Decidability of Higher-Order Subtyping with Intersection Types

Adriana B. Compagnoni*

Abstract

The combination of higher-order subtyping with intersection types yields a typed model of object-oriented programming with multiple inheritance [11]. 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, and is our subject of study.

Our main contribution is the proof that subtyping in $F^{\omega}_{\wedge}$ is decidable. This 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 [8] has no reductions on types. In the $F^{\omega}_{\wedge}$ subtyping system the presence of $\beta\wedge$-conversion – an extension of β-conversion with distributivity laws – drastically increases the complexity of proving the decidability of the subtyping relation. Our proof consists of, firstly, defining an algorithmic presentation of the subtyping system of $F^{\omega}_{\wedge}$, secondly, proving that this new presentation is sound and complete with respect to the original one, and finally, proving that the algorithm always terminates.

Moreover, we establish basic structural properties of the language of types of $F^{\omega}_{\wedge}$ such as strong normalization and Church-Rosser.

Among the novel aspects of the present solution is the use of term rewriting techniques to present intersection types, which clearly splits the computational semantics (reduction rules) from the syntax (inference rules) of the system. Another original feature is the use of a choice operator to model the behavior of variables during subtype checking.

1 Introduction

The system $F^{\omega}_{\wedge}$ (F-omega-meet) was first introduced in [11], where it was shown to be rich enough to provide a typed model of object oriented programming with multiple inheritance. $F^{\omega}_{\wedge}$ is an extension of F^{ω} [17] 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 [14, 21, 22, 8]. 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 [13, 1] and elsewhere (see [7] for background and further

*Laboratory for Foundations of Computer Science, Department of Computer Science, University of Edinburgh. United Kingdom. This research was supported by the Dutch organization for scientific research, NWO-SION project *Typed lambda calculus*, 612-316-030, and by EPSRC GRANT, GR/G 55792 *Constructive logic as a basis for program development.*

references). A second-order λ-calculus with intersection types was studied in [21]. Systems including subtyping were present in [6, 4]. Higher order generalizations of subtyping appear in [3, 12, 19, 2]. $F_{\leq}$ (F-sub), a second-order λ-calculus with bounded quantification, was studied in [15], and in [21] it was proved that subtyping in $F_{\leq}$ was undecidable and that undecidability was caused by the subtyping rule for bounded quantification.

In [8] an alternative rule for subtyping quantified types was presented and the decidability of subtyping was proved for an extension of system F with bounded polymorphism.

Allowing bounds of functional kind forces us to introduce a conversion rule to have invariance of subtyping under $\beta\wedge$-conversion of types. Therefore, our subtyping relation relates types of a more expressive type system than that presented in [8]. 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 [3] or Cardelli and Mitchell's system $F^{\omega}_{\leq}$ (F-omega-sub) [5, 19, 2].

Ghelli [16] remarked that the rule for subtyping between quantified types presented in [8] led to a well-behaved subtyping relation but that the typing relation fails to satisfy the minimal type property. This failure introduces serious problems in type checking and type inference. At the moment it is not clear how to solve them or, even more problematic, whether the typing relation is decidable. A possible solution to overcome this problem is to replace the subtyping rule between quantifiers by the corresponding rule of Cardelli and Wegner's kernel fun [6].

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. This immediately gives the decidability of subtyping for the $F^{\omega}_{\leq}$ fragment as the former is a conservative extension of the latter, i.e. each subtyping statement derivable in $F^{\omega}_{\wedge}$ that contains no intersections other than the empty ones is also derivable in $F^{\omega}_{\leq}$. The system $F^{\omega}_{\wedge}$ satisfies the minimal type property and its typing relation is decidable. These two results are beyond the scope of this paper and can be found in [9].

We present a definition of $F^{\omega}_{\wedge}$ that differs from the one introduced in [11] in two ways. First, Castagna and Pierce's quantifier rule has been replaced by the Cardelli and Wegner rule. Second, we introduce a richer notion of reduction on types, and thereby the four distributivity rules become particular cases of the conversion rule. This new reduction is shown to be confluent and strongly normalizing. The latter simplification was motivated by structural properties of the former presentation. Furthermore, this new presentation provides a different 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 restricting the subtyping rules of $F^{\omega}_{\wedge}$ to handle only types in normal form and replacing conversion by reflexivity. In section 3.1 there is an example

illustrating the problem to be solved with a subtyping statement which is not derivable without using conversion, i.e. without performing any β-reduction, even when the conclusion is in normal form.

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.

Our solution is divided into two main steps. First, we develop a *normal subtyping system*, $NF^\omega_\wedge$, in which only types in normal form are considered. We prove that derivations in $NF^\omega_\wedge$ can be normalized by eliminating transitivity and simplifying reflexivity. This simplification yields an algorithmic presentation, $AlgF^\omega_\wedge$. Moreover, we prove that $AlgF^\omega_\wedge$ is indeed an alternative presentation of the $F^\omega_\wedge$ subtyping relation, that is $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$ (proposition 3.3.3).

The second and last step towards the decidability of subtyping in $F^\omega_\wedge$ is to prove that the algorithm described by $AlgF^\omega_\wedge$ terminates, which is equivalent to showing that the definition of the $AlgF^\omega_\wedge$ is well-founded. We discuss this further in section 4.

Checking if $\Gamma \vdash_{Alg} S\,T \leq A$ is reduced to checking if $\Gamma \vdash_{Alg} lub_\Gamma(S\,T)^{\mathrm{nf}} \leq A$, where $lub_\Gamma(S\,T)$ substitutes the leftmost innermost variable of $S\,T$ 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 size of the types to be examined in the recursive call does not decrease. This indicates that the proof of termination of the algorithm is not immediate. In particular, the proof of termination presented in [8] cannot be modified to serve our purposes, because of the interaction between $\beta\wedge$-reduction and the substitution of type variables by their bounds in our system. We discuss this further in section 4.

2 The System $F^\omega_\wedge$

The *kinds* of $F^\omega_\wedge$ are those of F^ω, they are the kind $\star$ of proper types and the kinds $K_1 \to K_2$ of functions on types (sometimes called type operators). 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 ($T \to T'$), and polymorphic types ($\forall X \leq T{:}K.T'$), in which the bound type variable X ranges over all subtypes of the upper bound T. Moreover, like F^ω, we allow types to be abstracted on types ($\Lambda X{:}K.T$) and applied to argument types ($T\,T'$). 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. The empty intersection at kind K is written $\top^K$. We drop the maximal type *Top* of $F_\leq$, since its role is played here by $\top^\star$.

The reduction $\to_{\beta\wedge}$ on types consists of the β-reduction and distributivity rules for each constructor associated with intersections.

DEFINITION 2.1

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

The relation $\rightarrow_{\beta\wedge}$ is extended so as to become a compatible relation with respect to type formation, $\twoheadrightarrow_{\beta\wedge}$ is the transitive and reflexive closure of $\rightarrow_{\beta\wedge}$, and $=_{\beta\wedge}$ is the least equivalence relation containing $\rightarrow_{\beta\wedge}$. The capture-avoiding substitution of S for X in T is written $T[X \leftarrow S]$. Substitution is extended pointwise to contexts. The $\beta\wedge$-normal form of a type S is written S^{nf}, and is extended pointwise to contexts.

Strictly speaking the operational semantics of the system is captured by rules 1 through 5. Associativity, commutativity, and idempotency of intersection types exist in the system as derived subtyping judgements. We included rule 6 (associativity) here because it is convenient for the development of the subtyping algorithm in section 3.1.

THEOREM 2.2 (*Church-Rosser for* $\rightarrow_{\beta\wedge}$) If $S \twoheadrightarrow_{\beta\wedge} T_1$ and $S \twoheadrightarrow_{\beta\wedge} T_2$, then there exists T_3 such that $T_1 \twoheadrightarrow_{\beta\wedge} T_3$ and $T_2 \twoheadrightarrow_{\beta\wedge} T_3$.

The rules in $F^{\omega}_{\wedge}$ 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.

2.1 Context Formation

A *context* Γ is a finite sequence of typing and/or subtyping assumptions for a set of variables and/or type variables. The formation of contexts follow the three rules below.

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

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

In the type variable declaration $X \leq T{:}K$, T is called the *bound* of X. We write $\Gamma(Y)$ to denote the bound of Y in Γ. We say that Γ is included in Γ', $\Gamma \subseteq \Gamma'$, if every assumption in Γ is in Γ'. The set of variables declared in Γ is called the *domain* of Γ, $\mathrm{dom}(\Gamma)$.

2.1.1 Type Formation

For each type constructor, there is a rule specifying how it can be used to build well-formed type expressions. We use the abbreviation $X{:}K$ for $X \leq \top^K{:}K$.

$$\frac{\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash \text{ok}}{\Gamma_1,\ X{\leq}T{:}K,\ \Gamma_2 \vdash X \in K}\ \text{(K-TVar)} \qquad \frac{\Gamma \vdash T_1 \in \star \qquad \Gamma \vdash T_2 \in \star}{\Gamma \vdash T_1{\to}T_2 \in \star}\ \text{(K-Arrow)}$$

$$\frac{\Gamma,\ X{\leq}T_1{:}K_1 \vdash T_2 \in \star}{\Gamma \vdash \forall X{\leq}T_1{:}K_1.T_2 \in \star}\ \text{(K-All)} \qquad \frac{\Gamma,\ X{:}K_1 \vdash T_2 \in K_2}{\Gamma \vdash \Lambda X{:}K_1.T_2 \in K_1{\to}K_2}\ \text{(K-OAbs)}$$

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

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

Lemma 2.1.1 1. For any context Γ it is decidable whether $\Gamma \vdash \text{ok}$.

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

Lemma 2.1.2 (*Subject reduction*) 1. $\Gamma \vdash \text{ok}$ and $\Gamma \to_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash \text{ok}$.

2. $\Gamma \vdash S \in K$ and $S \to_{\beta\wedge} T$ implies $\Gamma \vdash T \in K$.

3. $\Gamma \vdash S \in K$ and $\Gamma \to_{\beta\wedge} \Gamma'$ implies $\Gamma' \vdash S \in K$. □

Proposition 2.1.3 (*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.

Theorem 2.1.4 (*Strong normalization for* $\to_{\beta\wedge}$) If $\Gamma \vdash T \in K$, then every $\beta\wedge$-reduction sequence starting from T terminates.

2.1.2 Subtyping

The rules defining the subtype relation are a natural extension of familiar calculi of bounded quantification. Aside from some extra well-formedness conditions, the rules S-Trans, S-TVar, and S-Arrow, are the same as in the usual, second-order case. F^ω's rule of type conversion (i.e. 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. Rules S-Meet-G and S-Meet-LB specify that an intersection of a set of types is their order-theoretic greatest lower bound.

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

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

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

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

$$\frac{\Gamma,\ X{\leq}\top^K{:}K \vdash S \leq T}{\Gamma \vdash \Lambda X{:}K.S \leq \Lambda X{:}K.T} \quad \text{(S-OAbs)} \qquad \frac{\Gamma \vdash S \leq T \qquad \Gamma \vdash S\,U \in K}{\Gamma \vdash S\,U \leq T\,U} \quad \text{(S-OApp)}$$

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

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

3 Decidability of subtyping

In this section we show that the subtyping relation of $F^\omega_\wedge$ is decidable. The solution is divided into two main parts. First, we develop a *normal subtyping system*, $NF^\omega_\wedge$, in which only types in normal form are considered. We prove that derivations in $NF^\omega_\wedge$ can be normalized by eliminating transitivity and simplifying reflexivity. This simplification yields an algorithmic presentation, $AlgF^\omega_\wedge$, whose rules are syntax directed. Moreover, we prove that $AlgF^\omega_\wedge$ is indeed an alternative presentation of the $F^\omega_\wedge$ subtyping relation. Formally, $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$ (proposition 3.3.3).

In the solution for the second order lambda calculus presented in [21], the distributivity rules for intersection types are not considered as rewrite rules. For that reason, new syntactic categories have to be defined (composite and individual canonical types) and an auxiliary mapping (flattening) transforms a type into a canonical type. Our solution does not need either new syntactic categories or elaborate auxiliary mappings, since the role played there by canonical types is performed here by types in normal form.

Independently Steffen and Pierce proved a similar result for $F^\omega_{\leq}$ [23]. There are several differences between our work and the proof of decidability of subtyping in [23]. First, our result is for a stronger system which also includes intersection types. Our proof of termination has the novel idea of using a choice operator to model the behavior of type variables during subtype checking. A second major difference is the choice of the intermediate subtyping system. We define the normal system $NF^\omega_\wedge$, which is not only the key to proving decidability of subtyping but helped understand the fine structure of subtyping, yielding the algorithm $AlgF^\omega_\wedge$. In [23] the intermediate system, called a reducing system, leads to a much more complicated proof which involves dealing with several notions of reduction and further reformulations of the intermediate system.

3.1 Normal Subtyping

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

Example 3.1.1 In $F^\omega_\wedge$ we can prove that if $\Gamma \equiv W{:}K,\ X{\leq}(\Lambda Y{:}K.Y){:}K{\to}K$,

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash \mathrm{ok}}{\Gamma \vdash X \leq \Lambda Y{:}K.Y}\,\text{S-TVar}}{\Gamma \vdash X\,W \leq (\Lambda Y{:}K.Y)\,W}\,\text{S-OApp} \qquad \dfrac{(\Lambda Y{:}K.Y)\,W =_{\beta\wedge} W}{(\Lambda Y{:}K.Y)\,W \leq W}\,\text{S-Conv}}{\Gamma \vdash X\,W \leq W.}\,\text{S-Trans}$$

This example shows that S-Trans erases information obtained by S-Conv that is not present in the conclusion. 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 ($NF^{\omega}_{\wedge}$) in which conversion is reduced to a minimum and transitivity can be eliminated (lemma 3.1.7). Both results are proved with a standard cut-elimination argument. This yields a syntax directed subtyping relation ($AlgF^{\omega}_{\wedge}$) which constitutes a decision procedure for the original system.

In this section, we present the subtyping system $NF^{\omega}_{\wedge}$ that uses the context and type formation rules of $F^{\omega}_{\wedge}$. We prove a generation lemma for subtyping (proposition 3.1.8) and define an algorithmic presentation, $AlgF^{\omega}_{\wedge}$ (see definition 3.3.1). Finally, we show that there is an equivalence between subtyping in $F^{\omega}_{\wedge}$ and subtyping in $AlgF^{\omega}_{\wedge}$, which is essential to prove the decidability of subtyping in $F^{\omega}_{\wedge}$ (see section 3.3).

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

Notation 3.1.2 A, B, and C range over types whose outermost constructor is not an intersection.

Remark 3.1.3 It is an immediate consequence of the $\beta\wedge$-reduction rules that, if T is in $\beta\wedge$-normal form, then T is either a variable X, an arrow type $S{\to}A$, a bounded quantification $\forall X{\leq}S{:}K.A$, an abstraction $\Lambda X{:}K.A$, an application $A\,S$ where A is not an abstraction, or an intersection $\bigwedge^{K}[A_1..A_n]$.

We now define $lub_{\Gamma}(S)$, and we prove in lemma 3.2.1 and corollary 3.2.9, that, when defined, it is the smallest type above S with respect to Γ.

Definition 3.1.4 (*Least strict Upper Bound*)

$$lub_{\Gamma}(X) \;=\; \Gamma(X), \qquad lub_{\Gamma}(T\,S) \;=\; lub_{\Gamma}(T)\,S.$$

Note that when S is a well-kinded type with respect to Γ, and it is a type variable or a type application in (head)normal form, then $lub_{\Gamma}(S)$ is defined.

Definition 3.1.5 ($\mathrm{NF}^{\omega}_{\wedge}$ *subtyping rules*)

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

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

$$\frac{\Gamma \vdash_n (lub_\Gamma(T\,S))^{\mathrm{nf}} \leq A \qquad \Gamma \vdash T\,S \in K \qquad T\,S \not\equiv A}{\Gamma \vdash_n T\,S \leq A} \tag{NS-OApp}$$

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

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

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

NS-Trans, NS-Arrow, NS-All, NS-OAbs are as in section 2.1.2.

As we mentioned in the introduction, an important factor in developing this system was to consider the distributivity rules of the presentation of $F^\omega_\wedge$ in [11] 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. However, as shown by the (counter)example 3.1.1, 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 3.1.1. Hence, the solution is not as simple as replacing S-Conv by NS-Refl.

In general, the interaction between S-Trans and S-Conv can be analyzed as follows. In S-Trans the metavariable T of the hypothesis is not present in the conclusion, but this is not a problem by itself (a similar situation appears in the simply typed lambda calculus in its application rule and the system is deterministic). The problem is that in the presence of S-Conv the *vanishing* T can be $\beta\wedge$-convertible to either S or U or to both S and U. What example 3.1.1 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 $F_\leq$ [15] transitivity is eliminated and hidden in a richer variable rule in which deciding whether $\Gamma \vdash X \leq T$ when $T \not\equiv X$ is reduced to deciding whether the bound of X is smaller than or equal to T. The bound of X has the particular property of being the least strict upper bound of X. This observation motivated the definition of our NS-OApp rule, in which we reduce the decision of whether $\Gamma \vdash T\,S \leq A$ when $A \not\equiv T\,S$, to checking whether the least strict upper bound of $T\,S$ is smaller than or equal to A (See lemma 3.2.1 and corollary 3.2.9). The least strict upper bound of $T\,S$ with respect to Γ, $lub_\Gamma(T\,S)$, is obtained from $T\,S$ by replacing its leftmost innermost variable by the corresponding bound in Γ. Consequently, $lub_\Gamma(T\,S)$ may be other than a normal form. That is the reason we normalize it. The strength of the conversion rule that is not captured by reflexivity is hidden in this normalization step. Since $T\,S$ is a well kinded type, the free type variables of $T\,S$ are included in $\mathrm{dom}(\Gamma)$. Therefore, $lub_\Gamma(T\,S)$ is defined. By lemma 3.2.1(1), $lub_\Gamma(T\,S)$ is well-kinded,

and since well-kinded types are strongly normalizing, its normal form exists. The rules S-MEET-LB and S-MEET-G are replaced by NS-$\exists$, NS-$\forall$, and NS-$\forall\exists$.

EXAMPLE 3.1.6 A derivation of the subtyping statement of example 3.1.1 in $NF^{\omega}_{\wedge}$. Let $\Gamma \equiv W{:}K,\ X{\leq}(\Lambda Y{:}K.Y){:}K{\to}K$.

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

We say that a proof is in *normal form* if it has no applications of NS-TRANS and in which NS-REFL is only applied to type variables and type applications.

LEMMA 3.1.7 If $\Gamma \vdash_n S \leq T$, then there exists a proof of $\Gamma \vdash_n S \leq T$ in normal form.

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

PROPOSITION 3.1.8 (*Generation 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. $\Gamma \vdash_n S{\to}A \leq B$ implies $B \equiv T{\to}C$, $\Gamma \vdash_n T \leq S$, $\Gamma \vdash_n A \leq C$, and $\Gamma \vdash S{\to}A \in \star$.
3. $\Gamma \vdash_n \forall X{\leq}S{:}K.A \leq B$ implies $B \equiv \forall X{\leq}S{:}K.C$, $\Gamma,\ X{\leq}S{:}K \vdash_n A \leq C$, and $\Gamma \vdash \forall X{\leq}S{:}K.A \in \star$.
4. $\Gamma \vdash_n \Lambda X{:}K.A \leq B$ implies $B \equiv \Lambda X{:}K.C$, and $\Gamma,\ X{\leq}\top^K{:}K \vdash_n A \leq C$.
5. $\Gamma \vdash_n A\,S \leq B$ implies $B \equiv A\,S$, or $\Gamma \vdash_n (lub_\Gamma(A\,S))^{\mathrm{nf}} \leq B$, and $\Gamma \vdash A\,S \in K$.
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.

3.2 Equivalence of ordinary and normal subtyping

In this section, we show that a subtyping statement is derivable in $F^\omega_\wedge$ if and only if the corresponding normalized statement is derivable in $NF^\omega_\wedge$. This equivalence is proved in theorem 3.2.8. As usual, we need some auxiliary properties and definitions, among which we can highlight propositions 3.2.2 and 3.2.7.

LEMMA 3.2.1 Let $lub_\Gamma(S)$ be defined. Then

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

PROPOSITION 3.2.2 (*Soundness*) If $\Gamma \vdash_n S \leq T$, then $\Gamma \vdash S \leq T$.

LEMMA 3.2.3
1. $\Gamma \vdash \mathrm{ok}$ implies $\Gamma^{\mathrm{nf}} \vdash \mathrm{ok}$.
2. $\Gamma \vdash T \in K$ implies $\Gamma^{\mathrm{nf}} \vdash T \in K$.
3. $\Gamma \vdash S \leq T$ implies $\Gamma^{\mathrm{nf}} \vdash S \leq T$.
4. Let $\Gamma_1, \Gamma_2 \vdash \mathrm{ok}$. Then $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash T \in K$ implies $\Gamma_1, \Gamma_2 \vdash T \in K$.
5. Let $\Gamma_1, \Gamma_2 \vdash \mathrm{ok}$. Then $\Gamma_1^{\mathrm{nf}}, \Gamma_2 \vdash S \leq T$ implies $\Gamma_1, \Gamma_2 \vdash S \leq T$.
6. Let $\Gamma \vdash S, T \in K$. Then $\Gamma^{\mathrm{nf}} \vdash S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$ if and only if $\Gamma \vdash S \leq T$.

The following lemma states that S-TVAR is an admissible rule in $NF^\omega_\wedge$.

LEMMA 3.2.4 If Γ is a context in normal form such that $\Gamma \vdash \mathrm{ok}$ and $Y \in \mathrm{dom}(\Gamma)$, then $\Gamma \vdash_n Y \leq \Gamma(Y)$.

LEMMA 3.2.5 (*Substitution*) If $\Gamma \vdash U \in K$ and $\Gamma, X{:}K, \Gamma' \vdash_n S \leq T$, then $\Gamma, (\Gamma'[X \leftarrow U])^{\mathrm{nf}} \vdash_n (S[X \leftarrow U])^{\mathrm{nf}} \leq (T[X \leftarrow U])^{\mathrm{nf}}$.

This substitution lemma is the key result we use in proving that S-OAPP has a corresponding admissible rule in $NF^\omega_\wedge$.

LEMMA 3.2.6 $\Gamma \vdash S\,U \in K$. Then $\Gamma \vdash_n S \leq T$ implies $\Gamma \vdash_n (S\,U)^{\mathrm{nf}} \leq (T\,U)^{\mathrm{nf}}$.

PROPOSITION 3.2.7 (*Completeness*) $\Gamma \vdash S \leq T \Rightarrow \Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

PROOF: By induction on the derivation of $\Gamma \vdash S \leq T$ using lemma 3.2.6 and the Church-Rosser theorem (2.2) for the case of S-OAPP. □

THEOREM 3.2.8 (*Equivalence of ordinary and normal subtyping*) Let $\Gamma \vdash S \in K$ and $\Gamma \vdash T \in K$. Then $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_n S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

PROOF: ($\Rightarrow$) By completeness (proposition 3.2.7). ($\Leftarrow$) By soundness (proposition 3.2.2), it follows that $\Gamma^{\mathrm{nf}} \vdash S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$, and, by lemma 3.2.3, it follows that $\Gamma \vdash S \leq T$. □

COROLLARY 3.2.9 Let $lub_\Gamma(S)$ be defined. Then $\Gamma \vdash S \leq T$ and $T \neq_{\beta\wedge} S$ implies $\Gamma \vdash lub_\Gamma(S) \leq T$.

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

As it stands, $NF^{\omega}_{\wedge}$ as defined in section 3.1 is not a deterministic algorithm, because its rules are not syntax directed. Fortunately, we are not far away from an algorithmic presentation. In fact, lemma 3.1.7 is the bridge to the algorithmic presentation of the subtyping relation, $AlgF^{\omega}_{\wedge}$, which states that transitivity steps can be eliminated and reflexivity steps can be simplified. $AlgF^{\omega}_{\wedge}$ is obtained from $NF^{\omega}_{\wedge}$ by removing NS-Trans and restricting NS-Refl to type variables and type applications.

Definition 3.3.1 We define the algorithmic system $AlgF^{\omega}_{\wedge}$ from $NF^{\omega}_{\wedge}$, by removing NS-Trans, and replacing NS-Refl by

$$\frac{\Gamma \vdash X \in K}{\Gamma \vdash_{Alg} X \leq X} \text{ (AS-TVarRefl)} \qquad \frac{\Gamma \vdash T\,S \in K}{\Gamma \vdash_{Alg} T\,S \leq T\,S} \text{ (AS-OAppRefl)}$$

Lemma 3.3.2 (*Equivalence of normal and algorithmic subtyping*)
Let $\Gamma \vdash S, T \in K$. Then $\Gamma \vdash_n S \leq T$ if and only if $\Gamma \vdash_{Alg} S \leq T$.

Proof: ($\Rightarrow$) By lemma 3.1.7. ($\Leftarrow$) Immediate. □

We have thereby proved that $AlgF^{\omega}_{\wedge}$ is indeed a sound and complete algorithm for the $F^{\omega}_{\wedge}$subtyping relation. We conclude the proof of decidability of subtyping in $F^{\omega}_{\wedge}$ by establishing in section 4 that $AlgF^{\omega}_{\wedge}$ always terminates.

Proposition 3.3.3 (*Equivalence of ordinary and algorithmic subtyping*)
Let $\Gamma \vdash S \in K$ and $\Gamma \vdash T \in K$. Then $\Gamma \vdash S \leq T$ if and only if $\Gamma^{\mathrm{nf}} \vdash_{Alg} S^{\mathrm{nf}} \leq T^{\mathrm{nf}}$.

Proof: By the equivalence of ordinary and normal subtyping (theorem 3.2.8) and the equivalence of normal and algorithmic subtyping (lemma 3.3.2). □

4 Termination of subtype checking

The last step in proving the decidability of the subtyping relation of $F^{\omega}_{\wedge}$ is proving the termination or well-foundedness of the relation defined by the $AlgF^{\omega}_{\wedge}$ subtyping rules. We show this by reducing the well-foundedness of $AlgF^{\omega}_{\wedge}$ to the strong normalization property of the $\rightarrow_{\beta\wedge+}$ relation.

We begin by extending the language of types with the constructor $+$ ($T+T'$). The reduction $\rightarrow_{\beta\wedge+}$ is obtained from $\rightarrow_{\beta\wedge}$ by adding the reductions associated with the choice operator $+$, $S + T \rightarrow_{\beta\wedge+} S$ and $S + T \rightarrow_{\beta\wedge+} T$. We also need to extend our kinding judgements (section 2.1.1) with a rule saying that $\Gamma \vdash_+ S + T \in K$ whenever $\Gamma \vdash_+ S, T \in K$. As far as we are aware, choice operators have not been used before to analyze subtyping.
As usual, $\rightarrow_{\beta\wedge+}$ is extended to become a compatible relation with respect to type formation, $\twoheadrightarrow_{\beta\wedge+}$ is the reflexive, transitive closure of $\rightarrow_{\beta\wedge+}$, and $=_{\beta\wedge+}$ is the least equivalence relation containing $\rightarrow_{\beta\wedge+}$. If $S \rightarrow_{\beta\wedge+} T$ in at least one step, we write $S \rightarrow_{\beta\wedge+}{}^{>0} T$.

PROPOSITION 4.1 (*Strong normalization for* $\rightarrow_{\beta\wedge+}$) If $\Gamma \vdash_+ T \in K$, then every $\beta\wedge+$-reduction sequence starting from T is finite.

Next, we define a measure for subtyping statements such that, for any subtyping rule, the measure of each hypothesis is smaller than that of the conclusion. Most measures for showing the well-foundedness of a relation defined by a set of inference rules involve a clever assignment of weights to judgements, often involving the number of symbols. We need a more sophisticated measure, since in NS-OAPP it is not necessarily the case that the size of the hypothesis is smaller than the size of the conclusion.

We introduce a new mapping from types to types in the extended language in order to define a new measure on subtyping statements. To motivate the definition of this new measure, we analyze the behavior of type variables during subtype checking. Assume that we want to check if $\Gamma \vdash_{Alg} S \leq T$, where S is a variable or a type application. It can be the case that the judgement is obtained with an application of NS-TVAR or NS-OAPP, in which case we have to consider a new statement $\Gamma \vdash_{Alg} S' \leq T$, where S' is obtained from S by replacing a variable by its bound (and possibly normalizing). However, we do not replace every variable by its bound, as this would constitute an unsound operation with respect to subtyping. This fact is illustrated in the following example.

EXAMPLE 4.2 Two unrelated variables may have the same bound.

$$X{\leq}\top^\star{:}\star,\, Y{\leq}\top^\star{:}\star \not\vdash X \leq Y, \text{ but } X{\leq}\top^\star{:}\star,\, Y{\leq}\top^\star{:}\star \vdash \top^\star \leq \top^\star.$$

Our new mapping, *plus*, includes in each type expression this nondeterministic behavior of its type variables.

DEFINITION 4.3 (*plus*)

1. $plus_{\Gamma_1,\, X\leq T:K\,,\,\Gamma_2}(X) = X + plus_{\Gamma_1}(T)$,
2. $plus_\Gamma(T{\rightarrow}S) = plus_\Gamma(T){\rightarrow}plus_\Gamma(S)$,
3. $plus_\Gamma(\forall X{\leq}T{:}K.S) = \forall X{\leq}plus_\Gamma(T){:}K.plus_{\Gamma,X\leq T:K}(S)$,
4. $plus_\Gamma(\Lambda X{:}K.S) = \Lambda X{:}K.plus_{\Gamma,X:K}(S)$,
5. $plus_\Gamma(S\,T) = plus_\Gamma(S)\,plus_\Gamma(T)$,
6. $plus_\Gamma(\bigwedge^K[S_1..S_n]) = \bigwedge^K[plus_\Gamma(S_1)..plus_\Gamma(S_n)]$.

EXAMPLE 4.4 $\quad plus_{X\leq\top^\star:\star,\, Y\leq X:\star,\, Z\leq Y:\star}(Z) = Z + Y + X + \top^\star$.

We need to show that *plus* is well defined on well-kinded arguments.

DEFINITION 4.5 (*Size*) The size of a type expression T, $size_t(T)$, is defined as follows.

- $size_t(X) = 1$,
- $size_t(\Lambda X{:}K.T) = size_t(\forall X{\leq}S{:}K.T) = size_t(T) + 1$,

- $size_t(S{\to}T) = size_t(S\,T) = size_t(S) + size_t(T) + 1$,
- $size_t(\bigwedge^K [T_1..T_n]) = 1 + \Sigma_{1\leq i\leq n} size_t(T_i)$.

LEMMA 4.6 (*Well-foundedness of plus*) If $\Gamma \vdash T \in K$, then $plus_\Gamma(T)$ is defined.

PROOF: The height of the derivation of the kinding judgements of the arguments strictly decreases in each recursive call. □

LEMMA 4.7 If $\Gamma \vdash T \in K$, then $\Gamma \vdash_+ plus_\Gamma(T) \in K$.

LEMMA 4.8 (*Strengthening for plus*) 1. Let $X \notin \mathrm{FTV}(\Gamma_2) \cup \mathrm{FTV}(S)$. Then $\Gamma_1, X{\leq}T_X{:}K_X, \Gamma_2 \vdash S \in K$ implies $plus_{\Gamma_1, X\leq T_X:K_X, \Gamma_2}(S) = plus_{\Gamma_1,\Gamma_2}(S)$.

2. $\Gamma_1, x{:}T, \Gamma_2 \vdash S \in K$ implies $plus_{\Gamma_1, x:T, \Gamma_2}(S) = plus_{\Gamma_1,\Gamma_2}(S)$.

LEMMA 4.9 (*Weakening for plus*) If $\Gamma' \vdash$ ok, $\Gamma \subseteq \Gamma'$, and $\Gamma \vdash S \in K$, then $plus_\Gamma(S) = plus_{\Gamma'}(S)$.

LEMMA 4.10 (*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 $\Gamma_1, X{\leq}U'{:}K, \Gamma_2 \vdash \Sigma$.

LEMMA 4.11 (*Substitution for plus*) Let $\Gamma_1, X{\leq}S{:}K_1, \Gamma_2 \vdash T_2 \in K_2$ and $\Gamma_1 \vdash T_1 \in K_1$. Then
$plus_{\Gamma_1, X\leq S:K_1, \Gamma_2}(T_2)[X{\leftarrow}plus_{\Gamma_1}(T_1)] \twoheadrightarrow_{\beta\wedge+} plus_{\Gamma_1,\Gamma_2[X\leftarrow T_1]}(T_2[X{\leftarrow}T_1])$.

LEMMA 4.12 (*Monotonicity of plus with respect to* $\to_{\beta\wedge}$) Let $\Gamma \vdash T \in K$. Then

1. $\Gamma \to_{\beta\wedge} \Gamma'$ implies $plus_\Gamma(T) \twoheadrightarrow_{\beta\wedge+} plus_{\Gamma'}(T)$.
2. $T \to_{\beta\wedge} T'$ implies $plus_\Gamma(T) \twoheadrightarrow_{\beta\wedge+}{}^{>0} plus_\Gamma(T')$.

LEMMA 4.13 Let $lub_\Gamma(S)$ be defined and $\Gamma \vdash S \in K$. Then $plus_\Gamma(S) \twoheadrightarrow_+{}^{>0}$ $plus_\Gamma(lub_\Gamma(S))$.

Our measure to show the well-foundedness of $AlgF^\omega_\wedge$ considers the $\beta\wedge+$-reduction paths of the *plus* versions of the types in the subtyping judgements. As we mentioned before, in NS-TVAR and NS-OAPP the types appearing in the hypothesis may be larger than those in their conclusions. Therefore, the well foundedness of the $AlgF^\omega_\wedge$ relation is not immediate. The next corollary gathers the previous results to serve our purposes.

COROLLARY 4.14 1. If $\Gamma \vdash X \in K$, then $plus_\Gamma(X) \twoheadrightarrow_{\beta\wedge+}{}^{>0} plus_\Gamma(\Gamma(X))$.

2. If $\Gamma \vdash A\,T \in K$ then $plus_\Gamma(A\,T) \twoheadrightarrow_{\beta\wedge+}{}^{>0} plus_\Gamma(lub_\Gamma(A\,T)^{\mathrm{nf}})$.

PROOF: Item 1 is a particular case of the previous lemma (lemma 4.13), and item 2 is a consequence of lemma 4.13 and the monotonicity of *plus* with respect to $\to_{\beta\wedge+}$ (lemma 4.12(2)). □

Finally, we can define our measure.

DEFINITION 4.15 (*Weight*)

1. $\mathit{weight}(\Gamma \vdash_{Alg} S \leq T) =$
   $<\text{max-red}(\mathit{plus}_\Gamma(S)) + \text{max-red}(\mathit{plus}_\Gamma(T)), \mathit{size}_t(S) + \mathit{size}_t(T)>$,
2. $\mathit{weight}(\Gamma \vdash T \in K) = <0, 0>$,

where max-red(S) is the length of a maximal $\beta\wedge+$-reduction path starting from S, and size_j is defined in definition 4.5.

Pairs are ordered lexicographically. Note that $<0, 0>$ is the least *weight*.

PROPOSITION 4.16 (*Well-foundedness of* $AlgF^\omega_\wedge$)
If $\dfrac{J_1 \,..\, J_n}{J}$ is an $AlgF^\omega_\wedge$ rule, then $\mathit{weight}(J_i) < \mathit{weight}(J)$, for each $i \in \{1..n\}$.

Finally, we can state our main result.

THEOREM 4.17 (*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 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.

References

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

[2] 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.

[3] 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.

[4] 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.

[5] Luca Cardelli. Notes about $F^\omega_{<:}$. Unpublished manuscript, October 1990.

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

[7] Felice Cardone and Mario Coppo. Two extensions of Curry's type inference system. In Odifreddi [20], pages 19–76.

[8] 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.

[9] Adriana Compagnoni. *Higher-Order Subtyping with Intersection Types*. PhD thesis, University of Nijmegen, The Netherlands, January 1995.

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

[11] Adriana B. Compagnoni and Benjamin C. Pierce. Multiple inheritance via intersection types. *Mathematical Structures in Computer Science*, 1995. To appear. Preliminary version available as University of Edinburgh technical report ECS-LFCS-93-275 and Catholic University Nijmegen computer science technical report 93-18, Aug. 1993.

[12] 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. Also in [18].

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

[14] 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. Also in [18].

[15] 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.

[16] Giorgio Ghelli, January 1994. Message to the `Types` mailing list.

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

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

[19] 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. Also in in [18].

[20] Piergiorgio Odifreddi, editor. *Logic and Computer Science*. Number 31 in APIC Studies in Data Processing. Academic Press, 1990.

[21] 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.

[22] Benjamin C. Pierce and David N. Turner. Simple type-theoretic foundations for object-oriented programming. *Journal of Functional Programming*, 4(2):207–247, April 1994.

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