Electronic Notes in Theoretical Informatics And Computer Science

ENTICS
https://entics.episciences.org

Volume 1
Proceedings of MFPS 2022

Revisiting Decidable Bounded Quantification, via Dinaturality

J. Laird

Department of Computer Science, University of Bath, UK

Abstract

We use a semantic interpretation to investigate the problem of defining an expressive but decidable type system with bounded quantification. Typechecking in the widely studied System $\mathsf{F}_{<:}$ is undecidable, thanks to an undecidable subtyping relation, for which the culprit is the rule for subtyping bounded quantification. Weaker versions of this rule, allowing decidable subtyping, have been proposed. One of the resulting type systems (Kernel $\mathsf{F}_{<:}$) lacks expressiveness, another (System $\mathsf{F}^{\top}_{<:}$) lacks the minimal typing property and thus has no evident typechecking algorithm.
We consider these rules as defining distinct forms of bounded quantification, one for interpreting type variable abstraction, and the other for type instantiation. By giving a semantic interpretation for both in terms of unbounded quantification, using the dinaturality of type instantiation with respect to subsumption, we show that they can coexist within a single type system. This does have the minimal typing property and thus a simple typechecking procedure.
We consider the fragments of this unified type system over types which contain only one form of bounded quantifier. One of these is Kernel $\mathsf{F}_{<:}$ while the other can type strictly more terms than System $\mathsf{F}^{\top}_{<:}$ but the same set of β-normal terms. We show decidability of typechecking for this fragment, and thus for System $\mathsf{F}^{\top}_{<:}$ typechecking of β-normal terms.

Keywords: Bounded quantification, Dinaturality

1 Introduction

By combining subtype and parametric polymorphism, type systems with bounded quantification may be used to write programs which are generic, but range over a constrained set of types (a program of type $\forall X <: S.T$ may be instantiated only with a subtype of S). They have been used to develop theories of key aspects of object oriented languages such as *inheritance* [4]. However, the problem of designing a tractable but expressive type system with bounded quantification is surprisingly difficult. The most natural and widely studied system based on the λ-calculus — System $\mathsf{F}_{<:}$ — has an undecidable subtyping relation [14], and thus an undecidable typing relation. It has nonetheless been influential in the development of subsequent type systems with bounded quantification such as the DOT (Dependent Object Types) calculus [12,15]. Undecidability of subtyping for a key fragment of this system has been shown by reduction to System $\mathsf{F}_{<:}$ [12,9].

Attempts to modify System $\mathsf{F}_{<:}$ to recover decidability of subtyping have been partially successful: amongst these, returning to the weaker subtyping rule for bounded quantification from the Fun calculus [4] ($\forall - \mathsf{Fun}$) gives a well-behaved system (Kernel $\mathsf{F}_{<:}$) with decidable typechecking, at the cost of a rather arbitrary restriction on the subtyping relation, leaving it unable to express some natural, and potentially useful, instances of subtyping. These *are* captured by System $\mathsf{F}^{\top}_{<:}$, a version of System $\mathsf{F}_{<:}$ with a different subtyping rule for bounded quantification ($\forall - \mathsf{Top}$) leading to a decidable and reasonably expressive

Published February 15, 2023
10.46298/entics.10474
Proceedings Available Online at
https://doi.org/10.46298/entics.proceedings.mfps38

$$\frac{}{\_ \vdash \top} \qquad \frac{\Theta \vdash S \qquad \Theta \vdash T}{\Theta, X <: S \vdash T}\ X \notin \mathsf{dom}(\Theta) \qquad \frac{\Theta \vdash S \qquad \Theta \vdash T}{\Theta, x : S \vdash T}\ x \notin \mathsf{dom}(\Theta)$$

$$\frac{\Theta, X <: T \vdash \top}{\Theta, X <: T \vdash X} \qquad \frac{\Theta \vdash S \qquad \Theta \vdash T}{\Theta \vdash S \to T} \qquad \frac{\Theta, X <: S \vdash T}{\Theta \vdash \forall (X <: S).T}$$

Table 1
Type and Context Formation Rules for System $\mathsf{F}_{<:}$

$$\frac{\Theta, X <: T, \Theta' \vdash \top}{\Theta, X <: T, \Theta' \vdash X <: T}\ \mathsf{Var} \qquad \frac{\Theta \vdash T}{\Theta \vdash T <: \top}\ \mathsf{Top} \qquad \frac{\Theta \vdash T}{\Theta \vdash T <: T}\ \mathsf{Refl} \qquad \frac{\Theta \vdash T <: T' \qquad \Theta \vdash T' <: T''}{\Theta \vdash T <: T''}\ \mathsf{Trans}$$

$$\frac{\Theta \vdash S' <: S \qquad \Theta \vdash T <: T'}{\Theta \vdash S \to T <: S' \to T'} \to \qquad \frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: T_0 \vdash S_1 <: T_1}{\Theta \vdash \forall (X <: S_0).S_1 <: \forall (X <: T_0).T_1}\ \forall - \mathsf{Orig}$$

Table 2
Subtyping Rules for System $\mathsf{F}_{<:}$

subtyping relation but to date no effective, sound and complete typechecking procedure [5]. A third potential replacement for the quantifier subtyping rule ($\forall - \mathsf{Loc}$) is intuitively appealing but does not have an evident subtyping algorithm [5]. Other decidable variants include the (heavy) restriction of not allowing quantification bounds to include the $\top$ type [10], and a family of structural *extensions* of System $\mathsf{F}_{<:}$ with decidable subtyping [16]. More recently, Strong Kernel $\mathsf{F}_{<:}$ has benn proposed [9] as a decidable subtyping system for $\mathsf{F}_{<:}$ types using two contexts of type bounds, which is strictly more expressive than Kernel $\mathsf{F}_{<:}$.

We revisit these problems from a semantic perspective, returning to the original, framework (single contexts, no restriction on which types can appear as bounds). We consider the two different rules from Kernel $\mathsf{F}_{<:}$ and System $\mathsf{F}^{\top}_{<:}$ for subtyping bounded quantification as defining different forms of bounded quantifier — one for typing abstraction of type variables, and one for typing instantiation, within a single type system: System $\mathsf{F}^{\mathsf{K}\top}_{<:}$.

Developing an interpretation of bounded types proposed in [13], we give interpretations of these two bounded quantifiers in terms of unbounded quantification and a meet operation on types, and derive a version of the $\forall - \mathsf{Loc}$ rule for inferring the subtyping relation between them. We extend this interpretation to a type system for System $\mathsf{F}_{<:}$ terms, and show soundess with respect to second-order β and η equality. This depends on the *dinaturality* property of quantifier instantiation with respect to subsumption (which was introduced to the equational theory of System $\mathsf{F}_{<:}$ by Cardelli et. al. [3]).

The subtyping and typechecking algorithms for System $\mathsf{F}_{<:}$ [7] adapt readily to our unified type system. Its most practically relevant fragments are those in which the quantifiers in the types annotating terms and contexts all satisfy the same subtyping rule. One is simply Kernel $\mathsf{F}_{<:}$ itself. The other is a modest extension of System $\mathsf{F}^{\top}_{<:}$ to include terms typable with the "missing" minimal types to allow a simple, terminating typechecking procedure. We establish that this fragment is nonetheless semantically equivalent to System $\mathsf{F}^{\top}_{<:}$, by showing that it types the same β-normal terms, for which System $\mathsf{F}^{\top}_{<:}$ typechecking is therefore decidable.

2 Background: Subtyping Bounded Quantification

We first review System $\mathsf{F}_{<:}$ [7,3] (and its subsystems, with their subtyping and typechecking problems). Its raw types are given by the grammar:

$$T ::= \top \mid X \mid T \to T \mid \forall (X <: T).T$$

where X ranges over a set of type-variables. We follow convention in defining the unbounded quantification $\forall X.S$ to be $\forall (X <: \top).S$ and identifying types up to α-conversion of bound variables.

A *context* Θ is a sequence of assumptions of the form $X <: T$ (the type-variable X has bound T) or $x : T$ (the term-variable x has type T). Judgments $\Theta \vdash T$ — "T is a well-defined type in the well-defined context Θ" — are derived according to the rules in Table 1. ($\Theta \vdash \top$ thus means that Θ is a well-defined context.)

Subtyping judgments — $\Theta \vdash S <: T$ (where $\Theta \vdash S$ and $\Theta \vdash T$) — are derived according to the rules in Table 2. This includes the following rule for subtyping bounded quantifications:

$$\frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: T_0 \vdash S_1 <: T_1}{\Theta \vdash \forall(X <: S_0).S_1 <: \forall(X <: T_0).T_1} \forall - \mathsf{Orig}$$

Although well-motivated semantically, this rule is problematic from an algorithmic point of view. Reading from the bottom up, the bound on instances of X occuring in S_1 changes from S_0 to T_0. This "re-bounding" prevents the sound and complete subtyping algorithm (deterministic search procedure for subtyping derivations) given in [7] from terminating on all inputs and indeed permits the encoding of a two-counter machine as a subtyping problem [14]. The subtyping relation for System $\mathsf{F}_{<:}$ is therefore undecidable.

Various proposals have been made to describe a more tractable subtyping relation on System $\mathsf{F}_{<:}$ types, including (*inter alia*) more restricted forms of the quantifier subtyping rule, which we now describe. Instead of viewing these variants as simply defining different type systems, we will also treat them as defining different quantifiers (potentially within a single type system), which we distinguish by decorating them with different superscripts.

Proposal 1 Restrict the subtype order on quantified types to those which have equal bounds. This corresponds to the rule $(\forall - \mathsf{Fun})$ for subtyping bounded quantification from Cardelli and Wegner's original *Fun* calculus [4]:

$$\frac{\Theta, X <: S \vdash T <: T'}{\Theta \vdash \forall^{\mathsf{K}}(X <: S).T <: \forall^{\mathsf{K}}(X <: S).T'} \forall - \mathsf{Fun}$$

This yields a type system — *Kernel* $\mathsf{F}_{<:}$ — which is algorithmically well-behaved (subtyping and type-checking are efficiently decidable) at a significant cost in expressiveness: quantified types may only be compared if their bounds are the same. For example, in System $\mathsf{F}_{<:}$ an unbounded quantification $\forall X.T$ is always a subtype of any bounded quantification $\forall(X <: S).T$ with the same body, whereas in Kernel $\mathsf{F}_{<:}$ it may only be a subtype of another unbounded quantification. Another example: if $S <: S'$ then the abstract data type $\exists(X <: S').T$ is "more abstract" than $\exists(X <: S).T$ in the sense that it is less constrained in the types that may be used to implement X. Representing $\exists(X <: S).T$ as $\forall Y.(\forall(X <: S).(T \to Y)) \to Y$, this is captured as a subtyping in the original system (i.e. $\exists(X <: S).T <: \exists(X <: S').T$) but not in Kernel $\mathsf{F}_{<:}$ — $\exists^{\mathsf{K}}(X <: S).T \not<: \exists^{\mathsf{K}}(X <: S').T$ if $S \not\equiv S'$.

Proposal 2 Ignore the bounds on the quantified variable when inferring the subtype relation on the bodies of quantified types. This corresponds to the following rule $(\forall - \mathsf{Top})$, which was proposed by Castagna and Pierce as the basis of System $\mathsf{F}^{\top}_{<:}$ [5]:

$$\frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: \top \vdash S_1 <: T_1}{\forall^{\top}(X <: S_0).S_1 <: \forall^{\top}(X <: T_0).T_1} \forall - \mathsf{Top}$$

While not strictly more expressive than Kernel $\mathsf{F}_{<:}$, this yields an expressive subtyping relation (e.g. capturing the relative abstractness of existential types) which has useful properties, including decidability and the existence of meets and joins for bounded types. However, it does not interact nicely with the typing rules of System $\mathsf{F}_{<:}$ for reasons we shall now discuss,

2.1 Typing Bounded Quantification

The raw terms of System $\mathsf{F}_{<:}$ are given by the grammar:

$$t ::= \mathsf{top} \mid x \mid \lambda(x : T).t \mid \Lambda(X <: T).t \mid tt \mid t\{T\}$$

where x ranges over the term variables, X over the type variables, and T over the raw types of System $\mathsf{F}_{<:}$. We write $\Lambda X.t$ for $\Lambda(X <: \top).t$ and identify terms up to α-conversion.

A term-in-context (or just a term, for short) $\Theta \vdash t$ consists of a well-formed context Θ and a raw term t. Typing judgments $\Theta \vdash t : T$, which are derived according to the rules in Table 3, associate a term-in-

$$\frac{\Theta \vdash \top}{\Theta \vdash \mathsf{top} : \top}\ \mathsf{top} \qquad \frac{\Theta, x : T, \Theta' \vdash \top}{\Theta, x : T, \Theta' \vdash x : T}\ \mathsf{var} \qquad \frac{\Theta \vdash t : T \qquad \Theta \vdash T <: T'}{\Theta \vdash t : T'}\ \mathsf{sub}$$

$$\frac{\Theta, x : S \vdash t : T}{\Theta \vdash \lambda x : S.t : S \to T} \to -\mathsf{i} \qquad \frac{\Theta \vdash t : S \to T \qquad \Theta \vdash s : S}{\Theta \vdash t\,s : T} \to -\mathsf{e}$$

$$\frac{\Theta, X <: S \vdash t : T}{\Theta \vdash \Lambda(X <: S).t : \forall(X <: S).T} \forall -\mathsf{i} \qquad \frac{\Theta \vdash t : \forall(X <: S).T \qquad \Theta \vdash S' <: S}{\Theta \vdash t\{S'\} : T[S'/X]} \forall -\mathsf{e}$$

Table 3
Typing Judgments for System $\mathsf{F}_{<:}$

context $\Theta \vdash t$ to a well-formed type over the same context $\Theta \vdash T$. Thanks to the typing rule of *subsumption* (sub), the problem of determining whether a given typing judgment is derivable (typechecking) depends on the subtyping problem: e.g. for any types $\Theta \vdash S, S', T$, the term $\Theta, x : S, f : S' \to T \vdash f\,x$ is typable (with T) if and only if $\Theta \vdash S <: S'$.

The (sound and complete) typechecking procedure for System $\mathsf{F}_{<:}$ (which is described in more detail in Section 6) is based on finding a *minimal type* for each typable term.

Definition 2.1 $\Theta \vdash T$ is a minimal type for the term $\Theta \vdash t$ if $\Theta \vdash t : T$, and if $\Theta \vdash t : T'$ then $\Theta \vdash T <: T'$.

System $\mathsf{F}_{<:}$ possesses the *minimal typing property* with respect to the original subtyping rule: every term which can be typed has a minimal type. Thus we may check the typing $\Theta \vdash t : T$ by finding a minimal type $\Theta \vdash S$ for $\Theta \vdash t$ (for which there is a sound and complete algorithm) and checking $\Theta \vdash S <: T$. The minimal typing algorithm for System $\mathsf{F}_{<:}$ restricts straightforwardly to Kernel $\mathsf{F}_{<:}$. Since the subtyping problem for Kernel $\mathsf{F}_{<:}$ is decidable, this gives an efficient typechecking procedure which terminates on all inputs. However:

Proposition 2.2 *System $\mathsf{F}^{\top}_{<:}$ does not possess the minimal typing property.*

Proof. The following counterexample was given by Ghelli ([5] — Appendix).
In System $\mathsf{F}^{\top}_{<:}$, the typing judgments $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y : \forall^{\top}(Z <: X).(Z \to Z)$ and $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y : \forall^{\top}(Z <: X).(Z \to X)$ are both derivable, but these types have no lower bound and so $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y$ can have no minimal type.

Informally, it is easy to see that such a lower bound would have the form $X <: \top \vdash \forall^{\top}(Z <: S).T \to T'$ where $X <: \top, Z <: \top \vdash T' <: X$ and $X <: \top, Z <: \top \vdash T' <: Z$, and that no such type exists. A more formal proof can be given using an alternative, algorithmic presentation of the subtyping relation as in Table 7. □

The problem is that the subtyping assumption $Z <: X$ may be used to derive the *typing* $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y : \forall^{\top}(Z <: X).(Z \to X)$ but not the *subtyping* $X <: \top \vdash \forall^{\top}(Z <: X).(Z \to Z) <: \forall^{\top}(Z <: X).(Z \to X)$. It is not known whether the type synthesis algorithm for System $\mathsf{F}_{<:}$ can be adapted to System $\mathsf{F}^{\top}_{<:}$, nor indeed whether its typechecking problem is decidable.

3 Semantics of Bounded Quantification

Following the suggestion [5] that an expressive yet tractable type system for bounded quantification should be grounded in semantic understanding, we seek an interpretation which relates the bounded quantifiers $\forall^{\top}$ and $\forall^{\mathsf{K}}$. Since System $\mathsf{F}^{\top}_{<:}$ and Kernel $\mathsf{F}_{<:}$ are subsystems of the original System $\mathsf{F}_{<:}$, they may both be interpreted in any semantics of the latter, examples of which include models based on modest sets and partial equivalence relations [2], and on games and strategies [6,11]. However, because these interpret the original subtyping rule for bounded quantification they unsurprisingly give few direct clues about decidable subtyping: the PER models are essentially realizability interpretations on an untyped model, whereas the game semantics in [11] interprets subtyping coercions as morphisms defined inductively on the derivation of the subtyping relation. On the other hand, the latter interpretation does depend crucially on the *dinaturality* of instantiation with respect to subtyping coercions (which does not hold with respect to terms in general [8]) and this will also be the basis for our interpretations. We will use the equational

formulation of dinaturality in System $\mathsf{F}_{<:}$, introduced by Cardelli et. al [3], to give related interpretations of $\forall^{\mathsf{K}}$ and $\forall^{\top}$ in terms of unbounded quantification and a meet operation ($\wedge$) for the subtyping relation.

We now define the target calculus for this translation. Let System $\mathsf{F}_{\wedge}$ (cf [13]) be System $\mathsf{F}_{<:}$, restricted to unbounded quantification, and extended with a binary meet operation on types — i.e. its raw types are given by the grammar:

$$T ::= \top \mid X \mid T \to T \mid \forall X.T \mid T \wedge T$$

The type-formation rules of Table 1 are extended with the rule:

$$\frac{\Theta \vdash S \qquad \Theta \vdash T}{\Theta \vdash S \wedge T}$$

and the subtyping rules of Table 2 with the rules:

$$\frac{\Theta \vdash S \wedge S'}{\Theta \vdash S \wedge S' <: S} \qquad \frac{\Theta \vdash S \wedge S'}{\Theta \vdash S \wedge S' <: S'} \qquad \frac{\Theta \vdash T <: S \qquad \Theta \vdash T <: S'}{\Theta \vdash T <: S \wedge S'}$$

The grammar of raw terms remains unchanged (except that annotating types range over System $\mathsf{F}_{\wedge}$ types) and the typing rules of Table 3 are also unchanged.

As observed in [13], a bounded variable $\Theta, X <: S \vdash X$ may be represented by the type $\Theta, X <: \top \vdash X \wedge S$. This is bounded above by S ($\Theta, X <: \top \vdash X \wedge S <: S$) and if $\Theta \vdash S' <: S$ then $\Theta \vdash S' <: S' \wedge S <: S'$, so that substituting S' for X in $T[X]$ is equivalent to substituting S' for X in $T[X \wedge S]$, up to the equivalence on types induced by the subtype preorder. This suggests an interpretation of the bounded quantification $\Theta \vdash \forall(X <: S).T$ as $\forall X.T[X \wedge S/X]$:

Lemma 3.1 *If* $\Theta, X <: S \vdash T <: T'$ *then* $\Theta \vdash \forall X.T[X \wedge S/X] <: \forall X.T'[X \wedge S/X]$.

Proof. We show that if $\Theta, X <: S, \Theta' \vdash T <: T'$ then $\Theta, X <: \top, \Theta'[X \wedge S/X] \vdash T[X \wedge S/X] <: T'[X \wedge S/X]$ by induction on the derivation of $\Theta, X <: S, \Theta' \vdash T' <: T'$. □

So if $\Theta, X <: S, \Theta' \vdash T <: T'$ then $\Theta \vdash \forall X.T[X \wedge S/X] <: \forall X.T'[X \wedge S/X]$ — i.e. this interpretation satisfies the subtyping rule $\forall - \mathsf{Fun}$. Accordingly, we define:

Definition 3.2 Let $\ulcorner \forall^{\mathsf{K}}(X <: S).T \urcorner \triangleq \forall X.T[X \wedge S/X]$.

3.1 Interpretation of $\forall^{\top}$

The above interpretation of bounded quantification does not satisfy the $\forall - \mathsf{Top}$ subtyping rule because it is not antitone in the variable bound — e.g. $S' <: S$ does not generally imply $\ulcorner \forall^{\mathsf{K}}(X <: S).X \to X \urcorner = \forall X.X \wedge S \to X \wedge S \not<: \forall X.X \wedge S' \to X \wedge S' = \ulcorner \forall^{\mathsf{K}}(X <: S').X \to X \urcorner$. The problem is that substitution into a type is not antitone with respect to the subtyping order: from a semantic viewpoint, types do not act as contravariant (nor covariant) functors with respect to the subtype preorder. A solution is to separate *positive* and *negative* occurrences of type-variables, such that substitution of the former is monotone, and of the latter is antitone with respect to the subtype preorder. In other words types act as mixed-variance functors with respect to substitution of type variables (in the next section, we will see that this leads to an interpretation of terms as dinatural transformations). Accordingly, we define a mixed substitution operation on types of System $\mathsf{F}_{\wedge}$ which acts separately on the positive and negative occurences of type variables.

Definition 3.3 Given raw types S_-, S_+, T, we assume, by α-conversion, that neither X nor any free variables of S_-, S_+ are quantified in T. Let $T[(S_-, S_+)/X]$ be the substitution of S_- and S_+ for the negative and positive occurrences of X in T, respectively, so that $T[S/X] = T[(S, S)/X]$. Formally:

$$Y[(S_-, S_+)/X] = \begin{cases} S_+ & \text{if } Y \equiv X \\ Y & \text{otherwise} \end{cases} \qquad \top[(S_-, S_+)/X] = \top$$

$$(T \to T')[(S_-, S_+/X] = T[(S_+, S_-)/X] \to T'[(S_-, S_+)/X] \qquad (\forall Y.T)[(S_-, S_+)/X] = \forall Y.T[(S_-, S_+)/X]$$

$$(T \wedge T')[(S_-, S_+)/X] = T[(S_-, S_+)/X] \wedge T'[(S_-, S_+)/X]$$

This is antitone in S_- and monotone in S_+ and T with respect to the subtyping preorder.

Lemma 3.4 *If* $\Theta, \Theta' \vdash S'_- <: S_-$ *and* $\Theta, \Theta' \vdash S_+ <: S'_+$ *then for any* $\mathsf{F}_\wedge$*-type* $\Theta, X <: \top \vdash T$*,*

$$\Theta, \Theta' \vdash T[(S_-, S_+)/X] <: T[(S'_-, S'_+)/X]$$

Proof. By induction on the size of T. □

Lemma 3.5 *For* $\mathsf{F}_\wedge$*-types* $\Theta, X <: \top \vdash T <: T'$ *and any types* $\Theta, \Theta' \vdash S_-, S_+$*,*

$$\Theta, \Theta' \vdash T[(S_-, S_+)/X] <: T'[(S_-, S_+)/X].$$

Proof. By induction on the derivation of $\Theta, X <: \top \vdash T_1 <: T_2$. □

Thus we may interpret $\forall^\top(X <: S).T$ in terms of unbounded quantification by substituting only negative occurrences of X in T with $X \wedge S$.

Definition 3.6 Writing $T[S/X^-]$ for $T[(S, X)/X]$, let $\ulcorner\forall^\top(X <: S).T\urcorner \triangleq \forall X.T[X \wedge S/X^-]$.

This interpretation of bounded quantification satisfies the $\forall - \mathsf{Top}$ subtyping rule:

Lemma 3.7 *If* $\Theta \vdash S' <: S$ *and* $\Theta, X <: \top \vdash T <: T'$ *then* $\Theta \vdash \forall X.T[X \wedge S/X^-] <: \forall X.T[X \wedge S'/X^-]$

Proof. $\Theta \vdash S' <: S$ implies $\Theta, X <: \top \vdash X \wedge S' <: X \wedge S$.
So $\Theta, X <: \top \vdash T[X \wedge S/X^-] <: T[X \wedge S'/X^-]$ by Lemma 3.4,
$\Theta, X <: \top \vdash T[X \wedge S'/X^-] <: T'[X \wedge S'/X^-]$ by Lemma 3.5
and so $\Theta \vdash \forall X.T[X \wedge S/X^-] <: \forall X.T'[X \wedge S'/X^-]$ as required. □

This rule does not satisfy $\forall - \mathsf{Fun}$ because positive occurences of a variable are not interpreted as subtypes of their bounds. For example, $\ulcorner\forall^\top(X <: S).X\urcorner$ is not a subtype of $\ulcorner\forall^\top(X <: S).S\urcorner$.

3.2 Relating $\forall^\mathsf{K}$ *and* $\forall^\top$

From the interpretations of $\forall^\mathsf{K}$ and $\forall^\top$ within the target calculus System $\mathsf{F}_\wedge$, we derive a subtyping rule which relates them.

Lemma 3.8 *If* $\Theta \vdash T_0 <: S_0$ *and* $\Theta, X <: S_0 \vdash S_1 <: T_1$ *then* $\Theta \vdash \forall X.S_1[X \wedge S_0/X] <: \forall X.T_1[X \wedge T_0/X^-]$.

Proof. $\Theta, X <: \top \vdash S_1[X \wedge S_0/X] <: T_1[X \wedge S_0/X]$ by Lemma 3.1.
$\Theta, X <: \top \vdash X \wedge T_0 <: X \wedge S_0$ and $\Theta, X <: \top \vdash X \wedge S_0 <: X$ implies
$\Theta, X <: \top \vdash T_1[X \wedge S_0/X] = T_1[(X \wedge S_0, X \wedge S_0)/X] <: T_1[(X \wedge T_0, X)/X] = T_1[X \wedge T_0/X^-]$ by Lemma 3.4.
So $\Theta \vdash \forall X.S_1[X \wedge S_0/X] <: \forall X.T_1[X \wedge T_0/X^-]$ as required. □

In other words, the semantics soundly interprets the following subtyping rule:

$$\frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: S_0 \vdash S_1 <: T_1}{\Theta \vdash \forall^\mathsf{K}(X <: S_0).S_1 <: \forall^\top(X <: T_0).T_1} \forall - \mathsf{Loc}$$

This is the rule $\forall - \mathsf{Loc}$ considered (without the decorating superscripts) as yet another candidate replacement for the original rule for subtyping bounded quantification in System $\mathsf{F}_{<:}$ [5]. However, the resulting type system lacks an evident subtyping algorithm, due to the failure of a key transitivity property which is essential to the subtyping algorithm for System $\mathsf{F}_{<:}$. In Section 6 we show that the decorated form of the rule avoids this problem, allowing adaptation of the $\mathsf{F}_{<:}$ subtyping algorithm to the decorated calculus.

$$\frac{\Theta, X <: T, \Theta' \vdash \top}{\Theta, X <: T, \Theta' \vdash X <: T}\ \mathsf{Var} \qquad \frac{\Theta \vdash T}{\Theta \vdash T <: \top}\ \mathsf{Top} \qquad \frac{\Theta \vdash T}{\Theta \vdash T <: T}\ \mathsf{Refl} \qquad \frac{\Theta \vdash T <: T' \qquad \Theta \vdash T' <: T''}{\Theta \vdash T <: T''}\ \mathsf{Trans}$$

$$\frac{\Theta \vdash S' <: S \quad \Theta \vdash T <: T'}{\Theta \vdash S \to T <: S' \to T'} \to \qquad \frac{\Theta, X <: S \vdash T <: T'}{\Theta \vdash \forall^{\mathsf{K}}(X <: S).T <: \forall^{\mathsf{K}}(X <: S).T'} \forall - \mathsf{Fun}$$

$$\frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: S_0 \vdash S_1 <: T_1}{\Theta \vdash \forall^{\mathsf{K}}(X <: S_0).S_1 <: \forall^{\top}(X <: T_0).T_1} \forall - \mathsf{Loc} \qquad \frac{\Theta \vdash T_0 <: S_0 \qquad \Theta, X <: \top \vdash S_1 <: T_1}{\Theta \vdash \forall^{\top}(X <: S_0).S_1 <: \forall^{\top}(X <: T_0).T_1} \forall - \top$$

Table 4
Subtyping Rules for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

$$\frac{\Theta \vdash \top}{\Theta \vdash \mathsf{top} : \top} \top \qquad \frac{\Theta, x : T, \Theta' \vdash \top}{\Theta, x : T, \Theta' \vdash x : T}\ \mathsf{var} \qquad \frac{\Theta \vdash t : T \qquad \Theta \vdash T <: T'}{\Theta \vdash t : T'}\ \mathsf{sub}$$

$$\frac{\Theta, x : S \vdash t : T}{\Theta \vdash \lambda x : S.t : S \to T} \to -\mathsf{i} \qquad \frac{\Theta \vdash t : S \to T \qquad \Theta \vdash s : S}{\Theta \vdash ts : T} \to -\mathsf{i}$$

$$\frac{\Theta, X <: S \vdash t : T}{\Theta \vdash \Lambda(X <: S).t : \forall^{\mathsf{K}}(X <: S).T} \forall - \mathsf{i} \qquad \frac{\Theta \vdash t : \forall^{\top}(X <: S).T \qquad \Theta \vdash S' <: S}{\Theta \vdash t\{S'\} : T[S'/X]} \forall - \mathsf{i}$$

Table 5
Typing Judgments for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

4 System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

We may now formally define a type system (System $\mathsf{F}^{\mathsf{K}\top}_{<:}$) with $\{\mathsf{K}, \top\}$-decorated bounded quantification. Raw types are given by the grammar:

$$T ::= \top \mid X \mid T \to T \mid \forall^{\mathsf{K}}(X <: T).T \mid \forall^{\top}(X <: T).T$$

Subtyping judgments are given by the rules in Table 4, which replace the single original typing rule for bounded quantification of System $\mathsf{F}_{<:}$ with the rules $\forall - \mathsf{Top}$, $\forall - \mathsf{Fun}$ and $\forall - \mathsf{Loc}$.

Raw terms are defined as in System $\mathsf{F}_{<:}$ (except that annotating types range over the raw $\mathsf{F}^{\mathsf{K}\top}_{<:}$-types). Type-variable abstraction is typed using $\forall^{\mathsf{K}}$ and instantiation is typed using $\forall^{\top}$: the typing rules (Table 5) are obtained by decorating the introduction and elimination rules for bounded quantification with K and $\top$, respectively. Introduction of $\forall^{\top}$ and elimination of $\forall^{\mathsf{K}}$ are derivable by subsumption, e.g.

$$\frac{\dfrac{\Theta, X <: S \vdash t : T}{\Theta \vdash \Lambda(X <: S).t : \forall^{\mathsf{K}}(X <: S).T} \forall - \mathsf{i} \qquad \dfrac{\dfrac{}{\Theta \vdash S <: S}\ \mathsf{Refl} \qquad \dfrac{}{\Theta, X <: S \vdash T <: T}\ \mathsf{Refl}}{\Theta \vdash \forall^{\mathsf{K}}(X <: S).T <: \forall^{\top}(X <: S).T} \forall - \mathsf{Loc}}{\Theta \vdash \Lambda(X <: S).t : \forall^{\top}(X <: S).T}\ \mathsf{sub}$$

The typing of Ghelli's example (Proposition 2.2) in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ illustrates how it repairs the failure of the minimal typing property in $\mathsf{F}^{\top}_{<:}$. Recall that the term-in-context $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y$ may be typed with either of the $\mathsf{F}^{\top}_{<:}$-types $X <: \top \vdash \forall^{\top}(Z <: X).(Z \to Z)$ and $X <: \top \vdash \forall^{\top}(Z <: X).(Z \to X)$, which are not bounded below by any $\mathsf{F}^{\top}_{<:}$-type.

In $\mathsf{F}^{\mathsf{K}\top}_{<:}$, $X <: \top \vdash \Lambda(Z <: X).\lambda(y : Z).y$ has the minimal type $X <: \top \vdash \forall^{\mathsf{K}}(Z <: X).(Z \to Z)$: using the $\forall - \mathsf{Loc}$ rule, we may derive both:

$$\frac{X <: \top \vdash X <: X \qquad X <: \top, Z <: X \vdash Z \to Z <: Z \to Z}{X <: \top \vdash \forall^{\mathsf{K}}(Z <: X).(Z \to Z) <: \forall^{\top}(Z <: X).(Z \to Z)} \forall - \mathsf{Loc}$$

and

$$\frac{X <: \top \vdash X <: X \qquad X <: \top, Z <: X \vdash Z \to Z <: Z \to X}{X <: \top \vdash \forall^{\mathsf{K}}(Z <: X).(Z \to Z) <: \forall^{\top}(Z <: X).(Z \to X)} \forall - \mathsf{Loc}$$

We will show that System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ possesses the minimal typing property in Section 6.

$$\frac{\Theta \vdash t : \top \qquad \Theta \vdash t' : \top}{\Theta \vdash t = t' : \top}\ \top \qquad \frac{\Theta \vdash t : T}{\Theta \vdash t = t : T}\ \mathit{refl} \qquad \frac{\Theta \vdash r = s : T \qquad \Theta \vdash s = t : T}{\Theta \vdash r = t : T}\ \mathit{trans} \qquad \frac{\Theta \vdash s = t : T}{\Theta \vdash t = s : T}\ \mathit{sym}$$

$$\frac{\Theta, x : S \vdash t : T \qquad \Theta \vdash s : S}{\Theta \vdash (\lambda x.t)\, s = t[s/x] : T}\ \beta_1 \qquad x \notin \mathsf{dom}(\Theta)\ \frac{\Theta \vdash t : S \to T}{\Theta \vdash \lambda x.(t\, x) = t : S \to T}\ \eta_1$$

$$\frac{\Theta, X <: S \vdash t : T \qquad \Theta \vdash R <: S}{\Theta \vdash (\Lambda(X <: S).t)\{R\} = t[S'/X] : T[S'/X]}\ \beta_2 \qquad Y \notin \mathsf{dom}(\Theta)\ \frac{\Theta \vdash t : \forall^{\top} X <: S.T}{\Theta \vdash \Lambda(Y <: S).(t\{Y\}) = t : \forall^{\top} X <: S.T}\ \eta_2$$

$$\frac{\Theta, X : S \vdash t = t' : T}{\Theta \vdash \lambda x : S.t = \lambda x.t' : S \to T}\ \mathit{abs}_1 \qquad \frac{\Theta, X <: S \vdash t = t' : T}{\Theta \vdash \Lambda(X <: S).t = \Lambda(X <: S).t' : \forall^{\mathsf{K}} X <: S.T}\ \mathit{abs}_2$$

$$\frac{\Theta \vdash t = t' : S \to T \qquad \Theta \vdash s = s' : S}{\Theta \vdash t\, s = t'\, s' : T}\ \mathit{app}_1$$

$$\frac{\Theta \vdash t = t' : \forall^{\top} X <: S.T \qquad \Theta \vdash R <: S \qquad \Theta \vdash R' <: S \qquad \Theta \vdash T[R/X] <: T' \qquad \Theta \vdash T[R'/X] <: T'}{\Theta \vdash t\{R\} = t'\{R'\} : T'}\ \mathit{app}_2$$

Table 6
Derivation Rules for Term Equivalence

5 Semantics of System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

We will consider subtyping and typechecking algorithms for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ in the following section, showing that these are quite well-behaved. First, we take its semantic justification further, by showing that the interpretation of $\forall^{\mathsf{K}}$ and $\forall^{\top}$ in System $\mathsf{F}_{\wedge}$ may be extended to terms.

We interpret the $\forall^{\top}$-elimination rule, using the fact that $\Theta \vdash S' <: S$ implies $\Theta \vdash S' \wedge S <: S' <: S' \wedge S$.

Proposition 5.1 *If* $\Theta \vdash t : \ulcorner \forall^{\top} X <: S.T \urcorner$ *and* $\Theta \vdash S' <: S$ *then* $\Theta \vdash t\{S'\} : T[S'/X]$.

Proof. From $\Theta \vdash S' <: S$ (and $\Theta \vdash S' <: S'$) we may infer $\Theta \vdash S' <: S' \wedge S$ and hence by Lemma 3.4 $\Theta \vdash T[(S' \wedge S, S')/X] <: T[S'/X] = T[(S', S')/X]$. Hence we have the following derivation of $\Theta \vdash t\{S'\} : T[S'/X]$:

$$\dfrac{\dfrac{\Theta \vdash t : \forall X <: \top.T[X \wedge S/X^{-}] \qquad \Theta \vdash S' <: \top}{\Theta \vdash t\{S'\} : T[(S' \wedge S, S')/X]}\ \forall - \mathsf{e} \qquad \Theta \vdash T[(S' \wedge S, S')/X] <: T[S'/X]}{\Theta \vdash t\{S'\} : T[S'/X]}\ \mathsf{sub}$$

□

To derive the $\forall^{\mathsf{K}}$-introduction rule, we extend the interpretation to bounded type-abstraction:

Definition 5.2 Suppose $\Theta, X <: S \vdash t : T$. Let $\ulcorner \Lambda(X <: S).t \urcorner \triangleq \Lambda(X <: \top).t[X \wedge S/X]$.

A straightforward induction establishes that:

Lemma 5.3 $\Theta, X <: S, \Theta' \vdash t : T$ *implies* $\Theta, X <: \top, \Theta'[X \wedge S/X] \vdash t[X \wedge S/X] : T[X \wedge S/X]$.

and hence:

Proposition 5.4 *If* $\Theta, X <: S \vdash t : T$ *then* $\Theta \vdash \ulcorner \Lambda(X <: S).t \urcorner : \ulcorner \forall^{\mathsf{K}} X <: S.T \urcorner$.

5.1 Soundness via Dinaturality

We show that these interpretations of type-abstraction and instantiation are sound with respect to β and η equivalences using the equational theory for System $\mathsf{F}_{<:}$ introduced by Cardelli et. al. [3], which we adapt to System $\mathsf{F}^{\mathsf{K}\top}_{<:}$. Derivation rules (for equational judgements in context $\Theta \vdash t = t' : T$, where $\Theta \vdash t : T$ and $\Theta \vdash t' : T$) are given in Table 6. The rules for original System $\mathsf{F}_{<:}$, and the target calculus System $\mathsf{F}_{\wedge}$, may be obtained by simply erasing the decorations on quantifiers. They axiomatize term-equality as a congruence containing β and η equalities for type and term variable abstraction, together with the rule app_2:

$$\frac{\Theta \vdash t = t' : \forall X <: S.T \qquad \Theta \vdash R <: S \qquad \Theta \vdash R' <: S \qquad \Theta \vdash T[R/X] <: T' \qquad \Theta \vdash T[R'/X] <: T'}{\Theta \vdash t\{R\} = t'\{R'\} : T'}$$

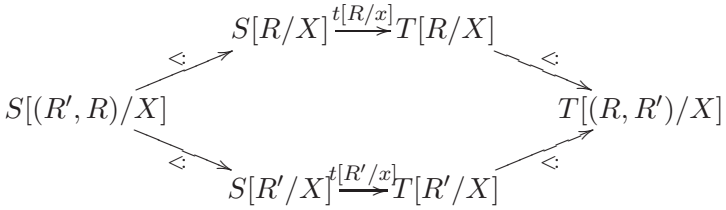
from which reflexivity and β-equivalence yield the derived rule:

$$\frac{\Theta, X <: S \vdash t : T \qquad \Theta \vdash R <: S \qquad \Theta \vdash R' <: S \qquad \Theta \vdash T[R/X] <: T' \qquad \Theta \vdash T[R'/X] <: T'}{\Theta \vdash t[R/X] = t[R'/X] : T'} \ (*)$$

This relates subtyping to parametricity by expressing the *extranaturality* of type instantiation with respect to subsumption. In the setting of the λ-calculus, this is equivalent to *dinaturality*: in category-theoretic terms, dinaturality generalizes the notion of natural transformation between covariant functors to mixed-variance functors [1]. Since second-order types correspond to mixed-variance functors on the preorder of subtypes in a given context (Lemma 3.4), dinaturality of instantiation may be captured within the equational theory for System $\mathsf{F}_\wedge$ by the derived rule:

$$\frac{\Theta, X <: \top \vdash t : S \to T \qquad \Theta \vdash R <: R'}{\Theta \vdash t[R/X] = t[R'/X] : S[(R', R)/X] \to T[(R, R')/X]}$$

Diagrammatically, this is the commuting hexagon:



The dinaturality expressed in equation $(*)$ is crucial to showing that interpretation in the target language System $\mathsf{F}_\wedge$ is sound with respect to the equational theory, because it equates terms which are instantiated with types which are equivalent up to subtyping equivalence (i.e. $S <: T$ and $T <: S$). Consider, for example, second-order β-equality:

Proposition 5.5 *If $\Theta, X <: S \vdash t : T$ and $\Theta \vdash S' <: S$ then $\Theta \vdash \ulcorner \Lambda(X <: S).t \urcorner\{S'\} = t[S'/X] : T[S'/X]$.*

Proof. If $\Theta, X <: S \vdash t : T$ then $\Theta, X <: \top \vdash t : T[X \wedge S/X]$ by Lemma 5.3, and the β-equivalence rule is:

$$\frac{\Theta, X <: \top \vdash t[X \wedge S/X] : T[S'/X] \qquad \Theta \vdash S' <: \top}{\Theta \vdash (\Lambda X.t[X \wedge S/X])\{S'\} = t[X \wedge S/X][S'/X]} \ \beta_2$$

where $t[X \wedge S/X][S'/X] \equiv t[S' \wedge S/X]$. So we need to show that $t[S' \wedge S/X]$ is equivalent to $t[S'/X]$ at type $T[S'/X]$.

$\Theta \vdash S' \wedge S <: S'$, and $\Theta \vdash S' <: S$ implies $\Theta \vdash S' <: S' \wedge S$, and so by Lemma 3.4 $\Theta \vdash T[S \wedge S'/X] <: T[S'/X]$. Thus we have the following instance of our derived rule $(*)$

$$\frac{\Theta, X <: S \vdash t : T \qquad \Theta \vdash S' \wedge S <: S \qquad \Theta \vdash S' <: S \qquad \Theta \vdash T[S \wedge S'/X] <: T[S'/X] \qquad \Theta \vdash T[S'/X] <: T[S'/X]}{\Theta \vdash t[S \wedge S'] = t[S'/X] : T[S'/X]}$$

and so by transitivity:

$$\frac{\Theta \vdash (\Lambda X.t[X \wedge S/X])\{S'\} = t[S' \wedge S/X] \qquad \Theta \vdash t[S \wedge S'] = t[S'/X] : T[S'/X]}{\Theta \vdash (\Lambda X.t[X \wedge S/X])\{S'\} = t[S'/X] : T[S'/X]} \ \mathit{trans}$$

□

Similarly, we use $(*)$ to show soundness with respect to second-order η-equality (at $\forall^\top$-types):

Proposition 5.6 *If $\Theta \vdash t : \ulcorner \forall^\top (X <: S).T \urcorner$ and $Y \notin \mathsf{dom}(\Theta)$, then $\Theta \vdash t = \ulcorner \Lambda(Y <: S).t\{Y\} \urcorner : \ulcorner \forall^\top (X <: S).T \urcorner$.*

$$\frac{\Theta \vdash T}{\Theta \vdash_A T <: \top} \qquad \frac{\Theta \vdash X}{\Theta \vdash_A X <: X} \qquad \frac{\Theta, X <: S, \Theta' \vdash_A S <: T}{\Theta, X <: T, \Theta' \vdash_A X <: T}\ T \not\equiv \top, X$$

$$\frac{\Theta \vdash_A S' <: S \quad \Theta \vdash_A T <: T'}{\Theta \vdash_A S \to T <: S' \to T'} \qquad \frac{\Theta, X <: S \vdash_A T <: T'}{\Theta \vdash_A \forall^{\mathsf{K}}(X <: S).T <: \forall^{\mathsf{K}}(X <: S).T'}$$

$$\frac{\Theta \vdash_A T_0 <: S_0 \qquad \Theta, X <: S_0 \vdash_A S_1 <: T_1}{\Theta \vdash_A \forall^{\mathsf{K}}(X <: S_0).S_1 <: \forall^{\top}(X <: T_0).T_1)} \qquad \frac{\Theta \vdash_A T_0 <: S_0 \qquad \Theta, X <: \top \vdash_A S_1 <: T_1}{\Theta \vdash_A \forall^{\top}(X <: S_0).S_1 <: \forall^{\top}(X <: T_0).T_1)}$$

Table 7
Algorithmic Derivation Rules for Subtyping Judgements

Proof. The second-order η-equality rule itself yields:

$$\frac{\Theta \vdash t : \forall X.T[X \wedge S/X^-]}{\Theta \vdash t = \Lambda Y.t\{Y\} : \forall X.T[X \wedge S/X^-]}\ \eta_2$$

However, $\ulcorner \Lambda(Y <: S).t\{Y\} \urcorner = \Lambda Y.t\{S \wedge Y\}$. Noting that $\Theta, Y <: S \vdash X \wedge S <: (X \wedge S) \wedge S$ and hence by Lemma 3.4, $\Theta, Y <: \top \vdash T[((X \wedge S) \wedge S, X \wedge S)/X] <: T[(Y \wedge S, Y)/X]$, we may use dinaturality to infer:

$$\frac{\dfrac{\Theta \vdash t : \forall X.T[X \wedge S/X^-]}{\Theta, Y <: \top \vdash t : \forall X.T[X \wedge S/X^-]} \qquad \Theta, Y <: \top \vdash T[(Y \wedge S, Y)/X], T[((Y \wedge S) \wedge S, Y \wedge S)/X] <: T[(Y \wedge S, Y)/X]\ app_2}{\Theta, Y <: \top \vdash t\{Y\} = t\{Y \wedge S\} : T[(Y \wedge S, Y)/X]}\ abs_2$$
$$\Theta \vdash \Lambda Y.t\{Y\} = \Lambda Y.t\{Y \wedge S\} : \forall X.T[X \wedge S/X^-]$$

and hence by transitivity:

$$\frac{\Theta \vdash t = \Lambda Y.t\{Y\} : \forall X.T[X \wedge S/X^-] \qquad \Theta \vdash \Lambda Y.t\{Y\} = \Lambda Y.t\{Y \wedge S\} : \forall X.T[X \wedge S/X^-]}{\Theta \vdash t = \Lambda Y.t\{Y \wedge S\} : \forall X.T[X \wedge S/X^-]}\ trans$$

□

Soundness with respect to the remaining rules (including app_2 itself) is straightforward.

6 Subtyping and Typechecking Algorithms for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

Having established a semantic basis for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$, we now describe procedures for solving its subtyping and typechecking problems. These adapt readily from their analogues for System $\mathsf{F}_{<:}$ [7]: we give sufficient details here to show how the difficulties previously associated with the rules $\forall - \mathsf{Loc}$ and $\forall - \mathsf{Top}$ are avoided.

The subtyping algorithm is given by defining an alternative "algorithmic" presentation of the subtyping relation, via a set of derivation rules (Table 7) with the property that any subtyping judgment is the consequence of at most one rule, so that the evident search procedure for the derivation of a subtyping judgment in this system is deterministic. To show that this is sound and complete, it suffices to establish that the subtyping judgments derivable according to the rules in Tables 2 and 7 are the same, by showing that each rule of one system is admissible in the other, and vice-versa. The only case that differs from the proof for System $\mathsf{F}_{<:}$ is to show that the transitivity rule for the subtyping relation (Trans) is admissible in the algorithmic system.

Lemma 6.1 *If $\Theta \vdash_A R <: S$ and $\Theta \vdash_A S <: T$ then $\Theta \vdash_A R <: T$.*

Proof. By induction on the size of the derivations of $\Theta \vdash_A R <: S$ and $\Theta \vdash_A S <: T$. It follows the proofs for Kernel $\mathsf{F}_{<:}$ or $\mathsf{F}^{\top}_{<:}$, except where R, S and T are differently quantified types (i.e. they are not all prefixed with quantifiers with the same decoration). Since it is never possible to infer $\forall^{\top}(X <: S_0).S_1 <: \forall^{\mathsf{K}}(X <: T_0).T_1$ (as this is not the consequence of any rule) the only possibilities are:

- $R \equiv \forall^{\mathsf{K}}(X <: R_0).R_1$, $S \equiv \forall^{\mathsf{K}}(X <: S_0).S_1$ and $T \equiv \forall^{\top}(X <: T_0).T_1$. Then $R_0 \equiv S_0$ and $\Theta \vdash_A T_0 <: S_0$ (so $\Theta \vdash_A T_0 <: R_0$) and $\Theta, X <: S_0 \vdash_A R_1 <: S_1$ and $\Theta, X <: S_0 \vdash_A S_1 <: T_1$. By induction hypothesis, $\Theta, X <: S_0 \vdash_A R_1 <: T_1$ and thus $\Theta \vdash_A \forall^{\mathsf{K}}(X <: R_0).R_1 <: \forall^{\top}(X <: T_0).T_1$ as required.
- $R \equiv \forall^{\mathsf{K}}(X <: R_0).R_1$, and $S \equiv \forall^{\top}(X <: S_0).S_1$ and $T \equiv \forall^{\top}(X <: T_0).T_1$. Then $\Theta \vdash_A S_0 <: R_0$ and $\Theta, X <: R_0 \vdash_A R_1 <: S_1$, and $\Theta \vdash_A T_0 <: S_0$, and $\Theta, X <: \top \vdash_A S_1 <: T_1$ (and hence $\Theta, X <: R_0 \vdash_A S_1 <: T_1$).
  By induction hypothesis, $\Theta \vdash_A T_0 <: R_0$, and $\Theta, X <: R_0 \vdash_A R_1 <: T_1$ and thus $\Theta \vdash_A \forall^{\mathsf{K}}(X <: R_0).R_1 <: \forall^{\top}(X <: T_0).T_1$ as required.

□

Admissibility of the transitivity rule is the property which fails for the restriction of (undecorated) System $\mathsf{F}_{<:}$ to $\forall - \mathsf{Loc}$: note that the proof of Lemma 6.1 depends on the fact that if $\Theta \vdash_A R <: S$ and $\Theta \vdash_A S <: T$ are derivable, at most one of these derivations may terminate with the rule $\forall - \mathsf{Loc}$. Using 6.1 we establish:

Proposition 6.2 *$\Theta \vdash S <: T$ if and only if $\Theta \vdash_A S <: T$.*

6.1 Typechecking

The algorithm for type-synthesis in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ is also an adaptation from the algorithm for System $\mathsf{F}_{<:}$. It follows a similar pattern to the subtyping algorithm: Table 8 gives rules for deriving a unique minimal type for each typable term $\Theta \vdash t$, via judgments of the form $\Theta \vdash_M t : T$. These make use of the following operation.

Lemma 6.3 *For any type $\Theta \vdash T$, there exists a minimal non-atomic type $\Theta \vdash \Theta^*(T)$ such that $\Theta \vdash T <: \Theta^*(T)$.*

Proof. Define $\Theta \vdash \Theta^*(T)$ by:

$$\Theta^*(T) = \begin{cases} \Theta^*(S) & \text{if } T \equiv X \text{ and } \Theta \equiv \Theta', X <: S, \Theta'' \\ T & \text{otherwise} \end{cases}$$

If T is a non-atomic type then it is immediate that $\Theta^*(T)$ is a $<:$-minimal type such that $\Theta \vdash T <: \Theta^*(T)$ (i.e. for any non-atomic type T', if $\Theta \vdash T <: T'$ then $\Theta \vdash \Theta^*(T) <: T'$). Otherwise, T is a type-variable X, where $\Theta \equiv \Theta', X <: S, \Theta''$ and we prove the lemma by induction on the length of Θ'. □

Any term-in-context may be pattern-matched to the conclusion of exactly one derivation rule in Table 8, yielding a deterministic algorithm for synthesizing a minimal type for each typable term-in-context. Soundness and completeness of this algorithm is established by showing that:

Proposition 6.4 *$\Theta \vdash t : T$ if and only if $\Theta \vdash_M t : S$ for some S such that $\Theta \vdash S <: T$.*

Proof. From right to left, it suffices to check that $\Theta \vdash t : T$ implies $\Theta \vdash t : S$ by observing that each rule of $\vdash_M$ is derivable in $\mathsf{F}^{\mathsf{K}\top}_{<:}$, so that $\Theta \vdash S <: T$ implies $\Theta \vdash t : T$.

The proof of the implication from left to right is by induction on the length of derivation of $\Theta \vdash t : T$. We consider the cases where the last rule in this derivation is introduction of $\forall^{\mathsf{K}}$ or elimination of $\forall^{\top}$.

Suppose the last rule applied is introduction of $\forall^{\mathsf{K}}$, so that $t \equiv \Lambda(X <: S).t'$ and $T \equiv \forall^{\mathsf{K}}(X <: S).T'$, where $\Theta, X <: S \vdash t' : T'$. By hypothesis, $\Theta, X <: S \vdash_M t' : R$ for some R such that $\Theta, X <: S \vdash R <: T'$. Then $\Theta \vdash_M t : \forall^{\mathsf{K}}(X <: S).R$ and $\Theta \vdash \forall^{\mathsf{K}}(X <: S).R <: T$ as required.

Suppose the last rule applied is the elimination of $\forall^{\top}$, so that $t \equiv t'\{S'\}$ and $T \equiv T'[S'/X]$, where $\Theta \vdash t : \forall^{\top} X <: S.T'$ and $\Theta \vdash S' <: S$. By inductive hypothesis, $\Theta \vdash_M t' : R$ for some R such that $\Theta \vdash R <: \forall^{\top}(X <: S).T'$. Then by Lemma 6.3, $\Theta \vdash \Theta^*(R) <: \forall^{\top}(X <: S).T'$, and hence $\Theta^*(R) \equiv \forall^{\mathsf{K}}(X <: S'').T''$ or $\Theta^*(R) \equiv \forall^{\top}(X <: S'').T''$ for some S'', T'' such that $\Theta \vdash S <: S''$ and $\Theta, X <: S'' \vdash T' <: T''$. Then $\Theta \vdash S' <: S''$ and $\Theta \vdash_M t\{S'\} : T''[S'/X]$, where $\Theta \vdash T''[S'/X] <: T = T'[S'/X]$ as required. □

$$\frac{}{\Theta, x : T, \Theta' \vdash_M x : T} \qquad \frac{\Theta, x : S \vdash_M t : T}{\Theta \vdash_M \lambda(x : S).t : S \to T} \qquad \frac{\Theta \vdash_M r : R \qquad \Theta \vdash_M s : S \qquad \Theta \vdash S <: S'}{\Theta \vdash_M r\, s : T}\ \Theta^*(R) = S' \to T$$

$$\frac{}{\Theta \vdash_M \mathsf{top} : \top} \qquad \frac{\Theta, X <: S \vdash_M t : T}{\Theta \vdash_M \Lambda(X <: S).t : \forall^{\mathsf{K}}(X <: S).T}$$

$$\frac{\Theta \vdash_M r : R \qquad \Theta \vdash S <: S'}{\Theta \vdash_M r\{S\} : T[S/X]}\ \Theta^*(R) = \forall^{\mathsf{K}}(X <: S').T \text{ or } \Theta^*(R) = \forall^{\top}(X <: S').T$$

Table 8
Derivation Rules for Minimal Typing Judgments

Hence we have a sound and complete typechecking procedure for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$, as for System $\mathsf{F}_{<:}$— accept the typing $\Theta \vdash t : T$ if the minimal typing algorithm produces a typing $\Theta \vdash_M t : S$ and the subtyping algorithm accepts $\Theta \vdash_A S <: T$.

7 Typing System $\mathsf{F}^{\top}_{<:}$ terms in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$

Given a type of System $\mathsf{F}_{<:}$, how should we decorate its bounded quantifiers? The obvious answer is to do so uniformly — i.e. choose either $\forall^{\mathsf{K}}$ or $\forall^{\top}$ for all of the quantifiers in the types of variables, giving two possible translations of undecorated types (and type-annotated terms) into System $\mathsf{F}^{\mathsf{K}\top}_{<:}$.

The first of these choices leads back to Kernel $\mathsf{F}_{<:}$. Writing $\Theta \vdash^{\mathsf{K}} T$ and $\Theta \vdash^{\mathsf{K}} t$ for Kernel $\mathsf{F}_{<:}$ types and terms (i.e. all of the quantifiers in Θ and T and t are instances of $\forall^{\mathsf{K}}$), and $\Theta \vdash^{\mathsf{K}} S <: T$ and $\Theta \vdash^{\mathsf{K}} t : T$ for Kernel $\mathsf{F}_{<:}$ subtyping and typing judgments (i.e. those derivable in Kernel $\mathsf{F}_{<:}$, if the decorating superscripts are erased) it is straightforward to show that:

- If $\Theta \vdash S <: T$, where $\Theta \vdash^{\mathsf{K}} S, T$, then $\Theta \vdash^{\mathsf{K}} S <: T$.
- If $\Theta \vdash_M t : S$, where $\Theta \vdash^{\mathsf{K}} t$, then $\Theta \vdash^{\mathsf{K}} t : S$.

and hence that for Kernel $\mathsf{F}_{<:}$ types and terms, $\Theta \vdash t : T$ if and only if $\Theta \vdash^{\mathsf{K}} t : T$. In other words, System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ is a conservative extension of Kernel $\mathsf{F}_{<:}$.

What of System $\mathsf{F}^{\top}_{<:}$? It is again straightforward to show that System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ conservatively extends the $\mathsf{F}^{\top}_{<:}$ subtyping relation: for $\mathsf{F}^{\top}_{<:}$-types, $\Theta \vdash S <: T$ implies $\Theta \vdash^{\top} S <: T$.[1] However, there are $\mathsf{F}^{\top}_{<:}$-terms which may be typed with a $\mathsf{F}^{\top}_{<:}$-type in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ but are not typable in System $\mathsf{F}^{\top}_{<:}$ itself. (Compare the following with the example given in Proposition 2.2.)

Proposition 7.1 *System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ is not a conservative extension of System $\mathsf{F}^{\top}_{<:}$.*

Proof. Let $X <: \top \vdash u$ be the $\mathsf{F}^{\top}_{<:}$-term $X <: \top \vdash^{\top} (\Lambda Y.\Lambda(Z <: X).\lambda(y : Y).y)\{X\}$. Then $X <: \top \vdash u : \forall^{\top}(Z <: X).Z \to X$, since we may derive the minimal type $X <: \top \vdash_M u : \forall^{\mathsf{K}}(Z <: X).X \to X$ in $\mathsf{F}^{\mathsf{K}\top}_{<:}$, and $X <: \top \vdash \forall^{\mathsf{K}}(Z <: X).X \to X <: \forall^{\top}(Z <: X).Z \to X$ by $\forall - \mathsf{Loc}$, so $X <: \top \vdash u : \forall^{\top}(Z <: X).Z \to X$ by subsumption.

However, this typing is not valid in System $\mathsf{F}^{\top}_{<:}$. Suppose $X <: \top \vdash^{\top} u : \forall^{\top}(Z <: X).Z \to X$. Then $X <: \top \vdash^{\top} \Lambda Y.\Lambda(Z <: X)\lambda(y : Y).y : \forall^{\top} Y.T$ for some $\mathsf{F}^{\top}_{<:}$-type $X <: \top, Y <: \top \vdash T$ such that $X <: \top \vdash T[X/Y] <: \forall^{\top}(Z <: X).Z \to X$ — i.e. $T[X/Y] \equiv \forall^{\top}(Z <: T_0).T_1 \to T_2$, where in particular $X <: \top, Y <: \top, Z <: \top \vdash Z <: T_1$, and so $T_1 \equiv Z$ or $T_1 \equiv \top$. But the only types for $\Lambda Y.\Lambda(Z <: X)\lambda(y : Y).y$ in System $\mathsf{F}^{\top}_{<:}$ are $X <: \top \vdash \forall^{\top} Y.\forall^{\top}(Z <: X).Y \to Y$, $X <: \top \vdash \forall^{\top} Y.\forall^{\top}(Z <: X).Y \to \top$, $X <: \top \vdash \forall^{\top} Y.\top$ and $X <: \top \vdash \top$.[2] □

Note that the β-normal form of u — $X <: \top \vdash^{\top} \Lambda(Z <: X).\lambda(y : X).y$ — *is* typable in System $\mathsf{F}^{\top}_{<:}$ with $X <: \top \vdash^{\top} \forall^{\top}(Z <: X).Z \to X$. In fact this is true in general: System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ typing is conservative over

[1] Writing $\Theta \vdash^{\top} T$ and $\Theta \vdash^{\top} t$ if $\Theta \vdash T$ and $\Theta \vdash t$ are $\mathsf{F}^{\top}_{<:}$ types and terms, and $\Theta \vdash^{\top} S <: T$ and $\Theta \vdash^{\top} t : T$ if these are $\mathsf{F}^{\top}_{<:}$ subtyping and typing judgments (derivable in System $\mathsf{F}^{\top}_{<:}$ if decorating superscripts are erased).

[2] Since this term is β-normal, by Proposition 7.3 below we may use the type-synthesis algorithm for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ to derive these types.

System $\mathsf{F}^{\top}_{<:}$ when restricted to β-normal forms, as we now show.

Lemma 7.2 *If $\Theta \vdash_M t : T$, where $\Theta \vdash^{\top} t$ is β-normal and not a (λ or Λ) abstraction, then $\Theta \vdash^{\top} T$.*

Proof. By induction on the length of t.

- If $t \equiv \mathsf{top}$, then $T \equiv \top$, which is a $\mathsf{F}^{\top}_{<:}$-type.
- If $t \equiv x$ for some variable x, then $\Theta \equiv \Theta', x : T, \Theta''$ and so T is a $\mathsf{F}^{\top}_{<:}$ type by assumption.
- If $t \equiv t't''$ then $\Theta \vdash_M t' : S \to T$ for some S. Since t is β-normal, t' is not an abstraction. By hypothesis, $S \to T$ (and hence also T) is a $\mathsf{F}^{\top}_{<:}$ type.
- If $t \equiv t'\{S\}$ then $\Theta \vdash_M t' : \forall^{\top}(X <: S').T'$, where T' is an $\mathsf{F}^{\top}_{<:}$ type (since t is β-normal and not an abstraction). So $T \equiv T'[S/X]$ is a $\mathsf{F}^{\top}_{<:}$-type.

□

Proposition 7.3 *If $\Theta \vdash t : T$, where $\Theta \vdash^{\top} t$ is β-normal and $\Theta \vdash^{\top} T$ then $\Theta \vdash^{\top} t : T$.*

Proof. By induction on the length of t. If $T = \top$ then evidently $\Theta \vdash^{\top} t : T$. Otherwise, suppose $\Theta \vdash_M t : S$ where $\Theta \vdash S <: T$ and $T \not\equiv \top$:

- If $t \equiv x$ then $\Theta \equiv \Theta', x : S, \Theta''$ and so $\Theta \vdash^{\top} t : S$ and $\Theta \vdash^{\top} t : T$ by conservativity of subtyping.
- If $t \equiv t't''$ then $\Theta \vdash_M t' : S'$, where $\Theta^*(S') = R \to S$ and $\Theta \vdash_M t'' : R$ for some types R, S'. Then t' is not an abstraction (as t is β-normal) and so by Lemma 7.2, S' is a $\mathsf{F}^{\top}_{<:}$ type, and hence so are $R \to S$ and R. By hypothesis $\Theta \vdash^{\top} t' : R \to S$. and $\Theta \vdash^{\top} t'' : R$, so $\Theta \vdash^{\top} t : T$ as required.
- If $t \equiv \lambda(x : R).t'$ then $S \equiv R \to S'$, where $\Theta, x : R \vdash_M t' : S'$, and $T \equiv R' \to T'$, where $\Theta \vdash R <: R'$ and $\Theta \vdash S' <: T'$. By hypothesis $\Theta, x : R \vdash^{\top} t' : S'$ and hence $\Theta, \vdash^{\top} t : T$ as required.
- If $t \equiv t'\{R\}$ then since t' is not an abstraction its minimal type is a $\mathsf{F}^{\top}_{<:}$ type by Lemma 7.2. So $\Theta \vdash_M t' : S'$, where $\Theta^*(S') = \forall^{\top}(X <: R').S''$ and $S \equiv S''[R'/X]$. By hypothesis (since S' is a $\mathsf{F}^{\top}_{<:}$ type), $\Theta \vdash^{\top} t' : S'$, and so $\Theta \vdash^{\top} t : T$ as required.
- If $t \equiv \Lambda(X <: R).t'$ then $S \equiv \forall^{\mathsf{K}}(X <: R).S'$ and $T \equiv \forall^{\top}(X <: R').T'$, for some R, R', S', T' such that $\Theta, X <: S' \vdash_M t' : R'$, $\Theta \vdash R <: R'$ and $\Theta, X <: R' \vdash S' <: T'$. Then by hypothesis $\Theta, X <: R' \vdash^{\top} t' : T'$ and hence $\Theta \vdash^{\top} t : T$ as required.

□

In semantic terms, System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ is thus a conservative extension of System $\mathsf{F}^{\top}_{<:}$. However, by supplying the missing minimal types it satisfies more cases of *subject expansion*.

8 Decidability of typechecking terms of $\mathsf{F}^{\top}_{<:}$ in $\mathsf{F}^{\mathsf{K}\top}_{<:}$

It is straightforward to show that (as in System $\mathsf{F}_{<:}$) the typechecking algorithm for System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ terminates on a given input if and only if every call made to the subtyping algorithm terminates. So decidability of typechecking boils down to termination of these calls. We do not know whether the subtyping algorithm determined by the rules in Table 7 terminates in general, nor whether a terminating algorithm exists. The culprit is the rule $\forall - \mathsf{Loc}$ used to infer the subtyping relation between types quantified by $\forall^{\mathsf{K}}$ and $\forall^{\top}$, which introduces a convoluted form of the rebounding problem encountered in System $\mathsf{F}_{<:}$ itself: $\forall - \mathsf{Loc}$ does not reduce the simple metrics on subtyping judgments used to prove termination for System $\mathsf{F}^{\top}_{<:}$ or Kernel $\mathsf{F}_{<:}$, but the arguments used to show undecidability of subtyping in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$ do not apply [5] either.

However, for uniformly decorated types this problem does not arise. Decidability of typechecking for Kernel $\mathsf{F}_{<:}$ terms follows by conservativity; here we show that typechecking of $\mathsf{F}^{\top}_{<:}$-terms with $\mathsf{F}^{\top}_{<:}$-types is also decidable, by showing that the proof of decidability of subtyping for System $\mathsf{F}^{\top}_{<:}$ shown in [5] extends to the minimal types inferred for $\mathsf{F}^{\top}_{<:}$-terms in System $\mathsf{F}^{\mathsf{K}\top}_{<:}$.

Definition 8.1 The *minimal types for* $\mathsf{F}^{\top}_{<:}$ are the $\mathsf{F}^{\mathsf{K}\top}_{<:}$ types given by the grammar:

$$T ::= S \mid \forall^{\mathsf{K}}(X <: S).T \mid S \to T$$

where S ranges over the $\mathsf{F}^{\top}_{<:}$-types.

The following lemma justifies the terminology.

Lemma 8.2 *For any* $\mathsf{F}^{\top}_{<:}$*-term* $\Theta \vdash^{\top} t$*, if* $\Theta \vdash_M t : T$ *then* T *is a minimal type for* $\mathsf{F}^{\top}_{<:}$*.*

Proof. By induction on the length of t:

- If t is a variable then its minimal type is that assigned to it in Θ, which is a $\mathsf{F}^{\top}_{<:}$-type.
- If $t \equiv \Lambda(x : S).t'$ then $\Theta, X : S \vdash_M t' : T'$, where $\Theta \vdash S$ is a $\mathsf{F}^{\top}_{<:}$-type and $\Theta \vdash T'$ is a minimal type for $\mathsf{F}^{\top}_{<:}$ by hypothesis, and so $T \equiv \forall^{\mathsf{K}}(X <: S).T'$ is a minimal type for $\mathsf{F}^{\top}_{<:}$.
- If $t \equiv t'\{S\}$ then $\Theta \vdash_M t' : \forall^{\mathsf{K}}(X <: S').T'$. By hypothesis $\forall^{\mathsf{K}}(X <: S').T'$ is a minimal type for $\mathsf{F}^{\top}_{<:}$ and hence so is T'. S is a $\mathsf{F}^{\top}_{<:}$-type, and it is straightforward to check that $T \equiv T'[S/X]$ is therefore a minimal type for $\mathsf{F}^{\top}_{<:}$.
- The cases $t \equiv \top$, $t \equiv \lambda(x : S).t'$ and $t \equiv t'\,t''$ are similar.

□

Proposition 8.3 *If* $\Theta \vdash S$ *is a minimal type for* $\mathsf{F}^{\top}_{<:}$*, and* $\Theta \vdash^{\top} T$ *is a* $\mathsf{F}^{\top}_{<:}$*-type, then the subtyping algorithm terminates on* $\Theta \vdash_A S <: T$*.*

Proof. By induction on the size of S. If it is a $\mathsf{F}^{\top}_{<:}$-type then the subtyping algorithm terminates by the proof of Castagna and Pierce [5]. The remaining cases are:

- $S \equiv \forall^{\mathsf{K}}(X <: S_0).S_1$. If $T \equiv \forall^{\top}(X <: T_0).T_1$ then by induction hypothesis the algorithm terminates on $\Theta \vdash_A T_0 <: S_0$ and $\Theta, X <: S_0 \vdash_A S_1 <: T_1$ and hence terminates on $\Theta \vdash_A S <: T$. Otherwise, either $T \equiv \top$, and so $\Theta \vdash_A S <: T$ is accepted, or $T \not\equiv \top$ and it is rejected immediately.
- $S \equiv S_0 \to S_1$ (similar).

□

Proposition 8.4 *The minimal typing algorithm terminates on any* $\mathsf{F}^{\top}_{<:}$*-term* $\Theta \vdash^{\top} t$*.*

Proof. By induction on the size of t, verifying that the minimal typing algorithm calls the subtyping algorithm only on terminating inputs.

Suppose, for example, that $t \equiv t'\{S\}$. By induction, the algorithm either rejects $\Theta \vdash t'$ or finds a minimal typing $\Theta \vdash_M t : T'$, where $\Theta \vdash T'$ is a $\mathsf{F}^{\top}_{<:}$-minimal type by Lemma 8.2, and hence so is $\Theta^*(T')$. If $\Theta^*(T') \equiv \forall^{\mathsf{K}}(X <: T_0).T_1$ or $\Theta^*(T') \equiv \forall^{\top}(X <: T_0).T_1$ then T_0 is a $\mathsf{F}^{\top}_{<:}$-type, and the subtyping algorithm either accepts or rejects $\Theta \vdash_A S <: T_0$ by Lemma 8.3: in the former case the minimal typing algorithm returns $T_1[S/X]$ as the minimal type of t, otherwise (or if $\Theta^*(T')$ is not a bounded quantification) it rejects. □

Proposition 8.5 *Typechecking of* $\mathsf{F}^{\top}_{<:}$*-terms in System* $\mathsf{F}^{\mathsf{K}\top}_{<:}$ *is decidable.*

Proof. For any $\mathsf{F}^{\top}_{<:}$-term $\Theta \vdash^{\top} t$ and $\mathsf{F}^{\top}_{<:}$-type $\Theta \vdash^{\top} T$, by Proposition 8.4 the minimal typing algorithm either rejects $\Theta \vdash t$ or produces a $\mathsf{F}^{\top}_{<:}$-minimal type $\Theta \vdash_M S$, in which case the the subtyping algorithm either rejects $\Theta \vdash_A S <: T$ or accepts it — and thus the typing $\Theta \vdash t : T$ — by Lemma 8.3. □

By Proposition 7.3 this extends to typechecking of β-normal terms in $\mathsf{F}^{\top}_{<:}$ itself.

Corollary 8.6 *Typechecking of* β*-normal terms in System* $\mathsf{F}^{\top}_{<:}$ *is decidable.*

9 Conclusions and Further Directions

We have described a semantics with two related interpretations of bounded quantification. Although this was presented via a simple syntactic reduction of bounded to unbounded quantification, its soundness depends fundamentally on a key semantic property, *dinaturality*, to relate subtype and parametric polymorphism, and arose from a more general investigation into the denotational semantics of bounded quantification.

These semantic insights were applied to give a type system which subsumes both Kernel $\mathsf{F}_{<:}$ and System $\mathsf{F}^{\top}_{<:}$. This sheds light on some of the troublesome aspects of the latter, in particular, by supplying its missing minimal types. The price for this more well-behaved system — having two forms of bounded quantification — need not be paid by the programmer: by restricting to programs annotated with $\mathsf{F}^{\top}_{<:}$-types, we arrive at a system in which typechecking is decidable and the same set of β-normal forms can be typed as in System $\mathsf{F}^{\top}_{<:}$ itself.

This treatment of bounded quantification is not dependent on the λ-calculus setting of System $\mathsf{F}^{\top}_{<:}$, and may transfer to related type systems such as the DOT calculus, where similar problems arise. Indeed, strong Kernel $\mathsf{F}_{<:}$ [9], which is similarly a fragment of System $\mathsf{F}_{<:}$ which achieves both decidability and greater expressiveness than Kernel $\mathsf{F}_{<:}$, is derived from an analogous fragment of the type system $\mathsf{D}_{<:}$, which is the part of DOT without self-referencing and intersection types. Strong Kernel $\mathsf{F}_{<:}$ avoids the rebounding problem by deriving subtyping judgments with respect to two contexts, which may have different bounds for the same variable. A semantic account of this calculus (which is part of the broader aim to develop an intensional denotational semantics of object-oriented programming) may shed light on its expressiveness and relation to $\mathsf{F}^{\mathsf{K}\top}_{<:}$.

Another conclusion could be drawn from our semantic analysis: since meet types and dinaturality may be used to interpret both forms of bounded quantifier in terms of unbounded quantification, why not interpret programs directly in such a system (for which subtyping and typechecking are straightforward)?

References

[1] Bainbridge, E. S., P. J. Freyd, A. Scedrov and P. Scott, *Functorial polymorphism*, Theoretical Computer Science **70**, pages 35–64 (1990).
https://doi.org/10.1016/0304-3975(90)90055-m

[2] Bruce, K. and G. Longo, *A modest model of records, inheritance and bounded quantification*, Information and Computation **87**, pages 196–240 (1990).
https://doi.org/10.1109/lics.1988.5099

[3] Cardelli, L., J. C. Mitchell, S. Martini and A. Scedrov, *An extension of System F with subtyping*, Information and Computation **109**, pages 4–56 (1994).
https://doi.org/10.1006/inco.1994.1013

[4] Cardelli, L. and P. Wegner, *On understanding types, data abstraction and polymorphism*, Computing Surveys **17**, pages 471 – 522 (1985).
https://doi.org/10.1145/6041.6042

[5] Castagna, G. and B. C. Pierce, *Decidable bounded quantification*, in: *Proceedings of POPL '94*, pages 1–29 (1994).
https://doi.org/10.1145/174675.177844

[6] Chroboczek, J., *Game semantics and subtyping*, in: *Proceedings of the fifteenth annual symposium on Logic in Computer Science*, pages 192–203, IEEE press (2000).
https://doi.org/10.1109/lics.2000.855769

[7] Curien, P.-L. and G. Ghelli, *Coherence of subsumption, minimum typing and type-checking in* $\mathsf{F}_{<:}$, Mathematical Structures in Computer Science **2**, pages 55 – 91 (1992).
https://doi.org/10.1007/3-540-52590-4_45

[8] de Lataillade, J., *Dinatural terms in System F*, in: *Proceedings of the 24th annual symposium on Logic in Computer Science, LICS '09*, IEEE Press (2009).
https://doi.org/10.1109/lics.2009.30

[9] Hu, J. Z. S. and O. Lhoták, *Undecidability of* $\mathsf{D}_{<:}$ *and its decidable fragments*, Proceedings of the ACM on Programming Languages (POPL) **4**, pages 1–30 (2020).
https://doi.org/10.1145/3371077

[10] Katiyar, D. and S. Sankar, *Completely bounded quantification is decidable*, in: *Proceedings of the ACM SIGPLAN Workshop on ML and its Applications*, pages 68–77 (1992).
https://www.researchgate.net/publication/2763874_Completely_Bounded_Quantification_is_Decidable

[11] Laird, J., *Game semantics for bounded polymorphism*, in: *Proceedings of FoSSaCS '16*, number 9634 in LNCS, Springer (2016).
https://doi.org/10.1007/978-3-662-49630-5_4

[12] N. Amin, S. Grütter, M. Odersky, T. Rompf and S. Stucki, *The essence of dependent object types.*, in: *A List of Successes That Can Change the World - Essays Dedicated to Philip Wadler on the Occasion of His 60th Birthday*, number 9600 in LNCS, pages 249 – 272, Springer (2016).
https://doi.org/10.1007/978-3-319-30936-1_14

[13] Pierce, B. C., *Programming with Intersection Types and Bounded Polymorphism*, Ph.D. thesis, Carnegie Mellon University (1991).
https://doi.org/10.5555/145640

[14] Pierce, B. C., *Bounded quantification is undecidable*, in: *POPL*, pages 305–315 (1992).
https://doi.org/10.1006/inco.1994.1055

[15] Rompf, T. and N. Amin, *From F to DOT: type soundness proofs with definitional interpreters*, CoRR **abs/1510.05216** (2015). 1510.05216.
http://arxiv.org/abs/1510.05216

[16] Vorobyov, S., *Structural decidable extensions of bounded quantification*, in: *Proceedings of POPL '95*, pages 164–175 (1995).
https://doi.org/10.1145/199448.199479