Structural Decidable Extensions of Bounded Quantification

Sergei Vorobyov

MPI-I-94-257          October 1994
Publication Notes


Acknowledgements

I am greatly indebted to Luca Cardelli, Benjamin Pierce, Martín Abadi, Roberto Amadio, Philippe de Groote, Didier Galmiche, Jean-Luc Rémy, Hubert Comon, Michel Parigot for invaluable remarks, ideas, and discussions. I am grateful to my anonymous referee for his substantial help. To produce proof trees I used Paul Taylor’s \LaTeX macro package. This work was mainly done when I was at CRIN (Centre National de Recherche en Informatique de Nancy, France), which provided me the excellent research opportunities.
Abstract

We show how the subtype relation of the well-known system $F\leq$, the second-order polymorphic $\lambda$-calculus with bounded universal type quantification and subtyping, due to Cardelli, Wegner, Bruce, Longo, Curien, Ghelli, proved undecidable by Pierce (POPL'92), can be interpreted in the (weak) monadic second-order theory of one (Büchi), two (Rabin), several, or infinitely many successor functions. These $(W)SnS$-interpretations show that the undecidable system $F_{\text{sub}}$ possesses consistent decidable extensions, i.e., $F_{\text{sub}}$ is not essentially undecidable (Tarski, 1949).

We demonstrate an infinite class of structural decidable extensions of $F\leq$, which combine traditional subtype inference rules with the above $(W)SnS$-interpretations. All these extensions, which we call systems $F^{SnS}_{\leq}$, are still more powerful than $F\leq$, but less coarse than the direct $(W)SnS$-interpretations.

The main distinctive features of the systems $F^{SnS}_{\leq}$ are: 1) decidability, 2) closure w.r.t. transitivity; 3) structuredness, e.g., they never subtype a functional type to a universal one or vice versa, 4) they all contain the powerful rule for subtyping boundedly quantified types.
Abstract

We show how the subtype relation of the well-known system $F_\leq$, the second-order polymorphic $\lambda$-calculus with bounded universal type quantification and subtyping, due to Cardelli, Wegner, Bruce, Longo, Curien, Ghelli [6, 2, 8], proved undecidable by Pierce [12], can be interpreted in the (weak) monadic second-order theory of one (Büchi), two (Rabin), several, or infinitely many successor functions [13, 14]. These $(W)\text{SnS}$-interpretations show that the undecidable system $F_\leq$ possesses consistent decidable extensions, i.e., $F_\leq$ is not essentially undecidable (Tarski et. al., 1949, [17]).

We demonstrate an infinite class of "structural" decidable extensions of $F_\leq$, which combine traditional subtype inference rules with the above $(W)\text{SnS}$-interpretations. All these extensions, which we call systems $F_\leq^{\text{SnS}}$, are still more powerful than $F_\leq$, but less coarse than the direct $(W)\text{SnS}$-interpretations:

$$F_\leq \subset F_\leq^{\text{SnS}} \subset (W)\text{SnS-interpreta}$$

The main distinctive features of the systems $F_\leq^{\text{SnS}}$ are: 1) decidability, 2) closure w.r.t. transitivity; 3) structuredness, e.g., they never subtype a functional type to a universal one or vice versa; 4) they all contain the powerful rule for subtyping boundedly quantified types:

$$\Gamma \vdash \tau_1 \leq \sigma_1, \Gamma, \alpha \leq \tau_1, \sigma_2 \leq \tau_2 \quad (\text{All})$$

Key words: second-order polymorphic typed $\lambda$-calculus, subtyping, system $F_\leq$, bounded universal type quantification, parametric and inheritance polymorphisms, (un-)decidability, essential undecidability, (weak) monadic second-order theory of several successor functions $(W)\text{SnS}$.

1 Introduction

The advantages and usefulness of strict typing disciplines in programming with static typing and rigid compile-time type control have been widely accepted, studied, and advocated in Software Engineering [10, 6, 4, 11] since creation of Simula-67, Algol-68, Pascal, Chu, Alphard, Modula, ML, Ada, etc. Typeful programming should be based on powerful and, preferably, decidable type systems.

The system $F_\leq$ is the polymorphic second-order typed $\lambda$-calculus with subtyping, combining the universal (or parametric) polymorphism of Girard’s system $F$ with Cardelli’s calculus of subtyping (inheritance polymorphism [3]). Introduced in [6], later improved, simplified, and investigated by many researchers [2, 1, 8, 12, 7, 5], the system $F_\leq$ serves a core calculus of type systems with subtyping and a model to represent polymorphic and object-oriented features in programming languages.

$F_\leq$ is an extension of $F$ with subtyping. In addition to the usual functional and universal type formation of $F$, the system $F_\leq$ allows one to form boundedly quantified types: $\nu \alpha \leq \text{bound body}$. Such type is a function on types transforming any subtype $\sigma$ of a bound into a type body[$\sigma/\alpha$].

As $F_\leq$ also contains the largest type $\top$, the unbounded type quantification of $F$ is included as a particular case: $\nu \alpha \leq \top$. The system $F_\leq$ consists of two components. The first one axiomatizes the subtyping relation on types $\Gamma \vdash \sigma \leq \tau$. The second generates the typing relation $\Gamma \vdash t : \sigma$. Both components interact by means of the rules as $\text{(Subsumption)}$, allowing one to derive the judgment $\Gamma \vdash t : \tau$ from $\Gamma \vdash t : \sigma$ and $\Gamma \vdash \sigma \leq \tau$.

In [12] Pierce proved that already the subtyping component of $F_\leq$ is undecidable, and hence the typing relation in $F_\leq$ is undecidable too. Using Ghelli’s example of divergence of $F_\leq$-subtyping algorithm (mainly due to the subtle interaction between the quantifier rule (All) above and transitivity), he succeeded to encode instances of the termination problem into $F_\leq$-subtyping judgments.

Given an undecidable theory $T$ one usually tries to weaken it to get a decidable subtheory $T_{dec} \subseteq T$. Accordingly, attempts were made to restrict $F_\leq$ to get decidable subsystems. In [7] the general quantifier rule (All) above was replaced by its weaker version:

$$\Gamma \vdash \tau_1 \leq \sigma_1, \Gamma, \alpha \leq \top \vdash \sigma_2 \leq \tau_2 \quad (\text{All-Top})$$

Subtyping in the resulting subsystem $F^{\nu \alpha \leq \text{bound body}}_{\leq}$ is decidable. In [9] a decidable subsystem of $F^{\nu \alpha \leq \text{bound body}}_{\leq}$ is obtained by
restoring bounds in bounded quantification to be \(\mathcal{F}\)-free
(with some relaxations to allow unbounded quantification).
An extensive discussion of different other weakenings of the
powerful rule (All) is contained in [7].

For an undecidable theory \(T\) there sometimes exists another
possibility, to reinforce it (instead of weakening) in order
to obtain a consistent decidable extension \(\textnormal{I}_{\text{dec}} \supseteq T\).
This works only if \(T\) is not essentially undecidable, i.e., possesses
consistent decidable extensions (A. Tarski, 1949, [17]).

Curiously enough, \(F_{\leq}\) appears to be undecidable, but not
essentially [18], with infinitely many nontrivial consistent
decidable extensions. This reopens the possibility for
obtaining good decidable systems relative to \(F_{\leq}\) without sacrificing
the general quantifier rule (All) or somehow restricting
the form of bounds in bounded quantification.

The first infinite class of such extensions was introduced in
[18], where it was shown that there exist infinitely many
ways to translate \(F_{\leq}\)-subtyping judgments into formulas of
Rabin’s \(\mathcal{S}\). Each such translation maps the \(F_{\leq}\) axioms to
valid \(\mathcal{S}\)-formulas, and each \(F_{\leq}\)-inference rule preserves
validity with respect to any \(\mathcal{S}\)-translation. It follows that
everything provable in \(\mathcal{S}\) is valid in any \(\mathcal{S}\)-interpretation.
Consequently, \(F_{\leq}\) is not essentially undecidable; any \(\mathcal{S}\)-
translation is a consistent decidable extension of \(F_{\leq}\).
\(\mathcal{S}\)-interpretations generalize for recursive types [19].

Precautions, however, should be taken concerning consistency.
For theories based on predicate calculus consistent means “do not prove everything”.
For theories, which are not based on predicate calculus, like \(F_{\leq}\), consistent might mean
“do not subtype any pair of types” (weak consistency) or “do not subtype too many types” (strong consistency).

\(\mathcal{S}\)-interpretations appeared to be weakly, but not strongly
consistent. They are coarse in the sense that they do not
make fine distinction between differently structured types,
and subtype too many of them, which is undesirable in strict
typing disciplines. In this paper we remedy this drawback by
combining our \(\mathcal{S}\)-interpretations with the traditional \(F_{\leq}\)-
like subtype inference rules. These rules guarantee the so-called “strict structural subtyping”, where the subtype relation
is defined by co(ntra)variant induction on type structure.
This prevents us from subtyping differently structured
types, e.g., universal and functional ones.

The main idea of our systems \(\mathcal{F}_{\leq}\) is that they disable
the infinite alternations of applications of the rule (All) and
the transitivity rule. This alternation is the source of non-
termination and undecidability of \(F_{\leq}\) [12]. Instead, we
prune proof tree branches, which may lead to infinite
alternations, and decide the remaining judgments by interpreting them in \(\mathcal{W}\)\(\mathcal{S}\). Of course, as \(F_{\leq}\) is undecidable,
and \(\mathcal{F}_{\leq}\) are decidable extensions of \(F_{\leq}\), sometimes they
accept \(F_{\leq}\)-unprovable judgments. But this is a reasonable
price for attaining decidability.

The scenario of the presentation is the following. Section 2
recalls the system \(F_{\leq}\). (Un)decidability results concerning
\(F_{\leq}\) are listed in Section 3. Section 4 introduces systems
\(\mathcal{F}_{\leq}\). Section 5 describes the decision procedure. In
Section 6 we show infinitely many ways to interpret the
subtype relation in any \(\mathcal{W}\)\(\mathcal{S}\). Section 7 discusses the consistency
of \(\mathcal{F}_{\leq}\). In Section 8 we explain the rule inversion
principle, the main tool of our proofs of the inclu-
sion \(F_{\leq} \subset \mathcal{F}_{\leq}\) and the transitivity of \(\mathcal{F}_{\leq}\). In Sections
9 and 10 we show that the inversion principle does not hold
for \(\mathcal{S}\)-interpretations, but holds for systems \(\mathcal{F}_{\leq}\).
In Sections 11, 12, and 13 we prove the inclusions
\(F_{\leq} \subset \mathcal{F}_{\leq} \subset (W)\mathcal{S}\) interpretations and the transitivity of all
\(\mathcal{F}_{\leq}\). Section 14 discusses further improvements of \(\mathcal{F}_{\leq}\).
In Section 15 we sketch problems for future research.
Appendices A and B contain the reference material on second-
order monadic theories and on Curien-Ghelli’s algorithmic
variant of \(F_{\leq}\). The proofs are collected in Appendix C.

In this paper we deal only with the subtyping relation.
Combinations with typing and related problems, like subject
reduction [20], typing proof normalization, the least type
property, strong normalization are in the course of study and will be
considered elsewhere.

Added in Proof. In [24] we continued the study of decidable
extensions of the \(F_{\leq}\) subtyping relation and developed the
general theory of converging hierarchies of structural decidable
extensions of the \(F_{\leq}\)-subtyping. The systems \(\mathcal{F}_{\leq}\)
presented in this paper form just the first level of the hierarchies from [21]. In [22] we combined these hierarchies
with the standard \(F_{\leq}\) term typing rules and obtained an
infinite family of the extensions of the polymorphic system
\(F_{\leq}\) where both subtyping and typing are decidable.

2 System \(F_{\leq}\)

For complete and exact reference see, e.g., [8, 12, 5].
We just briefly remind the essential definitions, retaining the
notation of [12].

Definition 2.1 (Types) The set of \(F_{\leq}\)-types is defined by
the following abstract grammar:

\[
\mathcal{T} \equiv_{df} \forall \mathcal{V} | \mathcal{T} | \mathcal{T} \rightarrow \mathcal{T} | \forall \mathcal{V} \mathcal{\subseteq} \mathcal{T}.
\]

where:

1. \(\mathcal{V}\) is a set of type variables denoted by Greek letters \(\alpha, \beta, \gamma\);
2. \(\mathcal{T}\) is the largest type majorizing any other type, \(\sigma \leq \mathcal{T}\);
3. \(\mathcal{T} \rightarrow \mathcal{T}\) is the functional type constructor, \(\sigma \rightarrow \tau\) is the
type of functions with domain of type \(\sigma\) and codomain of type \(\tau\);
4. \(\forall \alpha \leq \rho, \tau\) is a polymorphic boundedly quantified type,
i.e., a function assigning to each subtype \(\sigma\) of \(\rho, \sigma \leq \rho\),
the type \(\tau[\sigma/\alpha]\) obtained from \(\tau\) by substituting \(\sigma\) instead
of free occurrences of \(\alpha\) (with usual non-clashing
preconditions on free variables). In \(\forall \alpha \leq \rho, \tau\) the
bound \(\rho\) does not contain \(\alpha\) free.

The terms \(\tau, \sigma, \rho\) from the end of the Greek alphabet
denote arbitrary (variable or compound) \(F_{\leq}\)-types; \(\forall \beta, \tau\) abbreviates
\(\forall \beta \leq \mathcal{T}, \tau\); \(FV(\sigma)\) denotes the set of free variables in
\(\sigma\).

Definition 2.2 (Contexts) An \(F_{\leq}\)-context is an ordered
sequence \(\alpha_1 \leq \sigma_1, \ldots, \alpha_n \leq \sigma_n\) of \(\leq\)-relations between type
variables and \(F_{\leq}\)-types such that:

1. all \(\alpha_i\) are different type variables, and
2. for each \( i \), \( \text{FV}(\sigma_i) \subseteq \{\alpha_1, \ldots, \alpha_{i-1}\} \).

Contexts are denoted by capital Greek \( \Gamma \). \( \text{Dom}(\Gamma) \) is the set of type variables appearing to the left of \( \leq \) in \( \Gamma \). We write \( \Gamma(\alpha) = \sigma \) if \( \Gamma \) contains \( \alpha \leq \sigma \) and call \( \sigma \) a bound of \( \alpha \) in \( \Gamma \).

We define \( \Gamma(\alpha) \) as \( \Gamma(\alpha) \) if the latter is not a variable, and as \( \Gamma(\Gamma(\alpha)) \) otherwise.

**Definition 2.3 (Subtyping Judgments)**

An \( \preceq \)-subtyping judgment is a figure of the form:

\[
\Gamma \vdash \sigma \leq \tau,
\]

where \( \text{FV}(\sigma) \cup \text{FV}(\tau) \subseteq \text{Dom}(\Gamma) \).

The intuitive semantics of a judgment \( \Gamma \vdash \sigma \leq \tau \) is: \( \sigma \) is a subtype of \( \tau \) provided that all \( \alpha_i \) mentioned in \( \Gamma \) are subtypes of their respective bounds \( \sigma_i \).

**Definition 2.4 (Subtyping Rules)**

The \( \preceq \)-subtyping relation is generated by the system of 3 axioms and 3 inference rules, shown in Figure 1.

\[
\begin{align*}
\Gamma & \vdash \tau \leq \tau \quad \text{(Refl)} \\
\Gamma & \vdash \tau \leq \top \\
\Gamma & \vdash \alpha \leq \Gamma(\alpha) \\
\Gamma & \vdash \eta \leq \theta_1, \Gamma & \vdash \eta \leq \theta_2 \\
\Gamma & \vdash \eta \leq \theta \\
\Gamma & \vdash \eta \leq \theta_1, \Gamma & \vdash \eta \leq \theta_2 \\
\Gamma & \vdash \eta \leq \theta_1, \Gamma, \alpha \leq \eta & \vdash \sigma_1 \leq \eta \rightarrow \theta_1 \\
\Gamma & \vdash \eta \leq \theta_2, \Gamma, \alpha \leq \eta & \vdash \sigma_2 \leq \eta \rightarrow \theta_2 \\
\Gamma & \vdash \eta \leq \theta, \Gamma, \alpha \leq \eta & \vdash \sigma_1 \rightarrow \sigma_2 \leq \eta \rightarrow \theta \\
\Gamma & \vdash \eta \leq \theta, \Gamma, \alpha \leq \eta & \vdash \sigma_1 \rightarrow \sigma_2 \leq \eta \rightarrow \theta
\end{align*}
\]

Figure 1: \( \preceq \) subtyping axioms and inference rules

Let \( \vdash_{\preceq} \) denote the least three-place relation \( \Gamma \vdash \sigma \leq \tau \) containing all particular cases of the \( \preceq \)-axioms and closed with respect to the \( \preceq \)-inference rules. Sometimes, by abusing notation, we denote by \( \preceq \) the set of subtyping judgments provable in \( \preceq \).

**Definition 2.5 (Variants of \( \preceq \))**

1) Original \( \text{Fun} \) [6] replaces the (All) rule by the weaker rule (All-Fun) (Figure 2).

2) System \( \text{Fun}^\top \) [7] replaces the rule (All) by its particular case (All-Top) (Figure 2).

3) System \( \preceq_{\text{local}} \) [7] replaces the rule (All) by its modification (All-local) (Figure 2).

By \( \vdash_{\text{Fun}}, \vdash_{\text{Fun}^\top} \) and \( \vdash_{\preceq_{\text{local}}} \) we denote the corresponding subtyping relations.

\[
\begin{align*}
\Gamma, \alpha \leq \rho & \vdash \sigma_2 \leq \theta_2 \quad \text{(All-Fun)} \\
\Gamma & \vdash (\forall \alpha \leq \rho, \sigma_2) \leq (\forall \alpha \leq \rho, \theta_2) \\
\Gamma, \alpha \leq \tau & \vdash \sigma_2 \leq \theta_2 \\
\Gamma & \vdash (\forall \alpha \leq \tau, \sigma_2) \leq (\forall \alpha \leq \tau, \theta_2) \\
\Gamma, \alpha \leq \sigma_1, \Gamma & \vdash \sigma_2 \leq \theta_2 \\
\Gamma & \vdash (\forall \alpha \leq \sigma_1, \sigma_2) \leq (\forall \alpha \leq \sigma_1, \theta_2)
\end{align*}
\]

Figure 2: Variants of the (All) rule

3 (Un)Decidability

The interesting facts about \( \preceq \) are:

**Theorem 3.1 (Undecidability of \( \preceq \))** [12] The relation \( \vdash \) is undecidable.

The weakenings of \( \preceq \) are however decidable:

**Theorem 3.2 (Decidability of \( \text{Fun} \) and \( \text{Fun}^\top \))** [7] Both relations \( \vdash_{\text{Fun}} \) and \( \vdash_{\text{Fun}^\top} \) are decidable.

Nothing is known about decidability of \( \preceq_{\text{local}} \).

In [18] we demonstrated that the decidability of \( \preceq \) could be reached also by reinforcement, and not only by weakening, as opposed to systems \( \preceq_{\text{local}} \) and \( \text{Fun} \).

**Definition 3.3 (Essential Undecidability, [17])** A consistent theory \( T \) is essentially undecidable if it has no consistent decidable extensions \( T' \supseteq T \).

**Definition 3.4 (Consistency)** An extension of \( \preceq \) is consistent if it is closed with respect to the \( \preceq \)-inference rules and does not subtype any two types.

**Remarks.**

1) Further we replace “any two types” by “any two differently structured types” getting the stronger consistency.

2) As we are interested only in the extensions of \( \preceq \), the closure with respect to the \( \preceq \)-inference rules seems natural and meaningful. It would not be the case for \( \preceq_{\text{local}} \) and \( \text{Fun} \).

**Theorem 3.5 (\( \preceq \) Is Not Essentially Undecidable, [18])**

There exist infinitely many different consistent decidable extensions of \( \vdash_{\preceq} \).

This result was obtained by interpreting the \( \preceq \)-subtyping relation in \( \text{S2S} \), the monadic second-order logic of two successors due to M. Rabin [13, 14]. The corresponding infinite class of extensions of \( \preceq \) (which we call the \( \text{S2S} \)-interpretations) and their properties are studied in [18].

The main objection (by L. Cardelli and others) against these extensions was that they were too coarse and non-structural. \( \text{S2S} \)-interpretations subtype too many types, sometimes
and differently structured ones (i.e., universal and functional ones).

In this paper we introduce a new infinite class of decidable extensions of $F_{\leq}$ refining the $S_S$ interpretations. We call these extensions systems $F_{\leq}^{S_S}$. We also (re)introduce the $S_S$-interpretations in a slightly more general setting and call them $S_n$-interpretations (with $S_2$ being a particular case of $S_n$ for $n = 2$). We prove that all systems $F_{\leq}^{S_S}$ are more powerful than $F_{\leq}$, but being structural (they do not subtype differently structured types any more), they are less coarse than $S_n$-interpretations:

$$F_{\leq} \subset F_{\leq}^{S_S} \subset S_n$$

Again note that the decidable system $F_{\leq}^S$ introduced in [7] is weaker than $F_{\leq}$: $F_{\leq}^{S_S} \subset F_{\leq}$.

4 System $F_{\leq}^{S_S}$

Definition 4.1 The system $F_{\leq}^{S_S}$ is defined by the collection of subtyping axioms and inference rules shown in Figure 3, supposed to be applied bottom-up in the order of their presentation.

*** See Figure 3 ***

The $\text{DE\textsc{CIDE}}$ component in the rule ($\text{Var-All-Decide}$) and the whole $F_{\leq}^{S_S}$-decision procedure are described in the following Sections.

Roughly speaking, the system $F_{\leq}^{S_S}$ is $F_{\leq}$ without the general transitivity rule ($\text{Trans}$) replaced by a built-in decision procedure $\text{DE\textsc{CIDE}}$.

Remarks and Explanations

1. Our intention is to define the decision and not semidecision procedure for subtyping judgments. That is why we are going to apply rules bottom-up and introduce two constants $\text{TRUE}$ and $\text{FALSE}$ to treat both the accepting and rejecting cases.

2. Rules ($\text{Refl}$), ($\text{Top}$), and ($\text{TVar}$) correspond exactly to their $F_{\leq}$ counterparts. We formulate them as rules with the premises $\text{TRUE}$ just to be able to treat symmetrically the negative case $\text{FALSE}$ in other rules of $F_{\leq}^{S_S}$.

3. Rules ($\text{Arrow}$) and ($\text{All}$) are the same as in $F_{\leq}$.

4. Motivation for the rules ($\text{Top-L}$) and ($\text{TVar-R-2}$) is: the conclusions of these rules are NOT provable in $F_{\leq}$ (Proposition 4.2).

5. Motivation for the rules ($\forall \leq \rightarrow$) and ($\rightarrow \leq \forall$) is the same: the conclusions of these rules are underviable in $F_{\leq}$.

6. The ($\text{Var-Arrow}$) rule is just a half (with only arrow-types on the right of $\leq$) of Curien-Ghelli’s algorithmic transitivity rule ($\text{AlgTrans}$), see [8] and Appendix B.

7. The crucial difference with $F_{\leq}$ is the absence of the general rule ($\text{Trans}$) or of its algorithmic equivalent ($\text{AlgTrans}$) for universal types (see the rule ($\text{Var-All}$) below). Transitivity in this case is dealt separately, by means of a $\text{DE\textsc{CIDE}}$ procedure. Note that we do not weaken the general $F_{\leq}$ quantifier rule ($\text{All}$), which remains the same as in $F_{\leq}^{S_S}$.

8. The built-in procedure $\text{DE\textsc{CIDE}}$ appearing in the premise of the rule ($\text{Var-All-Decide}$) is a parameter of the system. Below we define infinitely many different such procedures. Note, in particular, that if we define the $\text{DE\textsc{CIDE}}$ procedure recursively, as $F_{\leq}^{S_S}$ plus the second half of Curien-Ghelli’s transitivity rule:

$$\Gamma \vdash \Gamma(\alpha) \leq (\forall \beta \leq \alpha, \tau)$$

then we will get exactly $F_{\leq}$!

\[ \text{(Var-All)} \]

Proposition 4.2 Subtyping judgments of the forms:

1. $\Gamma \vdash \top \leq \tau \quad (\tau \neq \top)$,

2. $\Gamma \vdash \sigma \leq \alpha \quad (\sigma \non-variable, \alpha \variable)$, where $\Gamma$ is any context, are not provable in $F_{\leq}$.

\[ \text{Proof} \] See Appendix C.1.

5 Decision Procedure

The rules of the system $F_{\leq}^{S_S}$ read bottom-up can be seen as a decision procedure (with a built-in $\text{DE\textsc{CIDE}}$ oracle). Given a subtyping judgment, the rules of $F_{\leq}^{S_S}$ apply deterministically in ordered manner (e.g., ($\text{Var-All-Decide}$) does not apply before ($\text{Var-All-2}$)). The rule application process always terminates, provided that the built-in $\text{DE\textsc{CIDE}}$ procedure is finitely terminating, and this is the fundamental difference with $F_{\leq}$, see [12].

Proposition 5.1 (Finite Termination of $F_{\leq}^{S_S}$) For every subtyping judgment $\Gamma \vdash \sigma \leq \tau$ any $F_{\leq}^{S_S}$-proof tree is finite.

\[ \text{Proof} \] The complexity of judgments decreases as one moves bottom-up. So the termination of the whole decision procedure depends on termination of its $\text{DE\textsc{CIDE}}$ component.

Irreducible leaves of $F_{\leq}^{S_S}$-proof trees are either:

1. $\text{TRUE}$ or

2. $\text{FALSE}$ or

3. of the form $\text{DE\textsc{CIDE}}(J)$, where $J$ is a subtyping judgment in the $F_{\leq}^{S_S}$-normal form, i.e.:

$$J \equiv \forall \alpha_1 \leq \sigma_1 \ldots \alpha_n \leq \sigma_n \vdash \beta \leq \tau.$$ (3)

where $\alpha_1, \ldots, \alpha_n, \beta$ are type variables, $\sigma_1, \ldots, \sigma_n$ are arbitrary types, and $\tau$ is a universal type.
\[
\begin{align*}
\text{TRUE} & \quad \frac{}{\Gamma \vdash \sigma \leq \sigma} \\
\text{TRUE} & \quad \frac{}{\Gamma \vdash \sigma \leq \top} \\
\text{FALSE} & \quad \frac{}{\Gamma \vdash \top \leq \tau \quad (\text{for } \tau \neq \top)} \\
\Gamma & \quad \frac{\Gamma \vdash \Gamma(\beta) \leq \alpha \quad \text{(for different variables } \alpha, \beta)}{\Gamma \vdash \beta \leq \alpha} \\
\text{FALSE} & \quad \frac{}{\Gamma \vdash \sigma \leq \alpha \quad (\sigma \text{ non-variable, } \alpha \text{ variable})} \\
\text{FALSE} & \quad \frac{}{\Gamma \vdash (\forall \alpha \leq \sigma_1 . \sigma_2) \leq (\tau_1 \rightarrow \tau_2)} \\
\text{FALSE} & \quad \frac{}{\Gamma \vdash (\sigma_1 \rightarrow \sigma_2) \leq (\forall \alpha \leq \tau_1 : \tau_2)} \\
\Gamma & \quad \frac{\Gamma \vdash \Gamma(\alpha) \leq \sigma \rightarrow \tau}{\Gamma \vdash \alpha \leq \sigma \rightarrow \tau} \\
\text{TRUE} & \quad \frac{}{\Gamma \vdash \alpha \leq \Gamma(\alpha)} \\
\Gamma & \quad \frac{\Gamma \vdash \Gamma(\alpha) \leq (\forall \beta \leq \sigma . \tau) \quad (\text{if } \Gamma(\alpha) \text{ is a variable})}{\Gamma \vdash \alpha \leq (\forall \beta \leq \sigma . \tau)} \\
\Gamma & \quad \frac{\Gamma \vdash \text{FALSE} \quad (\text{if } \Gamma(\alpha) \text{ is } \top \text{ or an } \rightarrow \text{-type})}{\Gamma \vdash \alpha \leq (\forall \beta \leq \sigma . \tau)} \\
\text{DECLIDE(} & \quad \frac{}{\Gamma \vdash \alpha \leq (\forall \beta \leq \sigma . \tau)} \\
\Gamma & \quad \frac{\Gamma \vdash \tau_1 \leq \sigma_1 \quad \Gamma \vdash \sigma_2 \leq \tau_2}{\Gamma \vdash \sigma_1 \rightarrow \sigma_2 \leq \tau_1 \rightarrow \tau_2} \\
\Gamma & \quad \frac{\Gamma \vdash \tau_1 \leq \sigma_1 \quad \Gamma, \alpha \leq \tau_1 \vdash \sigma_2 \leq \tau_2}{\Gamma \vdash (\forall \alpha \leq \sigma_1 : \sigma_2) \leq (\forall \alpha \leq \tau_1 : \tau_2)} \quad (\text{All})
\end{align*}
\]

Figure 3: System $\mathcal{F}_{\leq}^\text{SnS}$
In introduction an infinite class of direct interpretations of \( F_{\subseteq} \) into \( \mathbb{S} \mathbb{S} \), the monadic second-order arithmetic of two successor functions \([13, 14]\). These direct interpretations do not use any inference rules (as opposed to \( F_{\subseteq} \) or \( F_{\subseteq}^{snS} \)), immediately translating \( F_{\subseteq} \) judgments into \( \mathbb{S} \mathbb{S} \)-formulas. Like this we established that \( F_{\subseteq} \) is not essentially decidable, i.e., possesses consistent decidable extensions; any \( \mathbb{S} \mathbb{S} \)-translation of \( F_{\subseteq} \) satisfying the above properties is such an extension.

Therefore, instead of remaining in the undecidable \( F_{\subseteq} \) we forget it and work in the decidable \( F_{\subseteq}^{snS} \), which replaces the transitivity rule \( Trans \) by the transitivity implicitly present in a monadic second-order theory. As we show below, the proper choices of the \( DECIDE \) component lead to decidable extensions of \( F_{\subseteq} \) (Theorem 11.1), closed with respect to transitivity (Theorem 12.1).

### 6 Interpreting \( F_{\subseteq}^{snS} \)-Normal Forms in \( \mathbb{S} \mathbb{S} \)

In \([18]\) we introduced an infinite class of direct interpretations of \( F_{\subseteq} \) into \( \mathbb{S} \mathbb{S} \), the monadic second-order arithmetic of two successor functions \([13, 14]\). These direct \( \mathbb{S} \mathbb{S} \)-interpretations do not use any inference rules (as opposed to \( F_{\subseteq} \) or \( F_{\subseteq}^{snS} \)), immediately translating \( F_{\subseteq} \) judgments into \( \mathbb{S} \mathbb{S} \)-formulas. Like this we established that \( F_{\subseteq} \) possesses infinitely many different consistent decidable extensions, i.e., is not essentially undecidable.

The drawback of the direct \( \mathbb{S} \mathbb{S} \)-interpretations of \( F_{\subseteq} \) is that they subtype too many types (see \([18]\) and below), in particular, differently structured types. The systems \( F_{\subseteq}^{snS} \) are more subtle. By their very definition they do not subtype differently structured types. They cannot prove a subtyping between, say, an \( \rightarrow \)-type and a \( \forall \)-type. The systems \( F_{\subseteq}^{snS} \) apply the method of interpretations only to normal forms, i.e., to judgments of the form (3) inside the \( DECIDE \) procedure.

There is only a minor difference in defining the \( \mathbb{S} \mathbb{S} \)-interpretations only for normal forms (3) and for general \( F_{\subseteq} \)-subtyping judgments, so we give a complete definition of \( \mathbb{S} \mathbb{S} \)-interpretations of \( F_{\subseteq} \). Also, \( \mathbb{S} \mathbb{S} \)-interpretations generalize straightforwardly to \( \mathbb{S} \mathbb{S} \)-interpretations for arbitrary \( n \in \mathbb{N} \) or even \( \mathbb{S} \mathbb{S} \).

Choose and fix any monadic second-order theory of successor function(s), say, Büchi arithmetic \( \mathbb{S} \mathbb{S} \mathbb{S} \), Rabin’s arithmetic \( \mathbb{S} \mathbb{S} \mathbb{S} \), \( \mathbb{S} \mathbb{S} \mathbb{S} \), or their weak counterparts, with second-order quantifications restricted to finite sets (see Appendix A).

The intuition behind interpretations of \( F_{\subseteq} \) into \( \mathbb{S} \mathbb{S} \) is extremely simple. We interpret the \( F_{\subseteq} \) types as propositions of \( \mathbb{S} \mathbb{S} \). Each \( F_{\subseteq} \)-type \( \sigma \) is assigned a \( \mathbb{S} \mathbb{S} \)-formula \( S(x) \) with just one free object variable \( x \), and each subtyping relation \( \sigma \leq \tau \) is translated into \( \forall x(S(x) \supseteq T(x)) \), where \( S(x) \) and \( T(x) \) are \( \mathbb{S} \mathbb{S} \)-formulas assigned to types \( \sigma \) and \( \tau \).

Our translation satisfies the following properties:

1. all axioms of \( F_{\subseteq} \) are transformed into valid formulas of \( \mathbb{S} \mathbb{S} \);
2. all \( F_{\subseteq} \)-inference rules preserve validity with respect to any \( \mathbb{S} \mathbb{S} \), i.e., whenever both premises of a rule are translated into valid \( \mathbb{S} \mathbb{S} \)-formulas, then the conclusion of the rule is also translated into such formula.
3. consequently, by 1 and 2, any \( F_{\subseteq} \)-subtyping judgment is interpreted as a true formula of \( \mathbb{S} \mathbb{S} \), and, henceforth, \( F_{\subseteq} \) is not essentially decidable, i.e., possesses consistent decidable extensions; any \( \mathbb{S} \mathbb{S} \)-translation of \( F_{\subseteq} \) satisfying the above properties is such an extension.

It remains to show that the needed \( \mathbb{S} \mathbb{S} \)-translations of \( F_{\subseteq} \) with the above properties exist. We show it in the rest of this Section. The idea is quite simple: interpret type variables \( \alpha, \beta, \ldots \) as corresponding \( \mathbb{S} \mathbb{S} \)-atomic formulas \( A(x), B(x), \ldots \), choosing a new predicate variable for each new type variable. Then knowing that \( S(x) \) and \( T(x) \) interpret \( \sigma \) and \( \tau \) respectively, interpret:

- \( \sigma \rightarrow \tau \) as \( S(x) \supseteq T(x) \), or, more generally, as

  \[
  S(x) \supseteq T(f(x)),
  \]

- \( \forall x \leq \sigma, \tau \) as \( \forall^2 A \{ \forall^1 x[A(x) \supseteq S(x)] \supseteq T(x) \} \), or, more generally, as

  \[
  \forall^2 A \{ \forall^1 x[A(x) \supseteq S(x)] \supseteq T(g(x)) \},
  \]

where \( f, g \) are arbitrary strings composed of \( \mathbb{S} \mathbb{S} \)-successors.

Introduction of parameters \( f \) and \( g \) allows us to define infinitely many different interpretations of \( F_{\subseteq} \) in \( \mathbb{S} \mathbb{S} \), see \([18]\). Surprising, but it works! We now proceed to formal definitions.

### Definition 6.1 (\( \mathbb{S} \mathbb{S} \)\( F_{\subseteq} \)\( f, g \) -interpretations ) Let \( f \) and \( g \) be two arbitrary strings composed of successor function symbols of \( \mathbb{S} \mathbb{S} \). Both may be equal to the empty string \( \varepsilon \).

For an arbitrary type \( \rho \) of \( F_{\subseteq} \), the Types-As-Propositions-Interpretation of \( \rho \) in \( \mathbb{S} \mathbb{S} \) with parameters \( f \) and \( g \) (the \( \mathbb{S} \mathbb{S} \)\( [F_{\subseteq}]\)\( f, g \) -interpretation for short) is defined as an \( \mathbb{S} \mathbb{S} \)-formula \( \llbracket \rho \rrbracket^{fg}_l(x) \) with unique distinguished free object variable \( x \) by induction on the structure of \( \rho \):

Obviously:

- if all leaves of a \( F_{\subseteq}^{snS} \)-proof tree are \( TRUE \), we declare the input judgment valid;
- if one of the leaves of \( F_{\subseteq}^{snS} \)-proof tree is \( FALSE \), we declare the input judgment invalid;
- otherwise, before announcing our verdict we analyze \( F_{\subseteq}^{snS} \)-normal forms (3) using the built-in \( DECIDE \) procedure.
1. \[ \llbracket \alpha \rrbracket \Gamma g (x) \equiv_{df} A(x) \quad \text{(a new predicate variable A for each type variable \( \alpha \))} \]

2. \[ \llbracket \top \rrbracket \Gamma g (x) \equiv_{df} x = x; \]

3. \[ \llbracket \sigma \rightarrow \tau \rrbracket \Gamma g (x) \equiv_{df} \llbracket \sigma \rrbracket \Gamma g (x) \supset \llbracket \tau \rrbracket \Gamma g (f(x)); \]

4. \[ \llbracket \forall \alpha \leq \tau \rrbracket \Gamma g (x) \equiv_{df} \forall^2 A \{ \forall^1 x \left( A(x) \supset \llbracket \sigma \rrbracket \Gamma g (x) \right) \supset \llbracket \tau \rrbracket \Gamma g (g(x)) \}. \]

The SnS\([F_{\leq}](f, g)\)-interpretation is extended to all subtyping judgments by:

5. \[ \llbracket \sigma \leq \tau \rrbracket \Gamma g \equiv_{df} \forall^1 x (\llbracket \sigma \rrbracket \Gamma g (x) \supset \llbracket \tau \rrbracket \Gamma g (x)); \]

6. \[ \llbracket \alpha_1 \leq \alpha_2 \ldots \leq \alpha_n \leq \alpha \rrbracket \Gamma g \equiv_{df} \llbracket \alpha_1 \rrbracket \Gamma g \ldots \llbracket \alpha_n \rrbracket \Gamma g \llbracket \text{SnS} \llbracket \sigma \leq \tau \rrbracket \Gamma g . \]

**Definition 6.2 (Theory)** Define the SnS\([F_{\leq}](f, g)\)-theory as:

\[ \text{SnS}\left[F_{\leq}\right](f, g) \equiv_{df} \{ \Gamma \vdash \sigma \leq \tau \mid \Gamma \vdash \sigma \leq \tau \rrbracket \Gamma g \} \]

Further we will freely say that a typing judgment is true or valid in \( \text{SnS}\left[F_{\leq}\right](f, g) \)-interpretation if it belongs to the set \( \text{SnS}\left[F_{\leq}\right](f, g) \).

**Remarks.** In \( \text{SnS}\left[F_{\leq}\right](f, g) \)-interpretation we use just one-variable restricted fragment of SnS. If \( f = g = \varepsilon \), then this fragment is also function-free (and can be seen as the propositional second-order logic). \( x \) is the only free object variable of any \( \text{SnS}\left[F_{\leq}\right](f, g) \)-interpretation of any type. Subtyping judgments are interpreted as statements about SnS-semanntical consequence relation \( \llbracket \text{SnS} \rrbracket \Gamma g \) containing no free object variables at all. Any \( \text{SnS}\left[F_{\leq}\right](f, g) \) is decidable.

The SnS-interpretations enjoy the following important properties:

**Lemma 6.3 (Embedding)** 1) All axioms of \( F_{\leq} \) are valid with respect to any \( \text{SnS}\left[F_{\leq}\right](f, g) \).

2) All inference rules of \( F_{\leq} \) preserve validity with respect to any \( \text{SnS}\left[F_{\leq}\right](f, g) \), i.e., if both premises of a rule are valid in \( \text{SnS}\left[F_{\leq}\right](f, g) \), then so is the conclusion of the rule.

**Proof.** Straightforwardly rephrasing the proof from [18].

As a direct consequence we have, [18]:

**Theorem 6.4 (On Decidable Extensions of \( F_{\leq} \))** Any \( \text{SnS}\left[F_{\leq}\right](f, g) \) is a consistent decidable theory containing all \( F_{\leq} \)-derivable subtyping judgments. Henceforth, \( F_{\leq} \) is not essentially undecidable possessing consistent decidable extensions.

**Definition 6.5 \( (F_{\leq}^{SnS}\rrbracket(f, g)) \)** Define a system \( F_{\leq}^{SnS}\rrbracket(f, g) \) as a combination of the inference rules from Figure 3 and a DECIDE procedure for \( \text{SnS}\left[F_{\leq}\right](f, g) \).

Below, in Theorems 11.1 and 12.1 we show that all systems \( F_{\leq}^{SnS}\rrbracket(f, g) \) also extend \( F_{\leq} \) but are less coarse than \( \text{SnS}\)-interpretations, i.e.,

\[ F_{\leq} \subset F_{\leq}^{SnS}\rrbracket(f, g) \subset \text{SnS}\left[F_{\leq}\right](f, g) \quad (4) \]

7 Consistency and Well-Structuredness of \( F_{\leq}^{SnS}\rrbracket(f, g) \)

**Proposition 7.1** All systems \( F_{\leq}^{SnS}\rrbracket(f, g) \) are consistent: they do not prove, e.g., \( \vdash T \leq \neg (T \rightarrow T) \). Neither do they subtype any pair of differently structured types.

**Proof.** Immediate by definition.

8 Inversion Principle

The main tool of the proofs of inclusions (4) (Theorems 11.1 and 12.1) and of the transitivity of \( F_{\leq}^{SnS}\rrbracket(f, g) \) (Theorem 13.1) is the well-known inversion principle. The rule invertibility is the fundamental principle of the cut-free Gentzen-type derivation systems, see, e.g., [15].

The inversion principle is the key property needed to prove the minimal typing property for \( F_{\leq} \). In fact, this is almost all what is needed to reconstruct \( F_{\leq} \)-inferences into normal forms, [8].

The inversion principle can be formulated as follows: for an inference rule of a system \( S \)

\[ \Gamma \vdash \Phi \quad \Gamma \vdash \Psi \quad (\text{Rule}) \]

if a sequent \( \Gamma \vdash \Theta \) from the conclusion is derivable in \( S \) then the premises are also derivable in \( S \).

The inversion principle is important for goal-oriented proof-search procedures, which are guaranteed to be complete just stupidly applying inference rules bottom-up. Proofs in systems satisfying the inversion principle are direct, constructed from subproofs of subformulas of goal formulas, do not contain insights and roundabout ways.

The inversion principle is not evident, or even fails for systems with the CUT rule:

\[ \Gamma \vdash \Phi \quad \Gamma \vdash \Psi \quad (\text{Rule}) \]

\[ \Gamma \vdash \Phi \quad \Gamma \vdash \Psi \quad (\text{Cut}) \]

In the presence of (Cut), one cannot always be sure that a provable formula \( \Theta \) of the form \( A \supset B \) is obtained by some (Rule) or by the (Cut). But applying (Cut) requires ingenuity to find intermediate formulas \( C \), unattainable for mechanic theorem provers.
Note that the usual transitivity rule of $F_{\leq}$
\[
\frac{\Gamma \vdash \tau_1 \leq \tau_2 \quad \Gamma \vdash \tau_2 \leq \tau_3}{\Gamma \vdash \tau_1 \leq \tau_3} \tag{Trans}
\]
has the definite (Cut) form.

Proposition 8.1 (Inversion for $F_{\leq}$ [8]) In $F_{\leq}$ the rules
(Arrow) and (All) are invertible. □

This may be seen as a good structural property.

9 Failure of the Inversion Principle for $\text{SnS}[F_{\leq}](f, g)$

The inversion principle fails for $\text{SnS}$-interpretations. In
fact, we can have
\[
\frac{\Gamma \vdash (\sigma \to \tau) \leq (\sigma' \to \tau')}{\Gamma \vdash (f \; \text{has type} \; \sigma \to \tau \; \text{in} \; F_{\leq})}
\]
\[
\text{WITHOUT having}
\frac{\Gamma \vdash \sigma' \leq \sigma}{\text{and} \quad \frac{\Gamma \vdash \tau \leq \tau'}{f}}
\]
Take, for example, the judgment
\[
\alpha \leq \beta \vdash (\alpha \to \beta) \leq (\alpha \to \beta)
\]
with the valid $\text{SnS}$-translation, but the $\text{SnS}$-translation of
\[
\alpha \leq \beta \vdash T \leq \alpha
\]
is false: $\forall x(Ax \to x = x) \not\vdash \forall x(x = x \to Ax)$.

10 Inversion Principle for $F_{\leq}^{SnS}$

Inversion principle trivially holds for $F_{\leq}^{SnS}$:

Lemma 10.1 (Inversion Principle) In any $F_{\leq}^{SnS}(f, g)$:

- if $\Gamma \vdash \sigma_1 \to \sigma_2 \leq \tau_1 \to \tau_2$ is provable, then
  $\Gamma \vdash \tau_1 \leq \sigma_1$ and $\Gamma \vdash \sigma_2 \leq \tau_2$ are also provable;

- if $\Gamma \vdash (\forall x \leq \sigma_1 \to \sigma_2) \leq (\forall x \leq \tau_1 \to \tau_2)$ is provable, then $\Gamma \vdash \tau_1 \leq \sigma_1$ and $\Gamma \vdash \tau_2 \leq \sigma_2$ are also provable. □

Proof. Immediate by definition. In $F_{\leq}^{SnS}$ there are no
other ways to subtype two $\to$ or $V$-types except applying
(Arrow) or (All) (or by the (Ref)), in which case the conclusion is straightforward. □

The proofs in $F_{\leq}^{SnS}$ are direct, one needs not subtype anything which do not belong to a goal subtyping judgment, proofs are conducted without roundabout ways and insights, completely deterministically.

11 $F_{\leq}^{SnS}$ is More Powerful than $F_{\leq}$

Now we prove two strict inclusions:
\[
F_{\leq} \subset F_{\leq}^{SnS}(f, g) \subset \text{SnS}[F_{\leq}](f, g)
\]
So, the systems $F_{\leq}^{SnS}$ occupy an intermediate position be-
 tween $F_{\leq}$ and $\text{SnS}$-interpretations: they are more stron
ger than $F_{\leq}$ and more subtle than $\text{SnS}$-interpretations. Note
that the decidable system $F_{\leq}^{SnS}$ lies to the left of $F_{\leq}$ in
the above diagram.

Remark. $F_{\leq}^{SnS}$ is an infinite family of systems. To decide
normal forms each system uses a parametric $\text{SnS}[F_{\leq}](f, g)$-
interpretation . For each $f$ and $g$ we have different para-
metric $F_{\leq}^{SnS}(f, g)$. In fact, for the same $f, g$ we have the
above inclusion $F_{\leq}^{SnS}(f, g) \subset \text{SnS}[F_{\leq}](f, g)$. In general,
$F_{\leq}^{SnS}(f, g)$ and $\text{SnS}[F_{\leq}](f', g')$ are unrelated [18].

Theorem 11.1 ($F_{\leq} \subset F_{\leq}^{SnS}$) Each system $F_{\leq}^{SnS}(f, g)$ is
strictly more powerful than $F_{\leq}$: if a subtyping judgment
is provable in $F_{\leq}$ then it is also provable in $F_{\leq}^{SnS}(f, g)$; the
converse is not true in general. □

Proof. See Appendix C.2.

12 $F_{\leq}^{SnS}$ Are Less Coarse than $\text{SnS}$-Interpretations

We prove that $F_{\leq}^{SnS}(f, g)$ subtypes strictly less types than
the corresponding $\text{SnS}[F_{\leq}](f, g)$-interpretation:

Theorem 12.1 ($F_{\leq}^{SnS}(f, g) \subset \text{SnS}[F_{\leq}](f, g)$.) Each system
$F_{\leq}^{SnS}(f, g)$ is strictly less powerful than the corre-
ponding interpretation $\text{SnS}[F_{\leq}](f, g)$; whatever is provable in
$F_{\leq}^{SnS}(f, g)$ is also true in $\text{SnS}[F_{\leq}](f, g)$; the converse in
general does not hold. In particular, $F_{\leq}^{SnS}$ does not subtype dif-
erently structured types (e.g., a universally quantified and a
functional type). □

Proof. See Appendix C.3.

13 Transitivity of $F_{\leq}^{SnS}$

Changing $F_{\leq}$ for $F_{\leq}^{SnS}$ we gain decidability and do not lose
transitivity! Transitivity is an indispensable property needed
for many purposes, in particular, for proof normalization, see
[8, 21, 22].

Theorem 13.1 (Transitivity of $F_{\leq}^{SnS}$) All systems
$F_{\leq}^{SnS}(f, g)$ are closed with respect to the transitivity rule
(Trans):
\[
\text{whenever } \Gamma \vdash \sigma \leq \tau \text{ and } \Gamma \vdash \tau \leq \rho \text{ are provable in } F_{\leq}^{SnS}(f, g), \text{ then } \Gamma \vdash \sigma \leq \rho \text{ is also provable in } F_{\leq}^{SnS}(f, g).
\]

Proof. See Appendix C.4. □
The $F_{S}^{S_{w}}$-decision procedure may be obviously refined as follows: instead of pruning the $F_{S}^{A_{1}G}$-proof tree on the first application of $(V a r - A l l - D e c i d e)$, one may fix $k \in N$ and allow $k$ applications of $(V a r - A l l)$ on each branch of a subtyping proof tree before applying $(V a r - A l l - D e c i d e)$, which invokes the brute force $S n S$-decision procedure for normal forms. Denote the resulting system $F_{S}^{S_{w}}(f,g)(k)$.

Consider a similar example. The non-modified procedure analyzing the normal form

$$ \Gamma, \alpha \leq (\gamma: (T \rightarrow T) \rightarrow T) \vdash \alpha \leq (\gamma: T \rightarrow T) $$

returns TRUE. But if we allow just one application of $(V a r - A l l)$, we get $\Gamma \vdash (T \rightarrow T) \rightarrow T \leq T \rightarrow T$, then $\Gamma \vdash T \leq T \rightarrow T$, and, finally FALSE, which corresponds exactly to the $F_{S}$-proof.

With these modifications we still have for all $k \in \omega$

$$ F_{S} \subseteq F_{S}^{S_{w}}(f,g)(k) $$

It is not difficult to notice that

$$ F_{S}^{S_{w}}(f,g)(k + 1) \subseteq F_{S}^{S_{w}}(f,g)(k) $$

and

$$ F_{S}^{S_{w}}(f,g)(\infty) = F_{S}. $$

The general theory of the converging sequences

$$ \{ F_{S}^{S_{w}}(f,g)(k) \}_{k=0}^{\infty} $$

is systematically developed in [21].

15 Conclusion

In this paper we concentrated exclusively on the the subtyping relations more powerful than in $F_{S}$. When combined with the usual $F_{S}$-term typing rules, our subtyping extensions produce systems, which type strictly more terms than $F_{S}$. Let $\Gamma \vdash \sigma \rightarrow \tau$ be $F_{S}^{S_{w}}$-provable but $F_{S}$-unprovable. Then $\Gamma, x : \sigma, f : \tau \rightarrow \tau \vdash f x : \tau$ in $F_{S}^{S_{w}}$, but is untyppable in $F_{S}$.

Therefore, the problems of subject reduction, strong normalization, and minimal typing are nontrivial for our extensions. If the general answers appear to be negative, it might be interesting to investigate restricted classes and/or to modify senses in which we understand the above properties. It would also be interesting to construct models of $F_{S}^{S_{w}}$. The work on these problems has been started [21, 22, 20].

As shows the example in Section 14, the systems $F_{S}^{S_{w}}$ (and hence $S n S$-interpretations) do not separate the sets of $F_{S}$-provable and $F_{S}$-finitely disprovable subtyping judgments. So, the problem is: whether these two sets are recursively separable. If yes, the separating cover of $F_{S}$ will be a better substitute for the DECIDE component of the $F_{S}^{S_{w}}$-decision procedure.

In a particular case, when $f = g = \varepsilon$, our $S n S$-interpretations of $F_{S}$-subtyping are just interpretations into the second-order propositional logic. As it was established by Shamir [16], the class $P S A C E$ coincides with the class of languages recognizable by the so-called interactive proof systems. These systems are probabilistic algorithms exchanging messages in order to get convinced whether a given string belongs to a language with a given probability. It is challenging to introduce probabilistic algorithms in the domain of type systems.

Acknowledgments. I am greatly indebted to Luca Cardelli, Benjamin Pierce, Martin Abadi, Roberto Amadio, Philippe de Groote, Didier Galmiche, Jean-Luc Rémy, Hubert Comon, Michel Parigot for invaluable remarks, ideas, and discussions. I am grateful to my anonymous referee for his substantial help. To produce proof trees I used Paul Taylor’sEMPLATE macro package. This work was done when I was at CRIN (Centre National de Recherche en Informatique de Nancy, France), which provided me the excellent research opportunities.

References


Denote by $TS(SnS)$ or simply by $SnS$ the set of all formulas valid under the above interpretation.

Replacing the interpretation of the second-order quantifiers above by the following clause:

6) second-order quantifiers are interpreted as quantifiers over finite sets of nodes,

we get the weak monadic second order arithmetic of $n$ successors, denoted by $WSnS$.

All theories $WSnS$ and $SnS$ are decidable.

The most well known of all these are: Böhm's arithmetic $S1S$, Rabin's arithmetic $S2S$, and their weak counterparts $WS1S$, $WS2S$. The theory $S2S$ is strictly more powerful than $WS1S$, $S1S$, and easily encodes all $SnS$. For details see [13, 14].

B $F^{Alg}_\leq$: Curien-Ghelli's Algorithmic Variant of $F_\leq$

Curien and Ghelli [8], Sect. 6.1, suggested $F^{Alg}_\leq$, an alternative equivalent formulation of $F_\leq$. We present it following [12]:

$$\Gamma \vdash \top \leq \top \quad (Top)$$

$$\Gamma \vdash a \leq a \quad (Refl)$$

$$\Gamma \vdash \Gamma[a] \leq \tau \quad (AlgTrans)$$

$$\Gamma \vdash \top \leq \sigma_1 ; \sigma_2 \quad (Arrow)$$

$$\Gamma \vdash \sigma_1 \to \sigma_2 \leq \tau_1 \to \tau_2 \quad (Arr)$$

$$\Gamma \vdash (\forall a \leq \sigma_1 , \sigma_2) \leq (\forall a \leq \tau_1 , \tau_2) \quad (All)$$

Three differences of $F^{Alg}_\leq$, as compared to $F_\leq$, are: 1) reflexivity $(Ref1)$ is unlike $(Ref1)$ of $F_\leq$ is restricted to variables, 2) transitivity $(Trans)$ is replaced by $(AlgTrans)$; 3) rules are applied in ordered manner (e.g., $(AlgTrans)$ never applies if $(Ref1)$ is applicable).

Remark. Note that the inversion principle trivially holds for the $(Arrow)$ and $(All)$ of $F^{Alg}_\leq$: a conclusion of each rule is provable iff so are the premises. Proofs in $F^{Alg}_\leq$ are direct, without roundabout ways.

Lemma B.1 ($F^{Alg}_\leq \equiv F_\leq$) [8] The systems $F_\leq$ and $F^{Alg}_\leq$ are equivalent: a subtyping judgment is derivable in $F_\leq$ iff it is derivable in $F^{Alg}_\leq$.

As an immediate consequence we have the following

Lemma B.2 (Inversion Principle for $F_\leq$) In $F_\leq$:

- if $\Gamma \vdash \sigma_1 \to \sigma_2 \leq \tau_1 \to \tau_2$ is provable, then $\Gamma \vdash \tau_1 \leq \sigma_1$ and $\Gamma \vdash \sigma_2 \leq \tau_2$ are also provable;

- if $\Gamma \vdash (\forall a \leq \sigma_1 , \sigma_2) \leq (\forall a \leq \tau_1 , \tau_2)$ is provable, then $\Gamma \vdash \tau_1 \leq \sigma_1$ and $\Gamma , a \leq \tau_1 \vdash \sigma_2 \leq \tau_2$ are also provable.
The strictness of inclusion is simple: since $F_{\leq}^{S_{\text{SN}}} = \text{decidable}$ and $F_{\leq}$ is not, there should certainly exist $F_{\leq}^{S_{\text{SN}}}$-provable and not $F_{\leq}$-provable subtyping judgments. □

C.3 Proof of Theorem 12.1

Again applying Theorem 6.4 above, all $F_{\leq}^{S_{\text{SN}}}$-inference rules preserve validity with respect to any $\text{SN}$-interpretation. As normal forms of $F_{\leq}^{S_{\text{SN}}}$ are decided by the same $\text{SN}$-decision procedure, they are simultaneously true with respect to an $\text{SN}$-interpretation $\text{SN}[F_{\leq}^{S_{\text{SN}}}](f, g)$ and $F_{\leq}^{S_{\text{SN}}}(f, g)$. By definition, $F_{\leq}^{S_{\text{SN}}}(f, g)$ does not subtype differently structured types, whereas $\text{SN}$-interpretations do, e.g., $\vdash T \rightarrow \bot \leq \forall a.T$ is true in any $\text{SN}$-interpretation. □

C.4 Proof of Theorem 13.1

By induction on complexity of subtyping inference.

Suppose the premises of the theorem hold, i.e.,

$$
\Gamma \vdash \sigma \leq \tau \text{ and } \Gamma \vdash \tau \leq \rho \text{ are } F_{\leq}^{S_{\text{SN}}}-provable. 
$$

We must show that so is $\Gamma \vdash \sigma \leq \rho$.

We have to consider several cases:

1. $\rho$ is $T$;
2. $\rho$ is a type variable;
3. $\rho$ is an arrow or a universal type, both $\sigma$ and $\tau$ are type variables;
4. $\tau$ and $\rho$ are both arrow types and $\sigma$ is a type variable;
5. $\tau$ and $\rho$ are both universal types and $\sigma$ is a type variable;
6. $\sigma$, $\tau$, and $\rho$ are all arrow types;
7. $\sigma$, $\tau$, and $\rho$ are all universal types.

Case 1. Vacuous; $\Gamma \vdash \sigma \leq T$, always.

Case 2. If $\rho$ is a type variable then $\sigma$ and $\tau$ should also be type variables; otherwise the rule $(\text{TVar-R-2})$ would disagree one of the premises of the theorem.

So we should demonstrate that $F_{\leq}^{S_{\text{SN}}}$-provability of:

$$
\Gamma \vdash \sigma \leq \beta \text{, } \Gamma \vdash \beta \leq \gamma
$$

imply the $F_{\leq}^{S_{\text{SN}}}$-provability of

$$
\Gamma \vdash \sigma \leq \gamma
$$

for type variables $\alpha$, $\beta$, $\gamma$.

Note that the $F_{\leq}^{S_{\text{SN}}}$-proofs of (6) and (7) are just finite sequences of $(\text{TVar-R-1})$-applications finishing by an application of $(\text{Ref})$. These two sequences could be easily merged into just one such sequence proving (8). Indeed, starting from the judgment (8) by backward applications of $(\text{TVar-R-1})$ we are guaranteed (by provability of (6)) to reach $\beta$ on the left of $\leq$, i.e., we reach (7), which is provable by hypothesis.

Case 3. Suppose that $\rho$ is either an $\rightarrow$ or a $\forall$-type, $\sigma$ and $\tau$ are type variables $\alpha$ and $\beta$ respectively.

We transform the proofs of

$$
\Gamma \vdash \alpha \leq \beta, \quad \Gamma \vdash \beta \leq \rho
$$

into the proof of

$$
\Gamma \vdash \alpha \leq \rho
$$
as follows. Starting from the judgment (11) we first repeat (backwardly) exactly the same sequence of steps as in the proof of (9), which leads to \( \Gamma \vdash \beta \leq \beta \) (but applying \( Var \rightarrow \text{Arrow} \) or \( \text{Var-All-I} \) instead of \( \text{TVar} \)). This gives the inference of (11) from (10) used as axiom. We then repeat the proof of the latter judgment, which exists by assumption. The result is the desired proof.

**Case 4.** Suppose

\[
\begin{align*}
\Gamma & \vdash a \leq \tau_1 \rightarrow \tau_2, \quad (12) \\
\Gamma & \vdash \tau_1 \rightarrow \tau_2 \leq \rho_1 \rightarrow \rho_2, \quad (13)
\end{align*}
\]

are \( F_{\leq}^{SnS} \)-provable. We must prove that so is

\[
\Gamma \vdash a \leq \rho_1 \rightarrow \rho_2 \quad (14)
\]

The proof of (12) is a finite sequence of \( \text{Var} \rightarrow \text{Arrow} \) followed either a by \( \text{Ref} \) or b by \( \text{Arrow} \).

In the Case 4a we construct the proof of (14) (in a backward manner) first applying to (14) exactly the same sequence of \( \text{Var} \rightarrow \text{Arrow} \) applications until \( \text{Ref} \), as in the proof of (12). This gives a subsequence of (14) from (13) used as an axiom. We then complete the latter subsequence by including the proof of (13) (which is \( F_{\leq}^{SnS} \)-provable by assumption).

In the Case 4b we construct the proof of (14) as follows. Considering the final part of the inference of (12) till the first application of \( \text{Arrow} \):

\[
\begin{align*}
\Gamma & \vdash \sigma_1 \rightarrow \sigma_2 \leq \tau_1 \rightarrow \tau_2, \quad (\forall) \\
\Gamma & \vdash a' \leq \tau_1 \rightarrow \tau_2 \\
\vdots \\
\Gamma & \vdash a \leq \tau_1 \rightarrow \tau_2
\end{align*}
\]

we see that (12) is provable iff \( (\forall) \) is provable. By the inversion property for \( F_{\leq}^{SnS} \) (Theorem 10.1) this implies provability of

\[
\begin{align*}
\Gamma & \vdash \tau_1 \leq \sigma_1, \quad (16) \\
\Gamma & \vdash \sigma_2 \leq \tau_2
\end{align*}
\]

Similarly, provability of (13) implies provability of

\[
\begin{align*}
\Gamma & \vdash \rho_1 \leq \tau_1, \quad (18) \\
\Gamma & \vdash \rho_2 \leq \rho_2
\end{align*}
\]

Applying the inductive hypothesis to (18) and (16), then to (17) and (19) we get the \( F_{\leq}^{SnS} \)-provability of \( \Gamma \vdash \rho_1 \leq \sigma_1 \) and \( \Gamma \vdash \sigma_2 \leq \rho_2 \).

But this means that \( \sigma_1 \rightarrow \sigma_2 \leq \rho_1 \rightarrow \rho_2 \) is also \( F_{\leq}^{SnS} \)-provable. This allows us to transform the proof (15) into the proof of (14) by simple replacement of \( \tau_1 \rightarrow \tau_2 \) by \( \rho_1 \rightarrow \rho_2 \).

**Case 5.** Suppose

\[
\begin{align*}
\Gamma & \vdash a \leq (\forall \beta \leq \tau_1, \tau_2), \quad (20) \\
\Gamma & \vdash (\forall \beta \leq \tau_1, \tau_2) \leq (\forall \beta \leq \rho_1, \rho_2) \quad (21)
\end{align*}
\]

are \( F_{\leq}^{SnS} \)-provable. We have to prove that

\[
\Gamma \vdash a \leq (\forall \beta \leq \rho_1, \rho_2) \quad (22)
\]

The proof of (20) is a finite (possibly empty) sequence of \( \text{Var} \rightarrow \text{All-I} \) followed either a) by \( \text{TVar} \) or b) by \( \text{Var-All Decide} \).

In the Case 5a we construct the proof of (22) first applying to it the same sequence of \( \text{Var} \rightarrow \text{All-I} \) as in the proof of (20), until \( \text{TVar} \). This gives a subsequence of (22) from (21) used as axiom. We then complete the latter subsequence by including the proof of (21) (which is \( F_{\leq}^{SnS} \)-provable by assumption).

In the Case 5b we construct the proof of (22) as follows. Consider the final part of the inference of (20) till the application of \( \text{Var-All Decide} \):

\[
\begin{align*}
\Gamma & \vdash \text{DECIDE}(\Gamma \vdash a' \leq \forall \beta \leq \tau_1, \tau_2)(\forall) \\
\Gamma & \vdash a' \leq (\forall \beta \leq \tau_1, \tau_2) \\
& \vdots \\
\Gamma & \vdash a \leq (\forall \beta \leq \tau_1, \tau_2)
\end{align*}
\]

We see that (20) is provable iff the \( F_{\leq}^{SnS} \)-normal form in \( (\forall) \) is valid in a chosen theory \( \text{SnS}[f; g] \). As each \( \text{SnS}[f; g] \) is more powerful than the corresponding \( F_{\leq}^{SnS} \) (Theorem 12.1), the \( F_{\leq}^{SnS}[f; g] \)-provability of (20) implies:

\[
\left[ \Gamma \vdash \text{SnS}^\forall \varphi(x) \supset \left[ \forall \beta \leq \tau_1, \tau_2 \right] \varphi(x) \right]
\]

Similarly, the \( F_{\leq}^{SnS}[f; g] \)-provability of (21) implies

\[
\left[ \Gamma \vdash \text{SnS}^\forall \varphi(x) \supset \left[ \forall \beta \leq \rho_1, \rho_2 \right] \varphi(x) \right]
\]

Henceforth, by syllogistics, (24) and (25) imply

\[
\left[ \Gamma \vdash \text{SnS}^\forall \varphi(x) \supset \left[ \forall \beta \leq \rho_1, \rho_2 \right] \varphi(x) \right]
\]

Now, to construct the inference of (22) we start by the sequence of the same \( \text{Var-All} \) applications as in (23) till \( \Gamma \vdash a' \leq (\forall \beta \leq \rho_1, \rho_2) \). After that we should apply either the rule \( \text{TVar} \) (in this case we are done), or the rule \( \text{Var-All Decide} \) getting \( \text{DECIDE}(\Gamma \vdash a' \leq (\forall \beta \leq \rho_1, \rho_2)) \). But in the latter case \( \text{DECIDE} \) should necessarily return the result \( \text{TRUE} \) by (20), and the desired \( F_{\leq}^{SnS}[f; g] \)-proof is completed.

**Case 7.** Let

\[
\begin{align*}
\Gamma & \vdash (\forall \beta \leq \sigma_1, \sigma_2) \leq (\forall \beta \leq \tau_1, \tau_2), \quad (27) \\
\Gamma & \vdash (\forall \beta \leq \tau_1, \tau_2) \leq (\forall \beta \leq \rho_1, \rho_2)
\end{align*}
\]

We have to show

\[
\Gamma \vdash (\forall \beta \leq \sigma_1, \sigma_2) \leq (\forall \beta \leq \rho_1, \rho_2)
\]

By Inversion principle (Lemma 10.1) from (27) and (28) we get:

\[
\begin{align*}
\Gamma & \vdash \tau_1 \leq \sigma_1 \\
\Gamma & \vdash \beta \leq \tau_1 \rightarrow \tau_2 \\
\Gamma & \vdash \rho_1 \leq \tau_1 \\
\Gamma & \vdash \beta \leq \rho_1 \rightarrow \tau_2
\end{align*}
\]

From (29) and (30) by induction hypothesis we get

\[
\Gamma \vdash \rho_1 \leq \sigma_1
\]

From (31), (32) and (33) by induction hypothesis we get

\[
\Gamma, a \leq \rho_1 \rightarrow \tau_2
\]

each time instead of using the hypothesis \( \beta \leq \tau_1 \) we use the hypothesis \( \beta \leq \rho_1 \) and (32). But (34) and (35) imply (29).

**Case 6** is completely analogous to the preceding one. □
Below you find a list of the most recent technical reports of the research group *Logic of Programming* at the Max-Planck-Institut für Informatik. They are available by anonymous ftp from our ftp server `ftp.mpi-sb.mpg.de` under the directory `pub/papers/reports`. Most of the reports are also accessible via WWW using the URL `http://www.mpi-sb.mpg.de`. If you have any questions concerning ftp or WWW access, please contact `reports@mpi-sb.mpg.de`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik  
Library  
attn. Regina Kraemer  
Im Stadtwald  
D-66123 Saarbrücken  
GERMANY  
e-mail: kraemer@mpi-sb.mpg.de

<table>
<thead>
<tr>
<th>MPI-I-94-246</th>
<th>M. Hanus</th>
<th>On Extra Variables in (Equational) Logic Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI-I-94-240</td>
<td>P. Madden</td>
<td>Recursive Program Optimization Through Inductive Synthesis Proof Transformation</td>
</tr>
<tr>
<td>MPI-I-94-239</td>
<td>P. Madden, I. Green</td>
<td>A General Technique for Automatically Optimizing Programs through the Use of Proof Plans</td>
</tr>
<tr>
<td>MPI-I-94-238</td>
<td>P. Madden</td>
<td>Formal Methods for Automated Program Improvement</td>
</tr>
<tr>
<td>MPI-I-94-235</td>
<td>D. A. Plaisted</td>
<td>Ordered Semantic Hyper-Linking</td>
</tr>
<tr>
<td>MPI-I-94-234</td>
<td>S. Matthews, A. K. Simpson</td>
<td>Reflection using the derivability conditions</td>
</tr>
<tr>
<td>MPI-I-94-233</td>
<td>D. A. Plaisted</td>
<td>The Search Efficiency of Theorem Proving Strategies: An Analytical Comparison</td>
</tr>
<tr>
<td>MPI-I-94-232</td>
<td>D. A. Plaisted</td>
<td>An Abstract Program Generation Logic</td>
</tr>
<tr>
<td>MPI-I-94-229</td>
<td>Y. Dimopoulos</td>
<td>Classical Methods in Nonmonotonic Reasoning</td>
</tr>
<tr>
<td>MPI-I-94-228</td>
<td>H. J. Ohlbach</td>
<td>Computer Support for the Development and Investigation of Logics</td>
</tr>
<tr>
<td>MPI-I-94-225</td>
<td>H. J. Ohlbach</td>
<td>Synthesizing Semantics for Extensions of Propositional Logic</td>
</tr>
<tr>
<td>MPI-I-94-224</td>
<td>H. Aït-Kaci, M. Hanus, J. J. M. Navarro</td>
<td>Integration of Declarative Paradigms: Proceedings of the ICLP’94 Post-Conference Workshop Santa Margherita Ligure, Italy</td>
</tr>
<tr>
<td>MPI-I-94-223</td>
<td>D. M. Gabbay</td>
<td>LDS — Labelled Deductive Systems: Volume 1 — Foundations</td>
</tr>
<tr>
<td>MPI-I-94-218</td>
<td>D. A. Basin</td>
<td>Logic Frameworks for Logic Programs</td>
</tr>
<tr>
<td>MPI-I-94-216</td>
<td>P. Barth</td>
<td>Linear 0-1 Inequalities and Extended Clauses</td>
</tr>
<tr>
<td>MPI-I-94-209</td>
<td>D. A. Basin, T. Walsh</td>
<td>Termination Orderings for Rippling</td>
</tr>
</tbody>
</table>