\;..\;T_m]) \equiv \bigwedge^K[\varphi(T_1)\;..\;\varphi(S_1)\;..\;\varphi(S_n)\;..\;\varphi(T_m)]$.

Observe that φ reduces all underlined redexes.

Notation: $|T| \equiv S$ and $\varphi(T) \equiv S$ will be written:

$$T \xrightarrow{|-|} S \text{ and } T \xrightarrow{\varphi} S.$$

Lemma 2.2.3 *Given $T, S \in \mathbb{T}$ and $T' \in \underline{\mathbb{T}}$, if $|T'| \equiv T$ and $T \twoheadrightarrow_{\beta\wedge} S$, then there exists $S' \in \underline{\mathbb{T}}$ such that $T' \twoheadrightarrow_{\underline{\beta\wedge}} S'$ and $|S'| \equiv S$. Graphically:*

$$\begin{tikzcd}
T' \arrow[r, twoheadrightarrow, "\underline{\beta\wedge}"'] \arrow[d, "|-|"'] & S' \arrow[d, "|-|"] \\
T \arrow[r, twoheadrightarrow, "\beta\wedge"'] & S
\end{tikzcd}$$

Proof: By induction on the definition of $\twoheadrightarrow_{\beta\wedge}$.

1. $T \to_{\beta\wedge} S$ (in one step). Then S is obtained by contracting a redex in T. S' can be obtained by contracting the corresponding redex in T'.

2. $T \twoheadrightarrow_{\beta\wedge} T$. Take $S' \equiv T'$.

3. $T \twoheadrightarrow_{\beta\wedge} U$ and $U \twoheadrightarrow_{\beta\wedge} S$. Then by induction hypothesis and transitivity of $\twoheadrightarrow_{\underline{\beta\wedge}}$ the result follows.

Lemma 2.2.4 *Let $S, T, U \in \underline{\mathbb{T}}$. Then*

1. *Suppose $X \neq Y$ and $X \notin FV(U)$. Then*
   $S[X\leftarrow T][Y\leftarrow U] \equiv S[Y\leftarrow U][X\leftarrow T[Y\leftarrow U]]$.

2. $\varphi(S[X\leftarrow T]) \equiv \varphi(S)[X\leftarrow\varphi(T)]$.

3. *If $S \twoheadrightarrow_{\beta\wedge} S'$, then $S[X{\leftarrow}U] \twoheadrightarrow_{\beta\wedge} S'[X{\leftarrow}U]$.*
4. *Given $T, S \in \underline{\mathbb{T}}$, if $T \twoheadrightarrow_{\underline{\beta\wedge}} S$, then $\varphi(T) \twoheadrightarrow_{\beta\wedge} \varphi(S)$. Graphically:*

$$\begin{tikzcd} T \arrow[r, two heads, "\underline{\beta\wedge}"'] \arrow[d, "\varphi"'] & S \arrow[d, "\varphi"] \\ \varphi(T) \arrow[r, two heads, "\beta\wedge"'] & \varphi(S) \end{tikzcd}$$

Proof:

1. By induction on the structure of S.
2. It is enough to show the result for $\rightarrow_{\beta\wedge}$. The rest follows by induction. The interesting case is when S is a redex. If S is other than a β-redex, then the result follows easily using the definition of substitution. Otherwise the result follows easily using 1.
3. By induction on the structure of S using (1) in the case $S \equiv \underline{(\mathrm{Fun}\,(X{:}K)\,S_1)\,S_2}$.
4. By induction on the generation of $\twoheadrightarrow_{\underline{\beta\wedge}}$, using (2).

Lemma 2.2.5 *Let $T \in \underline{\mathbb{T}}$. Then $|T| \twoheadrightarrow_{\beta\wedge} \varphi(T)$. Graphically:*

$$\begin{tikzcd} & T \arrow[dl, "|-|"'] \arrow[dr, "\varphi"] & \\ |T| \arrow[rr, two heads, "\beta\wedge"] & & \varphi(T) \end{tikzcd}$$

Proof: By induction on the structure of T.

Lemma 2.2.6 (Strip lemma) *Let $S, T_1, T_2 \in \mathbb{T}$. 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 $T_2 \twoheadrightarrow_{\beta\wedge} T_3$. Graphically:*

$$\begin{tikzcd} S \arrow[r, two heads, "\beta\wedge"'] \arrow[d, "\beta\wedge"'] & T_2 \arrow[d, two heads, "\beta\wedge"] \\ T_1 \arrow[r, two heads, "\beta\wedge"'] & T_3 \end{tikzcd}$$

Proof: Suppose that T_1 is obtained from S by replacing the occurrence redex R by its reduct R'. Then we can write $S \equiv S[R]$ and $T_1 \equiv S[R']$. Let $S[\underline{R}]$ be obtained from S by replacing R by its underlined version $\underline{R}$. Observe that $|S[\underline{R}]| \equiv S[R]$ and $\varphi(S[\underline{R}]) \equiv S[R']$. Then, by lemma 2.2.3, there exists T_2', by lemma 2.2.4(4), $S[R'] \twoheadrightarrow_{\beta\wedge} \varphi(T_2')$, and, by lemma 2.2.5, $T_2 \twoheadrightarrow_{\beta\wedge} \varphi(T_2')$, which justify the following diagram.

$$
\begin{tikzcd}
S[R] \arrow[rr, two heads, "\beta\wedge"] \arrow[dd, "\beta\wedge"'] & & T_2 \arrow[dd, two heads, "\beta\wedge"] & \\
& S[\underline{R}] \arrow[ul, "|-|"'] \arrow[dl, "\varphi"] \arrow[rr, "\beta\wedge"] & & T_2' \arrow[ul, "|-|"'] \arrow[dl, "\varphi"] \\
S[R'] \arrow[rr, two heads, "\beta\wedge"] & & \varphi(T_2') &
\end{tikzcd}
$$

Now, to complete the proof let $T_3 \equiv \varphi(T_2')$.

Theorem 2.2.7 (Church-Rosser Theorem for $\rightarrow_{\beta\wedge}$) *Let $S, T_1, T_2 \in \mathbb{T}$. If $S \twoheadrightarrow_{\beta\wedge} T_1$ 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 $T_2 \twoheadrightarrow_{\beta\wedge} T_3$. Graphically:*

$$
\begin{tikzcd}
S \arrow[r, two heads, "\beta\wedge"'] \arrow[d, two heads, "\beta\wedge"'] & T_2 \arrow[d, two heads, "\beta\wedge"] \\
T_1 \arrow[r, two heads, "\beta\wedge"'] & T_3
\end{tikzcd}
$$

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

1. $S \rightarrow_{\beta\wedge} T_1$ By the strip lemma.
2. $S \equiv T_1$ Take $T_3 \equiv T_2$.
3. $S \twoheadrightarrow_{\beta\wedge} T_1'$ and $T_1' \twoheadrightarrow_{\beta\wedge} T_1$ By the induction hypothesis we can find first T_3' and then T_3, such that $T_1' \twoheadrightarrow_{\beta\wedge} T_3'$, $T_2 \twoheadrightarrow_{\beta\wedge} T_3'$, $T_1 \twoheadrightarrow_{\beta\wedge} T_3$, and $T_3' \twoheadrightarrow_{\beta\wedge} T_3$. Hence, by the transitivity of $\twoheadrightarrow_{\beta\wedge}$, the result follows.

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

Corollary 2.2.8 – *See [Bar84] – Let R be a reduction satisfying the CR 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.*

2.3 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 2.3.1 *If $\Gamma \vdash \Sigma$ and Γ_1 is a prefix of Γ, then $\Gamma_1 \vdash ok$ as a subderivation. Moreover, except for the case $\Gamma_1 \equiv \Gamma$ and $\Sigma \equiv ok$, the subderivation is strictly shorter.*

Lemma 2.3.2 (Syntax-directedness of context judgements)

1. *If $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash ok$, then $\Gamma_1 \vdash T \in K$ by a proper subderivation.*

2. *If $\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash ok$, then $\Gamma_1 \vdash T \in \star$ by a proper subderivation.*

Lemma 2.3.3 (Free variables lemma)

1. *$\Gamma \vdash T \in K$ implies $FTV(T) \subseteq dom(\Gamma)$.*

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

Lemma 2.3.4 (Weakening/Permutation) *Let Γ and Γ' be contexts such that $\Gamma \subseteq \Gamma'$ and $\Gamma' \vdash ok$. Then $\Gamma \vdash \Sigma$ implies $\Gamma' \vdash \Sigma$.*

Proof: By induction on the length of a derivation of $\Gamma \vdash \Sigma$.

K-ALL — We are given that $\Gamma \vdash T_1 \in K_1$ and that $\Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$. Applying K-MEET to $\Gamma' \vdash$ ok we obtain $\Gamma' \vdash \top^{K_1} \in K_1$; we can assume, without loss of generality, that $X \notin \mathrm{dom}(\Gamma')$. Then, by C-TVAR, $\Gamma',\, X{\leq}\top^{K_1}{:}K_1 \vdash$ ok. Now, by induction hypothesis, $\Gamma' \vdash T_1 \in K_1$ and $\Gamma',\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$, and applying K-ALL the result follows.

T-ABS — We are given that $\Gamma,\, x{:}T_1 \vdash e \in T_2$. By lemma 2.3.1 there exists a proper subderivation of $\Gamma,\, x{:}T_1 \vdash$ ok; by lemma 2.3.2, there is a yet shorter subderivation of $\Gamma \vdash T_1 \in \star$. Now we can apply the induction hypothesis to obtain $\Gamma' \vdash T_1 \in \star$. As before, we can assume $x \notin \mathrm{dom}(\Gamma')$; by C-VAR, $\Gamma',\, x{:}T_1 \vdash$ ok. By the induction hypothesis we have $\Gamma',\, x{:}T_1 \vdash e \in T_2$, and applying T-ABS yields the desired result.

Other cases — If $\Sigma \equiv$ ok there is nothing to prove. K-TVAR, S-TVAR, T-MEET and T-VAR applying the corresponding rule to $\Gamma' \vdash$ ok. K-OABS, S-ALL, and S-OABS similar to K-ALL. T-TABS similar to T-ABS. All the other cases follow by straightforward application of the induction hypothesis.

Lemma 2.3.5 (Context, kind, and term strengthening)

1. *If $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash ok$ and $X \notin FTV(\Gamma_2)$, then $\Gamma_1,\, \Gamma_2 \vdash ok$.*

2. *If $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash S \in K'$ and $X \notin FTV(\Gamma_2) \cup FTV(S)$, then $\Gamma_1,\, \Gamma_2 \vdash S \in K'$.*

3. If $\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \Sigma$ and $x \notin FV(\Sigma)$, then $\Gamma_1,\, \Gamma_2 \vdash \Sigma$.

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

Proof: 1 and 2 are proved by simultaneous induction on the length of derivations. 3 is proved by induction on the derivation of $\Gamma_1,\, x{:}T,\, \Gamma_2 \vdash \Sigma$.

1. C-EMPTY Vacuously true.

   C-VAR By part 2 of the induction hypothesis and C-VAR.

   C-TVAR $\Gamma_2 \equiv \emptyset$ The result follows from lemma 2.3.1.
   $\Gamma_2 \not\equiv \emptyset$ By part 2 of the induction hypothesis and C-TVAR.

2. K-TVAR By part 1 of the induction hypothesis and K-TVAR.

   K-ARROW By part 2 of the induction hypothesis and K-ARROW.

   K-ALL We are given $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2 \vdash T_1 \in K_1$, and $\Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2,\, Y{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$, and $X \notin \mathrm{FTV}(\Gamma_2) \cup \mathrm{FTV}(\mathrm{All}\,(Y{\leq}T_1{:}K_1)\,T_2)$. Observe that, by lemma 2.3.3, $X \not\equiv Y$. Then $X \notin \mathrm{FTV}(\Gamma_2) \cup \mathrm{FTV}(T_1)$ and $X \notin \mathrm{FTV}(\Gamma_2,\, Y{\leq}\top^{K_1}{:}K_1) \cup \mathrm{FTV}(T_2)$. Now, applying part 2 of the induction hypothesis and K-ALL the result follows.

   K-OABS Similar to the case K-ALL.

   K-OAPP By part 2 of the induction hypothesis and K-OAPP.

   K-MEET By parts 1 and 2 of the induction hypothesis and K-OAPP.

3. Except for the cases we will consider below and the case for C-EMPTY, which is trivially true, the result follows by straightforward application of part 3 of the induction hypothesis and the corresponding rule in each case.

   C-VAR $\Gamma_2 \equiv \emptyset$ The result follows by lemma 2.3.1.
   $\Gamma_2 \not\equiv \emptyset$ By part 3 of the induction hypothesis and C-VAR.

   T-ABS Using lemma 2.3.3, part 3 of the induction hypothesis, and T-ABS.

   T-FOR Using that $\mathrm{FV}(\mathrm{for}\,(X \in T_1..T_n)\,e) = \mathrm{FV}(e) = \mathrm{FV}(e[X{\leftarrow}S])$, part 3 of the induction hypothesis, and T-FOR.

Proposition 2.3.6 (Syntax-directedness of kinding/Generation lemma for kinding) *1. $\Gamma \vdash X \in K$ implies $\Gamma \equiv \Gamma_1,\, X{\leq}T{:}K,\, \Gamma_2$ for some Γ_1, T, and Γ_2.*

2. $\Gamma \vdash T_1{\rightarrow}T_2 \in K$ implies $K \equiv \star$ and $\Gamma \vdash T_1, T_2 \in \star$.

3. $\Gamma \vdash All\,(X{\leq}T_1{:}K_1)\,T_2 \in K$ implies $K \equiv \star$ and $\Gamma \vdash T_1 \in K_1$ and $\Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$.

4. $\Gamma \vdash Fun\,(X{:}K_1)\,T_2 \in K$ implies $K \equiv K_1{\rightarrow}K_2$ and $\Gamma,\, X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in K_2$, for some K_2.

5. $\Gamma \vdash S\,T \in K$ *implies* $\Gamma \vdash S \in K' \rightarrow K$ *and* $\Gamma \vdash T \in K'$*, for some* K'*.*

6. $\Gamma \vdash \bigwedge^K [T_1..T_n] \in K'$ *implies* $K \equiv K'$ *and* $\Gamma \vdash ok$ *and* $\Gamma \vdash T_i \in K$ *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 2.3.7 (Uniqueness of kinds) $\Gamma \vdash T \in K$ *and* $\Gamma \vdash T \in K'$*, then* $K \equiv K'$*.*

Definition 2.3.8 (Size) 1. *The size of a type expression* T*,* $\mathrm{size}_t(T)$*, is defined as follows.*

(a) $\mathrm{size}_t(X) = 3$,

(b) $\mathrm{size}_t(S \rightarrow T) = \mathrm{size}_t(All\,(X \leq S{:}K)\,T) = \mathrm{size}_t(S\,T) = \mathrm{size}_t(S) + \mathrm{size}_t(T) + 1$,

(c) $\mathrm{size}_t(Fun\,(X{:}K)\,T) = \mathrm{size}_t(T) + 3$,

(d) $\mathrm{size}_t(\bigwedge^K [T_1..T_n]) = 2 + \Sigma_{1 \leq i \leq n} \mathrm{size}_t(T_i)$.

2. *The homomorphic extension to contexts,* $\mathrm{size}_c(\Gamma)$*, is defined as follows.*

(a) $\mathrm{size}_c(\emptyset) = 0$,

(b) $\mathrm{size}_c(\Gamma,\, X \leq T{:}K) = \mathrm{size}_c(\Gamma,\, x{:}T) = \mathrm{size}_c(\Gamma) + \mathrm{size}_t(T)$.

3. *The size of a kinding and ok judgement* J*,* $\mathrm{size}_j(J)$*, is defined as follows.*

(a) $\mathrm{size}_j(\Gamma \vdash ok) = \mathrm{size}_c(\Gamma) + 1$,

(b) $\mathrm{size}_j(\Gamma \vdash T \in K) = \mathrm{size}_c(\Gamma) + \mathrm{size}_t(T)$.

Lemma 2.3.9 *Let* T *be a type expression. Then* $\mathrm{size}_t(\top^K) \leq \mathrm{size}_t(T)$ *for any kind* K*.*

Lemma 2.3.10 (Well-foundedness of context formation and kinding rules) 1. *For every kinding or ok judgement* J*,* $\mathrm{size}_j(\emptyset \vdash ok) \leq \mathrm{size}_j(J)$*.*

2. *If* $\dfrac{J_1\ ..\ J_n}{J}$ *is a kinding rule or a context formation rule, then* $\mathrm{size}_j(J_i) < \mathrm{size}_j(J)$ *for each* $i \in \{1..n\}$ *.*

Corollary 2.3.11 1. *For any context* Γ *it is decidable whether* $\Gamma \vdash ok$*.*

2. *For any context* Γ*, type expression* T*, and kind* K*, it is decidable whether* $\Gamma \vdash T \in K$*.*

Proof: Lemma 2.3.2 and proposition 2.3.6 imply that context formation rules and kinding rules determine an algorithm to check context judgements and kinding judgements and lemma 2.3.10 implies that the algorithm terminates.

Lemma 2.3.12 (Type substitution) *Let* $\Gamma_1 \vdash T \in K_U$*. Then*

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

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

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. Now take the case of K-TVar, where $S \equiv Y$ for some variable Y. By proposition 2.3.6(1) $Y{\leq}T_Y{:}K_S \in (\Gamma_1, X{\leq}U{:}K_U, \Gamma_2)$ for some T_Y. There are three cases to consider.

$Y{\leq}T_Y{:}K_S \in \Gamma_1$	Then we also have $Y{\leq}T_Y{:}K_S \in (\Gamma_1, \Gamma_2[X{\leftarrow}T])$. By part 2 of the induction hypothesis, $\Gamma_1, \Gamma_2[X{\leftarrow}T] \vdash$ ok. Applying K-TVar, we get $\Gamma_1, \Gamma_2[X{\leftarrow}T] \vdash Y \in K_S$.
$Y{\leq}T_Y{:}K_S \equiv X{\leq}U{:}K_U$	We know that $\Gamma_1 \vdash T \in K_S \equiv K_U$. From the premise of K-TVar and part 2 of the induction hypothesis, we have $\Gamma_1, \Gamma_2[X{\leftarrow}T] \vdash$ ok. The result follows by weakening (2.3.4).
$Y{\leq}T_Y{:}K_S \in \Gamma_2$	Then we have $Y{\leq}T_Y[X{\leftarrow}T]{:}K_S \in (\Gamma_1, \Gamma_2[X{\leftarrow}T])$. By part 2 of the induction hypothesis, $\Gamma_1, \Gamma_2[X{\leftarrow}T] \vdash$ ok, from which the result follows from K-TVar.

Lemma 2.3.13 (Subject reduction for kinding judgements) *If* $S \twoheadrightarrow_{\beta\wedge} T$ *and* $\Gamma \vdash S \in K$*, then* $\Gamma \vdash T \in K$*.*

Proof sketch: In order to prove this result it is enough to prove the following statement by induction on the derivation of $\Gamma \vdash S \in K$. The rest follows by induction on the definition of $\twoheadrightarrow_{\beta\wedge}$. $\Gamma \vdash S{\in}K$ and $S \rightarrow_{\beta\wedge} T$ implies $\Gamma \vdash T{\in}K$.

Theorem 2.3.14 (Kind invariance under type conversion) *If* $\Gamma \vdash S \in K_S$ *and* $\Gamma \vdash T \in K_T$*, with* $S =_{\beta\wedge} T$*, then* $K_S \equiv K_T$*.*

Proof: By the Church-Rosser theorem 2.2.7, there exists U such that $S \twoheadrightarrow_{\beta\wedge} U$ and $T \twoheadrightarrow_{\beta\wedge} U$, and by subject reduction and unicity of kinds the result follows.

Lemma 2.3.15 *Let* $\Gamma \vdash S, T{\in}K$*. Then* $\Gamma \vdash S \leq T \iff \Gamma \vdash S^{nf} \leq T^{nf}$*.*

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

$\Rightarrow$) By subject reduction, we have that $\Gamma \vdash S^{nf}{\in}K$, then, by S-Conv, $\Gamma \vdash S^{nf} \leq S$. By similar reasoning we have $\Gamma \vdash T \leq T^{nf}$. Finally, the result follows applying twice S-Trans.

Lemma 2.3.16 (Context modification) *If* $\Gamma_1 \vdash U' \in K$ *and* Σ *is either ok or* $T \in K'$*, then* $\Gamma_1, X{\leq}U{:}K, \Gamma_2 \vdash \Sigma$ *implies* $\Gamma_1, X{\leq}U'{:}K, \Gamma_2 \vdash \Sigma$*.*

Proposition 2.3.17 (Well-kindedness of subtyping) *If* $\Gamma \vdash S \leq T$, *then* $\Gamma \vdash S \in K$ *and* $\Gamma \vdash T \in K$ *for some* K.

Proof: By induction on the derivation of $\Gamma \vdash S \leq T$.

S-CONV We are given that $\Gamma \vdash S \in K$ and $\Gamma \vdash T \in K'$ and $S =_\beta T$. By lemma 2.3.14, $K \equiv K'$.

S-TRANS By induction hypothesis and uniqueness of kinds (2.3.7).

S-TVAR We are given that $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash \text{ok}$. By K-TVAR we get $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash X \in K$. Moreover, by lemma 2.3.2 we have $\Gamma_1 \vdash T \in K$, and by lemma 2.3.4, $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash T \in K$.

S-ARROW We are given $\Gamma \vdash T_1 \leq S_1$ and $\Gamma \vdash S_2 \leq T_2$ and $\Gamma \vdash S_1{\rightarrow}S_2 \in \star$. By proposition 2.3.6 $\Gamma \vdash S_1, S_2 \in \star$. Further, by the induction hypothesis together with uniqueness of kinds (2.3.7), we have $\Gamma \vdash T_1, T_2 \in \star$. Finally, applying K-ARROW, the result follows.

S-ALL We are given that $\Gamma \vdash T_1 \leq S_1$ and $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash S_2 \leq T_2$ and also $\Gamma \vdash \text{All}\,(X{\leq}S_1{:}K_1)\,S_2 \in \star$. By proposition 2.3.6 $\Gamma \vdash S_1 \in K_1$ and $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash S_2 \in \star$. Then, applying the induction hypothesis and lemma 2.3.7, we obtain $\Gamma \vdash T_1 \in K_1$ and $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$ from which, by K-ALL, the result follows.

S-OABS By induction hypothesis and K-OABS.

S-OAPP Similar to S-ALL.

S-MEET-G Using the induction hypothesis, lemma 2.3.7, and K-MEET.

S-MEET-LB We are given $\Gamma \vdash \bigwedge^K[T_1..T_n] \in K$, which, by proposition 2.3.6, implies $\Gamma \vdash T_i \in K$ for each i.

Proposition 2.3.18 (Well-kindedness of typing) *If* $\Gamma \vdash e \in T$, *then* $\Gamma \vdash T \in \star$.

Proof: By induction on the derivation of $\Gamma \vdash e \in T$.

T-VAR We are given $\Gamma_1,\ x{:}T,\ \Gamma_2 \vdash \text{ok}$. The result follows by lemma 2.3.2 and lemma 2.3.4.

T-ABS We are given $\Gamma,\ x{:}T_1 \vdash e \in T_2$. By induction hypothesis $\Gamma,\ x{:}T_1 \vdash T_2 \in \star$. By lemma 2.3.5, it follows that $\Gamma \vdash T_2 \in \star$. On the other hand, by lemmas 2.3.1 and 2.3.2, $\Gamma \vdash T_1 \in \star$. Hence, K-ARROW yields $\Gamma \vdash T_1{\rightarrow}T_2 \in \star$.

T-APP By induction hypothesis on $\Gamma \vdash f \in T_1{\rightarrow}T$ and proposition 2.3.6.

T-TABS We are given $\Gamma,\ X{\leq}T_1{:}K_1 \vdash e \in T_2$. By induction hypothesis $\Gamma,\ X{\leq}T_1{:}K_1 \vdash T_2 \in \star$ and, by the context-modification lemma (2.3.16), $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash e \in T_2$. Hence, applying K-ALL, we obtain $\Gamma \vdash \text{All}\,(X{\leq}T_1{:}K_1)\,T_2 \in \star$.

T-TApp We know that $\Gamma \vdash f \in \mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2$ and also $\Gamma \vdash S \leq T_1$. By the induction hypothesis, $\Gamma \vdash \mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2 \in \star$ and, by proposition 2.3.6, $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in \star$ and $\Gamma \vdash T_1 \in K_1$. Now, by the well-kindedness of subtyping (2.3.17) and uniqueness of kinds (2.3.7), we have $\Gamma \vdash S \in K_1$. Then, by the type substitution lemma (2.3.12),$\Gamma \vdash T_2[X{\leftarrow}S] \in \star$.

T-For By induction hypothesis.

T-Meet We are given that $\Gamma \vdash$ ok and that $\Gamma \vdash e \in T_i$ for each i. We have to consider two cases.

$n = 0$ Applying K-Meet to $\Gamma \vdash$ ok we obtain $\Gamma \vdash \top^{\star} \in \star$.

$n \neq 0$ By the induction hypothesis, $\Gamma \vdash T_i \in \star$ for every i and, applying K-Meet, the result follows.

T-Sub By induction hypothesis using proposition 2.3.17 and lemma 2.3.7.

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

To prove that every type that has a kind in $F^{\omega}_{\wedge}$ is strongly normalizing, we divide the proof in three steps. We first prove that $\rightarrow_a$ and also $\rightarrow_{\beta\wedge^-}$ are strongly normalizing. 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. 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 2.4.1 $\rightarrow_a$ *is strongly normalizing.*

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

The interpretation of a kind K, notation $[\![K]\!]$, is defined as follows.

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

Definition 2.4.2 (Saturated set) $\mathbb{S} \subseteq \mathrm{SN}$ *is* saturated *if*

1. If $R_1..R_n \in \mathrm{SN}$, *then* $X R_1..R_n \in \mathbb{S}$.

2. *If* $R_1..R_n, Q \in \mathrm{SN}$*, Then*

   (a) $P[X \leftarrow Q]R_1..R_n \in \mathbb{S} \Rightarrow (Fun\,(X{:}K)\;P)QR_1..R_n \in \mathbb{S}$*, for every* K *and*

   (b) $\bigwedge^{K_2}[T_1Q,\,..,\,T_mQ]R_1,\,..,\,R_n \in \mathbb{S} \Rightarrow \bigwedge^{K_1\to K_2}[T_1,\,..,\,T_m]QR_1,\,..,\,R_n \in \mathbb{S}$*, for every* K_1*.*

Lemma 2.4.3 *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 2.4.4 *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* $FV(T) = \{X_1..X_n\}$*.*

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

4. *A context* Γ satisfies $T \in K$*, written* $\Gamma \models T \in K$*, if* $\forall\rho[\rho \models \Gamma \;\Rightarrow\; \rho \models T \in K]$*.*

Proposition 2.4.5 (Soundness) *If* $\Gamma \vdash T \in K$*, then* $\Gamma \models T \in K$*.*

Proof: By induction on the derivation of $\Gamma \vdash T \in K$. 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 $\Gamma \vdash T_i \in K$ for each $i{\in}\{1..n\}$, and, by induction hypothesis $\Gamma \models T_i \in K$. Let ρ be a valuation such that $\rho \models \Gamma$. Then $[\![T_i]\!]_\rho{\in}K$. Now the proof follows by induction on the structure of K.

- $K \equiv \star$ — Then, by definition of $[\![K]\!]$, $[\![T_i]\!]_\rho{\in}SN$ for each $i{\in}\{1..n\}$, and every reduction starting from $[\![T]\!]_\rho$ is a reduction inside some $[\![T_j]\!]_\rho$.
- $K \equiv K_1{\to}K_2$ — Then $\Gamma \vdash T_j \in K_1{\to}K_2$ for each j, and, by induction hypothesis, $[\![T_j]\!]_\rho{\in}[\![K_1{\to}K_2]\!]$. If $A{\in}[\![K_1]\!]$, then $[\![T]\!]_\rho A = \bigwedge^{K_1\to K_2}[[\![T_1]\!]_\rho..[\![T_n]\!]_\rho]A \to_{\beta\wedge} \bigwedge^{K_2}[[\![T_1]\!]_\rho A..[\![T_n]\!]_\rho A]$. By the definition of $\to$, $[\![T_j]\!]_\rho A{\in}[\![K_2]\!]$. Now, as K_2 is smaller that $K_1{\to}K_2$, $\bigwedge^{K_2}[[\![T_1]\!]_\rho A..[\![T_n]\!]_\rho A]{\in}[\![K_2]\!]$, by induction hypothesis. Finally, by the saturation of $[\![K_2]\!]$, we conclude that $[\![T]\!]_\rho A{\in}K_2$, which proves that $[\![T]\!]_\rho{\in}[\![K_1{\to}K_2]\!]$.

$n \equiv 0$ $T \equiv \top^K$. Again we proceed by induction on the structure of K.

- $K \equiv \star$ — $[\![K]\!] \equiv SN$ and $\top^\star$ is in normal form.
- $K \equiv K_1{\to}K_2$ — Let $A{\in}[\![K_1]\!]$, then we have that $[\![T]\!]_\rho A = \top^{K_1\to K_2}A \to_{\beta\wedge} \top^{K_2}$, which, by induction hypothesis, is in $[\![K_2]\!]$. Now, by the saturation of $[\![K_2]\!]$, $[\![T]\!]_\rho A{\in}[\![K_2]\!]$, which proves that $[\![T]\!]_\rho{\in}[\![K_1{\to}K_2]\!]$.

Theorem 2.4.6 (Strong normalization for $\rightarrow_{\beta\wedge^-}$) *$\Gamma \vdash T \in K$ implies that every $(\beta\wedge^-)$-reduction sequence starting from T is finite.*

Proof: By soundness, $\Gamma \models T \in K$. 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 2.4.7 *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.

Now we can prove strong normalization of the largest reduction.

Theorem 2.4.8 (Strong normalization for $\rightarrow_{\beta\wedge}$) *$\Gamma \vdash T \in K$ implies that every $(\beta\wedge)$-reduction sequence starting from T is finite.*

Proof: Let $\Gamma \vdash T \in K$. Suppose that there is an infinite $\beta\wedge$-reduction sequence starting from T and derive a contradiction using 2.4.1, 2.4.6, and 2.4.7.

3 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.

In $F^\omega_\wedge$ we can prove

$$W{:}K, X \leq \mathrm{Fun}\,(Y{:}K)\,Y{:}K\rightarrow K,\ Z \leq X{:}K\rightarrow K \vdash X(Z\,W) \leq W \tag{1}$$

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 its derivations. For that we define a set of rules in which conversion is reduced to a minimum and transitivity can be eliminated with a standard cut-elimination argument in such a way that subtyping is syntax directed and we can still have a decision procedure for the original system. In subsection 3.2 we present the subtyping system $NF^\omega_\wedge$ that uses the context and type formation rules of $F^\omega_\wedge$. We define rewriting rules for derivations and prove that there exists a terminating procedure to normalize proofs, which gives, as a consequence, a generation lemma for subtyping (proposition 3.4.4). Finally, we show that there is an equivalence between subtyping in $F^\omega_\wedge$ and subtyping in $NF^\omega_\wedge$, which is essential to prove the decidability of subtyping in $F^\omega_\wedge$ (see section 4).

3.1 Syntax of $NF^\omega_\wedge$

Definition 3.1.1 *The sets of* normal types *and* simple normal types *are defined by the following abstract grammar.*

$$\begin{array}{rcl} \mathbb{A} & ::= & X \mid \mathbb{F}\rightarrow\mathbb{A} \mid \mathit{Fun}\,(X{:}\mathbb{K})\,\mathbb{A} \mid \mathbb{A}\mathbb{F} \mid \mathit{All}\,(X{\leq}\mathbb{F}{:}\mathbb{K})\,\mathbb{A} \\ \mathbb{F} & ::= & \bigwedge^K[\mathbb{A}..\,\mathbb{A}] \mid \mathbb{A} \end{array}$$

Furthermore, all types are in $\beta\wedge$-normal form.

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: A, B, C and D range over simple normal types, S, T, F, G and H, range over normal types.

Definition 3.1.2 (Least strict Upper Bound)

$$\begin{array}{rcl} LUB(\Gamma, X) & = & \Gamma(X), \\ LUB(\Gamma, A\,F) & = & LUB(\Gamma, A)\,F. \end{array}$$

3.2 Normal subtyping rules

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

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

$$\frac{\Gamma \vdash_n \Gamma(X) \leq A}{\Gamma \vdash_n X \leq A} \qquad \text{(NS-TVar)}$$

$$\frac{\Gamma \vdash_n F \leq G \qquad \Gamma \vdash_n A \leq B \qquad \Gamma \vdash G \rightarrow A \in \star}{\Gamma \vdash_n G \rightarrow A \leq F \rightarrow B} \qquad \text{(NS-Arrow)}$$

$$\frac{\Gamma \vdash_n F \leq G \qquad \Gamma, X \leq \top^K{:}K \vdash_n A \leq B \qquad \Gamma \vdash \mathrm{All}\,(X \leq G{:}K)\,A \in \star}{\Gamma \vdash_n \mathrm{All}\,(X \leq G{:}K)\,A \leq \mathrm{All}\,(X \leq F{:}K)\,B} \qquad \text{(NS-All)}$$

$$\frac{\Gamma, X \leq \top^K{:}K \vdash_n A \leq B}{\Gamma \vdash_n \mathrm{Fun}\,(X{:}K)\,A \leq \mathrm{Fun}\,(X{:}K)\,B} \qquad \text{(NS-OAbs)}$$

$$\frac{\Gamma \vdash_n (\mathrm{LUB}(\Gamma, A\,F))^{nf} \leq B \qquad \Gamma \vdash A\,F \in K}{\Gamma \vdash_n A\,F \leq B} \qquad \text{(NS-OApp)}$$

$$\frac{\Gamma \vdash A \in K \qquad \forall i {\in} \{1..m\}\, \Gamma \vdash_n A \leq B_i}{\Gamma \vdash_n A \leq \bigwedge^K [B_1..B_m]} \qquad \text{(NS-}\forall\text{)}$$

$$\frac{\forall k {\in} \{1..n\}\, \Gamma \vdash A_k \in K \qquad \exists j {\in} \{1..n\}\, \Gamma \vdash_n A_j \leq B}{\Gamma \vdash_n \bigwedge^K [A_1..A_n] \leq B} \qquad \text{(NS-}\exists\text{)}$$

$$\frac{\forall k {\in} \{1..n\}\, \Gamma \vdash A_k \in K \qquad \forall i {\in} \{1..m\}\, \exists j {\in} \{1..n\}\, \Gamma \vdash_n A_j \leq B_i}{\Gamma \vdash_n \bigwedge^K [A_1..A_n] \leq \bigwedge^K [B_1..B_m]} \qquad \text{(NS-}\forall\exists\text{)}$$

As we mentioned in the introduction, an important factor to develop the present system was to consider the distributivity rules of the presentation of $F^\omega_\wedge$ in [CP93] 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 (1). Hence, the solution is not as simple as replacing S-CONV by NS-REFL. This example illustrates the interaction between S-TRANS and S-CONV. (See section 3.6 for a derivation.)

In general terms such interaction can be analized 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 (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. 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 $F_{\leq}$ [Ghe90] 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 decide 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 $\Gamma \vdash A\,F \leq T$ when $T \not\equiv A\,F$, to check if the least strict upper bound of $A\,F$ is smaller than or equal to T (See lemma 3.5.1 and corollary 3.5.10). LUB$(\Gamma, A\,F)$ is obtained from $A\,F$ by replacing its leftmost innermost variable by the corresponding bound in Γ. Consequently, LUB$(\Gamma, A\,F)$ may be other than a normal form. That is the reason why we normalize the bound and here is where the strength of the conversion rule that is not captured by reflexivity is hidden. The rules S-MEET-LB and S-MEET-G are replaced by NS-$\exists$, NS-$\forall$, and NS-$\forall\exists$.

3.3 Basic properties of $NF^{\omega}_{\wedge}$

This section establishes a number of structural properties of $NF^{\omega}_{\wedge}$. The proofs of lemmas 3.3.1 and 3.3.2 are similar to those of the corresponding properties for $F^{\omega}_{\wedge}$.

Lemma 3.3.1 *If $\Gamma \vdash_n F \leq G$ and Γ_1 is a prefix of Γ, then $\Gamma_1 \vdash ok$ as a subderivation. Moreover, the subderivation is strictly shorter.*

Lemma 3.3.2 (Weakening/Permutation) *Let Γ and Γ' be contexts such that $\Gamma \subseteq \Gamma'$ and $\Gamma' \vdash ok$. Then $\Gamma \vdash_n S \leq T$ implies $\Gamma' \vdash_n S \leq T$.*

3.4 Normal derivations

In this subsection 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 are either the ones appearing as hypothesis in the redex or a proper subderivation of those missing ones as we proved in the generation lemma for kinding(2.3.6). The derivations of the kinding judgements of each reduct of the reflexivity rules are proper subderivations of the kinding judgements in its redex.

3.4.1 Reflexivity simplification rules

1. $$\boxed{\dfrac{\Gamma \vdash F{\to}A \in \star}{\Gamma \vdash_n F{\to}A \leq F{\to}A}\,\textsc{NS-Refl}}$$

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

2. $$\boxed{\dfrac{\Gamma \vdash \mathrm{All}\,(X{\leq}F{:}K)\,A \in \star}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}F{:}K)\,A \leq \mathrm{All}\,(X{\leq}F{:}K)\,A}\,\textsc{NS-Refl}}$$

$$\Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma \vdash F \in K}{\Gamma \vdash_n F \leq F}\,\textsc{NS-Refl} \qquad \dfrac{\Gamma,\ X{:}K \vdash A \in \star}{\Gamma,\ X{:}K \vdash_n A \leq A}\,\textsc{NS-Refl}}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}F{:}K)\,A \leq \mathrm{All}\,(X{\leq}F{:}K)\,A}\,\textsc{NS-All}}$$

3. $$\boxed{\dfrac{\Gamma \vdash \mathrm{Fun}\,(X{:}K)\,A \in K{\to}K'}{\Gamma \vdash_n \mathrm{Fun}\,(X{:}K)\,A \leq \mathrm{Fun}\,(X{:}K)\,A}\,\textsc{NS-Refl}} \Rightarrow_R \boxed{\dfrac{\dfrac{\Gamma,\ X{:}K \vdash A \in K'}{\Gamma,\ X{:}K \vdash_n A \leq A}\,\textsc{NS-Refl}}{\Gamma \vdash_n \mathrm{Fun}\,(X{:}K)\,A \leq \mathrm{Fun}\,(X{:}K)\,A}\,\textsc{NS-OAbs}}$$

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

3.4.2 Transitivity elimination rules

1. $$\boxed{\dfrac{\dfrac{\Gamma \vdash F \in K}{\Gamma \vdash_n F \leq F}\,\textsc{NS-Refl} \qquad \Gamma \vdash_n F \leq G}{\Gamma \vdash_n F \leq G}\,\textsc{NS-Trans}} \Rightarrow_T \boxed{\Gamma \vdash_n F \leq G}$$

2. $$\boxed{\dfrac{\Gamma \vdash_n F \leq G \qquad \dfrac{\Gamma \vdash G \in K}{\Gamma \vdash_n G \leq G}\,\textsc{NS-Refl}}{\Gamma \vdash_n F \leq G}\,\textsc{NS-Trans}} \Rightarrow_T \boxed{\Gamma \vdash_n F \leq G}$$

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. $$\boxed{\dfrac{\dfrac{\Gamma \vdash_n G \leq F \quad \Gamma \vdash_n A \leq B}{\Gamma \vdash_n F \rightarrow A \leq G \rightarrow B}\,\textsc{NS-Arrow} \qquad \dfrac{\Gamma \vdash_n H \leq G \quad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n G \rightarrow B \leq H \rightarrow C}\,\textsc{NS-Arrow}}{\Gamma \vdash_n F \rightarrow A \leq H \rightarrow C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma \vdash_n H \leq G \quad \Gamma \vdash_n G \leq F}{\Gamma \vdash_n H \leq F}\,\textsc{NS-Trans} \qquad \dfrac{\Gamma \vdash_n A \leq B \quad \Gamma \vdash_n B \leq C}{\Gamma \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n F \rightarrow A \leq H \rightarrow C}\,\textsc{NS-Arrow}}$$

5. $$\boxed{\dfrac{\dfrac{\Gamma \vdash_n G \leq F \quad \Gamma,\, X{:}K \vdash_n A \leq B}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}F{:}K)\,A \leq \mathrm{All}\,(X{\leq}G{:}K)\,B}\,\textsc{NS-All} \qquad \dfrac{\Gamma \vdash_n H \leq G \quad \Gamma,\, X{:}K \vdash_n B \leq C}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}G{:}K)\,B \leq \mathrm{All}\,(X{\leq}H{:}K)\,C}\,\textsc{NS-All}}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}F{:}K)\,A \leq \mathrm{All}\,(X{\leq}H{:}K)\,C}\,\textsc{NS-Trans}}$$

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma \vdash_n H \leq G \quad \Gamma \vdash_n G \leq F}{\Gamma \vdash_n H \leq F}\,\textsc{NS-Trans} \qquad \dfrac{\Gamma,\, X{:}K \vdash_n A \leq B \quad \Gamma,\, X{:}K \vdash_n B \leq C}{\Gamma,\, X{:}K \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n \mathrm{All}\,(X{\leq}F{:}K)\,A \leq \mathrm{All}\,(X{\leq}H{:}K)\,C}\,\textsc{NS-All}}$$

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

$$\Rightarrow_T \boxed{\dfrac{\dfrac{\Gamma,\, X{:}K \vdash_n A \leq B \quad \Gamma,\, X{:}K \vdash_n B \leq C}{\Gamma,\, X{:}K \vdash_n A \leq C}\,\textsc{NS-Trans}}{\Gamma \vdash_n \mathrm{Fun}\,(X{:}K)\,A \leq \mathrm{Fun}\,(X{:}K)\,C}\,\textsc{NS-OAbs}}$$

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

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

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

$$\Rightarrow_T \boxed{\dfrac{\Gamma \vdash_n A \le A_j \quad \Gamma \vdash_n A_j \le B}{\Gamma \vdash_n A \le B}\,\text{NS-Trans}}$$

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

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

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

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

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

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

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

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

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

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

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

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 transitivity nor reflexivity redexes. The elimination of NS-Trans follows a standard cut-elimination argument.

Lemma 3.4.1 (Transitivity elimination) *Let D be a derivation of a subtyping statement in refl-normal form 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 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 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 amount of times the lemma is applied is equal to the number of occurrences of NS-Trans in D. Furthermore, each reflexivity-rule yields, if any, reflexivity redexes of smaller complexity. Hence 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 3.4.2 (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 3.4.3 1. *A derivation in normal form whose last rule is* NS-Refl *is either a proof of $\Gamma \vdash_n X \leq X$ or $\Gamma \vdash_n A\,F \leq A\,F$*

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 induction hypothesis or because the last rule of at least one of the derivations of the premises of D constitutes a redex.

Proposition 3.4.4 (Generation lemma for normal subtyping)

1. $\Gamma \vdash_n X \leq B$ *implies* $X \equiv B$ *and* $\Gamma \vdash X \in K$ *for some* K, *or* $\Gamma \vdash_n \Gamma(X) \leq B$.

2. $\Gamma \vdash_n F \rightarrow A \leq B$ *implies* $B \equiv H \rightarrow C$, $\Gamma \vdash_n H \leq F$, $\Gamma \vdash_n A \leq C$, *and* $\Gamma \vdash F \rightarrow A \in \star$.

3. $\Gamma \vdash_n All(X{\leq}F{:}K)\, A \leq B$ *implies* $B \equiv All(X{\leq}H{:}K)\, C$, $\Gamma \vdash_n H \leq F$, $\Gamma, X{\leq}\top^K{:}K \vdash_n A \leq C$, *and* $\Gamma \vdash All(X{\leq}F{:}K)\, A \in \star$.

4. $\Gamma \vdash_n Fun(X{:}K)\, A \leq B$ *implies* $B \equiv Fun(X{:}K)\, C$, *and* $\Gamma, X{\leq}\top^K{:}K \vdash_n A \leq C$.

5. $\Gamma \vdash_n A\, F \leq B$ *implies* $B \equiv A\, F$, *or* $\Gamma \vdash_n LUB(\Gamma, A\, F)^{nf} \leq B$, *and* $\Gamma \vdash A\, F \in K$.

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* $\forall k{\in}\{1..m\}\, \Gamma \vdash A_k \in K$.

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* $\Gamma \vdash A \in K$.

8. $\Gamma \vdash_n \bigwedge^K[A_1..A_m] \leq \bigwedge^K[B_1..B_n]$ *implies for each* $i{\in}\{1..n\}$ *there exists* $j{\in}\{1..m\}$ *such that* $\Gamma \vdash_n A_j \leq B_i$ *and* $\forall k{\in}\{1..m\}\, \Gamma \vdash A_k \in K$.

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 3.4.3(2), such derivation cannot end with an application of NS-TRANS, and, because of lemma 3.4.3(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 3.4.5 *1.* $\Gamma \vdash_n F \leq \bigwedge^K[A_1..A_n]$ *if and only if* $\Gamma \vdash_n F \leq A_k$ *for each* $k \in \{1..n\}$.

2. $\Gamma \vdash_n F \leq \bigwedge^K[A_1..A_n]$ *if and only if* $\Gamma \vdash_n F \leq \bigwedge^K[A_k]$ *for each* $k \in \{1..n\}$.

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

Proof: By induction on derivations using lemma 3.4.2 and generation lemma.

3.5 Equivalence of ordinary and normal subtyping

In this section we will 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 will be proved in theorem 3.5.8. As usual, we need some auxiliary properties and definitions among which we can highlight propositions 3.5.2 and 3.5.7.

Lemma 3.5.1 *Let $LUB(\Gamma, S)$ be defined. Then*

1. $\Gamma \vdash S {\in} K$ *implies* $\Gamma \vdash LUB(\Gamma, S) {\in} K$.
2. $\Gamma \vdash S \leq LUB(\Gamma, S)$.

Proof: 1 follows by induction on derivations, while 2 follows by induction on the structure of S.

Proposition 3.5.2 (Soundness) $\Gamma \vdash_n S \leq T \Rightarrow \Gamma \vdash S \leq T$.

Proof: By induction on the derivation of $\Gamma \vdash_n S \leq T$.

NS-Refl By S-Conv.

NS-TVar By induction hypothesis, S-TVar and S-trans.

NS-OApp By induction hypothesis, lemma 3.5.1(2), S-Conv and S-Trans.

NS-∃ We are given that for each k in $\{1..n\}$ $\Gamma \vdash A_k \in K$, and there is a j in $\{1..n\}$ such that $\Gamma \vdash_n A_j \leq B$. By K-Meet, $\Gamma \vdash \bigwedge^K[A_1..A_n] \in K$, and, by S-Meet-LB $\Gamma \vdash \bigwedge^K[A_1..A_n] \leq A_k$ for each k, in particular for j. Now, by induction hypothesis and S-Trans the result follows.

NS-∀ We are given that $\Gamma \vdash A \in K$, and for each i in $\{1..m\}$ $\Gamma \vdash_n A \leq B_i$. Now, by induction hypothesis and S-Meet-G the result follows.

NS-∀∃ We are given that for each k in $\{1..n\}$ $\Gamma \vdash A_k \in K$, 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, $\Gamma \vdash \bigwedge^K[A_1..A_n] \in K$, and, by S-Meet-LB, $\Gamma \vdash \bigwedge^K[A_1..A_n] \leq A_k$ for each k. Now, by induction hypothesis, S-Trans and S-Meet-G the result follows.

Other cases By induction hypothesis and the corresponding rule in the other system.

Lemma 3.5.3 *Let* $\Gamma \vdash ok$. *Then*

1. $\Gamma \vdash T \in K$ *implies* $\Gamma \vdash_n T \leq \top^K$.
2. $\Gamma \vdash T \in K$ *implies* $\Gamma \vdash T \leq \top^K$.
3. $\Gamma^{nf} \vdash ok$.
4. $\Gamma^{nf} \vdash T \in K$ *if and only if* $\Gamma \vdash T \in K$.

5. *$\Gamma^{nf} \vdash S \leq T$ if and only if $\Gamma \vdash S \leq T$.*

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

Proof: 1 by the cases $m = 0$ in NS-$\forall$ and NS-$\forall\exists$. 2 is the case $n = 0$ in S-Meet-G. 3 and 4(if) by simultaneous induction on derivations. 4(only if) by induction on derivations. 5 by induction on derivations using 4. 6 by 5 and lemma 2.3.15.

Lemma 3.5.4 *Let Γ be a context in normal form such that $\Gamma \vdash ok$ and $Y \in dom(\Gamma)$. Then $\Gamma \vdash_n Y \leq \Gamma(Y)$.*

Proof: Let $\Gamma \equiv \Gamma_1, Y \leq T{:}K, \Gamma_2$. By lemma 2.3.2, $\Gamma_1 \vdash T \in K$. If T is not an intersection, then, by NS-Refl and NS-TVar, we have $\Gamma \vdash_n Y \leq T$. If $T \equiv \bigwedge^{K'}[B_1..B_m]$. By generation lemma for kinding and unicity of kinds, $\Gamma \vdash B_i \in K$ 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-$\forall$, $\Gamma \vdash_n Y \leq T$.

Lemma 3.5.5 (Substitution lemma) *Let $\Gamma \vdash U \in K$. Then*
$\Gamma, X{:}K, \Gamma' \vdash_n S \leq T$ implies $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n (S[X \leftarrow U])^{nf} \leq (T[X \leftarrow U])^{nf}$.

Proof: By induction on the derivation of $\Gamma, X{:}K, \Gamma' \vdash_n S \leq T$. For the sake of clarity, sometimes we leave out kinding judgements and their justifications which follow easily from the structural properties in section 2.3. Let $\Gamma'' \equiv \Gamma, X{:}K, \Gamma'$.

NS-Refl: By the type substitution lemma 2.3.12, lemma 3.5.3(4), subject reduction for kinds 2.3.13, and NS-Refl.

NS-Trans: By induction hypothesis and NS-Trans.

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, $\Gamma \vdash U^{nf} \in K$, by 3.5.3(1), $\Gamma \vdash_n U^{nf} \leq \top^K$. By weakening, it follows that $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n U^{nf} \leq \top^K$ and, by induction hypothesis, that $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n \top^K \leq (A[X \leftarrow U])^{nf}$. Finally, by NS-Trans, the result follows.
2. $Y \in \mathrm{dom}(\Gamma)$. By the free variables lemma, $X \notin \mathrm{FV}(\Gamma(Y))$ and $X \not\equiv Y$. By lemmas 2.3.12, 3.5.3(3), and 3.5.4, $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n Y \leq \Gamma(Y)$, and, by induction hypothesis, it follows that $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n \Gamma(Y) \leq (A[X \leftarrow U])^{nf}$. Finally, by NS-Trans, the result follows.
3. $Y \in \mathrm{dom}(\Gamma')$. By induction hypothesis, it follows that$\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n (\Gamma'(Y)[X \leftarrow U])^{nf} \leq (A[X \leftarrow U])^{nf}$. By lemmas 2.3.12, 3.5.3(3), and 3.5.4, $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n Y \leq (\Gamma, (\Gamma'[X \leftarrow U])^{nf})(Y)$. On the other hand, $(\Gamma, (\Gamma'[X \leftarrow U])^{nf})(Y) = (\Gamma'(Y)[X \leftarrow U])^{nf}$. Hence, by NS-Trans, the result follows.

NS-ARROW We are given that $\Gamma'' \vdash_n F \leq G$ and $\Gamma'' \vdash_n C \leq B$. By induction hypothesis, it follows that $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n (F[X \leftarrow U])^{nf} \leq (G[X \leftarrow U])^{nf}$ and $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n (C[X \leftarrow U])^{nf} \leq (B[X \leftarrow U])^{nf}$. There are four cases to consider, because $(B[X \leftarrow U])^{nf}$ and $(C[X \leftarrow U])^{nf}$ may be intersections or not. We shall consider only two of them to illustrate the proof method.

1. $(B[X \leftarrow U])^{nf}$ and $(C[X \leftarrow U])^{nf}$ are not intersections. Then the result follows applying NS-ARROW.
2. $(C[X \leftarrow U])^{nf} = \bigwedge^*[C_1..C_n]$ and $(B[X \leftarrow U])^{nf}$ is not an intersection. Then we have that $((F \to B)[X \leftarrow U])^{nf} = (F[X \leftarrow U])^{nf} \to (B[X \leftarrow U])^{nf}$ and also that $((G \to C)[X \leftarrow U])^{nf} = \bigwedge^*[(G[X \leftarrow U])^{nf} \to C_1..(G[X \leftarrow U])^{nf} \to C_n]$. By 3.4.4, $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n C_i \leq (B[X \leftarrow U])^{nf}$for some i. By NS-ARROW, $\Gamma, (\Gamma'[X \leftarrow U])^{nf} \vdash_n (G[X \leftarrow U])^{nf} \to C_i \leq (F[X \leftarrow U])^{nf} \to (B[X \leftarrow U])^{nf}$. Finally, by NS-$\exists$, the result follows.

Other cases NS-OAPP is similar to the case for NS-TVAR using lemma 2.2.4(3) and uniqueness of normal forms. All other cases are similar to that of NS-ARROW.

Lemma 3.5.6 $\Gamma \vdash S\,F \in K$. *Then* $\Gamma \vdash_n S \leq T$ *implies* $\Gamma \vdash_n (S\,F)^{nf} \leq (T\,F)^{nf}$.

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 $\Gamma \vdash S\,F \in K$.

NS-REFL By subject reduction for kinds and NS-REFL.

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

$(AU)^{nf} \equiv B$	By NS-OAPP.
$(AU)^{nf} \equiv \bigwedge^K[A_1..A_n]$	By lemma 3.4.5, it follows that $\Gamma \vdash_n (\Gamma(X)U)^{nf} \leq A_k$ 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 $\Gamma \vdash_n X\,U \leq (AU)^{nf}$.

NS-OABS We are given $\Gamma, X{:}K \vdash_n S_1 \leq T_1$. By the substitution lemma 3.5.5, it follows that $\Gamma \vdash_n (S_1[X \leftarrow F])^{nf} \leq (T_1[X \leftarrow F])^{nf}$. On the other hand, we have that $(\mathrm{Fun}\,(X{:}K)\,S_1)F \to_{\beta\wedge} S_1[X \leftarrow F]$ and $(\mathrm{Fun}\,(X{:}K)\,T_1)F \to_{\beta\wedge} T_1[X \leftarrow F]$. Finally, by unicity of normal forms, the result follows.

NS-OAPP Similar to case NS-TVAR.

NS-$\forall\exists$ By induction hypothesis and NS-$\forall\exists$ using the generation lemma for subtyping.

NS-$\exists$ and NS-$\forall$ By induction hypothesis using lemma 3.4.5.

Proposition 3.5.7 (Completeness) $\Gamma \vdash S \leq T \Rightarrow \Gamma^{nf} \vdash_n S^{nf} \leq T^{nf}$.

Proof: By induction on the derivation of $\Gamma \vdash S \leq T$ using lemma 3.5.6 for the case of S-OApp.

Theorem 3.5.8 (Equivalence of ordinary and normal subtyping) *Let* $\Gamma \vdash S{\in}K$ *and* $\Gamma \vdash T{\in}K$. *Then* $\Gamma \vdash S \leq T \iff \Gamma^{nf} \vdash_n S^{nf} \leq T^{nf}$.

Proof:

$\Rightarrow$) By completeness (3.5.7).

$\Leftarrow$) By soundness (3.5.2), it follows that $\Gamma^{nf} \vdash S^{nf} \leq T^{nf}$, and, by lemma 3.5.3(6), it follows that $\Gamma \vdash S \leq T$.

3.5.1 Least strict upper bound. Why least?

So far we only used that $\mathrm{LUB}(\Gamma, S)$ is an upper bound of S in the context Γ (See lemma 3.5.1(2)). Now, we are in conditions to give the final motivation of the name we chose, showing that if $\mathrm{LUB}(\Gamma, S)$ is defined and $T \neq_{\beta\wedge} S$, then $\Gamma \vdash S \leq T$ implies $\Gamma \vdash \mathrm{LUB}(\Gamma, S) \leq T$. To prove the latter we prove first the following lemma which establishes the corresponding property for the normalized system.

Lemma 3.5.9 *Let* $LUB(\Gamma, S)$ *be defined. Then*

1. $S \twoheadrightarrow_{\beta\wedge} S'$ *and* $\Gamma \twoheadrightarrow_{\beta\wedge} \Gamma'$ *implies* $LUB(\Gamma, S) \twoheadrightarrow_{\beta\wedge} LUB(\Gamma', S')$.
2. $\Gamma \vdash_n S \leq T$ *implies* $\Gamma \vdash_n LUB(\Gamma, S)^{nf} \leq T$ *or* $S \equiv T$.

Proof:

1. By induction on the structure of S observing that if $\mathrm{LUB}(\Gamma, S)$ is defined, so is $\mathrm{LUB}(\Gamma', S')$.
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 $\mathrm{LUB}(\Gamma, S)$ is not defined; for the other rules the result follows using the induction hypothesis.

Corollary 3.5.10 *Let* $LUB(\Gamma, S)$ *be defined. Then* $\Gamma \vdash S \leq T$ *and* $T \neq_{\beta\wedge} S$ *implies* $\Gamma \vdash LUB(\Gamma, S) \leq T$.

Proof: By completeness, $\Gamma^{nf} \vdash_n S^{nf} \leq T^{nf}$, then, by lemma 3.5.9(2), $\Gamma^{nf} \vdash_n \mathrm{LUB}(\Gamma^{nf}, S^{nf})^{nf} \leq T^{nf}$, because $S^{nf} \not\equiv T^{nf}$. By soundness, it follows that $\Gamma^{nf} \vdash \mathrm{LUB}(\Gamma^{nf}, S^{nf})^{nf} \leq T^{nf}$, which, by lemma 3.5.3(6), is equivalent to $\Gamma \vdash \mathrm{LUB}(\Gamma^{nf}, S^{nf}) \leq T$. Finally, (using lemmas 3.5.1(1 and 2.3.13, and proposition 2.3.17 to get the corresponding kinding judgements) by lemma 3.5.9(1), S-Conv and S-Trans, it follows that $\Gamma \vdash \mathrm{LUB}(\Gamma, S) \leq T$.

3.6 Example

In this section we give the derivations in $F^{\omega}_{\wedge}$ and in $NF^{\omega}_{\wedge}$ of the example(1) mentioned in the introduction and in section 3.

Let $\Gamma \equiv X \leq \text{Fun}\,(Y{:}K)\,Y{:}K{\rightarrow}K,\ Z \leq X{:}K{\rightarrow}K$. We present a proof of $\Gamma \vdash X(Z\,W) \leq W$ and a proof of its translation in the normal system $\Gamma^{nf} \vdash_n X(Z\,W)^{nf} \leq W^{nf}$.

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.

A derivation in $F^{\omega}_{\wedge}$.

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

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

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

And now, a derivation in $NF^{\omega}_{\wedge}$, which is substantially shorter. [1]

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash W \in K}{\Gamma \vdash_n ((\text{Fun}\,(Y{:}K)\,Y)W)^{nf} \leq W}\,\text{NS-REFL}}{\Gamma \vdash_n X\,W \leq W}\,\text{NS-OAPP}}{\Gamma \vdash_n ((\text{Fun}\,(Y{:}K)\,Y)(Z\,W))^{nf} \leq W}\,\text{NS-OAPP}}{\Gamma \vdash_n X(Z\,W) \leq W}\,\text{NS-OAPP}$$

4 Subtype Checking

In this section we present a terminating algorithm that is sound and complete with respect to $NF^{\omega}_{\wedge}$. In section 3.5 we proved the equivalence of $F^{\omega}_{\wedge}$ and $NF^{\omega}_{\wedge}$ (theorem 3.5.8) which means that the algorithm *check* of subsection 4.1 provides a strategy for the subtyping relation in $F^{\omega}_{\wedge}$. Hence, the termination of the algorithm *check* implies the decidability of subtyping in $F^{\omega}_{\wedge}$.

[1] It is worth noting that this is the derivation found by the algorithm *check* of section 4.

$$
\begin{aligned}
check(\Gamma \vdash_n S \leq T) = {} & \text{if } S \equiv T \\
& \quad \text{then } true \\
& \text{elseif } S \equiv X \text{ and } T \equiv A \\
& \quad \text{then } check(\Gamma \vdash_n \Gamma(X) \leq A) \\
& \text{elseif } S \equiv F{\to}A \text{ and } T \equiv G{\to}B \\
& \quad \text{then } check(\Gamma \vdash_n G \leq F) \text{ and } check(\Gamma \vdash_n A \leq B) \\
& \text{elseif } S \equiv \mathrm{All}\,(X{\leq}F{:}K_1)\,A \text{ and } T \equiv \mathrm{All}\,(X{\leq}G{:}K_1)\,B \\
& \quad \text{then } check(\Gamma \vdash_n G \leq F) \text{ and } check(\Gamma,\, X{:}K_1 \vdash_n A \leq B) \\
& \text{elseif } S \equiv \mathrm{Fun}\,(X{:}K_1)\,A \text{ and } T \equiv \mathrm{Fun}\,(X{:}K_1)\,B \\
& \quad \text{then } check(\Gamma,\, X{:}K_1 \vdash_n A \leq B) \\
& \text{elseif } S \equiv A\,F \text{ and } T \equiv B \\
& \quad \text{then } check(\Gamma \vdash_n \mathrm{LUB}(\Gamma, A\,F)^{nf} \leq B) \\
& \text{elseif } S \equiv \textstyle\bigwedge^K[A_1..A_m] \text{ and } T \equiv \textstyle\bigwedge^K[B_1..B_n] \\
& \quad \text{then } \forall i{\in}\{1..n\}\ \exists j{\in}\{1..m\}\ \ [check(\Gamma \vdash_n A_j \leq B_i)] \\
& \text{elseif } S \equiv \textstyle\bigwedge^K[A_1..A_m] \text{ and } T \equiv B \\
& \quad \text{then } \exists j{\in}\{1..m\}\ \ [check(\Gamma \vdash_n A_j \leq B)] \\
& \text{elseif } S \equiv A \text{ and } T \equiv \textstyle\bigwedge^K[B_1..B_n] \\
& \quad \text{then } \forall i{\in}\{1..n\}\ \ [check(\Gamma \vdash_n A \leq B_i)] \\
& \text{else } false
\end{aligned}
$$

4.1 Termination of subtype-checking

There is a delicate point to analize, namely whether $\mathrm{LUB}(\Gamma, A\,F)^{nf}$ is defined. Part of the problem is solved in section 2.4 where we show that $\to_{\beta\wedge}$ is strongly normalizing. On the other hand, proposition 2.3.17 says that it is enough to consider well kinded types. Then we will assume that $\Gamma \vdash A\,F{\in}K$, in which case $\mathrm{LUB}(\Gamma, A\,F)$ is well defined. Hence, this problem can be solved by defining the function $\underline{check}$ as follows:

$$
\begin{aligned}
\underline{check}(\Gamma \vdash_n S \leq T\ ) = {} & \text{if } kind(\Gamma, S) = kind(\Gamma, T) \neq NoKind \\
& \text{then } check(\Gamma \vdash_n S \leq T) \\
& \text{else } false,
\end{aligned}
$$

where the function *kind* can be defined straightforwardly using proposition 2.3.6, such that $kind(\Gamma, S) = K$ if $\Gamma \vdash S{\in}K$, and gives a constant *NoKind* otherwise. Moreover, lemma 2.3.10 implies that the function *kind* is total. In this case $\mathrm{LUB}(\Gamma, A\,F)$ is always defined. The next step is to prove that given $\Gamma \vdash S, T \in K$, then $check(\Gamma \vdash_n S \leq T)$ terminates.

The algorithm *check* with input $\Gamma \vdash_n S \leq T$ visits in parallel all the subexpressions of S and T in each level of their underlying trees. *check* eventually stops or replaces a variable by its bound or an application by the normal form of its least strict upper bound (LUB from now on). We can trace the transformations of the original types in the successive steps. For example, while checking $W{:}K, Z \leq W{:}K, Y \leq \mathrm{Fun}\,(X{:}K)\,X{:}K{\to}K \vdash_n W{\to}Y\,Z \leq Z{\to}W$, we can construct the following traces.

$$W{\to}Y\,Z \Rightarrow W{\to}Y\,Z \Rightarrow W{\to}Z \Rightarrow W{\to}Z \Rightarrow W{\to}W \ \text{ and}$$

$$Z{\to}W \Rightarrow Z{\to}W \Rightarrow W{\to}W \Rightarrow W{\to}W \Rightarrow W{\to}W.$$

If there are finitely many replacements of a variable by its bound or of an application by its LUB, the algorithm terminates because the size of the visited subexpressions decreases in each recursive call. Hence, if the algorithm constructs infinite traces it means that there are infinitely many replacements of a variable by its bound or of an application by the normal form of its LUB. To continue with the argument to show termination we define the following function *plus* that recursively replaces every variable of a given type by itself *and* its bound with respect to a given context. For that we need to extend our language of types so that if $S \in \mathbb{T}$ and $T \in \mathbb{T}$, then $S + T \in \mathbb{T}$. Consequently, we need the following kinding rule.

$$\frac{\Gamma \vdash_+ S \in K \qquad \Gamma \vdash_+ T \in K}{\Gamma \vdash_+ S + T \in K} \tag{K-Plus}$$

1. $plus(X; \Gamma_1, X \leq F{:}K, \Gamma_2) = X + plus(F; \Gamma_1)$,
2. $plus(F \to A; \Gamma) = plus(F; \Gamma) \to plus(A; \Gamma)$,
3. $plus(\mathrm{All}\,(X \leq F{:}K)\,A; \Gamma) = \mathrm{All}\,(X \leq plus(F; \Gamma){:}K) plus(A; \Gamma, X \leq \top^K{:}K)$,
4. $plus(\mathrm{Fun}\,(X{:}K)\,A; \Gamma) = \mathrm{Fun}\,(X{:}K)\,plus(A; \Gamma, X{:}K)$,
5. $plus(A\,F; \Gamma) = plus(A; \Gamma)\,plus(F; \Gamma)$,
6. $plus(\bigwedge^K[A_1..A_n]; \Gamma) = \bigwedge^K[plus(A_1; \Gamma)..plus(A_n; \Gamma)]$.

Each variable in a type expression may become its bound, but this is not necessarily the case. What the mapping *plus* does is to include in the type this nondeterministic behavior of its variables, in such a way that the transformations that *check* performs in its input types are latent in their *plus* versions.

If $\Gamma \vdash S \in K$, then $plus(S; \Gamma)$ is well defined. A way to prove that *plus* is well-founded is considering $weight(plus(S; \Gamma)) = size_j(\Gamma \vdash S \in kind(\Gamma, S))$ where $size_j(\Gamma \vdash S \in K)$ is the size of the derivation of the kinding judgement (see definition 2.3.8). Given that $\Gamma \vdash S \in K$, by lemmas 2.3.2(1) and 2.3.6, the weight decreases in each recursive call and the least value is that of $size(\vdash \top^K \in K)$.

Now, we define a new reduction $\to_{\beta\wedge+}$ which is obtained from $\to_{\beta\wedge}$ by adding: $S + T \to S$ and $S + T \to T$.
$\to_{\beta\wedge+}$ is proved to be strongly normalizing in the same way $\to_{\beta\wedge}$ is proved to be strongly normalizing on well kinded types. We only need to add to the definition of saturated sets the following closure condition. Given $T, U, R_1..R_n \in SN$. Then $T R_1..R_n \in S$ and $U R_1..R_n \in S$ imply $(T + U) R_1..R_n \in S$.

Lemma 4.1.1 *1. Let* $\Gamma \vdash S \in K$, $\Gamma' \vdash ok$ *and* $\Gamma \subseteq \Gamma'$. *Then* $\mathrm{plus}(S; \Gamma) = \mathrm{plus}(S; \Gamma')$.

2. Let $\Gamma \vdash S_1 \in K_1$ *and* $\Gamma \vdash S_2 \in K_2$. *Then*

$$\mathrm{plus}(S_1; \Gamma)[X \leftarrow \mathrm{plus}(S_2; \Gamma)] \twoheadrightarrow_{\beta\wedge+} \mathrm{plus}(S_1[X \leftarrow S_2]; \Gamma).$$

3. Let $\Gamma \vdash S \in K$. *Then* $S \twoheadrightarrow_{\beta\wedge} S_1$ *implies* $\mathrm{plus}(S;\Gamma) \twoheadrightarrow_{\beta\wedge+} \mathrm{plus}(S_1;\Gamma)$.

Proof:

1. The assumptions ensure that *plus* is defined, then we can proceed by induction on the unfolding steps.

2. By induction on the structure of S_1 using 1.

3. By induction on the definition of $\twoheadrightarrow_{\beta\wedge}$, using 2 when S is a β-redex and S_1 its reduct.

Lemma 4.1.2 *Let* $\Gamma \vdash T \in K$, *and* F *and* G *are consecutive types in a trace starting from* T. *Then*

1. $\Gamma \vdash_+ \mathrm{plus}(T;\Gamma) \in K$.

2. $\mathrm{plus}(F;\Gamma) \twoheadrightarrow_{\beta\wedge+} \mathrm{plus}(G;\Gamma)$. *Moreover, if* G *is obtained from* F *by replacing a variable by its bound or an application subexpression by the normal form of its LUB, then the reduction has at least one step.*

Proof:

1. By induction on the derivation of $\Gamma \vdash T \in K$, observing that $\Gamma \vdash T \in K$ implies $\Gamma \vdash_+ T \in K$. It is straightforward to verify that $\vdash_+$ satisfies weakening (see lemma 2.3.4). We consider here the case for K-TVAR, the rest follows by straightforward induction. We are given, $\Gamma_1, X \leq T{:}K, \Gamma_2 \vdash \mathrm{ok}$. By lemma 2.3.2, there is a proper subderivation of $\Gamma_1 \vdash T \in K$. Now by induction hypothesis, weakening and K-PLUS the result follows.

2. There are three possibilities for F and G. Either $F \equiv G$ or G is obtained form F by replacing a variable by its bound (note that the bound can be in the original Γ or inside T), or G is obtained form F by replacing an application by the normal form of its least strict upper bound. In the first case the result follows immediately. For the other two cases the result follows by induction on the structure of F. It is enough to prove that:

   (a) $plus(X;\Gamma) \twoheadrightarrow_{\beta\wedge+}{}^{>0}\ plus(\Gamma(X);\Gamma)$.
   (b) $plus(A\,F;\Gamma) \twoheadrightarrow_{\beta\wedge+}{}^{>0}\ plus(\mathrm{LUB}(A\,F)^{nf};\Gamma)$.

   2a follows by one step of $\rightarrow_{\beta\wedge+}$ and lemma 4.1.1(1). 2b follows by induction on the structure of A.

   $A \equiv X$ Let $\Gamma \equiv \Gamma_1, X \leq T{:}K, \Gamma_2$.

$$
\begin{array}{rcll}
plus(X\,F;\Gamma) & = & (X + plus(T;\Gamma_1))plus(F;\Gamma) & \\
 & = & (X + plus(T;\Gamma))plus(F;\Gamma) & \text{by lemma 4.1.1(1)} \\
 & \rightarrow_{\beta\wedge+} & plus(T;\Gamma)plus(F;\Gamma) & \\
 & = & plus(T\,F;\Gamma) & \\
 & = & plus((T\,F)^{nf};\Gamma) & \text{by lemma 4.1.1(3).} \\
 & = & plus(\mathrm{LUB}(A\,F)^{nf};\Gamma). &
\end{array}
$$

$A \equiv A_1\, F_1$ By induction hypothesis using lemma 4.1.1(3), and unicity of $\beta\wedge$-normal forms.

A corollary of the last lemma is that an infinite trace starting from a well kinded type would yield an infinite $\beta\wedge+$-reduction starting from a well kinded type, which is a contradiction, because $\rightarrow_{\beta\wedge+}$ is strongly normalizing. Hence we have proved the termination of the algorithm *check*.

Proposition 4.1.3 *For any context* Γ *and any two type expressions* S *and* T *in normal form,* $\underline{\text{check}}(\Gamma \vdash_n S \leq T)$ *terminates.*

4.2 Soundness and completeness of *check*

Lemma 4.2.1 (Soundness and completeness) $\underline{\text{check}}(\Gamma \vdash_n S \leq T) = \text{true} \iff \Gamma \vdash_n S \leq T$.

Proof:

- ⇒ $\underline{\mathit{check}}(\Gamma \vdash_n S \leq T) = \mathit{true}$ when $\mathit{kind}(\Gamma, S) = \mathit{kind}(\Gamma, T) = K$, for some K, and $\mathit{check}(\Gamma \vdash_n S \leq T) = \mathit{true}$. Given that *check* terminates we can reason by induction on the number of recursive calls. We analize one case here. The rest follows by similar reasoning. Let $S \equiv \bigwedge^K[A_1..A_m]$, $T \equiv \bigwedge^K[B_1..B_n]$, and for each $i \in \{1..n\}$ there exists $j \in \{1..m\}$ such that $\mathit{check}(\Gamma \vdash_n A_j \leq B_i)$. By induction hypothesis, it follows that $\Gamma \vdash A_j \leq B_i$ and, finally, by NS-∀∃, we have that $\Gamma \vdash_n \bigwedge^K[A_1..A_m] \leq \bigwedge^K[B_1..B_n]$.

- ⇐ Using proposition 2.3.17 and lemma 2.3.7 and the fact that $\Gamma \vdash_n S \in \mathit{kind}(\Gamma, S)$, we reduce our problem to show that $\mathit{check}(\Gamma \vdash_n S \leq T) = \mathit{true}$. We proceed by induction on the derivation of $\Gamma \vdash_n S \leq T$, using the fact that there is also a derivation in normal form.

  Finally, we have proved our main result.

Theorem 4.2.2 (Decidability of subtyping in $F^\omega_\wedge$) *For any context* Γ *and for any two types* S *and* T *it is decidable whether* $\Gamma \vdash S \leq T$.

5 A Related System

An alternative system of higher-order subtyping with intersection types can be obtained from $F^\omega_\wedge$ by replacing the quantifier rule S-All by the ω-version of the quantifier rule of Cardelli and Wegner's original *Fun* calculus [CW85].

$$\frac{\Gamma,\, X{\leq}U{:}K \vdash S_2 \leq T_2 \qquad \Gamma \vdash \mathrm{All}\,(X{\leq}U{:}K)\, S_2 \in \star}{\Gamma \vdash \mathrm{All}\,(X{\leq}U{:}K)\, S_2 \leq \mathrm{All}\,(X{\leq}U{:}K)\, T_2} \qquad \text{(S-All-Fun)}$$

The differences from S-All are that the bounds are equal in the conclusion, which immediately eliminates the need for one of its premises, and that the remaining premise has the binding $X{\leq}U{:}K$ instead of $X{\leq}\top^K{:}K$. (C.f. [CP94] for a discussion of the pragmatic differences between these rules.)

For the sake of consistency, we also need to modify the kinding rule K-ALL.

$$\frac{\Gamma,\, X{\leq}U{:}K \vdash T \in \star}{\Gamma \vdash \mathrm{All}\,(X{\leq}U{:}K)\,T \in \star} \qquad \text{(K-ALL-FUN)}$$

Subtyping in this "*Fun*-variant" is decidable. Moreover, all the results proved in sections 2, 3, and 4 also hold for the *Fun*-variant after some straightforward modifications to accommodate the new rules. For the reader interested in the details here are the most significant changes:

1. To prove lemma 2.3.13, subject reduction for kinding judgements, we prove by simultaneous induction on the derivation of the antecedent that

   (a) $\Gamma \vdash S \in K$ and $S \rightarrow_{\beta\wedge} T$ implies $\Gamma \vdash T \in K$,

   (b) $\Gamma \vdash S \in K$ and $\Gamma \rightarrow_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash S \in K$, and

   (c) $\Gamma \vdash \mathrm{ok}$ and $\Gamma \rightarrow_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash \mathrm{ok}$.

2. In lemma 3.5.3, the *left-to-right* cases of items 4 and 5 use the context modification lemma 2.3.16. On the other hand, the *right-to-left* cases must be proved in a more general form.

   Lemma Let $\Gamma_1^{nf},\, \Gamma_2 \vdash \mathrm{ok}$. Then

   (a) $\Gamma_1^{nf},\, \Gamma_2 \vdash T \in K$ implies $\Gamma_1,\, \Gamma_2 \vdash T \in K$, and

   (b) $\Gamma_1^{nf},\, \Gamma_2 \vdash S \leq T$ implies $\Gamma_1,\, \Gamma_2 \vdash S \leq T$.

3. Lemma 4.1.2(1) uses a context modification lemma for the $\Gamma \vdash_{+} S \in K$ kinding relation analogous to lemma 2.3.16.

6 Semantics

The present section is essentially based on the work done in [CP93]. The differences come from having replaced the distributivity subtyping rules by reduction rules.

Among simplest models for typed λ-calculi are those based on partial equivalence relations (PERs). A model in this style is essentially untyped: terms are interpreted by erasing all type information and interpreting the resulting pure λ-term as an element of the model. A type, in this setting, is just a subset of the model along with an appropriate notion of equivalence of elements. Coercions between types are interpreted as inclusion of PERs.

Our PER model for $F^{\omega}_{\wedge}$ extends the model of $F_{\wedge}$ given in [Pie91], which is based on Bruce and Longo's model for $F_{\leq}$ [BL90]. The usual interpretation of a quantified type $\mathrm{All}\,(X)\,T$ in a PER model is the PER-indexed intersection of all possible instances of T. Bruce and Longo refined this definition to interpret $\mathrm{All}\,(X{\leq}S)\,T$ as the intersection of all the instances of T where X is interpreted as a sub-PER of the interpretation of S. This intuition also serves for intersection types: $\bigwedge^{\star}[T_1..T_n]$ is interpreted as the intersection of the PERs interpreting each of the T_i's. We generalize this model to ω-order polymorphism (and subtyping) by interpreting type operators as functions over PERs.

To deal correctly with intersection types, we need to make one significant technical departure here from PER models of ordinary bounded quantification: instead of allowing the elements of our PERs to be drawn from the carrier of an arbitrary partial combinatory algebra $\mathcal{D}$, we require that $\mathcal{D}$ be a *total* combinatory algebra. This restriction is needed to validate instances of S-Conv, which have the form $\Gamma \vdash \top^\star \leq S \rightarrow \top^\star$. For example, let $S = \top^\star$. The empty intersection $\top^\star$ is interpreted by the everywhere-defined PER, i.e., $[\![\top^\star]\!]$ relates every m to itself. To validate the distributivity law, it must therefore be the case that $[\![\top^\star \rightarrow \top^\star]\!]$ relates every element to itself. But this will only be true if the application of any element to any other element is defined. This observation is due to QingMing Ma.

Cardelli and Longo [CL91] and Bruce and Mitchell [BM92] have given related models for variants of F^ω including subtyping, but without intersections.

The notation and fundamental definitions used here are based on papers of Bruce and Longo [BL90], Freyd, Mulry, Rosolini, and Scott [FMRS90], and others. A helpful basic reference for PER models of second-order λ-calculi is [Mit90b]; also see [BMM90] for more general discussion of second-order models and [Bar84, HS86] for general discussion of combinatory models.

6.1 Total combinatory algebras

A *total combinatory algebra* is a tuple $\mathcal{D} = \langle D, \cdot, k, s \rangle$ comprising a set D of *elements*, an *application function* $\cdot$ with type $D \rightarrow (D \rightarrow D)$, and distinguished elements k, $s \in D$ such that, for all d_1, d_2, $d_3 \in D$,

$$
\begin{array}{l}
k \cdot d_1 \cdot d_2 = d_1 \\
s \cdot d_1 \cdot d_2 \cdot d_3 = (d_1 \cdot d_3) \cdot (d_2 \cdot d_3).
\end{array}
$$

Throughout this section, we work with a fixed, but unspecified, total combinatory algebra $\mathcal{D}$. (C.f. [Sco76] for examples.)

The set of *pure λ-terms* is defined by the following grammar:

$$M ::= x \mid \mathrm{fun}\,(x)\,M \mid M_1\; M_2$$

The set of *combinator terms* is:

$$C ::= x \mid C_1\; C_2 \mid K \mid S$$

The *bracket abstraction* of a combinator term C with respect to a variable x, written $\mathrm{fun}^\star\,(x)\,C$, is defined as follows:

$$
\begin{array}{lcll}
\mathrm{fun}^\star\,(x)\,C & = & K\,C & \text{when } x \notin \mathrm{FV}(C) \\
\mathrm{fun}^\star\,(x)\,x & = & S\,K\,K & \\
\mathrm{fun}^\star\,(x)\,C_1\,C_2 & = & S\,(\mathrm{fun}^\star\,(x)\,C_1)\,(\mathrm{fun}^\star\,(x)\,C_2) & \text{when } x \in \mathrm{FV}(C_1\,C_2)
\end{array}
$$

The *combinator translation* of a pure λ-term M, written $|M|$, is defined as follows:

$$
\begin{array}{lcl}
|x| & = & x \\
|\mathrm{fun}\,(x)\,M| & = & \mathrm{fun}^\star\,(x)\,|M| \\
|M_1\,M_2| & = & |M_1|\;|M_2|
\end{array}
$$

A *term environment* η is a finite function from term variables to elements of D. When $x \notin \mathrm{dom}(\eta)$, we write $\eta[x{\leftarrow}d]$ for the environment that maps x to d and agrees with η everywhere else. We write $\eta\backslash x$ for the environment like η except that $\eta(x)$ is undefined; $\eta\backslash\Gamma$ is like η but undefined on all the variables in $\mathrm{dom}(\Gamma)$. We say that η' *extends* η when $\mathrm{dom}(\eta) \subseteq \mathrm{dom}(\eta')$ and η and η' agree on $\mathrm{dom}(\eta)$.

Let C be a combinator term and η a term environment such that $\mathrm{FV}(C) \subseteq \mathrm{dom}(\eta)$. Then the *interpretation* of C under η, written $[\![C]\!]_\eta$, is defined as follows:

$$
\begin{array}{lcl}
[\![x]\!]_\eta & = & \eta(x) \\
[\![C_1\,C_2]\!]_\eta & = & [\![C_1]\!]_\eta \cdot [\![C_2]\!]_\eta \\
[\![K]\!]_\eta & = & k \\
[\![S]\!]_\eta & = & s.
\end{array}
$$

Lemma 6.1.1

1. *If η' extends η and $FV(C) \subseteq dom(\eta)$, then $[\![C]\!]_\eta = [\![C]\!]_{\eta'}$.*
2. *$[\![fun^\star\,(x)\,C]\!]_\eta \cdot m = [\![C]\!]_{\eta[x\leftarrow m]}$.*

Proof: Standard.

6.2 Higher-order partial equivalence relations

A *partial equivalence relation* (PER) on a combinatory algebra $\mathcal{D}$ is a symmetric and transitive relation A on D. We write $m\ \{A\}\ n$ when A relates m and n. The *domain* of A, written $\mathrm{dom}(A)$, is the set $\{n \mid n\ \{A\}\ n\}$. (Note that $m\ \{A\}\ n$ implies $m \in \mathrm{dom}(A)$.)

We write **PER** for the class of all PERs.

If A and B are relations, then $A \to B$ is the relation where $m\ \{A \to B\}\ n$ iff, for all $p, q \in D$, $p\ \{A\}\ q$ $m{\cdot}p\ \{B\}\ n{\cdot}q$. It is not hard to show that $A \to B$ is a PER when A and B are PERs, and that the intersection of any set of PERs is a PER. To interpret type operators, we need to consider not only PERs, but arbitrary function spaces built on PER. An element of such a function space (including, as a special case, an element of PER itself), is called a *higher-order PER* (HOPER). The interpretation of a kind K is a suitable space of HOPERs: $[\![\star]\!] = \mathbf{PER}$ and $[\![K_1{\to}K_2]\!] = [\![K_1]\!] \to [\![K_2]\!]$.

We generalize the familiar graph-inclusion of relations to HOPERs as follows:

$$
\begin{array}{lll}
A \subseteq^\star B & \text{iff} & A, B \in [\![\star]\!] \text{ and } m\ \{A\}\ n \text{ implies } m\ \{B\}\ n \text{ for all } m, n \in D; \\
A \subseteq^{K_1\to K_2} B & \text{iff} & A, B \in [\![K_1{\to}K_2]\!] \text{ and } A\,P \subseteq^{K_2} B\,P \text{ for all } P \in [\![K_1]\!].
\end{array}
$$

Let $\{A_i \in [\![K]\!]\}_{i\in I}$ be a set of HOPERs indexed by a set I. Then $\bigcap^K_{i\in I} A_i$ is the HOPER defined by

$$
\begin{array}{lll}
m\ \{\bigcap^\star_{i\in I} A_i\}\ n & \text{iff} & \text{for every } i,\ m\ \{A_i\}\ n \\
\bigcap^{K_1\to K_2}_{i\in I} A_i & = & \lambda P{\in}[\![K_1]\!].\ \bigcap^{K_2}_{i\in I} A_i\,P
\end{array}
$$

Lemma 6.2.1

1. *Each* $\subseteq^K$ *is transitive.*
2. *If* $A_j \in [\![K]\!]$ *for each* $j{\in}I$*, then* $\bigcap^K_{i\in I} A_i \in [\![K]\!]$.
3. *If* $A \subseteq^K B_j$ *for each* j*, then* $A \subseteq^K \bigcap^K_{i\in I} B_i$.
4. $\bigcap^K_{i\in I} A_i \subseteq^K A_j$ *for each* j.
5. $\bigcap^\star_{i\in I} A{\rightarrow}B_i \subseteq^\star A \rightarrow \bigcap^\star_{i\in I} B_i$.
6. $\bigcap^\star_{i\in I}\bigcap^\star_{P\subseteq^K A} B_i \subseteq^\star \bigcap^\star_{P\subseteq^K A}\bigcap^\star_{i\in I} B_i$.
7. $\bigcap^{K_1\rightarrow K_2}_{i\in I} \lambda P{\in}[\![K_1]\!].\, B_i\, P \subseteq^{K_1\rightarrow K_2} \lambda P{\in}[\![K_1]\!].\bigcap^{K_2}_{i\in I} B_i\, P$, *when each* $B_i \in [\![K_1{\rightarrow}K_2]\!]$.
8. $\bigcap^{K_2}_{i\in I} A\; B_i \subseteq^{K_2} A\;(\bigcap^{K_1\rightarrow K_2}_{i\in I} B_i)$.

Indeed, in cases 4 through 8 the inclusions are equalities.

Proof: Straightforward.

Each collection $[\![K]\!]$ of HOPERs has a maximal element under the ordering $\subseteq^K$. This element is written $\top^K$ and can be calculated as follows: $\top^\star$ is the total relation on $\mathcal{D}$ and $\top^{K_1\rightarrow K_2} = \lambda P{\in}[\![K_1]\!].\, \top^{K_2}$.

Fact 6.2.2 *Let* $A \in [\![K]\!]$*. Then:*

1. $A \subseteq^K \top^K$.
2. $\top^K \subseteq^K A$ *implies* $A = \top^K$.

6.3 HOPER interpretation of $F^\omega_\wedge$

An *environment* η is a finite function from type variables to HOPERs and from term variables to elements of D. The notations for environment extension, restriction, and agreement are carried over from term environments. By an abuse of notation, type environments are used in place of term environments from now on.

The *erasure* of an $F^\omega_\wedge$ term e, written erase(e), is the pure λ-term defined as follows:

$$
\begin{array}{lcl}
\mathrm{erase}(x) & = & x \\
\mathrm{erase}(\mathrm{fun}\,(x{:}T)\,e) & = & \mathrm{fun}\,(x)\,\mathrm{erase}(e) \\
\mathrm{erase}(e_1\; e_2) & = & \mathrm{erase}(e_1)\;\mathrm{erase}(e_2) \\
\mathrm{erase}(\mathrm{fun}\,(X{\leq}T{:}K)\,e) & = & \mathrm{erase}(e) \\
\mathrm{erase}(e\; T) & = & \mathrm{erase}(e) \\
\mathrm{erase}(\mathrm{for}\,(X{\in}T_1..T_n)\,e) & = & \mathrm{erase}(e).
\end{array}
$$

Let η be a term environment and e an expression such that $\mathrm{FV}(e) \subseteq \mathrm{dom}(\eta)$. Then the *interpretation* of e under η, written $[\![e]\!]_\eta$, is $[\![|\mathrm{erase}(e)|]\!]_\eta$.

If η is an environment and T a type expression such that $\mathrm{FTV}(T) \subseteq \mathrm{dom}(\eta)$, then the *interpretation* of T under η, written $[\![T]\!]_\eta$, is the HOPER definedas follows:

$$
\begin{array}{lcl}
[\![X]\!]_\eta & = & \eta(X) \\
[\![T_1 \to T_2]\!]_\eta & = & [\![T_1]\!]_\eta \to [\![T_2]\!]_\eta \\
[\![\mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2]\!]_\eta & = & \bigcap^\star_{P \subseteq^{K_1} [\![T_1]\!]_\eta} [\![T_2]\!]_{\eta[X \leftarrow P]} \\
[\![\bigwedge^K [T_1..T_n]]\!]_\eta & = & \bigcap^K_{1 \leq i \leq n} [\![T_i]\!]_\eta \\
[\![S\,T]\!]_\eta & = & [\![S]\!]_\eta\, [\![T]\!]_\eta \\
[\![\mathrm{Fun}\,(X{:}K)\,T]\!]_\eta & = & \lambda P \in [\![K]\!].\, [\![T]\!]_{\eta[X \leftarrow P]}
\end{array}
$$

We say that an environment η *satisfies* a context Γ, written $\eta \models \Gamma$, if $\mathrm{dom}(\eta) = \mathrm{dom}(\Gamma)$ and

1. $\Gamma \equiv \emptyset$; or
2. $\Gamma \equiv \Gamma_1,\ x{:}T$, where $\eta\backslash x$ satisfies Γ_1 and either $[\![T]\!]_{\eta\backslash x}\uparrow$ or $\eta(x) \in \mathrm{dom}([\![T]\!]_{\eta\backslash x})$; or
3. $\Gamma \equiv \Gamma_1,\ X{\leq}T{:}K$, where $\eta\backslash X$ satisfies Γ_1 and either $[\![T]\!]_{\eta\backslash X}\uparrow$ or $\eta(X) \subseteq^K [\![T]\!]_{\eta\backslash X}$.

Iterating the definition immediately yields that either $[\![T]\!]_{\eta\backslash\Gamma_2\backslash X}\uparrow$ or $[\![T]\!]_{\eta\backslash\Gamma_2\backslash X} \in [\![K]\!]$, whenever $\eta \models \Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2$. Also, note that if η' extends η and $\mathrm{FTV}(T) \subseteq \mathrm{dom}(\eta)$, then either $[\![T]\!]_\eta\uparrow$ and $[\![T]\!]_{\eta'}\uparrow$, or else both are defined and $[\![T]\!]_\eta = [\![T]\!]_{\eta'}$.

Lemma 6.3.1 *Let T be a type, Γ a context, and η an environment such that $FTV(T) \subseteq$ $dom(\eta)$ and $\eta \models \Gamma$. If $\Gamma \vdash T \in K$, then $[\![T]\!]_\eta\downarrow$ and $[\![T]\!]_\eta \in [\![K]\!]$.*

Proof: We need to check the desired result together with an additional fact:

1. If $\Gamma \vdash T \in K$, then $[\![T]\!]_\eta$ is defined and $[\![T]\!]_\eta \in [\![K]\!]$.
2. If $\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash \mathrm{ok}$, then $[\![T]\!]_{\eta\backslash\Gamma_2\backslash X}\downarrow$ and $[\![T]\!]_{\eta\backslash\Gamma_2\backslash X} \in [\![K]\!]$.

The two are proved by simultaneous induction on derivations. We give only the interesting cases; the rest follow by straightforward use of the induction hypothesis and simple properties of HOPERs.

1. K-TVar We are given that $\Gamma \equiv \Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2$ and that $\Gamma \vdash \mathrm{ok}$. By part 2 of the induction hypothesis, $[\![T]\!]_{\eta\backslash\Gamma_2\backslash X}\downarrow$. By the definition of satisfaction, $(\eta\backslash\Gamma_2)(X) \subseteq^K [\![T]\!]_{\eta\backslash\Gamma_2\backslash X}$, which implies in particular that $\eta(X) \in [\![K]\!]$. By the definition of interpretation, $[\![X]\!]_\eta = \eta(X) \in [\![K]\!]$.

   K-All We are given that $K \equiv \star$ and $T \equiv \mathrm{All}\,(X{\leq}T_1{:}K_1)\,T_2$, and that $\Gamma \vdash T_1 \in K_1$ and $\Gamma,\ X{\leq}\top^{K_1}{:}K_1 \vdash T_2 \in K$. The induction hypothesis yields $[\![T_1]\!]_\eta\downarrow$ and $[\![T_1]\!]_\eta \in [\![K_1]\!]$. Now, suppose $P \subseteq^{K_1} [\![T_1]\!]_\eta$. Since $P \subseteq^{K_1} \top^{K_1}$, the definition of satisfaction yields $\eta[X \leftarrow P] \models \Gamma,\ X{\leq}\top^{K_1}{:}K_1$. By the induction hypothesis, $[\![T_2]\!]_{\eta[X \leftarrow P]}\downarrow$ and $[\![T_2]\!]_{\eta[X \leftarrow P]} \in [\![\star]\!]$. Since PER is closed under intersections, $[\![T]\!]_\eta = \bigcap^\star_{P \subseteq^{K_1} [\![T_1]\!]_\eta} [\![T_2]\!]_{\eta[X \leftarrow P]} \in [\![\star]\!]$.

K-OAbs Similar.

2. C-Var We are given that $\Gamma \equiv \Gamma', x{:}S$, where $\Gamma' \equiv \Gamma_1, X{\leq}T{:}K, \Gamma'_2$ and that $\Gamma' \vdash S \in \star$. By lemma 2.3.1, $\Gamma' \vdash \text{ok}$. By the definition of satisfaction, $\eta\backslash x \models \Gamma'$. By the induction hypothesis, the result follows.

   C-TVar The case where $\Gamma_2 \not\equiv \emptyset$ is similar to the previous case. When $\Gamma_2 \equiv \emptyset$, we are given that $\Gamma \equiv \Gamma_1, X{\leq}T{:}K$ and that $\Gamma_1 \vdash T \in K$. By the definition of satisfaction, $\eta\backslash X \models \Gamma'$. By the induction hypothesis, the result follows.

In order to prove the soundness of subtyping we need some technical results about substitution and β-conversion.

Lemma 6.3.2 *Let η be an environment with $X \notin dom(\eta)$ and such that $FTV(S[X{\leftarrow}T]) \subseteq dom(\eta)$ and $[\![S]\!]_{\eta[X\leftarrow[\![T]\!]_\eta]}\downarrow$. Then $[\![S[X{\leftarrow}T]]\!]_\eta = [\![S]\!]_{\eta[X\leftarrow[\![T]\!]_\eta]}$.*

Proof: By induction on the structure of S.

Lemma 6.3.3 *Let η be an environment such that $FTV(S) \subseteq dom(\eta)$. If $S =_{\beta\wedge} T$ and $[\![S]\!]_\eta\downarrow$, then $[\![S]\!]_\eta = [\![T]\!]_\eta$.*

Proof: By induction on the definition of $\beta\wedge$-conversion, it is easy to see that it suffices to show the statement for a one-step reduction $S \rightarrow_{\beta\wedge} T$. This is proved by induction on the structure of S. The only interesting cases are when S is a $\beta\wedge$-redex and T its reduct. Let $S \equiv (\text{Fun}\,(X{:}K)\,T_2)T_1$ and $T \equiv T_1[X{\leftarrow}T_2]$. Then

$$\begin{array}{rcll}
[\![(\text{Fun}\,(X{:}K)\,T_2)T_1]\!]_\eta & = & (\lambda P{\in}[\![K]\!].\,[\![T_2]\!]_{\eta[X\leftarrow P]})[\![T_1]\!]_\eta & \text{by definition of } [\![\,-\,]\!]_\eta \\
& = & [\![T_2]\!]_{\eta[X\leftarrow[\![T_1]\!]_\eta]} & \\
& = & [\![T_2[X{\leftarrow}T_1]]\!]_\eta & \text{by lemma 6.3.2.}
\end{array}$$

For the other redexes the result follows from lemma 6.2.1 given that for items 4 through 8 the inclusions are equalities.

Our main semantic results are the soundness of subtyping and typing.

Theorem 6.3.4 (Soundness of subtyping) *If $\Gamma \vdash S \leq T$ and $\Gamma \vdash S \in K$, and if $\eta \models \Gamma$, then $[\![S]\!]_\eta \subseteq^K [\![T]\!]_\eta$.*

Proof: The proof proceeds by induction on the structure of a derivation of $\Gamma \vdash S \leq T$. For the sake of readability, we often make implicit use of the fact that if $\Gamma \vdash S \leq T$ and $\Gamma \vdash S \in K$, then, by Proposition 2.3.17 and lemmas 2.3.7 and 6.3.1, $[\![S]\!]_\eta$ and $[\![T]\!]_\eta$ are defined and belong to $[\![K]\!]$ whenever $\eta \models \Gamma$.

S-Conv We are given that $S =_\beta T$. Then, by lemma 6.3.3, $[\![S]\!]_\eta = [\![T]\!]_\eta$

S-Trans By the induction hypothesis and the transitivity of $\subseteq^K$ (6.2.1).

S-TVar We are given $\Gamma_1,\ X{\le}T{:}K,\ \Gamma_2 \vdash \mathrm{ok}$. Now, $[\![X]\!]_\eta = \eta(X) \subseteq^K [\![T]\!]_{\eta\backslash X}$, because $\eta \models \Gamma$ and, as η extends $\eta\backslash X$, $[\![T]\!]_{\eta\backslash X} = [\![T]\!]_\eta$.

S-Arrow We are given $\Gamma \vdash T_1 \le S_1$ and $\Gamma \vdash S_2 \le T_2$, with $\Gamma \vdash S_1{\rightarrow}S_2 \in \star$. By the uniqueness of kinds (2.3.7), $K \equiv \star$. Now by the well-kindedness of subtyping (2.3.17) and syntax directedness of kinds (2.3.6) we have $\Gamma \vdash S_2, T_1 \in \star$. By the induction hypothesis, $[\![S_2]\!]_\eta \subseteq^\star [\![T_2]\!]_\eta$ and $[\![T_1]\!]_\eta \subseteq^\star [\![S_1]\!]_\eta$. Hence, by the covariance on the right and contravariance on the left of the function space constructor on PERs (easily verified from its definition), $[\![S_1]\!]_\eta{\rightarrow}[\![S_2]\!]_\eta \subseteq^\star [\![T_1]\!]_\eta{\rightarrow}[\![T_2]\!]_\eta$, i.e. $[\![S_1{\rightarrow}S_2]\!]_\eta \subseteq^\star [\![T_1{\rightarrow}T_2]\!]_\eta$.

S-All We are given that $\Gamma \vdash T_1 \le S_1$ and $\Gamma,\ X{\le}\top^{K_1}{:}K_1 \vdash S_2 \le T_2$ and that $\Gamma \vdash \mathrm{All}\,(X{\le}S_1{:}K_1)\,S_2 \in \star$. Let $P \in [\![K_1]\!]_\eta$. Then $\eta[X{\leftarrow}P] \models \Gamma,\ X{\le}\top^{K_1}{:}K_1$. By Proposition 2.3.6, $\Gamma,\ X{\le}\top^{K_1}{:}K_1 \vdash S_2 \in \star$. Now, by the induction hypothesis, $[\![S_2]\!]_{\eta[X\leftarrow P]} \subseteq^\star [\![T_2]\!]_{\eta[X\leftarrow P]}$. By the well-kindedness of subtyping and syntax-directedness of kinds, $\Gamma \vdash T_1 \in K_1$. Applying the induction hypothesis we obtain $[\![T_1]\!]_\eta \subseteq^{K_1} [\![S_1]\!]_\eta$. Hence $\bigcap^\star_{P\subseteq^{K_1}[\![S_1]\!]_\eta} [\![S_2]\!]_{\eta[X\leftarrow P]} \subseteq^\star \bigcap^\star_{P\subseteq^{K_1}[\![T_1]\!]_\eta} [\![T_2]\!]_{\eta[X\leftarrow P]}$, i.e. $[\![\mathrm{All}\,(X{\le}S_1{:}K_1)\,S_2]\!]_\eta \subseteq^\star [\![\mathrm{All}\,(X{\le}T_1{:}K_1)\,T_2]\!]_\eta$.

S-OAbs We are given $\Gamma,\ X{\le}\top^{K_1}{:}K_1 \vdash S \le T$ and $\Gamma \vdash \mathrm{Fun}\,(X{:}K_1)\,S \in K$. By the syntax-directedness of kinding, $K \equiv K_1{\rightarrow}K_2$ and $\Gamma,\ X{\le}\top^{K_1}{:}K_1 \vdash S \in K_2$ for some K_2. Let $P \in [\![K_1]\!]_\eta$. Then $\eta[X{\leftarrow}P] \models \Gamma,\ X{\le}\top^{K_1}{:}K_1$. Now by the induction hypothesis, $[\![S]\!]_{\eta[X\leftarrow P]} \subseteq^{K_2} [\![T]\!]_{\eta[X\leftarrow P]}$. Then $\lambda P \in [\![K_1]\!].\ [\![S]\!]_{\eta[X\leftarrow P]} \subseteq^{K_1\rightarrow K_2} \lambda P{\in}[\![K_1]\!].\ [\![T]\!]_{\eta[X\leftarrow P]}$, i.e. $[\![\mathrm{Fun}\,(X{:}K_1)\,S]\!]_\eta \subseteq^{K_1\rightarrow K_2} [\![\mathrm{Fun}\,(X{:}K_1)\,T]\!]_\eta$.

S-OApp By induction hypothesis, using the syntax-directedness of kinding.

S-Meet-G By the induction hypothesis and lemma 6.2.1(3).

S-Meet-LB By lemma 6.2.1(4).

The *type context* Γ/TV *obtained from a context* Γ is defined in the obvious way:

$$\begin{aligned} \emptyset/TV &= \emptyset, \\ (\Gamma,\ x{:}T)/TV &= \Gamma/TV, \\ (\Gamma,\ X{\le}T{:}K)/TV &= \Gamma/TV,\ X{\le}T{:}K. \end{aligned}$$

Theorem 6.3.5 *Let $\eta_1 \models \Gamma$ and $\eta_2 \models \Gamma$, such that $\eta_1/TV = \eta_2/TV$ and, for all $x \in dom(\Gamma)$, $\eta_1(x)\ \{[\![\Gamma(x)]\!]_\eta\}\ \eta_2(x)$, where $\eta = \eta_1/TV$. Then $[\![e]\!]_{\eta_1}\ \{[\![T]\!]_\eta\}\ [\![e]\!]_{\eta_2}$.*

Proof: From $\Gamma \vdash e \in T$ it follows by the well-kindedness of typing (2.3.18) that $\Gamma \vdash T \in \star$. Note that $\mathrm{FTV}(T) \subseteq \mathrm{dom}(\Gamma/TV)$; then, by strengthening (2.3.5), $\Gamma/TV \vdash T \in \star$. Note also that $\eta \models \Gamma/TV$; by lemma 6.3.1, $[\![T]\!]_\eta$ is defined and in $[\![\star]\!]$. We often use this fact implicitly in the following. The proof now proceeds by induction on a derivation of $\Gamma \vdash e \in T$.

T-Var From the assumption that for every x in $\mathrm{dom}(\Gamma)$. $\eta_1(x)\ \{[\![\Gamma(x)]\!]_\eta\}\ \eta_2(x)$.

T-Abs We are given that $\Gamma,\, x{:}T_1 \vdash e \in T_2$. Suppose that $p \in D$ and $q \in D$ are such that $p\ \{\llbracket T_1 \rrbracket_\eta\}\ q$. Then $\eta_1[x \leftarrow p] \models \Gamma,\, x{:}T_1$ and $\eta_2[x \leftarrow q] \models \Gamma,\, x{:}T_1$. By the induction hypothesis, $\llbracket e \rrbracket_{\eta_1[x\leftarrow p]}\ \{\llbracket T_2 \rrbracket_\eta\}\ \llbracket e \rrbracket_{\eta_2[x\leftarrow q]}$, that is, $\llbracket |\mathrm{erase}(e)| \rrbracket_{\eta_1[x\leftarrow p]}\ \{\llbracket T_2 \rrbracket_\eta\}$ $\llbracket |\mathrm{erase}(e)| \rrbracket_{\eta_2[x\leftarrow q]}$. From lemma 6.1.1(2) it follows that $\llbracket \mathrm{fun}^\star\,(x)\,|\mathrm{erase}(e)| \rrbracket_{\eta_1} \cdot p\ \{\llbracket T_2 \rrbracket_\eta\}$ $\llbracket \mathrm{fun}^\star\,(x)\,|\mathrm{erase}(e)| \rrbracket_{\eta_2} \cdot q$, that is, $\llbracket \mathrm{fun}\,(x{:}T_1)\, e) \rrbracket_{\eta_1} \cdot p\ \{\llbracket T_2 \rrbracket_\eta\}\ \llbracket \mathrm{fun}\,(x{:}T_1)\, e \rrbracket_{\eta_2} \cdot q$. Since p and q were chosen freely, we have $\llbracket \mathrm{fun}\,(x{:}T_1)\, e \rrbracket_{\eta_1}\ \{\llbracket T_1 \rrbracket_\eta \rightarrow \llbracket T_2 \rrbracket_\eta\}\ \llbracket \mathrm{fun}\,(x{:}T_1)\, e \rrbracket_{\eta_2}$, in other words $\llbracket \mathrm{fun}\,(x{:}T_1)\, e \rrbracket_{\eta_1}\ \{\llbracket T_1 \rightarrow T_2 \rrbracket_\eta\}\ \llbracket \mathrm{fun}\,(x{:}T_1)\, e \rrbracket_{\eta_2}$.

T-App By the induction hypothesis, using the fact that $\llbracket f\, a \rrbracket_\eta = \llbracket f \rrbracket_\eta \llbracket a \rrbracket_\eta$.

T-TAbs We are given that $\Gamma,\, X{\leq}T_1{:}K_1 \vdash e \in T_2$. Suppose that $P \subseteq^{K_1} \llbracket T_1 \rrbracket_\eta$. Then $\eta_1[X \leftarrow P] \models \Gamma,\, X{\leq}T_1{:}K_1$ and $\eta_2[X \leftarrow P] \models \Gamma,\, X{\leq}T_1{:}K_1$. Since $\llbracket e \rrbracket_{\eta_1} = \llbracket e \rrbracket_{\eta_1[X\leftarrow P]}$, we have by the induction hypothesis that $\llbracket e \rrbracket_{\eta_1}\ \{\llbracket T_2 \rrbracket_{\eta[X\leftarrow P]}\}\ \llbracket e \rrbracket_{\eta_2}$. Since P was chosen freely, we have that $\llbracket e \rrbracket_{\eta_1}\ \{\bigcap^\star_{P \subseteq^{K_1} \llbracket T_1 \rrbracket_\eta} \llbracket T_2 \rrbracket_{\eta[X\leftarrow P]}\}\ \llbracket e \rrbracket_{\eta_2}$. Now the result follows from the definition $\llbracket - \rrbracket_\eta$ and the fact that $\llbracket \mathrm{fun}\,(X{\leq}T_1{:}K_1)\, e \rrbracket_\eta = \llbracket e \rrbracket_\eta = \llbracket e \rrbracket_{\eta[X\leftarrow P]}$.

T-TApp We are given that $\Gamma \vdash f \in \mathrm{All}\,(X{\leq}T_1{:}K_1)\, T_2$ and that $\Gamma \vdash S \leq T_1$. By the induction hypothesis, $\llbracket f \rrbracket_{\eta_1}\ \{\llbracket \mathrm{All}\,(X{\leq}T_1{:}K_1)\, \rrbracket_\eta\}\ \llbracket f \rrbracket_{\eta_2}$, that is, $\llbracket f \rrbracket_{\eta_1}\ \{\bigcap^\star_{P \subseteq^{K_1}} \llbracket T_2 \rrbracket_{\eta[X\leftarrow P]}\}$ $\llbracket f \rrbracket_{\eta_2}$. By the well-kindedness of typing, syntax directedness of kinds, and well-kindedness of subtyping, $\Gamma \vdash S \in K_1$, and, by soundness of subtyping, $\llbracket S \rrbracket_\eta \subseteq^{K_1}$ $\llbracket T_1 \rrbracket_\eta$. So we have $\llbracket f \rrbracket_{\eta_1}\ \{\llbracket T_2 \rrbracket_{\eta[X\leftarrow \llbracket S \rrbracket_\eta]}\}\ \llbracket f \rrbracket_{\eta_2}$, which, by lemma 6.3.2 and the fact that $\llbracket f \rrbracket_\eta = \llbracket f\, S \rrbracket_\eta$ for any η, is $\llbracket f\, S \rrbracket_{\eta_1}\ \{\llbracket T_2[X \leftarrow S] \rrbracket_\eta\}\ \llbracket f\, S \rrbracket_{\eta_2}$.

T-For Immediate from the induction hypothesis, since $\mathrm{erase}(e[X \leftarrow S]) = \mathrm{erase}(e)$.

T-Meet We are given that $\Gamma \vdash \mathrm{ok}$ and also $\Gamma \vdash e \in T_i$ for each i in $\{1\,..\,n\}$. The result follows by the induction hypothesis and the definitions of $\bigcap^\star$ and $\llbracket - \rrbracket_\eta$.

T-Sub We are given that $\Gamma \vdash e \in S$ and that $\Gamma \vdash S \leq T$. By the induction hypothesis, $\llbracket e \rrbracket_{\eta_1}\ \{\llbracket S \rrbracket_\eta\}\ \llbracket e \rrbracket_{\eta_2}$. By well-kindedness of typing $\Gamma \vdash S \in \star$, and by the soundness of subtyping $\llbracket S \rrbracket_\eta \subseteq^\star \llbracket T \rrbracket_\eta$. Hence, $\llbracket e \rrbracket_{\eta_1}\ \{\llbracket T \rrbracket_\eta\}\ \llbracket e \rrbracket_{\eta_2}$.

Corollary 6.3.6 (Soundness of typing) *Let η be an environment such that $\eta \models \Gamma$. Then $\Gamma \vdash e \in T$ implies $\llbracket e \rrbracket_\eta {\in} dom(\llbracket T \rrbracket_\eta)$.*

Proof: Take $\eta_1 = \eta_2 = \eta$.

7 Related work

Independently, M. Steffen and B. Pierce proved decidability of subtyping for the $F^\omega_\leq$ fragment. They prove termination of their algorithm by proving strong normalization of a notion of reduction indexed with contexts ($\rightarrow_\Gamma$), while we reduce the proof of termination of *check* to prove strong normalization of $\rightarrow_{\beta\wedge+}$. A major difference with the work presented in [SP94] is the choice of the intermediate subtyping system. We define the normal system $NF^\omega_\wedge$ which not only is the key to proving decidability of subtyping but helped understanding the fine structure of subtyping, yielding, for example, a generation lemma, and revealing the role of bounds. In [SP94] the intermediate system (called reducing system), leads to a much more complicated proof which involves dealing with several notions of reduction and further reformulation of the intermediate system.

We can compare type variables and their bounds with definitions as follows. While checking subtyping statements, a variable can be replaced by its bound. Similarly, in a λ-calculus with definitions, during the evaluation of a term, a constant can be replaced by the body of its definition. Moreover, type variable declarations in a context correspond to global definitions, while bounds in quantifiers and abstractions[2] correspond to local definitions. Recently, Paula Severi and Erik Poll proved that extending all known strongly normalizing pure type systems, PTSs, – in particular, the systems of Barendregt's cube – with definitions preserves strong normalization. The proof uses an extended calculus of constructions [SP93]. Our approach to proving the termination of *check* deals uniformly with free and bound variables. Consequently, our solution suggests an alternative strategy to prove strong normalization of the systems studied [SP93].

8 Conclusions and future work

We proved the decidability of subtyping in higher-order systems containing a generalization of the rule suggested by G. Castagna in [CP94], namely $F^\omega_\wedge$ and $F^\omega_\leq$. What made this possible was a detailed analysis and deep understanding of syntactical properties of $F^\omega_\wedge$. The results presented here constitute the foundations for future study of typechecking in $F^\omega_\wedge$. We also showed that a simple semantics of intersections can be given by a straightforward extension of a model of higher-order polymorphism.

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 Benjamin Pierce, Henk Barendregt, and Ugo de'Liguoro.

This work was supported by funding from the Netherlands Organization for Scientific Research (N.W.O.).

[2] In Fun $(X{:}K)\,T$ the implicitly declared bound is $\top^K$.

References

[Bar84] H. P. Barendregt. *The Lambda Calculus.* North Holland, revised edition, 1984.

[BCD83] H. 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.

[BL90] Kim B. Bruce and Giuseppe Longo. A modest model of records, inheritance, and bounded quantification. *Information and Computation*, 87:196–240, 1990. To appear in [GM93]. An earlier version appeared in the proceedings of the IEEE Symposium on Logic in Computer Science, 1988.

[BM92] Kim Bruce and John 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.

[BMM90] Kim B. Bruce, Albert R. Meyer, and John C. Mitchell. The semantics of second-order lambda calculus. In Huet [Hue90], pages 213–272. Also appeared in Information and Computation 84, 1 (January 1990).

[Car] Luca Cardelli. Notes about $f^{\omega}_{<:}$. Technical report.

[Car88] Luca 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.

[CC90] Felice Cardone and Mario Coppo. Two extensions of Curry's type inference system. In Piergiorgio Odifreddi, editor, *Logic and Computer Science*, number 31 in APIC Studies in Data Processing, pages 19–76. Academic Press, 1990.

[CCH+89] Peter Canning, William Cook, Walter Hill, Walter Olthoff, and John Mitchell. F-bounded quantification for object-oriented programming. In *Fourth International Conference on Functional Programming Languages and Computer Architecture*, pages 273–280, September 1989.

[CDC78] M. Coppo and M. Dezani-Ciancaglini. A new type-assignment for λ-terms. *Archiv. Math. Logik*, 19:139–156, 1978.

[CG92] Pierre-Louis Curien and Giorgio Ghelli. Coherence of subsumption: Minimum typing and type-checking in $F_{\leq}$. *Mathematical Structures in Computer Science*, 2:55–91, 1992. To appear in [GM93].

[CHC90] William R. Cook, Walter L. Hill, and Peter S. Canning. Inheritance is not subtyping. In *Seventeenth Annual ACM Symposium on Principles of Programming Languages*, pages 125–135, San Francisco, CA, January 1990. To appear in [GM93].

[CL91] Luca Cardelli and Giuseppe Longo. A semantic basis for Quest. *Journal of Functional Programming*, 1(4):417–458, October 1991. Preliminary version in ACM Conference on Lisp and Functional Programming, June 1990. Also available as DEC SRC Research Report 55, Feb. 1990.

[CP93] Adriana B. Compagnoni and Benjamin C. Pierce. Multiple inheritance via intersection types. Technical Report ECS-LFCS-93-275, LFCS, University of Edinburgh, August 1993. Also available as Catholic University Nijmegen computer science technical report 93-18.

[CP94] Giuseppe Castagna and Benjamin Pierce. Decidable bounded quantification. In *Proceedings of Twenty-First Annual ACM Symposium on Principles of Programming Languages, Portland, OR.* ACM, January 1994. To appear.

[CW85] Luca Cardelli and Peter Wegner. On understanding types, data abstraction, and polymorphism. *Computing Surveys*, 17(4), December 1985.

[dB72] Nicolas 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.

[FMRS90] P. Freyd, P. Mulry, G. Rosolini, and D. Scott. Extensional PERs. In *Fifth Annual Symposium on Logic in Computer Science (Philadelphia, PA)*, pages 346–354. IEEE Computer Society Press, June 1990.

[Ghe90] Giorgio 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.

[Gir72] Jean-Yves Girard. *Interprétation fonctionelle et élimination des coupures de l'arithmétique d'ordre supérieur.* PhD thesis, Université Paris VII, 1972.

[GM93] Carl A. Gunter and John C. Mitchell. *Theoretical Aspects of Object-Oriented Programming: Types, Semantics, and Language Design.* The MIT Press, 1993. To appear.

[HS86] J. Roger Hindley and Jonathan P. Seldin. *Introduction to Combinators and λ-Calculus*, volume 1 of *London Mathematical Society Student Texts.* Cambridge University Press, 1986.

[Hue90] Gérard Huet, editor. *Logical Foundations of Functional Programming.* University of Texas at Austin Year of Programming Series. Addison-Wesley, 1990.

[Mit90a] John 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. To appear in [GM93].

[Mit90b] John C. Mitchell. A type-inference approach to reduction properties and semantics of polymorphic expressions. In Huet [Hue90], pages 195–212.

[Pie91] Benjamin 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.

[PT93] Benjamin C. Pierce and David N. Turner. Simple type-theoretic foundations for object-oriented programming. *Journal of Functional Programming*, 1993. To appear; 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".

[Sco76] Dana Scott. Data types as lattices. *SIAM Journal on Computing*, 5(3):522–587, 1976.

[SP93] Paula Severi and Erik Poll. Pure type systems with definitions. Computing Science Note 93/24, Eindhoven University of Technology, September 1993.

[SP94] Martin Steffen and Benjamin C. Pierce. Higher-order subtyping. Draft technical report, January 1994.