Noname manuscript No.
(will be inserted by the editor)

Judgmental subtyping systems with intersection types and modal types

Jeongbong Seo · Sungwoo Park

Received: date / Accepted: date

Abstract We study how to extend modal type systems based on intuitionistic modal logic S4 or S5 with a subtyping system based on intersection types. In the presence of four type constructors $\rightarrow$, $\wedge$, $\Box$, and $\Diamond$, the traditional approach using a binary subtyping relation does not work well because of lack of orthogonality in subtyping rules and presence of a transitivity rule. We adopt the idea from the judgmental formulation of modal logic [15] and use subtyping judgments whose definitions express those notions internalized into type constructors directly at the level of judgments. The resultant judgmental subtyping systems admit cut rules similarly to a sequent calculus for intuitionistic logic and play a key role in designing and verifying the relational subtyping systems based on the binary subtyping relation. We use the proof assistant Coq to prove the admissibility of the cut rules and the equivalence between the two kinds of subtyping systems. The lesson from our study is that by using subtyping judgments instead of the binary subtyping relation, we can overcome the limitation usually associated with the syntactic approach to formulating subtyping systems.

1 Introduction

In designing programming languages for network environments, the spatial interpretation of modal logic has inspired a number of type systems via the Curry-Howard correspondence. The basic idea is to interpret necessity modal types $\Box A$ as types of mobile terms which are valid at every node in the network, and possibility modal types $\Diamond A$ as types of local terms which are valid at

Jeongbong Seo · Sungwoo Park
Pohang University of Science and Technology
San 31 Hyojadong Namgu Pohang Gyungbuk, 790-784, Republic of Korea
Tel.: +82-54-279-2386
Fax.: +82-54-279-2299
E-mail: {baramseo,gla}@postech.ac.kr

some node in the network. The underlying modal logic, such as S4 or S5, regulates the interaction between modal types, and we obtain via the Curry-Howard correspondence a modal type system that is suitable for distributed programming languages such as λ_{rpc}[8] and Lambda 5 [12] or task description languages such as MTSN (Modal Type System for Networks) [2].

This paper studies how to extend such modal type systems based on intuitionistic modal logic S4 or S5 with a subtyping system based on intersection types. (Throughout the paper, we assume intuitionistic modal logics S4 and S5 rather than their classical counterparts.) An intersection type $A \wedge B$ is inhabited by those terms that can have both types A and B simultaneously, and thus has A and B as its supertypes. It provides a limited form of polymorphism that is a good compromise between expressiveness and simplicity in the presence of modal types, as illustrated below:

- A distributed programming language may assign a modal intersection type $\Box((\mathsf{int} \to \mathsf{int}) \wedge (\mathsf{float} \to \mathsf{float}))$ to a mobile polymorphic function in order to specialize it for two base types int and float. Assigning a modal polymorphic type $\Box(\forall\alpha.\alpha \to \alpha)$ instead would increase the utility of the function, but would also considerably complicate type inference because polymorphic types are no longer in prenex form.
- A task description language may use a modal intersection type $\Diamond((\mathsf{PDF} \to \mathsf{PS}) \wedge (\mathsf{PS} \to \mathsf{PDF}))$ for a local service that converts between PDF and PS files. Without intersection types, we would instead use a modal sum type $\Diamond((\mathsf{PDF} \vee \mathsf{PS}) \to (\mathsf{PDF} \vee \mathsf{PS}))$, which is less accurate.

Thus a subtyping system based on intersection types is particularly useful as an extension to modal type systems into which a general form of polymorphism is too expensive to incorporate.

The use of a subtyping relation $A \leq B$ as in the traditional approach, however, is not an ideal decision for designing such a subtyping system because two inherent problems, lack of orthogonality in subtyping rules and presence of a transitivity rule, are seriously exacerbated by four independent type constructors ($\to$, $\wedge$, $\Box$, and $\Diamond$). Lack of orthogonality arises because a single subtyping rule may have to analyze multiple type constructors that interact with each other. When mixing the four type constructors, however, we find it difficult even to recognize the interaction between different type constructors, let alone to propose a corresponding subtyping rule. In our study of a subtyping system based on modal logic S4, for example, we discover the need for the following subtyping rule only during the proof of its equivalence with another subtyping system:

$$\frac{\Box C \wedge A \leq B}{\Box C \wedge \Diamond A \leq \Diamond B}\ \textsf{R-box-dia-K}$$

Moreover the use of a subtyping relation gives rise to distributivity rules, in conjunction with which a transitivity rule makes it particularly difficult to reason about the subtyping system and to derive a decision procedure for the subtyping relation.

Instead of using a binary relation between two types, we adopt the idea from the judgmental formulation of modal logic [15] and use subtyping judgments whose definitions express those notions internalized into type constructors directly at the level of judgments and thus do not depend on any type constructors. Our subtyping system based on modal logic S4 uses two subtyping judgments where subtyping contexts Γ and Σ are collections of types:

- A subtyping judgment $\Gamma \mid \Sigma \preceq C$ can be interpreted as corresponding to a subtyping relation $(\bigwedge_{A_i \in \Gamma} \Box A_i) \wedge (\bigwedge_{B_j \in \Sigma} B_j) \leq C$ and thus exploits those notions internalized into $\Box$ and $\wedge$.
- Another subtyping judgment $\Gamma \mid \Sigma \div C$ can be interpreted as corresponding to a subtyping relation $(\bigwedge_{A_i \in \Gamma} \Box A_i) \wedge (\bigwedge_{B_j \in \Sigma} B_j) \leq \Diamond C$ and thus exploits those notions internalized into $\Box$, $\Diamond$, and $\wedge$.

Our subtyping system based on modal logic S5 uses a single subtyping judgment $\Delta \mid \Gamma \mid \Sigma \preceq C$ which uses Δ as a collection of subtyping contexts and can be interpreted as corresponding to a subtyping relation

$$(\bigwedge_{\Sigma_k \in \Delta} \Diamond(\bigwedge_{C_l^k \in \Sigma_k} C_l^k)) \wedge (\bigwedge_{A_i \in \Gamma} \Box A_i) \wedge (\bigwedge_{B_j \in \Sigma} B_j) \leq C.$$

By designing every subtyping rule so that it deals with at most one type constructor, we obtain a subtyping system in which all subtyping rules are orthogonal. Transitivity is also implicit and deriving a decision procedure is straightforward.

Given a set of type constructors and their logical properties, we formulate two subtyping systems: a *relational* subtyping system using the subtyping relation $A \leq B$ and a *judgmental* subtyping system using subtyping judgments. The relational subtyping system explicitly includes a transitivity rule whereas the judgmental subtyping system admits cut rules similarly to a sequent calculus for intuitionistic logic. Then we prove the admissibility of the cut rules and the equivalence between the two subtyping systems. In addition, we use the proof assistant Coq to mechanize these proofs.

Instead of presenting all subtyping systems at once, we develop pairs of relational and judgmental subtyping systems in an incremental way.

- We begin with base subtyping systems *R-Base* and *J-Base* which include only two type constructors $\rightarrow$ and $\wedge$ (Section 2). The study of the base subtyping systems illustrates the disadvantage of the relational subtyping system *R-Base* and the difficulty of embedding the transitivity rule, as well as the advantage of the judgmental subtyping system *J-Base* due to orthogonality in subtyping rules and the admissibility of the cut rule. It also outlines the development of the metatheory in subsequent subtyping systems, in particular the proof of the admissibility of the cut rules.
- Next we extend the base subtyping systems to develop subtyping systems based on modal logic S4 (Section 3), first considering modalities $\Box$ and $\Diamond$ individually (subtyping systems *R-S4-*$\Box$ and *J-S4-*$\Box$ in Section 3.1 and subtyping systems *R-S4-*$\Diamond$ and *J-S4-*$\Diamond$ in Section 3.2) and then putting

both together to account for their interaction (subtyping systems *R-S4*-$\Box\Diamond$ and *J-S4*-$\Box\Diamond$ in Section 3.3). Here we explain how axioms in modal logic S4 relate to subtyping relations, and show that the judgmental subtyping systems (*J-S4*-$\Box$, *J-S4*-$\Diamond$, and *J-S4*-$\Box\Diamond$) are easier to verify than the relational subtyping systems (*R-S4*-$\Box$, *R-S4*-$\Diamond$, and *R-S4*-$\Box\Diamond$).
- Finally we develop subtyping systems *R-S5*-$\Box\Diamond$ and *J-S5*-$\Box\Diamond$ based on modal logic S5 (Section 4). Here we reuse the relational subtyping system *R-S4*-$\Box\Diamond$ based on modal logic S4 by extending it with two new subtyping rules, but design the judgmental subtyping system *J-S5*-$\Box\Diamond$ afresh by introducing a new subtyping judgment.

We discuss related work in Section 5 and conclude in Section 6. The lesson from our study is that by using subtyping judgments instead of subtyping relations, we can take full advantage of the syntactic approach to formulating a subtyping system (which develops a system of inference rules) while simultaneously overcoming the limitation associated with it. Figure 1 shows the complete relational subtyping system with all the type constructors developed in this incremental way.

We prove all the theorems in Sections 2, 3.3, and 4 in the proof assistant Coq. (We do not mechanize the theorems in Sections 3.1 and 3.2 which can be derived as special cases of the corresponding theorems in Section 3.3.) Our Coq scripts consist of about 9400 lines of code (including blank lines and comments): about 1300 lines for the base subtyping systems, about 2700 lines for the subtyping systems based on S4, and about 5300 lines for the subtyping systems based on S5. As an integral part of the development, we write the Coq scripts in parallel with designing the subtyping systems, rather than after verifying their correctness, so that we can take full advantage of programming in the proof assistant Coq. We believe that the Coq scripts are interesting in their own right, but this paper does not present their details because its focus is on the design and verification of the subtyping systems. All our Coq scripts are available at `http://pl.postech.ac.kr/subtyping/`.

Throughout this paper, we use conventional precedence rules for type constructors: $\Box = \Diamond > \wedge > \rightarrow$.

2 Base subtyping systems *R-Base* and *J-Base*

In this section, we present the base subtyping systems with primitive types P, function types $A \rightarrow B$, and intersection types $A \wedge B$:

$$\text{type} \qquad A, B, C, \cdots \ ::= \ P \mid A \rightarrow B \mid A \wedge B$$

First we develop the relational subtyping system *R-Base*. Then we develop the judgmental subtyping system *J-Base* and prove the admissibility of the cut rule. Finally we compare the two subtyping systems and prove their equivalence.

R-Base, base relational subtyping system with $\to$ and $\wedge$:

$$\frac{}{A \le A}\ \textsf{R-refl} \qquad \frac{A \le C \quad C \le B}{A \le B}\ \textsf{R-trans}$$

$$\frac{C \le A \quad B \le D}{A \to B \le C \to D}\ \textsf{R-fun} \qquad \frac{}{(A \to B) \wedge (A \to C) \le A \to (B \wedge C)}\ \textsf{R-fun-dist}$$

$$\frac{}{A \wedge B \le A}\ \textsf{R-and-l}_1 \qquad \frac{}{A \wedge B \le B}\ \textsf{R-and-l}_2 \qquad \frac{A \le B \quad A \le C}{A \le B \wedge C}\ \textsf{R-and-r}$$

R-S4-$\Box$, extension with $\Box$ in modal logic S4:

$$\frac{}{\Box A \le A}\ \textsf{R-box-T} \qquad \frac{}{\Box A \le \Box\Box A}\ \textsf{R-box-4} \qquad \frac{A \le B}{\Box A \le \Box B}\ \textsf{R-box-K}$$

$$\frac{}{\Box A \wedge \Box B \le \Box(A \wedge B)}\ \textsf{R-box-dist}$$

R-S4-$\Diamond$, extension with $\Diamond$ in modal logic S4:

$$\frac{}{A \le \Diamond A}\ \textsf{R-dia-T} \qquad \frac{}{\Diamond\Diamond A \le \Diamond A}\ \textsf{R-dia-4} \qquad \frac{A \le B}{\Diamond A \le \Diamond B}\ \textsf{R-dia-K}$$

R-S4-$\Box\Diamond$, with the interaction between $\Box$ and $\Diamond$ in modal logic S4:

$$\frac{\Box C \wedge A \le B}{\Box C \wedge \Diamond A \le \Diamond B}\ \textsf{R-box-dia-K}$$

R-S5-$\Box\Diamond$, extension with $\Box$ and $\Diamond$ in modal logic S5:

$$\frac{}{\Diamond A \le \Box\Diamond A}\ \textsf{R-dia-5} \qquad \frac{}{\Diamond\Box A \le \Box A}\ \textsf{R-box-5}$$

Fig. 1 Relational subtyping systems with four type constructors $\to$, $\wedge$, $\Box$, and $\Diamond$

2.1 Relational subtyping system *R-Base*

The topmost box of Figure 1 shows the subtyping rules in the relational subtyping system *R-Base*. A subtyping relation $A \le B$ means that a term of type A also has type B. The rules R-refl and R-trans state reflexivity and transitivity of the subtyping relation, respectively. By the rule R-fun, the type constructor $\to$ is contravariant in the argument and covariant in the result. The rule R-fun-dist is the distributivity rule for $\to$ over intersection types. Since the rule R-fun-dist is unsound in the presence of computational effects [4], our subtyping system assumes no computational effects in the underlying programming language. The rules R-and-l$_1$, R-and-l$_2$, and R-and-r are standard for intersection types and allow us to assume the idempotency, commutativity, and associativity of the type constructor $\wedge$. (Not every type system with intersection types, however, permits all these rules, as in System-$\mathbb{I}$ [9] where A and $A \wedge A$ are not equivalent).

While all the subtyping rules make sense, it is not easy to reason about the relational subtyping system and prove its meta-properties, especially because

$$\frac{}{A \leq A}\ \textsf{R-refl-t} \qquad \frac{C \leq A \quad B \leq D}{A \to B \leq C \to D}\ \textsf{R-fun-t}$$

$$\frac{A \leq C}{A \wedge B \leq C}\ \textsf{R-and-l}_1\textsf{-t} \qquad \frac{B \leq C}{A \wedge B \leq C}\ \textsf{R-and-l}_2\textsf{-t} \qquad \frac{A \leq B \quad A \leq C}{A \leq B \wedge C}\ \textsf{R-and-r-t}$$

$$\frac{C \leq (D \to E) \wedge (D \to F) \quad A \leq D \quad E \wedge F \leq B}{C \leq A \to B}\ \textsf{R-fun-dist-t}$$

Fig. 2 Base relational subtyping system without a transitivity rule by Laurent [10]

of the transitivity rule R-trans. For example, the following two propositions appear to hold, but it is not immediately obvious how to prove them:

Proposition 1 *If $A \to B \leq C \to D$, then $C \leq A$ and $B \leq D$.*

Proposition 2 *If $(A \to B) \wedge E \leq C \to D$, then for any type A' such that $C \leq A'$, we have $(A' \to B) \wedge E \leq C \to D$.*

Applying a structural induction on the derivation of $A \to B \leq C \to D$ or $(A \to B) \wedge E \leq C \to D$ fails in the case of the rule R-trans, whose premise may introduce a type that does not allow us to generate induction hypotheses. Proving these propositions requires us to generalize their statements, which does not seem to be trivial.

We can eliminate the transitivity rule R-trans by embedding it into all other subtyping rules, but the resultant system is no better because of the distributivity rule R-fun-dist. The system in Figure 2 by Laurent [10] is the result of embedding the rule R-trans. Although it has no transitivity rule, the new distributivity rule R-fun-dist-t essentially suffers from the same problem as the rule R-trans: the premise introduces three new types (D, E, and F) not present in the conclusion. On the other hand, if the underlying programming language allows computational effects and we exclude the rule R-fun-dist, embedding the rule R-trans is trivial: a fragment of the system in Figure 2 without the rule R-fun-dist-t is the result of embedding the rule R-trans. Hence we attribute the complexity of the relational subtyping system mainly to the distributivity rule R-fun-dist, which is the rule that destroys its orthogonality.

The above problem with the relational subtyping system is primarily due to its use of a binary relation which compares only two types to decide the subtyping relation. By adopting the idea from the judgmental formulation of modal logic [15], our judgmental subtyping system relaxes this restriction and uses a subtyping judgment which directly expresses the notion internalized into type constructor $\wedge$ and compares a collection of types against another type to decide the subtyping relation.

2.2 Judgmental subtyping system *J-Base*

Our judgmental subtyping system *J-Base* uses a subtyping judgment $\Sigma \preceq A$ where subtyping context Σ is an unordered collection of types:

$$\text{subtyping context} \qquad \Sigma \ ::= \ \cdot \mid \Sigma, A$$

$$\frac{}{\Sigma, P \preceq P}\ \textsf{J-refl} \qquad \frac{C \preceq A_i \quad B_1, \cdots, B_n \preceq D \quad 1 \leq i \leq n}{\Sigma, A_1 \to B_1, \cdots, A_n \to B_n \preceq C \to D}\ \textsf{J-fun}$$

$$\frac{\Sigma, A, B \preceq C}{\Sigma, A \wedge B \preceq C}\ \textsf{J-and-l} \qquad \frac{\Sigma \preceq A \quad \Sigma \preceq B}{\Sigma \preceq A \wedge B}\ \textsf{J-and-r}$$

Fig. 3 *J-Base*, base judgmental subtyping system with type constructors $\to$ and $\wedge$

A subtyping judgment $A_1, \cdots, A_n \preceq B$ means that if a term has types $A_1, \cdots, A_n$, it also has type B. Hence we may think of it as implicitly using $\wedge$ between all types in the subtyping context and corresponding to $A_1 \wedge \cdots \wedge A_n \leq B$ in the relational subtyping system *R-Base*. Note, however, that the meaning of the subtyping judgment itself does not depend on any type constructor such as $\wedge$, which follows the design principle underlying the judgmental subtyping system.

Figure 3 shows the rules in the judgmental subtyping system *J-Base*. The rule J-refl can be generalized to another rule that deduces $\Sigma, A \preceq A$ for any type A. The interaction between $\to$ and $\wedge$, which is expressed explicitly by the distributivity rule R-fun-dist in the relational subtyping system *R-Base*, now arises in the rule J-fun as a consequence of implicitly using $\wedge$ between all types in the subtyping context. To see how the rule J-fun works, assume a function that has types $A_i \to B_i$ for $i = 1, \cdots, n$. When applied to a term of type C such that $C \preceq A_i$, it returns a term that is known to have types B_i. Then the term can also have type D because of $B_1, \cdots, B_n \preceq D$, which implies that the function also has type $C \to D$. Note that as n, the number of function types chosen from the subtyping context, increases, we have to prove more subtyping judgments $C \preceq A_i$ from the first premise, but the second premise $B_1, \cdots, B_n \preceq D$ becomes easier to prove. The rules for $\wedge$ are presented in the style of the sequent calculus: the (left) rule J-and-l analyzes $A \wedge B$ in the subtyping context while the (right) rule J-and-r analyzes the type in the right side of the subtyping judgment. The judgmental subtyping system is orthogonal as every rule involves at most one type constructor.

Although the judgmental subtyping system *J-Base* has no transitivity rule, we can prove the admissibility of the cut rule (Theorem 1) which states that the following cut rule is redundant:

$$\frac{\Sigma \preceq A \quad \Sigma', A \preceq C}{\Sigma, \Sigma' \preceq C}\ \mathit{Cut}$$

Its proof uses two structural properties, weakening and contraction, as the proof of the same theorem in a sequent calculus for intuitionistic logic.

Lemma 1 (Weakening) *If $\Sigma \preceq C$, then $\Sigma, A \preceq C$.*

Proof By structural induction on the derivation of $\Sigma \preceq C$.

Lemma 2 (Contraction) *If $\Sigma, A, A \preceq C$, then $\Sigma, A \preceq C$.*

Proof By nested structural induction on type A and the derivation of $\Sigma, A, A \preceq C$.

Theorem 1 (Admissibility of the cut rule in *J-Base*)
If $\Sigma \preceq A$ and $\Sigma', A \preceq C$, then $\Sigma, \Sigma' \preceq C$.

Our proof of Theorem 1 proceeds in three steps, consisting of Lemmas 3, 4, and 5, as in the proof of cut elimination in display logic [1]. It refines the idea of structural cut elimination for intuitionistic propositional logic [14] which proceeds by nested structural induction on type A, the derivation of $\Sigma \preceq A$, and the derivation of $\Sigma', A \preceq C$ (without using the size of derivations at all). Specifically each lemma is designed to focus on analyzing a group of similar cases in such a monolithic proof, thereby contributing to a modular structure of the whole proof. Below we say that $/A/$ holds if $\Sigma \preceq A'$ and $\Sigma', A' \preceq C$ imply $\Sigma, \Sigma' \preceq C$ for any proper constituent type of A (or any proper subformula of A if we call A a formula). We also write $\Sigma, \boxed{\mathrm{A}} \preceq C$ and $\Sigma \preceq \boxed{\mathrm{A}}$ to indicate that A is a principal type of the last inference rule in their derivations. (A principal type of an inference rule is a type in the conclusion that is decomposed into its constituent types in the premise. The rule R-fun has multiple principal types whereas all the other rules have a single principal type.) The proof of Lemma 4 uses Lemma 3 while the proof of Lemma 5 uses Lemma 4. Theorem 1 follows from Lemma 5.

Lemma 3
Suppose that $/A/$ holds. Then $\Sigma \preceq \boxed{A}$ and $\Sigma', \boxed{A} \preceq C$ imply $\Sigma, \Sigma' \preceq C$.

Proof By case analysis of type A (which is a principal type in both derivations). Note that we do not use a structural induction on type A.

Lemma 4
Suppose that $/A/$ holds. Then $\Sigma \preceq \boxed{A}$ and $\Sigma', A \preceq C$ imply $\Sigma, \Sigma' \preceq C$.

Proof By structural induction on the derivation of $\Sigma', A \preceq C$. If C is a principal type, we first use induction hypothesis and then apply the last inference rule in the derivation. If A is a principal type, using Lemma 3 immediately completes the proof. Hence the proof needs to analyze only those cases in which principal types are in Σ'.

Lemma 5
Suppose that $/A/$ holds. Then $\Sigma \preceq A$ and $\Sigma', A \preceq C$ imply $\Sigma, \Sigma' \preceq C$.

Proof By structural induction on the derivation of $\Sigma \preceq A$. If A is a principal type, using Lemma 4 immediately completes the proof. Hence the proof needs to analyze only those cases in which principal types are in Σ.

To prove a similar result for an extension of the base subtyping system *J-Base* with modalities $\Box$ and $\Diamond$ from modal logic S4 or S5, we first adapt the proofs of Lemmas 3, 4, and 5 and then add new cases in Theorem 1. This strategy works well because in addition to achieving a modular structure, dividing the proof of Theorem 1 into three steps has another advantage that

the proof of each individual lemma is easily reusable for such an extension. In our study, maintaining the modular structure is crucial to completing the whole proof because the goal statement itself becomes much more complex and a direct proof attempt by nested induction is impractical (see Theorems 10 and 13). Reusing the proofs of the three lemmas is also equally crucial because we mechanize the whole proof in the proof assistant Coq. Our Coq scripts often reuse existing proofs just by slightly revising custom tactics, which would be practically impossible if we directly proved Theorem 1.

The judgmental subtyping system *J-Base* is decidable because the premise of a subtyping rule either is empty or consists of subtyping judgments strictly smaller than the subtyping judgment in the conclusion (where we define the size of a subtyping judgment as the number of occurrences of type constructors in it). We can also obtain an algorithmic subtyping system as follows. To prove a subtyping judgment $\Sigma \preceq A$, we first apply the rules J-and-l and J-and-r in the bottom-up way until no more instances of the type constructor $\wedge$ can be eliminated. Then we apply either the rule J-refl to complete the proof, or the rule J-fun for the type constructor $\rightarrow$ to initiate the proofs of smaller subtyping judgments.

2.3 Equivalence between the two subtyping systems *R-Base* and *J-Base*

The judgmental subtyping system *J-Base* is equivalent to the structural subtyping system *R-Base*. Theorem 2 proves the soundness of the judgmental subtyping system with respect to the relational subtyping system. It states precisely the intuition behind the subtyping judgment. Theorem 3 proves the completeness of the judgmental subtyping system with respect to the relational subtyping system. Now we may use $A \leq B$ and $A \preceq B$ interchangeably.

Theorem 2 (Soundness of *J-Base* with respect to *R-Base*)
If $A_1, \cdots, A_n \preceq B$, then $A_1 \wedge \cdots \wedge A_n \leq B$.

Proof By structural induction on the derivation of $A_1, \cdots, A_n \preceq B$.

Theorem 3 (Completeness of *J-Base* with respect to *R-Base*)
If $A \leq B$, then $A \preceq B$.

Proof By structural induction on the derivation of $A \leq B$. For the case of the rule R-trans, we use Theorem 1.

We close this section by proving Propositions 1 and 2 to demonstrate that it is easier to reason about the judgmental subtyping system *J-Base* than the relational subtyping system *R-Base*. Proposition 1 follows immediately from the fact that the only way to prove $A \rightarrow B \preceq C \rightarrow D$ is by applying the rule J-fun:

$$\frac{C \preceq A \quad B \preceq D}{A \rightarrow B \preceq C \rightarrow D}\ \textsf{J-fun}$$

Proposition 2 follows from Lemmas 6 and 7.

Lemma 6 *If $A \wedge B \preceq C$, then $A, B \preceq C$.*

Proof By structural induction on the derivation of $A \wedge B \preceq C$.

Lemma 7 *If $A \to B, \Sigma \preceq C \to D$, then for any type A' such that $C \leq A'$, we have $A' \to B, \Sigma \preceq C \to D$.*

Proof By structural induction on the derivation of $A \to B, \Sigma \preceq C \to D$. In the case of the rule J-fun, if the premise contains a subtyping judgment $C \preceq A$, we replace it by $C \preceq A'$ and apply the rule J-fun to deduce $A' \to B, \Sigma \preceq C \to D$. Otherwise we reuse the premise to deduce $A' \to B, \Sigma \preceq C \to D$.

3 Subtyping systems based on modal logic S4

In this section, we extend the base subtyping systems with modalities $\Box$ and $\Diamond$ from modal logic S4:

$$\text{type} \qquad A, B, C, \cdots \ ::= \ P \mid A \to B \mid A \wedge B \mid \Box A \mid \Diamond A$$

Necessity modal types $\Box A$ are assigned to mobile terms which are valid at every accessible node in the network, and possibility modal types $\Diamond A$ are assigned to local terms which are valid at some accessible node in the network. As in modal logic S4, we assume a reflexive and transitive accessibility relation between nodes in the network, which determines a unique set of properties of modal types.

We design the new subtyping systems in such a way that for each axiom $A \supset B$ in modal logic S4 (where $\supset$ denotes logical implication), a corresponding subtyping relation $A \leq B$ is provable. Recall that in a system of logic, the truth of $A \supset B$ ensures via the Curry-Howard correspondence that one can consume a term of type A to produce another term of type B. In a subtyping system, a subtyping relation $A \leq B$ means that one can implicitly convert the type of a term from A to B. Hence, by replacing the notion internalized into $\supset$ with the notion of an implicit conversion of types, we can derive a subtyping system from a system of logic. The case of modal logic S4 (as well as modal logic S5 in Section 4) is particularly interesting because of non-trivial interactions between logical implication and modalities.

We first extend the base subtyping systems with necessity modality $\Box$ (subtyping systems *R-S4-*$\Box$ and *J-S4-*$\Box$ in Section 3.1). Then we independently consider possibility modality $\Diamond$ (subtyping systems *R-S4-*$\Diamond$ and *J-S4-*$\Diamond$ in Section 3.2). Finally we put both modalities together to complete the subtyping systems (*R-S4-*$\Box\Diamond$ and *J-S4-*$\Box\Diamond$ in Section 3.3). For each extension, we develop both a relational subtyping system and a judgmental subtyping system and prove their equivalence in the same way as for the base subtyping systems in Section 2. The key idea for developing the judgmental subtyping systems, which is originally from the judgmental formulation of modal logic [15], is to introduce another (global) subtyping context for $\Box$ and another (possibility) subtyping judgment for $\Diamond$. Figure 4 shows the complete judgmental subtyping system based on modal logic S4.

J-S4-$\Box$, judgmental subtyping system with $\to$, $\wedge$, and $\Box$ in modal logic S4:

$$\frac{}{\Gamma, P \mid \Sigma \preceq P}\ \textsf{J-refl-v4} \qquad \frac{}{\Gamma \mid \Sigma, P \preceq P}\ \textsf{J-refl4}$$

$$\frac{\cdot \mid C \preceq A_i \quad 1 \le i \le n \quad \cdot \mid B_1, \cdots, B_n \preceq D}{\Gamma, A_1 \to B_1, \cdots, A_m \to B_m \mid \Sigma, A_{m+1} \to B_{m+1}, \cdots, A_n \to B_n \preceq C \to D}\ \textsf{J-fun4}$$

$$\frac{\Gamma, A, B \mid \Sigma \preceq C}{\Gamma, A \wedge B \mid \Sigma \preceq C}\ \textsf{J-and-v4} \qquad \frac{\Gamma \mid \Sigma, A, B \preceq C}{\Gamma \mid \Sigma, A \wedge B \preceq C}\ \textsf{J-and-l4}$$

$$\frac{\Gamma \mid \Sigma \preceq A \quad \Gamma \mid \Sigma \preceq B}{\Gamma \mid \Sigma \preceq A \wedge B}\ \textsf{J-and-r4}$$

$$\frac{\Gamma, A \mid \Sigma \preceq B}{\Gamma, \Box A \mid \Sigma \preceq B}\ \textsf{J-box-v4} \qquad \frac{\Gamma, A \mid \Sigma \preceq B}{\Gamma \mid \Sigma, \Box A \preceq B}\ \textsf{J-box-l4} \qquad \frac{\Gamma \mid \cdot \preceq A}{\Gamma \mid \Sigma \preceq \Box A}\ \textsf{J-box-r4}$$

J-S4-$\Box\Diamond$, extension with $\Diamond$ in modal logic S4:

$$\frac{\Gamma, A, B \mid \Sigma \div C}{\Gamma, A \wedge B \mid \Sigma \div C}\ \textsf{J-and-vp4} \qquad \frac{\Gamma \mid \Sigma, A, B \div C}{\Gamma \mid \Sigma, A \wedge B \div C}\ \textsf{J-and-lp4}$$

$$\frac{\Gamma, A \mid \Sigma \div B}{\Gamma, \Box A \mid \Sigma \div B}\ \textsf{J-box-vp4} \qquad \frac{\Gamma, A \mid \Sigma \div C}{\Gamma \mid \Sigma, \Box A \div C}\ \textsf{J-box-lp4}$$

$$\frac{\Gamma \mid \Sigma \preceq A}{\Gamma \mid \Sigma \div A}\ \textsf{J-poss-p4}$$

$$\frac{\Gamma, \Diamond A \mid A \div B}{\Gamma, \Diamond A \mid \Sigma \div B}\ \textsf{J-dia-vp4} \qquad \frac{\Gamma \mid A \div B}{\Gamma \mid \Sigma, \Diamond A \div B}\ \textsf{J-dia-lp4} \qquad \frac{\Gamma \mid \Sigma \div A}{\Gamma \mid \Sigma \preceq \Diamond A}\ \textsf{J-dia-r4}$$

Fig. 4 Judgmental subtyping systems based on modal logic S4

3.1 Extension with necessity modality $\Box$

3.1.1 Relational subtyping system R-S4-$\Box$

The second box in Figure 1 shows four subtyping rules to be added to the base subtyping system *R-Base*. We can justify each subtyping rule according to the accessibility relation between nodes in modal logic S4.

- The rule R-box-T is based on the reflexivity of the accessibility relation. It corresponds to the axiom $\Box A \supset A$ in S4.
- The rule R-box-4 is based on the transitivity of the accessibility relation. It corresponds to the axiom $\Box A \supset \Box\Box A$ in S4.
- The rule R-box-K corresponds to the axiom $\Box(A \supset B) \supset (\Box A \supset \Box B)$ in S4. Note that its premise can be thought of as corresponding to a formula $\Box(A \supset B)$ in S4 because it uses no hypotheses.
- The rule R-box-dist, the distributivity rule for $\Box$ over intersection types, corresponds to the axiom $\Box A \wedge \Box B \supset \Box(A \wedge B)$ in S4 (where we interpret $\wedge$ as logical conjunction). It allows us to use $\Box(A \wedge B)$ and $\Box A \wedge \Box B$ interchangeably because $\Box(A \wedge B)$ is also a subtype of $\Box A \wedge \Box B$.

Note that necessity modality $\Box$ does not interact with function types as neither $\Box(A \to B) \le \Box A \to \Box B$ nor $\Box A \to \Box B \le \Box(A \to B)$ holds. Modal type

$\Box(A \to B)$ describes mobile terms of function type $A \to B$ whereas function type $\Box A \to \Box B$ describes terms that only manipulates mobile terms and are not necessarily mobile. Thus the two types cannot be compared in a subtyping relation.

3.1.2 Judgmental subtyping system J-S4-$\Box$

The judgmental subtyping system *J-S4-$\Box$* uses a new form of subtyping judgment $\Gamma \mid \Sigma \preceq A$ which uses a *global subtyping context* Γ and a *local subtyping context* Σ; both subtyping contexts are unordered collections of types:

$$\begin{array}{lll} \text{global subtyping context} & \Gamma & ::= \ \cdot \mid \Gamma, A \\ \text{local subtyping context} & \Sigma & ::= \ \cdot \mid \Sigma, A \end{array}$$

A subtyping judgment $A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$ means that if a term has types $A_1, \cdots, A_m$ at every accessible node and types $B_1, \cdots, B_n$ at the current node, it also has type C at the current node. Hence it can be thought of as corresponding to $(\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq C$ in the relational subtyping system *R-S4-$\Box$*. Note, however, that the meaning of the subtyping judgment itself does not depend on any type constructor, in particular $\Box$.

The upper box in Figure 4 shows the judgmental subtyping system *J-S4-$\Box$* with modality $\Box$. We justify the rule J-refl-v4 with the reflexivity of the accessibility relation, and the rule J-refl4 with the definition of the subtyping judgment. The rule J-fun4 explains the interaction between $\to$ and $\wedge$ using the same idea as in the rule J-fun in Figure 3. It simultaneously inspects m function types from the global subtyping context and $n - m$ function types from the local subtyping context. The rule J-and-v4 is based on the implicit use of $\wedge$ between those types in the global subtyping context and explains the interaction between $\Box$ and $\wedge$. The rules J-and-l4 and J-and-r4 extend their counterparts in Figure 3 with a global typing context. We justify the rule J-box-v4 with the transitivity of the accessibility relation, and the rule J-box-l4 with the notion internalized into $\Box$. The premise of the rule J-box-r4 describes a situation in which a given term can be assigned type A at an arbitrary accessible node about which nothing is known. Hence the term can be assigned type $\Box A$ as stated in the conclusion. The judgmental subtyping system *J-S4-$\Box$* is orthogonal as every rule involves at most one type constructor.

As the subtyping judgment uses two separate subtyping contexts, we need two cut rules. Accordingly we state the admissibility of the cut rules with two clauses:

Theorem 4 (Admissibility of the cut rules in *J-S4-$\Box$*)
If $\Gamma \mid \cdot \preceq A$ and $\Gamma', A \mid \Sigma \preceq C$, then $\Gamma, \Gamma' \mid \Sigma \preceq C$.
If $\Gamma \mid \Sigma \preceq A$ and $\Gamma' \mid \Sigma', A \preceq C$, then $\Gamma, \Gamma' \mid \Sigma, \Sigma' \preceq C$.

Note that like the premise of the rule J-box-r4, the first clause uses an empty local subtyping context in $\Gamma \mid \cdot \preceq A$ in order to prove that a given term has type A at every accessible node. We prove the two clauses simultaneously because the rules J-fun4 and J-box-r4 can mix subtyping judgments of both forms $\Gamma \mid \cdot \preceq A$ and $\Gamma \mid \Sigma \preceq A$.

3.1.3 Equivalence between the two subtyping systems R-S4-□ and J-S4-□

The equivalence between the two subtyping systems are stated in Theorems 5 and 6. Theorem 5 states precisely the intuition behind the subtyping judgment.

Theorem 5 (Soundness of *J-S4*-□ with respect to *R-S4*-□)
If $A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$, *then* $(\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq C$.

Proof By structural induction on the derivation of $A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$.

Theorem 6 (Completeness of *J-S4*-□ with respect to *R-S4*-□)
If $A \leq B$, *then* $\cdot \mid A \preceq B$.

Proof By structural induction on the derivation of $A \leq B$.

3.2 Extension with possibility modality ◇

3.2.1 Relational subtyping system R-S4-◇

The third box in Figure 1 shows three subtyping rules to be added to the base subtyping system *R-Base*. We can justify each subtyping rule according to the accessibility relation between nodes in modal logic S4.

- The rule R-dia-T is based on the reflexivity of the accessibility relation. It corresponds to the axiom $A \supset \Diamond A$ in S4.
- The rule R-dia-4 is based on the transitivity of the accessibility relation. It corresponds to the axiom $\Diamond\Diamond A \supset \Diamond A$ in S4.
- The rule R-dia-K corresponds to the axiom $\Box(A \supset B) \supset (\Diamond A \supset \Diamond B)$ in S4. The premise of the rule R-dia-K can be thought of as corresponding to a formula $\Box(A \supset B)$ in S4 because it uses no hypotheses.

Possibility modality ◇ interacts with neither intersection types nor function types. For example, there is no distributivity rule for ◇ over intersection types such as $\Diamond A \wedge \Diamond B \leq \Diamond(A \wedge B)$, or over function types such as $\Diamond(A \to B) \leq \Diamond A \to \Diamond B$.

3.2.2 Judgmental subtyping system J-S4-◇

In addition to an ordinary subtyping judgment $\Sigma \preceq A$, the judgmental subtyping system now uses a *possibility subtyping judgment* $\Sigma \div A$. A possibility subtyping judgment $A_1, \cdots, A_n \div B$ means that if a term has types A_1, $\cdots$, A_n at the current node, it also has type B at some node accessible from the current node. Hence it can be thought of as corresponding to $A_1 \wedge \cdots \wedge A_n \leq \Diamond B$ in the relational subtyping system *R-S4-◇*. Note, however, that the meaning of the possibility subtyping judgment itself does not depend on any type constructor, in particular ◇.

$$\frac{\Sigma, A, B \div C}{\Sigma, A \wedge B \div C}\ \textsf{J-and-lp} \qquad \frac{\Sigma \preceq A}{\Sigma \div A}\ \textsf{J-poss-p} \qquad \frac{A \div B}{\Sigma, \Diamond A \div B}\ \textsf{J-dia-lp} \qquad \frac{\Sigma \div A}{\Sigma \preceq \Diamond A}\ \textsf{J-dia-r}$$

Fig. 5 Additional subtyping rules for $\Diamond$ in modal logic S4 in the judgmental subtyping system *J-S4-*$\Diamond$ which strictly extends *J-Base*

Figure 5 shows subtyping rules for possibility modality $\Diamond$ which are to be added to the base judgmental subtyping system *J-Base* in Figure 3. The rule J-and-lp shows that like an ordinary subtyping judgment, a possibility subtyping judgment implicitly uses $\wedge$ between all types in its subtyping context. We justify the rule J-poss-p from the reflexivity of the accessibility relation. The conclusion of the rule J-dia-lp describes a situation in which a term has type $\Diamond A$ at the current node, or equivalently, type A at some accessible node; the premise of the rule J-dia-lp analyzes the same term at this unknown node. The rule J-dia-r shows that $\Diamond$ internalizes the knowledge embedded in a possibility subtyping judgment. The judgmental subtyping system *J-S4-*$\Diamond$ continues to be orthogonal as every rule in Figure 5 involves at most one type constructor.

The two kinds of subtyping judgments give rise to four different cut rules and we state the admissibility of the cut rules with four clauses:

Theorem 7 (Admissibility of the cut rules in *J-S4-*$\Diamond$)
If $\Sigma \preceq A$ *and* $\Sigma', A \preceq C$, *then* $\Sigma, \Sigma' \preceq C$.
If $\Sigma \preceq A$ *and* $\Sigma', A \div C$, *then* $\Sigma, \Sigma' \div C$.
If $\Sigma \div A$ *and* $A \preceq C$, *then* $\Sigma \div C$.
If $\Sigma \div A$ *and* $A \div C$, *then* $\Sigma \div C$.

Note that in the last two clauses, the second assumption uses a singleton subtyping context consisting only of type A. This is because the first assumption $\Sigma \div A$ means that a give term has type A at some accessible node, about which nothing else is known. We prove the four clauses simultaneously because proofs of ordinary subtyping judgments may require proofs of possibility subtyping judgments (by the rule J-dia-r) and vice versa (by the rule J-poss-p).

3.2.3 Equivalence between the two subtyping systems R-S4-$\Diamond$ *and J-S4-*$\Diamond$

The equivalence between the two subtyping systems are stated in Theorems 8 and 9. The second clause in Theorem 8 states precisely the intuition behind the possibility subtyping judgment.

Theorem 8 (Soundness of *J-S4-*$\Diamond$ with respect to *R-S4-*$\Diamond$)
If $A_1, \cdots, A_n \preceq B$, *then* $A_1 \wedge \cdots \wedge A_n \leq B$.
If $A_1, \cdots, A_n \div B$, *then* $A_1 \wedge \cdots \wedge A_n \leq \Diamond B$.

Proof By simultaneous structural induction on the derivation of $A_1, \cdots, A_n \preceq B$ and $A_1, \cdots, A_n \div B$.

Theorem 9 (Completeness of *J-S4*-$\Diamond$ with respect to *R-S4*-$\Diamond$)
If $A \leq B$, then $A \preceq B$.

Proof By structural induction on the derivation of $A \leq B$.

3.3 Putting both modalities $\Box$ and $\Diamond$ together

We finally complete the subtyping systems based on modal logic S4 by merging all the subtyping systems developed so far. The resultant relational subtyping system *R-S4*-$\Box\Diamond$ consists of the upper four boxes in Figure 1 and the corresponding judgmental subtyping system *J-S4*-$\Box\Diamond$ is shown in Figure 4.

3.3.1 Relational subtyping system R-S4-$\Box\Diamond$

Our relational subtyping system *R-S4*-$\Box\Diamond$ is not just the sum of the previous two relational subtyping systems *R-S4*-$\Box$ and *R-S4*-$\Diamond$ because it includes a new subtyping rule R-box-dia-K. This rule explicitly expresses the interaction between modalities $\Box$ and $\Diamond$ in the subtyping relation. Consider a term that has types $\Box C$ and $\Diamond A$ at the current node. In order to show that it also has type $\Diamond B$, we analyze the term at a remote node where it is known to have type A. Because of the transitivity of the accessibility relation, the term should have type $\Box C$ at the remote node as well. Hence the premise of the rule R-box-dia-K inherits the $\Box C$ from the conclusion.

We may think of the rule R-box-dia-K as a generalization of the rule R-dia-K. Strictly speaking, the rule R-dia-K is not a special case of the rule R-box-dia-K in which the inclusion of type $\Box C$ is mandatory. If, however, we extend the subtyping system so that every type A has a necessity modal type $\Box A'$ such that $A \leq \Box A'$, we can derive the rule R-dia-K from the rule R-box-dia-K. For example, it suffices to introduce a special type $\top$, the unit for $\wedge$, such that $A \leq \Box\top$ for every type A:

$$\dfrac{\dfrac{\dfrac{}{\Diamond A \leq \Box\top}\quad\dfrac{}{\Diamond A \leq \Diamond A}\,\textsf{R-refl}}{\Diamond A \leq \Box\top \wedge \Diamond A}\,\textsf{R-and-r} \qquad \dfrac{\dfrac{\dfrac{}{\Box\top \wedge A \leq A}\,\textsf{R-and-l}_2 \qquad \begin{array}{c}\vdots\\ A \leq B\end{array}}{\Box\top \wedge A \leq B}\,\textsf{R-trans}}{\Box\top \wedge \Diamond A \leq \Diamond B}\,\textsf{R-box-dia-K}}{\Diamond A \leq \Diamond B}\,\textsf{R-trans}$$

The rule R-box-dia-K corresponds to a formula $\Box(\Box C \wedge A \supset B) \supset (\Box C \wedge \Diamond A \supset \Diamond B)$ in modal logic S4. Even though there is no axiom in S4 that explicitly expresses the interaction between modalities $\Box$ and $\Diamond$, this formula is still true essentially because we use the same logical connective $\supset$ both when connecting $\Box C \wedge A$ to B and $\Box C \wedge \Diamond A$ to $\Diamond B$ and when connecting $\Box(\Box C \wedge A \supset B)$ to $\Box C \wedge \Diamond A \supset \Diamond B$. In contrast, the rule R-box-dia-K is not derivable because it uses the subtyping relation when connecting $\Box C \wedge A$ to B and $\Box C \wedge \Diamond A$ to $\Diamond B$, but a logical inference when connecting $\Box(\Box C \wedge A \supset B)$ to $\Box C \wedge \Diamond A \supset \Diamond B$.

If we omit the rule R-box-dia-K in our relational subtyping system, the interaction between modalities $\Box$ and $\Diamond$ disappears altogether.

The need for the rule R-box-dia-K is not obvious when we combine the previous two relational subtyping systems. It becomes obvious only when we develop the judgmental subtyping system *J-S4-*$\Box\Diamond$ and prove its soundness with respect to the relational subtyping system.

3.3.2 Judgmental subtyping system J-S4-$\Box\Diamond$

The judgmental subtyping system *J-S4-*$\Box\Diamond$ combines the ideas from the previous two systems, namely of global subtyping context and possibility subtyping judgment. Thus it uses two kinds of subtyping judgments both of which use two subtyping contexts:

global subtyping context	$\Gamma \ ::= \ \cdot \mid \Gamma, A$
local subtyping context	$\Sigma \ ::= \ \cdot \mid \Sigma, A$
ordinary subtyping judgment	$\Gamma \mid \Sigma \preceq A$
possibility subtyping judgment	$\Gamma \mid \Sigma \div A$

The meaning of an ordinary subtyping judgment is the same as in Section 3.1. A possibility subtyping judgment $A_1, \cdots, A_m \mid B_1, \cdots, B_n \div C$, a direct extension of its definition in Section 3.2, means that if a term has types $A_1, \cdots, A_m$ at every accessible node and types $B_1, \cdots, B_n$ at the current node, it also has type C at some node accessible from the current node. Hence it can be thought of as corresponding to $(\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq \Diamond C$ in the relational subtyping judgment.

Figure 4 shows the judgmental subtyping system *J-S4-*$\Box\Diamond$ with both modalities $\Box$ and $\Diamond$ from modal logic S4. We reuse all the rules in the upper box from Section 3.1. We can explain the first four rules in the lower box (J-and-vp4, J-and-lp4, J-box-vp4, and J-box-lp4) in the same way as we explain the rules for $\wedge$ and $\Box$ in Section 3.1. The rules J-poss-p4, J-dia-lp4, and J-dia-r4 generalize the rules J-poss-p, J-dia-lp, and J-dia-r in Figure 5, respectively, by adding a global typing context in every subtyping judgment.

The rule J-dia-vp4 is distinct from all the other rules in that its principal type $\Diamond A$ remains in the premise after producing a new type A in the ordinary subtyping context. This can be problematic in proof search because the rule J-dia-vp4 can be applied indefinitely without making progress, as in:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Diamond A \mid A \div C}}{\Diamond A \mid A \div C}\ \textsf{J-dia-vp4}}{\Diamond A \mid \Sigma \div C}\ \textsf{J-dia-vp4} \qquad \dfrac{\dfrac{\dfrac{\vdots}{\Diamond\Box A, A, A \mid \cdot \div C}}{\Diamond\Box A, A \mid \Box A \div C}\ \textsf{J-box-lp4}}{\Diamond\Box A, A \mid \cdot \div C}\ \textsf{J-dia-vp4}$$

Still, however, we choose to leave $\Diamond A$ in the premise because the presence of type A in the ordinary subtyping context (which means that a given term has type A only at the current node) is not enough to reproduce the principal type $\Diamond A$ in the global subtyping context (which means that a given term has

type $\Diamond A$ at every accessible node) when the accessibility relation may not be symmetric. For example, $\cdot \mid \Box\Diamond A \preceq \Diamond(\Box\Diamond A \wedge A)$ no longer holds if we omit $\Diamond A$ in the premise of the rule J-dia-vp4.

The admissibility of the cut rules combines Theorems 4 and 7, and consists of six clauses. Its proof uses the weakening and contraction properties.

Lemma 8 (Weakening) *If $\Gamma \mid \Sigma \preceq C$, then $\Gamma, A \mid \Sigma \preceq C$ and $\Gamma \mid \Sigma, A \preceq C$.*

Proof By structural induction on the derivation of $\Gamma \mid \Sigma \preceq C$.

Lemma 9 (Contraction)
If $\Gamma, A, A \mid \Sigma \preceq C$, then $\Gamma, A \mid \Sigma \preceq C$.
If $\Gamma \mid \Sigma, A, A \preceq C$, then $\Gamma \mid \Sigma, A \preceq C$.

Proof By simultaneous nested induction on (i) the structure of type A and (ii) the structure of the derivation of $\Sigma, A, A \preceq C$ and $\Gamma \mid \Sigma, A, A \preceq C$.

Theorem 10 (Admissibility of the cut rules in *J-S4*-$\Box\Diamond$)
(1) If $\Gamma \mid \cdot \preceq A$ and $\Gamma', A \mid \Sigma \preceq C$, then $\Gamma, \Gamma' \mid \Sigma \preceq C$.
(2) If $\Gamma \mid \cdot \preceq A$ and $\Gamma', A \mid \Sigma \div C$, then $\Gamma, \Gamma' \mid \Sigma \div C$.
(3) If $\Gamma \mid \Sigma \preceq A$ and $\Gamma' \mid \Sigma', A \preceq C$, then $\Gamma, \Gamma' \mid \Sigma, \Sigma' \preceq C$.
(4) If $\Gamma \mid \Sigma \preceq A$ and $\Gamma' \mid \Sigma', A \div C$, then $\Gamma, \Gamma' \mid \Sigma, \Sigma' \div C$.
(5) If $\Gamma \mid \Sigma \div A$ and $\Gamma' \mid A \preceq C$, then $\Gamma, \Gamma' \mid \Sigma \div C$.
(6) If $\Gamma \mid \Sigma \div A$ and $\Gamma' \mid A \div C$, then $\Gamma, \Gamma' \mid \Sigma \div C$.

Clause (1) is from Theorem 4, and clause (2) uses $\Gamma \mid \cdot \preceq A$ because type A is in the global subtyping context in $\Gamma', A \mid \Sigma \div C$. Clauses (3) to (6) are obtained from Theorem 7 by adding a global typing context in every subtyping judgment. As in the proof of Theorem 7, we prove all the clauses simultaneously because proofs of ordinary subtyping judgments may require proofs of possibility subtyping judgments and vice versa. The proof exploits the strategy described in Section 2.2.

The judgmental subtyping system *J-S4*-$\Box\Diamond$ is decidable because it satisfies the subformula property in the following sense: in any derivation of a subtyping judgment $\Gamma \mid \Sigma \preceq A$ or $\Gamma \mid \Sigma \div A$, only those types in Γ, Σ, and A or their constituent types can appear. In conjunction with the contraction property, the subformula property implies that any derivation of a goal subtyping judgment needs to consider a database consisting only of a finite number of subtyping judgments. Since a set of subtyping judgments already known to be provable determines another unique set of subtyping judgments, we can eventually decide the provability of every subtyping judgment in the database, including the goal subtyping judgment itself.

3.3.3 Equivalence between the two subtyping systems R-S4-$\Box\Diamond$ and J-S4-$\Box\Diamond$

The equivalence between the two subtyping systems *R-S4*-$\Box\Diamond$ and *J-S4*-$\Box\Diamond$ are stated in Theorems 11 and 12. Theorem 11 states precisely the intuition behind the two subtyping judgments.

Theorem 11 (Soundness of *J-S4*-□◇ with respect to *R-S4*-□◇)
If $A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$, *then* $(\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq C$.
If $A_1, \cdots, A_m \mid B_1, \cdots, B_n \div C$, *then* $(\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq \Diamond C$.

Proof By simultaneous structural induction on the derivation of $A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$ and $A_1, \cdots, A_m \mid B_1, \cdots, B_n \div C$.

Theorem 12 (Completeness of *J-S4*-□◇ with respect to *R-S4*-□◇)
If $A \leq B$, *then* $\cdot \mid A \preceq B$.

Proof By structural induction on the derivation of $A \leq B$.

In our study, we identify the need for the rule R-box-dia-K in the relational subtyping system *R-S4*-□◇ during an attempt to prove Theorem 11. For a relational subtyping system without the rule R-box-dia-K (consisting of the upper three boxes in Figure 1), we cannot prove Theorem 11 unless we replace the rules J-dia-vp4 and J-dia-lp4 with stronger rules that use an empty global subtyping context in their premise:

$$\frac{\cdot \mid A \div B}{\Gamma, \Diamond A \mid \Sigma \div B}\ \textsf{J-dia-vp4}' \qquad \frac{\cdot \mid A \div B}{\Gamma \mid \Sigma, \Diamond A \div B}\ \textsf{J-dia-lp4}'$$

In order to prove Theorem 11 without changing the rules J-dia-vp4 and J-dia-lp4, we are led to introduce the rule R-box-dia-K in the relational subtyping system, which can be thought of as a generalization of the already existing rule R-dia-K as explained in Section 3.3.1.

The discovery of the rule R-box-dia-K during the proof of Theorem 11 also testifies to the advantage of subtyping judgments over subtyping relations for designing a subtyping system. In the case of the relational subtyping system *R-S4*-□◇, we start with an incomplete system and the comparison with the judgmental subtyping system later reveals the need to extend it with the rule R-box-dia-K, which is not easy to discover in the beginning. In the case of the judgmental subtyping system *J-S4*-□◇, the opposite occurs: we start with a complete system and the comparison with the relational subtyping system only convinces us of its correctness (because it is clear that the rules J-dia-vp4′ and J-dia-lp4′ are unnecessarily strong). On the whole, the judgmental subtyping system is easier to verify because the key step in the design lies not in formulating subtyping rules but in figuring out a right set of subtyping judgments themselves. In the next section, we study the subtyping systems based on modal logic S5 to draw the same conclusion.

4 Subtyping systems based on modal logic S5

In this section, we extend the base subtyping systems *R-Base* and *J-Base* with modalities □ and ◇ from modal logic S5. The interpretation of necessity modal

types $\Box A$ and possibility modal types $\Diamond A$ is the same as in the previous subtyping systems based on modal logic S4, but the accessibility relation between nodes in the network is additionally assumed to be symmetric: if node w is accessible from node w', then node w' is also accessible from node w. Now new subtyping relations hold that exploit the symmetry of the accessibility relation. For example, a subtyping relation $A \wedge \Diamond B \leq \Diamond(\Diamond A \wedge B)$ is expected to hold: if a term of type A at the current node is known to have type B at some accessible node, it must be a term of types $\Diamond A$ and B from the point of view of the remote node.

As in the previous section, we separately develop the relational subtyping system *R-S5*-$\Box\Diamond$ (Section 4.1) and the judgmental subtyping system *J-S5*-$\Box\Diamond$ (Section 4.2). Then we prove their equivalence (Section 4.3).

4.1 Relational subtyping system *R-S5*-$\Box\Diamond$

The relational subtyping system *R-S5*-$\Box\Diamond$ based on modal logic S5 is a strict extension of the previous system *R-S4*-$\Box\Diamond$ based on modal logic S4. This is because every accessibility relation valid in S4 is also valid in S5 while the interpretation of modal types remains the same in both systems. Hence every subtyping relation provable in the previous system, which exploits only the reflexivity and transitivity of the accessibility relation, is also provable in the new system, which additionally allows the symmetry of the accessibility relation.

Figure 1 shows the relational subtyping system *R-S5*-$\Box\Diamond$ based on modal logic S5. In comparison with the previous system *R-S4*-$\Box\Diamond$ based on modal logic S4, it has two new rules R-dia-5 and R-box-5, both of which are based on the symmetry of the accessibility relation:

- The rule R-dia-5 says that if a term is known to have type A at some accessible node, every accessible node shares this knowledge. It corresponds to the axiom $\Diamond A \supset \Box\Diamond A$ in S5.
- The rule R-box-5 says that if a term has type $\Box A$ at some accessible node, it has type A at every accessible node. It corresponds to the axiom $\Diamond\Box A \supset \Box A$ in S5.

It turns out that unlike in the previous system *R-S4*-$\Box\Diamond$ based on modal logic S4, we do not need to introduce additional subtyping rules that correspond to some formulas in modal logic S5. This, however, remains hard to verify until we develop the judgmental subtyping system *J-S5*-$\Box\Diamond$ and prove the equivalence result.

4.2 Judgmental subtyping system *J-S5*-$\Box\Diamond$

In designing the judgmental subtyping system *J-S5*-$\Box\Diamond$ based on modal logic S5, the symmetry of the accessibility relation requires a subtyping judgment to

maintain a subtyping context for *every* known accessible node in the network. Intuitively, even when inspecting a term at a remote node, we may still need those types that the term has at the original node, which is accessible from the remote node because of the symmetry of the accessibility relation. Since the accessibility relation is also transitive, we have to remember every node that a term has previously visited, as well as those types that can be assigned to the term at such a node. This observation leads to the use of a new form of subtyping judgment that analyzes a set of subtyping contexts, similarly to the label-free sequent calculus for intuitionistic modal logic S5 [6].

Our judgmental subtyping system *J-S5-*$\Box\Diamond$ uses a single subtyping judgment $\Delta \mid \Gamma \mid \Sigma \preceq C$ which uses a *remote node context* Δ in addition to a global subtyping context Γ and a local subtyping context Σ; a remote node context is an unordered collection of local subtyping contexts:

$$\begin{array}{ll} \text{remote node context} & \Delta ::= \cdot \mid \Delta; \Sigma \\ \text{global subtyping context} & \Gamma ::= \cdot \mid \Gamma, A \\ \text{local subtyping context} & \Sigma ::= \cdot \mid \Sigma, A \end{array}$$

As a direct extension of the ordinary subtyping judgment from the previous system *J-S4-*$\Box\Diamond$ based on modal logic S4, the new subtyping judgment $\Delta \mid A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$ means that a term has type C at the current node whenever the following three conditions hold:

- For each subtyping context $\Sigma_k = C_1^k, \cdots, C_l^k$ in Δ, there exists some accessible node where the term has types C_1^k, $\cdots$, C_l^k.
- The term has types A_1, $\cdots$, A_m at every accessible node.
- The term has types B_1, $\cdots$, B_n at the current node.

Unlike in the previous system *J-S4-*$\Box\Diamond$, however, we do not use another subtyping judgment similar to a possibility subtyping judgment, since the use of a remote node context enables us not just to assert the existence of a remote node where a term has a particular type, but even to explicitly specify such a node with its own local subtyping context. We can think of $\Delta \mid \Gamma \mid \Sigma \preceq C$ as corresponding to a subtyping relation $(\bigwedge_{\Sigma_k \in \Delta} \Diamond(\bigwedge_{C_l^k \in \Sigma_k} C_l^k)) \wedge (\bigwedge_{A_i \in \Gamma} \Box A_i) \wedge (\bigwedge_{B_j \in \Sigma} B_j) \leq C$ in the relational subtyping system.

Figure 6 shows the judgmental subtyping system *J-S5-*$\Box\Diamond$ with both modalities $\Box$ and $\Diamond$ from modal logic S5. We can explain the first seven rules J-refl-v5 to J-and-r5 in a similar way to their corresponding rules in Figure 4. The rules J-box-lc5, J-box-v5, and J-box-l5 analyze a necessity modal type $\Box A$ to assert that a given term has type A at every accessible node, and augment the global subtyping context with type A. Similarly to the rule J-box-r4 in Figure 4, the premise of the rule J-box-r5 describes a situation in which a given term can be assigned type A at an arbitrary accessible node. Unlike the rule J-box-r4, however, the premise leaves the local subtyping context Σ in the remote node context, instead of discarding it, because of the symmetry of the accessibility relation. The rules J-dia-lc5, J-dia-v5, and J-dia-l5 analyze a possibility modal type $\Diamond A$ to assert the existence of a remote node where a given term has type A, and augment the remote node context with a new local subtyping context

$$\frac{}{\Delta \mid \Gamma, P \mid \Sigma \preceq P}\ \textsf{J-refl-v5} \qquad \frac{}{\Delta \mid \Gamma \mid \Sigma, P \preceq P}\ \textsf{J-refl5}$$

$$\frac{\cdot \mid \cdot \mid C \preceq A_i \quad 1 \leq i \leq n \quad \cdot \mid \cdot \mid B_1, \cdots, B_n \preceq D}{\Delta \mid \Gamma, A_1 \to B_1, \cdots, A_m \to B_m \mid \Sigma, A_{m+1} \to B_{m+1}, \cdots, A_n \to B_n \preceq C \to D}\ \textsf{J-fun5}$$

$$\frac{\Delta; \Sigma, A, B \mid \Gamma \mid \Sigma' \preceq C}{\Delta; \Sigma, A \wedge B \mid \Gamma \mid \Sigma' \preceq C}\ \textsf{J-and-lc5} \qquad \frac{\Delta \mid \Gamma, A, B \mid \Sigma \preceq C}{\Delta \mid \Gamma, A \wedge B \mid \Sigma \preceq C}\ \textsf{J-and-v5}$$

$$\frac{\Delta \mid \Gamma \mid \Sigma, A, B \preceq C}{\Delta \mid \Gamma \mid \Sigma, A \wedge B \preceq C}\ \textsf{J-and-l5} \qquad \frac{\Delta \mid \Gamma \mid \Sigma \preceq A \quad \Delta \mid \Gamma \mid \Sigma \preceq B}{\Delta \mid \Gamma \mid \Sigma \preceq A \wedge B}\ \textsf{J-and-r5}$$

$$\frac{\Delta; \Sigma \mid \Gamma, A \mid \Sigma' \preceq B}{\Delta; \Sigma, \Box A \mid \Gamma \mid \Sigma' \preceq B}\ \textsf{J-box-lc5} \qquad \frac{\Delta \mid \Gamma, A \mid \Sigma \preceq B}{\Delta \mid \Gamma, \Box A \mid \Sigma \preceq B}\ \textsf{J-box-v5}$$

$$\frac{\Delta \mid \Gamma, A \mid \Sigma \preceq B}{\Delta \mid \Gamma \mid \Sigma, \Box A \preceq B}\ \textsf{J-box-l5} \qquad \frac{\Delta; \Sigma \mid \Gamma \mid \cdot \preceq A}{\Delta \mid \Gamma \mid \Sigma \preceq \Box A}\ \textsf{J-box-r5}$$

$$\frac{\Delta; \Sigma; A \mid \Gamma \mid \Sigma' \preceq B}{\Delta; \Sigma, \Diamond A \mid \Gamma \mid \Sigma' \preceq B}\ \textsf{J-dia-lc5} \qquad \frac{\Delta; A \mid \Gamma \mid \Sigma \preceq B}{\Delta \mid \Gamma, \Diamond A \mid \Sigma \preceq B}\ \textsf{J-dia-v5}$$

$$\frac{\Delta; A \mid \Gamma \mid \Sigma \preceq B}{\Delta \mid \Gamma \mid \Sigma, \Diamond A \preceq B}\ \textsf{J-dia-l5} \qquad \frac{\Delta; \Sigma \mid \Gamma \mid \Sigma' \preceq A}{\Delta; \Sigma' \mid \Gamma \mid \Sigma \preceq \Diamond A}\ \textsf{J-dia-r5}$$

$$\frac{\Delta \mid \Gamma \mid \Sigma \preceq A}{\Delta \mid \Gamma \mid \Sigma \preceq \Diamond A}\ \textsf{J-dia-r5}'$$

Fig. 6 *J-S5-*$\Box\Diamond$, judgmental subtyping system based on modal logic S5

consisting of type A. In the rules J-dia-r5 and J-dia-r5′, the local subtyping context in the premise explicitly specifies the remote node where a given term can be assigned type A, which is the reason why we do not need another subtyping judgment similar to a possibility judgment.

Unlike in the previous judgmental subtyping systems, the inclusion of a remote node context in a subtyping judgment calls for weakening and contraction not only at the level of types but also at the level of subtyping contexts:

Lemma 10 (Weakening)
If $\Delta \mid \Gamma \mid \Sigma \preceq C$, *then* $\Delta \mid \Gamma \mid \Sigma, A \preceq C$ *and* $\Delta \mid \Gamma, A \mid \Sigma \preceq C$.
If $\Delta; \Sigma' \mid \Gamma \mid \Sigma \preceq C$, *then* $\Delta; \Sigma', A \mid \Gamma \mid \Sigma \preceq C$.
If $\Delta \mid \Gamma \mid \Sigma \preceq C$, *then* $\Delta; \Sigma' \mid \Gamma \mid \Sigma \preceq C$.

Lemma 11 (Contraction)
(1) If $\Delta \mid \Gamma \mid \Sigma, A, A \preceq C$, *then* $\Delta \mid \Gamma \mid \Sigma, A \preceq C$.
(2) If $\Delta \mid \Gamma, A, A \mid \Sigma \preceq C$, *then* $\Delta \mid \Gamma, A \mid \Sigma \preceq C$.
(3) If $\Delta; \Sigma', A, A \mid \Gamma \mid \Sigma \preceq C$, *then* $\Delta; \Sigma', A \mid \Gamma \mid \Sigma \preceq C$.
(4) If $\Delta; \Sigma; \Sigma \mid \Gamma \mid \Sigma' \preceq C$, *then* $\Delta; \Sigma \mid \Gamma \mid \Sigma' \preceq C$.
(5) If $\Delta; \Sigma \mid \Gamma \mid \Sigma \preceq C$, *then* $\Delta \mid \Gamma \mid \Sigma \preceq C$.

Clause (5) in Lemma 11 states the contraction of the local subtyping context and another local subtyping context in the remote node context, which is necessary when a remote node (described by Σ in the remote node context) happens to be the current node itself (described by the local subtyping context Σ) by the reflexivity of the accessibility relation.

We prove all the clauses in Lemma 10 simultaneously by structural induction on the derivation of the subtyping judgment. We prove all the clauses in Lemma 11 simultaneously by nested induction on (i) the size of type A or local subtyping context Σ to be combined and (ii) the structure of the derivation of the subtyping judgments. Here we define the size of a type as the number of occurrences of type constructors in it, and the size of a local subtyping context as the sum of the size of all types in it.

The use of three kinds of contexts in a subtyping judgment gives rise to three cut rules and we state the admissibility of the cut rules with three clauses. Its proof uses the weakening and contraction properties.

Theorem 13 (Admissibility of the cut rules in *J-S5-*$\Box\Diamond$)
If $\Delta \mid \Gamma \mid \cdot \preceq A$ *and* $\Delta' \mid \Gamma', A \mid \Sigma \preceq C$, *then* $\Delta; \Delta' \mid \Gamma, \Gamma' \mid \Sigma \preceq C$.
If $\Delta \mid \Gamma \mid \Sigma \preceq A$ *and* $\Delta' \mid \Gamma' \mid \Sigma', A \preceq C$, *then* $\Delta; \Delta' \mid \Gamma, \Gamma' \mid \Sigma, \Sigma' \preceq C$.
If $\Delta \mid \Gamma \mid \Sigma \preceq A$ *and* $\Delta'; \Sigma', A \mid \Gamma' \mid \Sigma'' \preceq C$, *then* $\Delta; \Delta'; \Sigma, \Sigma' \mid \Gamma, \Gamma' \mid \Sigma'' \preceq C$.

The first clause uses an empty local subtyping context in $\Delta \mid \Gamma \mid \cdot \preceq A$ in order to prove that a given term has type A at every accessible node. In the third clause, the current node described by Σ in $\Delta \mid \Gamma \mid \Sigma \preceq A$ is the same as the remote node described by Σ', A in $\Delta'; \Sigma', A \mid \Gamma' \mid \Sigma'' \preceq C$. We prove the three clauses simultaneously and exploit the strategy described in Section 2.2.

Similarly to the base judgmental subtyping system *J-Base* in Section 2, the judgmental subtyping system *J-S5-*$\Box\Diamond$ is decidable because the premise of a subtyping rule either is empty or consists of subtyping judgments strictly smaller than the subtyping judgment in the conclusion.

4.3 Equivalence between the two subtyping systems *R-S5-*$\Box\Diamond$ and *J-S5-*$\Box\Diamond$

The equivalence between the two subtyping systems *R-S5-*$\Box\Diamond$ and *J-S5-*$\Box\Diamond$ are stated in Theorems 14 and 15. Theorem 14 states precisely the intuition behind the subtyping judgment.

Theorem 14 (Soundness of *J-S5-*$\Box\Diamond$ with respect to *R-S5-*$\Box\Diamond$)
If $\Delta \mid A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$,
then $(\bigwedge_{\Sigma_k \in \Delta} \Diamond(\bigwedge_{C_l^k \in \Sigma_k} C_l^k)) \wedge (\Box A_1 \wedge \cdots \wedge \Box A_m) \wedge (B_1 \wedge \cdots \wedge B_n) \leq C$.

Proof By structural induction on the derivation of $\Delta \mid A_1, \cdots, A_m \mid B_1, \cdots, B_n \preceq C$.

Theorem 15 (Completeness of *J-S5-*$\Box\Diamond$ with respect to *R-S5-*$\Box\Diamond$)
If $A \leq B$, *then* $\cdot \mid \cdot \mid A \preceq B$.

Proof By structural induction on the derivation of $A \leq B$.

Although we are already aware of the rule R-box-dia-K in the relational subtyping system, the proof of Theorem 14 could have equally identified the need for it. Moreover we find it easy to design each subtyping rule in Figure 6 because of the clarity of the meaning of the subtyping judgment $\Delta \mid \Gamma \mid \Sigma \preceq C$.

Thus we draw exactly the same conclusion as in Section 3: the judgmental subtyping system is easier to verify than the relational subtyping system because the crux of the design lies in figuring out a suitable form of subtyping judgment rather than in formulating subtyping rules.

5 Related work

There are two approaches to formulating a subtyping system: the semantic approach and the syntactic approach. The semantic approach introduces some semantic interpretation in order to define the subtyping relation. Typically one introduces a semantic mapping from types to sets of values and defines a subtyping relation between two types as a subset relation between their corresponding sets, as in the semantic subtyping for XDuce [7]. In the presence of higher-order functions, such a set-theoretic interpretation of types does not work and one should define a subtyping relation indirectly by specifying a semantic constraint on it (thus without having to explicitly specify the set of values corresponding to each type), as proposed by Frisch *et al.* [5] and further extended by Castagna and Xu [3]. In contrast, the syntactic approach develops a system of inference rules for deducing the subtyping relation. One may present the system in the axiomatic style where the transitivity rule is included as part of the system, or in the proof-theoretic style where the admissibility of the cut rule implies the transitivity of the subtyping relation, as in the sequent calculus for subtyping by Longo *et al.* [11]. Our work advocates the use of a new style of the syntactic approach which uses subtyping judgments whose definitions express those notions internalized into type constructors directly at the level of judgments.

The idea of using subtyping judgments is proposed in a type system for probabilistic computation by the second author [13]. This type system uses the same subtyping judgment as in Section 3.1 for function types, intersection types, and necessity modal types. Its interpretation of modal type $\Box A$ as denoting point-mass distributions and ordinary type A as denoting all kinds of probability distributions over the same probability domain naturally leads to a subtyping system based on modal logic S4, which is presented both as a relational subtyping system and as a judgmental subtyping system. Our work completes this previous study, albeit with different motivations, by considering both necessity and possibility modal types and developing subtyping systems based on each of modal logics S4 and S5. The focus is also shifted from the proof of the admissibility of the cut rules to the design of various subtyping systems (whose properties are now all proven in Coq).

The design of the subtyping system based on modal logic S4 is inspired by the judgmental formulation of intuitionistic modal logic S4 by Pfenning and Davies [15], which introduces a new kind of context for necessity modality $\Box$ and a new form of judgment for possibility modality $\Diamond$. The idea of using remote node contexts in the subtyping system based on modal logic S5 also appears in the label-free sequent calculus for intuitionistic modal logic S5 by

Galmiche and Salhi [6], in which a sequent uses a set of contexts in order to record those formulas true at remote nodes. Because it additionally uses global subtyping contexts, our subtyping system is simpler than their sequent calculus and has fewer inference rules. A comparison with these systems for intuitionistic modal logics S4 and S5 shows that our judgmental subtyping systems differ primarily in the treatment of function types $A \supset B$: a subtyping rule for function types tests if a term of several function types can have another function type, whereas its corresponding inference rules in a system of logic are based solely on logical entailment.

6 Conclusion

When several type constructors interact with each other in a non-trivial way, the syntactic approach often fails to reach a decidable subtyping system, especially because of the difficulty of finding appropriate distributivity rules and embedding the transitivity rule. As a result, researchers usually choose an alternative approach, namely the semantic approach, when designing a subtyping system involving many type constructors, as illustrated in [7, 5, 3]. Our work on judgmental subtyping systems, however, sheds new light on the syntactic approach — it is not that the syntactic approach is inherently a bad choice for designing such a complex subtyping system; rather it is just the use of a binary relation between two types that unnecessarily complicates the whole design. By using subtyping judgments instead of subtyping relations, we can overcome the limitation of the traditional style of the syntactic approach and also develop a subtyping system by taking full advantage of the syntactic approach.

Future work includes designing a base language into which the judgmental subtyping systems can be incorporated. Another direction to pursue is to apply the new style of the syntactic approach to reformulate existing subtyping systems that are based on the traditional style of the syntactic approach or the semantic approach.

Acknowledgements This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2008-0062609) and Mid-career Researcher Program through NRF funded by the MEST (2010-0022061).

References

1. N.D. Belnap. Display logic. *Journal of philosophical logic*, 11(4):375–417, 1982.
2. Tijn Borghuis and Loe Feijs. A constructive logic for services and information flow in computer networks. *The Computer Journal*, 43(4):275–289, 2000.
3. Giuseppe Castagna and Zhiwu Xu. Set-theoretic foundation of parametric polymorphism and subtyping. In *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming*, pages 94–106. ACM, 2011.
4. Rowan Davies and Frank Pfenning. Intersection types and computational effects. In *Proceedings of the ACM SIGPLAN International Conference on Functional Programming*, pages 198–208. ACM Press, 2000.

5. Alain Frisch, Giuseppe Castagna, and Véronique Benzaken. Semantic subtyping: Dealing set-theoretically with function, union, intersection, and negation types. *Journal of the ACM*, 55(4):19:1–19:64, September 2008.
6. Didier Galmiche and Yakoub Salhi. Label-free proof systems for intuitionistic modal logic IS5. In *Proceedings of the 16th international conference on Logic for Programming, Artificial intelligence, and Reasoning*, LPAR'10, pages 255–271. Springer-Verlag, 2010.
7. Haruo Hosoya and Benjamin C. Pierce. XDuce: A statically typed XML processing language. *ACM Transactions on Internet Technology*, 3(2):117–148, 2003.
8. Limin Jia and David Walker. Modal proofs as distributed programs (extended abstract). In *Proceedings of the European Symposium on Programming, LNCS 2986*, pages 219–233. Springer, 2004.
9. Assaf J. Kfoury and J. B. Wells. Principality and decidable type inference for finite-rank intersection types. In *Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 161–174. ACM, 1999.
10. Olivier Laurent. Intersection types with subtyping by means of cut-elimination. Unpublished note, January 2005.
11. Giuseppe Longo, Kathleen Milsted, and Sergei Soloviev. A logic of subtyping. In *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 292–299. IEEE Computer Society Press, 1995.
12. Tom Murphy, VII, Karl Crary, Robert Harper, and Frank Pfenning. A symmetric modal lambda calculus for distributed computing. In *Proceedings of the 19th IEEE Symposium on Logic in Computer Science*, pages 286–295. IEEE Press, 2004.
13. Sungwoo Park. A calculus for probabilistic languages. In *Proceedings of the 2003 ACM SIGPLAN International Workshop on Types in Language Design and Implementation*, pages 38–49. ACM Press, 2003.
14. Frank Pfenning. Structural cut elimination. In *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 156–166. IEEE, 1995.
15. Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11(4):511–540, 2001.