Deciding Type Equivalence in a Language with Singleton Kinds

Christopher A. Stone Robert Harper

September 15, 1999
CMU-CS-99-155

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Work on the TILT compiler for Standard ML led us to study a language with *singleton kinds*: $S(A)$ is the kind of all types provably equivalent to the type A. Singletons are interesting because they provide a very general form of definitions for type variables and allow fine-grained control of type computations.

Internally, TILT represents programs using a predicative variant of Girard's F_ω enriched with singleton kinds, dependent product and function kinds (Σ and Π), and a sub-kinding relation. An important benefit of using a typed language as the representation of programs is that typechecking can detect many common compiler implementation errors. However, the decidability of typechecking for our particular representation is not obvious. In order to typecheck a term, we must be able to determine whether two type constructors are provably equivalent. But in the presence of singleton kinds, the equivalence of type constructors depends both on the typing context in which they are compared and on the kind at which they are compared.

In this paper we concentrate on the key issue for decidability of typechecking: determining the equivalence of well-formed type constructors. We show this problem decidable by presenting a sound, complete, and terminating decision algorithm. These properties are established by a novel Kripke-style logical relations argument inspired by Coquand's result for type theory.

This research was sponsored in part by the Advanced Research Projects Agency CSTO under the title "The Fox Project: Advanced Languages for Systems Software," ARPA Order No. C533, issued by ESC/ENS under Contract No. F19628-95-C-0050.

Keywords: singleton kinds, subkinding, logical relations, type theory

1 Introduction

1.1 Motivation

The TIL compiler for core Standard ML [17] was structured as a series of translations between explicitly-typed intermediate languages. Each pass of the compiler (e.g., common subexpression elimination or closure conversion) transformed the program and its type, preserving well-typedness. One advantage of this framework is that typechecking the intermediate representation can detect a wide variety of common compiler implementation errors. The typing information on terms can also be used to support type-based optimizations and efficient data representations; TIL used a type-passing interpretation of polymorphism in which types were passed and analyzed at run-time [12]. In the future, it should be possible to use such typing information for annotating binaries with a certification of safety [13, 14].

The results from TIL were very encouraging, but the compiler implementation was inefficient and could only handle complete programs written without use of modules. The Fox Project group at Carnegie Mellon therefore decided to completely re-engineer TIL to produce TILT (TIL Two), a more practical compiler which could handle separate compilation and the complete SML language.

One challenge in scaling up the compiler was properly handling the propagation of type information. For example, in the Standard ML module language we can have a structure `Set` with the signature

```
sig
   type item = int
   type set
   type setpair = set * set

   val empty     : set
   val insert    : set * item -> set
   val member    : set * item -> bool
   val union     : setpair -> set
   val intersect : setpair -> set
end
```

From this interface it is apparent that the module `Set` has three type components: the type `Set.item` known to be equal to `int`, the type `Set.set` about which nothing is known, and the type `Set.set` which is the type of pairs of `Set.set`'s.

There are two important points to note about this example. First, equivalences such as the one between `Set.item` and `int` are *open-scope* definitions available to "the rest of the program", which may not even be written when this module is compiled. Second, because of type-passing these type components really are computed and stored by the run-time code. Although it is possible get rid of type definitions in signatures by replacing all references to these components with their definitions [16] we do not wish to do so; such substitutions could substantially increase the number of type computations performed at run-time.

The choice we made was to use an typed intermediate language based on F_ω with the following kind structure (recall that kinds classify type constructors):

- A kind T classifying ordinary types;
- Singleton kinds $S(A)$ classifying all types of kind T provably equivalent to A;
- Dependent record kinds classifying records of type constructors and dependent function kinds classifying functions mapping type constructors to type constructors[1];
- A sub-kinding relation induced by $S(A) \leq T$.

Modules are represented in this language using a phase-splitting interpretation [7, 16]. The main idea is that modules can be split into type constructor and a term, while signatures split in a parallel way into a kind and a type. Singleton kinds are used to model definitions and type sharing specifications in module

[1]A record of type constructors should not be confused with a record type, which would have kind T. Similarly, functions of type constructors are not function types, which would also have kind T.

signatures, dependent record kinds model the type parts of structure signatures, dependent function kinds model the type parts of functor signatures, and subkinding models (non-coercive) signature matching.

For example, the kind corresponding to the above signature is a dependent record kind saying that there are three type components: the first component *item* has kind $S(\texttt{int})$ because its definition is known; the second component *set* has kind T because its definition is not known; finally the third component *setpair* has kind $S(\mathit{set}\times\mathit{set})$, which takes advantage of the record kind being dependent.

Singletons are used to describe and control the propagation of type definitions and sharing in the compiler. The constructor A has kind $S(B)$ if and only if the constructors A and B are provably equivalent. Thus, the hypothesis that the variable α has type $S(A)$ essentially says that α is a type variable with definition A. This models open-scope definitions in the source language.

Furthermore, singletons provide "partial" definitions for variables. If α is a pair of types with kind $S(\texttt{int})\times T$ this tells us that the first component of this pair, $\pi_1\alpha$, is $\texttt{int}$. However, this kind tells us nothing about the identity of the $\pi_2\alpha$. As in the above example, partial definitions allow natural modeling of definitions in a modular system, where some components of a module have known definitions and others remain abstract.

Interestingly, in a language with singleton kinds we can additionally express with delimited scope (*closed-scope*) definitions. The expression $\mathsf{let}\ \alpha{:}T = \texttt{int}\times\texttt{int}\ \mathsf{in}\ \texttt{id}[\alpha](3,4)\ \mathsf{end}$ does not typecheck when expressed as a function application $(\Lambda\alpha{:}T.\texttt{id}[\alpha](3,4))[\texttt{int}\times\texttt{int}]$; the application of $\texttt{id}[\alpha]$ to a pair of integers is only well-formed if α is known to be $\texttt{int}\times\texttt{int}$, which is not apparent while checking the abstraction. We can express this information, however, by annotating the argument with a singleton kind to get the well-formed term $(\Lambda\alpha{:}S(\texttt{int}\times\texttt{int}).\texttt{id}[\alpha](3,4))[\texttt{int}\times\texttt{int}]$. Now let-bindings of types could be directly added to our calculus, but the general ability to turn types into function arguments (particularly into new arguments of pre-existing functions) is necessary for a low-level description of type-preserving closure-conversion in the type-passing framework [11]. It also enables finer control of when type computations occur at run time, permitting optimizations such as improved common subexpression elimination of types.

Given that we wish to typecheck our intermediate representation, the question that arises is whether typechecking is decidable. This question reduces to the decidability of equivalence for well-formed type constructors. This latter question is non-trivial because the equivalence of two constructors can depend both on the singletons (definitions) in the context and — less obviously — on the kind at which the constructors are being compared. (See Section 2.2.) The common method of implementing equivalence via context-insensitive rewrite rules is thus completely inapplicable for our calculus. The goal of this paper is to show that constructor equivalence is nevertheless decidable.

1.2 Outline

In Section 2 we introduce the $\lambda_{\leq}^{\Pi\Sigma S}$ calculus (a formalization of the key features of the type constructors and kinds of the TILT intermediate representation). We explain some of the more interesting aspects of this calculus, including the dependency of equivalence on the typing context and the classifying kind. We show that singletons for constructors of higher kinds are definable, and show that every constructor has a principal (most-specific) kind.

In Section 3 we present a sound algorithm for determining equivalence of well-formed constructors. We were inspired by Coquand's approach to $\beta\eta$-equivalence for a type theory with Π types and one universe [3]. Coquand worked with an algorithm which directly decides equivalence, rather than using a confluent and strongly-normalizing reduction relation. However, in contrast to Coquand's system we cannot compare terms by their shape alone; we must take account of both the context and the classifier. Where Coquand maintains a set of bound variables, we maintain a full typing context. Similarly, he uses shapes to guide the algorithm where we maintain a classifying kind. (For example, when he would check whether either constructor is a lambda-abstraction, we check whether the classifying kind is a function kind.) Although the natural presentation of our algorithm defines a relation of the form $\Gamma \vdash A_1 \Leftrightarrow A_2 : K$, we cannot analyze the correctness of this algorithm directly. Asymmetries in the formulation preclude a direct proof of such simple properties as symmetry and transitivity, both of which are immediately evident in Coquand's case. Instead we analyze a related algorithm which restores symmetry by maintaining two typing context and two classifying kinds, with the form $\Gamma_1 \vdash A_1 : K_1 \Leftrightarrow \Gamma_2 \vdash A_2 : K_2$.

Contexts	$, , \Delta ::=$	$\bullet$	Empty context
	$\mid$	$, , \alpha{:}K$	Context extension
Kinds	$K, L ::=$	T	Kind of types
	$\mid$	$S(A)$	Singleton kind
	$\mid$	$\Pi\alpha{:}K_1.K_2$	Dependent function kind
	$\mid$	$\Sigma\alpha{:}K_1.K_2$	Dependent product kind
Constructors	$A, B, C ::=$	b_i	Base types
	$\mid$	$\alpha, \beta, \ldots$	Variables
	$\mid$	$\lambda\alpha{:}K.A$	Function
	$\mid$	AA'	Application
	$\mid$	$\langle A, A'\rangle$	Pair
	$\mid$	$\pi_i A$	Projection

Figure 1: Syntax of $\lambda_{\leq}^{\Pi\Sigma S}$

Our main technical result is the proof in Section 4 that the algorithm of Section 3 is both complete and terminating. Our proof of completeness is inspired by Coquand's use of Kripke logical relations, but our proof differs substantially from his. Our "worlds" are full contexts rather than sets of bound variables. More importantly, we make use of a novel form of Kripke logical relations in which we employ two worlds, rather than one.

In Section 5 we use this completeness result to show the correctness of the natural algorithm. This yields the practical algorithm used in the TILT implementation.

Finally we discuss related work and conclude.

Appendix A contains the full set of rules for the $\lambda_{\leq}^{\Pi\Sigma S}$ calculus, and Appendix B contains a collection of important but standard properties of the calculus.

2 The $\lambda_{\leq}^{\Pi\Sigma S}$ calculus

2.1 Overview

The syntax of $\lambda_{\leq}^{\Pi\Sigma S}$ is shown in Figure 1. The constants b_i of kind T represent base types such as `int`. As usual, we use the usual notation of $K_1 \times K_2$ for $\Sigma\alpha{:}K_1.K_2$ and $K_1 \to K_2$ for $\Pi\alpha{:}K_1.K_2$ when α is not free in K_2.

There is a natural notion of size for kinds where $\mathit{size}(T) = 1$, $\mathit{size}(S(A)) = 2$, and $\mathit{size}(\Pi\alpha{:}K.K') = \mathit{size}(\Sigma\alpha{:}K.K') = \mathit{size}(K) + \mathit{size}(K') + 2$. The size of a kind is preserved under substitution of terms for variables.

The declarative rules defining the kinding and equivalence system of $\lambda_{\leq}^{\Pi\Sigma S}$ are given in Appendix A. For the most part, these are the usual rules for a dependently-typed lambda calculus with $\beta\eta$ equivalence. We concentrate here on presenting the less common rules.

Since we restrict constructors within singletons to be types (constructors of kind T), we have the following well-formedness rule for singleton kinds:

$$\frac{, \vdash A : T}{, \vdash S(A)}.$$

However, Section 2.3 shows that singletons of constructors of higher kind are definable in this language.

There are two introduction rules for singletons:

$$\frac{, \vdash A : T}{, \vdash A : S(A)} \qquad \frac{, \vdash A \equiv B : T}{, \vdash A \equiv B : S(A)}$$

and a corresponding elimination rule:

$$\frac{,\ \vdash A : S(B)}{,\ \vdash A \equiv B : T}.$$

The calculus includes implicit subsumption, where the subkinding relation is generated by the rules

$$\frac{,\ \vdash A : T}{,\ \vdash S(A) \leq T} \qquad \frac{,\ \vdash A_1 \equiv A_2 : T}{,\ \vdash S(A_1) \leq S(A_2)}$$

and is lifted to subkinding at Π and Σ kinds with the usual co- and contravariance rules. Under this ordering, the singleton introduction rule above allows a constructor A of kind T to be viewed as a constructor of the subkind $S(A)$. Symmetrically, by subsumption any constructor of a singleton kind can be viewed as having the superkind T.

Constructor equivalence includes β and η rules for functions and pairs. We express the η rules as extensionality principles:

$$\frac{\begin{array}{c},\ \vdash A_1 : \Pi\alpha{:}K'.K''_1 \\ ,\ \vdash A_2 : \Pi\alpha{:}K'.K''_2 \\ ,\ ,\alpha{:}K' \vdash A_1\alpha \equiv A_2\alpha : K''\end{array}}{,\ \vdash A_1 \equiv A_2 : \Pi\alpha{:}K'.K''} \qquad \frac{\begin{array}{c},\ \vdash \Sigma\alpha{:}K'.K'' \\ ,\ \vdash \pi_1 A_1 \equiv \pi_1 A_2 : K' \\ ,\ \vdash \pi_2 A_1 \equiv \pi_2 A_2 : \{\alpha\mapsto\pi_1 A_1\}K''\end{array}}{,\ \vdash A_1 \equiv A_2 : \Sigma\alpha{:}K'.K''}$$

The constructor well-formedness rules may be seen as reflexive instances of equivalence rules. For example, we have the following two non-standard kinding rules corresponding to the extensionality rules:

$$\frac{\begin{array}{c},\ \vdash A : \Pi\alpha{:}K'.K''_1 \\ ,\ ,\alpha{:}K' \vdash A\alpha : K''\end{array}}{,\ \vdash A : \Pi\alpha{:}K'.K''} \qquad \frac{\begin{array}{c},\ \vdash \Sigma\alpha{:}K'.K'' \\ ,\ \vdash \pi_1 A : K' \\ ,\ \vdash \pi_2 A : \{\alpha\mapsto\pi_1 A\}K''\end{array}}{,\ \vdash A : \Sigma\alpha{:}K'.K''}$$

Similar rules have previously appeared in the literature, including the non-standard structure-typing rule of Harper, Mitchell, and Moggi [7], the "VALUE" rules of Harper and Lillibridge's translucent sums [6], the strengthening operation of Leroy's manifest type system [8], and the "self" rule of Leroy's applicative functors [9]. In the presence of singletons, these rules give constructors more precise kinds than would otherwise be possible. (See Section 2.3.)

A number of straightforward properties of the $\lambda^{\Pi\Sigma S}_{\leq}$ calculus, used in the following proofs, are given in Appendix B.

2.2 Examples of Term Equivalence

As mentioned in the introduction, singletons in the context can act as definitions and partial definitions for variables. So the provable judgments include:

$$\begin{array}{l}\alpha : S(b_i) \vdash \alpha \equiv b_i : T \\ \alpha : S(b_i) \vdash \langle \alpha, b_i \rangle \equiv \langle b_i, \alpha \rangle : T{\times}T \\ \alpha : T{\times}S(b_i) \vdash \pi_2\alpha \equiv b_i : T \\ \alpha : \Sigma\beta{:}T.S(\beta) \vdash \pi_1\alpha \equiv \pi_2\alpha : T \\ \alpha : \Sigma\beta{:}T.S(\beta) \vdash \alpha \equiv \langle \pi_1\alpha, \pi_1\alpha \rangle : T{\times}T.\end{array}$$

In the last two of these equations, the assumption governing α gives a definition to $\pi_2\alpha$ (namely $\pi_1\alpha$) without specifying what the two equal components actually are.

Singletons behave like terminal types, so by extensionality we can prove equivalences such as:

$$\begin{array}{l}\alpha : S(b_i){\to}T \vdash \alpha \equiv \lambda\beta{:}S(b_i).(\alpha b_i) : S(b_i){\to}T \\ \alpha : T{\to}S(b_i) \vdash \alpha \equiv \lambda\beta{:}T.b_i : T{\to}T\end{array}$$

Notice that in the first of these equations, the right-hand side is not simply an η-expansion of the left-hand side.

$$\begin{array}{lcl}
S(A : T) & := & S(A) \\
S(A : S(A')) & := & S(A) \\
S(A : \Pi\alpha{:}K_1.K_2) & := & \Pi\alpha{:}K_1.(S(A\alpha : K_2)) \\
S(A : \Sigma\alpha{:}K_1.K_2) & := & (S(\pi_1 A : K_1))\times(S(\pi_2 A : \{\alpha\mapsto\pi_1 A\}K_2))
\end{array}$$

Figure 2: Encodings of Labelled Singletons

Because of subkinding, constructors do not have unique kinds. The equivalence of two constructors depends on the kind at which they are compared; they may be equivalent at one kind but not at another. For example, one *cannot* prove

$$\vdash \lambda\alpha{:}T.\alpha \equiv \lambda\alpha{:}T.b_i : T\to T$$

as the identity function and constant function have distinct behaviors. However, by subsumption these two functions also have kind $S(b_i)\to T$ and the judgment

$$\vdash \lambda\alpha{:}T.\alpha \equiv \lambda\alpha{:}T.b_i : S(b_i)\to T$$

is provable using extensionality.

The classifying kind at which constructors are compared depends on the context of their occurrence. For example, from this last equation it follows that

$$\beta : (S(b_i)\to T)\to T \vdash \beta(\lambda\alpha{:}T.\alpha) \equiv \beta(\lambda\alpha{:}T.b_i) : T$$

2.3 Labelled Singletons

In our calculus $S(A)$ is well-formed if and only if A is of type T. Aspinall [1] has studied equivalence in a lambda calculus with labelled singletons of the form $S(A : K)$.[2] This represents the kind of all constructors equivalent to A at kind K. Because equivalence depends on the classifier, the label K in these labelled singletons does matter. It follows from the examples of the previous section that $S(\lambda\alpha{:}T.b_i : \Pi\alpha{:}S(b_i).T)$ and $S(\lambda\alpha{:}T.b_i : T\to T)$ are not equivalent; only the former classifies the identity function $\lambda\alpha{:}T.\alpha$.

Our system does not contain such labelled singletons as a primitive notion because they are all definable; Figure 2 gives an inductive definition.

For example, if β has kind $T\to T$, then $S(\beta : T\to T)$ is defined to be $\Pi\alpha{:}T.S(\beta\alpha)$. This can be interpreted as "the kind of all functions which, when applied, yield the same answer as β does". The non-standard kinding rules mentioned in Section 2.1 are vital in proving that β has this kind.

The following proposition shows that the definitions of Figure 2 do have properties analogous to Aspinall's labelled singletons.

Proposition 2.1

1. *Let γ be a substitution mapping variables to terms, extended in the obvious way to constructors and kinds. Then $\gamma(S(A : K)) = S(\gamma A : \gamma K)$.*
2. *If , $\vdash A_2 : K$ and , $\vdash A_1 : S(A_2 : K)$ then , $\vdash A_1 \equiv A_2 : K$.*
3. *If , $\vdash A_1 \equiv A_2 : K$ then , $\vdash A_1 \equiv A_2 : S(A_1 : K)$.*
4. *If , $\vdash A : K$ then , $\vdash S(A : K)$ and , $\vdash A : S(A : K)$.*
5. *If , $\vdash A : K$ then , $\vdash S(A : K) \leq K$.*
6. *If , $\vdash A_1 \equiv A_2 : K_1$ and , $\vdash K_1 \leq K_2$ then , $\vdash S(A_1 : K_1) \leq S(A_2 : K_2)$.*

[2] Aspinall's notation for our $S(A : K)$ is $\{A\}_K$. Our $S(A)$ is not the same as Aspinall's unlabelled singleton $\{A\}$, but rather would correspond to $\{A\}_T$.

Proof:

1. By induction on K.
2. By induction on the size of K.
   - Case $K = T$ and $S(A_2 : K) = S(A_2)$. Then , $\vdash A_1 \equiv A_2 : T$ by Rule 34.
   - Case $K = S(B)$ and $S(A_2 : K) = S(A_2)$. Then , $\vdash A_1 \equiv A_2 : T$ and , $\vdash A_2 \equiv B : T$, so , $\vdash A_1 \equiv A_2 : S(B)$.
   - Case $K = \Pi\alpha{:}K_1.K_2$ and $S(A_2 : K) = \Pi\alpha{:}K_1.S(A_2\alpha : K_2)$. Then , $,\alpha{:}K_1 \vdash A_1\alpha : S(A_2\alpha : K_2)$. By the inductive hypothesis, , $,\alpha{:}K_1 \vdash A_1\alpha \equiv A_2\alpha : K_2$. Therefore by Rule 30 we have , $\vdash A_1 \equiv A_2 : \Pi\alpha{:}K_1.K_2$.
   - $K = \Sigma\alpha{:}K_1.K_2$ and $S(A_2 : K) = (S(\pi_1 A_2 : K_1)){\times}(S(\pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_2\}K_2))$. Then , $\vdash \pi_1 A_1 : S(\pi_1 A_2 : K_1)$ and , $\vdash \pi_2 A_1 : S(\pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_1\}K_2)$. By the inductive hypothesis, , $\vdash \pi_1 A_1 \equiv \pi_1 A_2 : K_1$ and , $\vdash \pi_2 A_1 \equiv \pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_1\}K_2$. Therefore by Rule 31 we have , $\vdash A_1 \equiv A_2 : \Sigma\alpha{:}K_1.K_2$.
3. By induction on the size of K.
   - Case $K = T$ and $S(A_1 : K) = S(A_1)$, and , $\vdash A_1 \equiv A_2 : S(A_1)$.
   - Case $K = S(B)$ and $S(A_2 : K) = S(A_2)$. Straightforward.
   - Case $K = \Pi\alpha{:}K'.K''$ and $S(A_1 : K) = \Pi\alpha{:}K'.S(A_1\alpha : K'')$. Then , $,\alpha{:}K' \vdash A_1\alpha \equiv A_2\alpha : K''$. By the inductive hypothesis, , $,\alpha{:}K' \vdash A_1\alpha \equiv A_2\alpha : S(A_1\alpha : K'')$. Therefore by Rule 30, , $\vdash A_1 \equiv A_2 : \Pi\alpha{:}K''.S(A_1\alpha : K'')$.
   - $K = \Sigma\alpha{:}K'.K''$ and $S(A_1 : K) = (S(\pi_1 A_1 : K')){\times}(S(\pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K''))$. Then , $\vdash \pi_1 A_1 \equiv \pi_1 A_2 : K'$ and , $\vdash \pi_2 A_1 \equiv \pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_1\}K''$. By the inductive hypothesis, , $\vdash \pi_1 A_1 \equiv \pi_1 A_2 : S(\pi_1 A_1 : K')$ and , $\vdash \pi_2 A_1 \equiv \pi_2 A_2 : S(\pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K'')$. Therefore by Rule 31 we have , $\vdash A_1 \equiv A_2 : (S(\pi_1 A_1 : K')){\times}(S(\pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K''))$

   (Note the crucial use of extensionality in the Π and Σ cases.)
4. By reflexivity of equivalence, Part 3, and Lemma B.1.
5. By induction on the size of K.
   - Case $K = T$ and $S(A : K) = S(A)$. Assume , $\vdash A : T$. By Rule 9 we have , $\vdash S(A : T) \leq T$.
   - Case $K = S(B)$ and $S(A : K) = S(A)$. Assume , $\vdash A : S(B)$. Then , $\vdash A \equiv B : T$ so , $\vdash S(A) \leq S(B)$.
   - Case $K = \Pi\alpha{:}K_1.K_2$ and $S(A : K) = \Pi\alpha{:}K_1.S(A\alpha : K_2)$. Then , $\vdash K_1$ and , $,\alpha{:}K_1 \vdash A\alpha : K_2$. By the inductive hypothesis, , $,\alpha{:}K_1 \vdash S(A\alpha : K_2) \leq K_2$. Therefore, , $\vdash \Pi\alpha{:}K_1.S(A\alpha : K_2) \leq \Pi\alpha{:}K_1.K_2$.
   - Case $K = \Sigma\alpha{:}K'.K''$ and $S(A : K) = (S(\pi_1 A : K')){\times}(S(\pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K''))$. Then , $\vdash \pi_1 A : K'$ so by the inductive hypothesis, , $\vdash S(\pi_1 A : K') \leq K'$. Furthermore, , $\vdash \pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K''$. By the inductive hypothesis, , $\vdash S(\pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K'') \leq \{\alpha{\mapsto}\pi_1 A\}K''$. Also, by Lemma B.1 and Weakening, , $,\alpha{:}S(\pi_1 A : K') \vdash K'' \leq K''$ and by Part 4 , $,\alpha{:}S(\pi_1 A : K') \vdash \alpha \equiv \pi_1 A : K'$ so by Lemma B.11 , $,\alpha{:}S(\pi_1 A : K') \vdash \{\alpha{\mapsto}\pi_1 A\}K'' \leq K''$. Therefore, , $\vdash (S(\pi_1 A : K')){\times}(S(\pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K'')) \leq \Sigma\alpha{:}K'.K''$.
6. By induction on the size of K_1.
   - Case $K_1 = T$ or $S(A_1)$ and $K_2 = T$ or $S(A_2)$. $S(A_1 : K_1) = S(A_1)$, $S(A_2 : K_2) = S(A_2)$, and the desired conclusion follows by Rule 11.
   - Case $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$. $S(A_i : K_i) = \Pi\alpha{:}K_i'.S(A_i\alpha : K_i'')$. By inversion , $\vdash K_2' \leq K_1'$ and , $,\alpha{:}K_2' \vdash K_1'' \leq K_2''$. Now , $,\alpha{:}K_2' \vdash A_1\alpha \equiv A_2\alpha : K_1''$. By the inductive hypothesis, , $,\alpha{:}K_2' \vdash S(A_1\alpha : K_1'') \leq S(A_2\alpha : K_2'')$. The conclusion follows by Rule 12.
   - Case $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$. $S(A_1 : K_1) = \Sigma\alpha{:}S(\pi_1 A_1 K_1' : .)S(\pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K_1'')$ and $S(A_2 : K_2) = \Sigma\alpha{:}S(\pi_1 A_2 K_2' : .)S(\pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_2\}K_2'')$. Now , $\vdash \pi_1 A_1 \equiv \pi_1 A_2 : K_1'$ and , $\vdash \pi_2 A_1 \equiv \pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_1\}K_1''$. By the inductive hypothesis, , $\vdash S(\pi_1 A_1 : K_1') \leq S(\pi_1 A_2 : K_2')$. Since , $\vdash \{\alpha{\mapsto}\pi_1 A_1\}K_1'' \leq \{\alpha{\mapsto}\pi_1 A_2\}K_2''$, the inductive hypothesis applies yielding , $\vdash S(\pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K_1'') \leq S(\pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_2\}K_2'')$. (Here it is important that the induction is on the size of K_1 and not by induction on the proof , $\vdash K_1 \leq K_2$.) The desired result follows by Weakening and Rule 13.

$$\begin{array}{ll}
, \vdash b_i \Uparrow S(b_i) & \\
, \vdash \alpha \Uparrow S(\alpha : , (\alpha)) & \\
, \vdash \lambda\alpha{:}K'.A \Uparrow \Pi\alpha{:}K'.K'' & \text{where } , , \alpha : K' \vdash A \Uparrow K'' \\
, \vdash AA' \Uparrow \{\alpha{\mapsto}A'\}K'' & \text{where } , \vdash A \Uparrow \Pi\alpha{:}K'.K'' \\
, \vdash \langle A', A''\rangle \Uparrow K' {\times} K'' & \text{where } , \vdash A' \Uparrow K' \text{ and } , \vdash A'' \Uparrow K''. \\
, \vdash \pi_1 A \Uparrow K' & \text{where } , \vdash A \Uparrow \Pi\alpha{:}K'.K'' \\
, \vdash \pi_2 A \Uparrow \{\alpha{\mapsto}\pi_1 A\}K' & \text{where } , \vdash A \Uparrow \Pi\alpha{:}K'.K''
\end{array}$$

Figure 3: Algorithm for Principal Kind Synthesis

$\blacksquare$

It is curious to note that in our system, as in Aspinall's, β-rules become *admissible* in the presence of singletons. This can be easily seen using Proposition 2.1; for example,

$$\cfrac{\cfrac{\cfrac{\cfrac{, , \alpha{:}K_2 \vdash A : K}{, , \alpha{:}K_2 \vdash A : S(A : K)}}{, \vdash \lambda\alpha{:}K_2.A : \Pi\alpha{:}K_2.S(A : K)} \qquad , \vdash A_2 : K_2}{, \vdash (\lambda\alpha{:}K_2.A)A_2 : S(\{\alpha{\mapsto}A_2\}A : \{\alpha{\mapsto}A_2\}K)}}{, \vdash (\lambda\alpha{:}K_2.A)A_2 \equiv \{\alpha{\mapsto}A_2\}A : \{\alpha{\mapsto}A_2\}K}.$$

For convenience we have chosen to formulate the system with a stronger form of the β-rules (though we conjecture this does not change the system) and we do not use this admissibility result in the remainder of the paper.

2.4 Principal Kinds

Figure 3 gives an algorithm for determining the principal kind of a well-formed constructor. Correctness is shown by the following lemma:

Lemma 2.2
If $, \vdash A : L$ *then* $, \vdash A \Uparrow K$, $, \vdash A : K$, *and* $, \vdash K \leq S(A : L)$.

Proof: [By induction on the proof of the assumption.]

- Case: Rule 18.

$$\frac{, \vdash \text{ok}}{, \vdash b_i : T}$$

  $, \vdash b_i \Uparrow S(b_i)$ and $, \vdash b_i : S(b_i)$. $S(b_i : T) = S(b_i)$. $, \vdash b \equiv b : T$, so $, \vdash S(b) \leq S(b)$.

- Case: Rule 19.

$$\frac{, \vdash \text{ok}}{, \vdash \alpha : , (\alpha)}$$

  1. $, \vdash \alpha \Uparrow S(\alpha : , (\alpha))$ by definition.
  2. By Proposition 2.1, $, \vdash S(\alpha : , (\alpha))$
  3. and $, \vdash \alpha : S(\alpha : , (\alpha))$.
  4. Thus by reflexivity, $, \vdash S(\alpha : , (\alpha)) \leq S(\alpha : , (\alpha))$.

- Case: Rule 20.

$$\frac{, , \alpha{:}K' \vdash A : L''}{, \vdash \lambda\alpha{:}K'.A : \Pi\alpha{:}K'.L''}$$

  1. By the inductive hypothesis $, , \alpha{:}K' \vdash A \Uparrow K''$,
  2. $, , \alpha{:}K' \vdash A : K''$,

3. and $, \, , \alpha{:}K' \vdash K'' \leq S(A : L'')$.
4. Then $, \vdash \lambda\alpha{:}K'.A \Uparrow \Pi\alpha{:}K'.K''$
5. and $, \vdash \lambda\alpha{:}K'.A : \Pi\alpha{:}K'.K''$.
6. Now $, \, , \alpha{:}K' \vdash (\lambda\alpha{:}K'.A)\alpha \equiv A : L''$,
7. so $, \, , \alpha{:}K' \vdash S(A : L'') \leq S((\lambda\alpha{:}K'.A)\alpha : L'')$ by Proposition 2.1.
8. Since $S(\lambda\alpha{:}K'.A : \Pi\alpha{:}K'.L'') = \Pi\alpha{:}K'.S((\lambda\alpha{:}K'.A)\alpha : L'')$
9. and $, \vdash K' \leq K'$,
10. we have $, \vdash \Pi\alpha{:}K'.K'' \leq S(\lambda\alpha{:}K'.A : \Pi\alpha{:}K'.L'')$.

- Case: Rule 21.

$$\frac{, \vdash A : \Pi\alpha{:}L'.L'' \qquad , \vdash A' : L'}{, \vdash AA' : \{\alpha\mapsto A'\}L''}$$

1. By the inductive hypothesis $, \vdash A \Uparrow K$
2. $, \vdash A : K$
3. and $, \vdash K \leq S(A : \Pi\alpha{:}L'.L'')$.
4. Now $S(A : \Pi\alpha{:}L'.L'') = \Pi\alpha{:}L'.S(A\alpha : L'')$.
5. By inversion $K = \Pi\alpha{:}K'.K''$,
6. $, \vdash L' \leq K'$,
7. and $, \, , \alpha{:}L' \vdash K'' \leq S(A\alpha : L'')$.
8. Then $, \vdash AA' \Uparrow \{\alpha\mapsto A'\}K''$.
9. By subsumption, $, \vdash A' : K'$, so
10. $, \vdash AA' : \{\alpha\mapsto A'\}K''$.
11. Finally, by Lemma B.4 and Proposition 2.1 we have $, \vdash \{\alpha\mapsto A'\}K'' \leq S(AA' : \{\alpha\mapsto A'\}L'')$.

- Case: Rule 22

$$\frac{, \vdash A : \Sigma\alpha{:}L'.L''}{, \vdash \pi_1 A : L'}$$

1. By the inductive hypothesis, $, \vdash A \Uparrow K$,
2. $, \vdash A : K$,
3. and $, \vdash K \leq S(A : \Sigma\alpha{:}L'.L'')$.
4. Now $S(A : \Sigma\alpha{:}L'.L'') = S(\pi_1 A : L') \times S(\pi_2 A : \{\alpha\mapsto\pi_1 A\}L'')$.
5. By inversion, $K = \Sigma\alpha{:}K'.K''$,
6. and $, \vdash K' \leq S(\pi_1 A : L')$.
7. Finally, $, \vdash \pi_1 A \Uparrow K'$
8. and $, \vdash \pi_1 A : K'$.

- Case: Rule 23

$$\frac{, \vdash A : \Sigma\alpha{:}L'.L''}{, \vdash \pi_2 A : \{\alpha\mapsto\pi_1 A\}L''}$$

1. By the inductive hypothesis, $, \vdash A \Uparrow K$,
2. $, \vdash A : K$,
3. and $, \vdash K \leq S(A : \Sigma\alpha{:}L'.L'')$.
4. Now $S(A : \Sigma\alpha{:}L'.L'') = S(\pi_1 A : L') \times S(\pi_2 A : \{\alpha\mapsto\pi_1 A\}L'')$.
5. By inversion, $K = \Sigma\alpha{:}K'.K''$,
6. $, \vdash K' \leq S(\pi_1 A : L')$,
7. and $, \, , \alpha{:}K' \vdash K'' \leq S(\pi_2 A : \{\alpha\mapsto\pi_1 A\}L'')$.
8. Then $, \vdash \pi_1 A : K'$.

9. so by Lemma B.4 and Proposition 2.1, $, \vdash \{\alpha\mapsto\pi_1 A\}K'' \leq S(\pi_2 A : \{\alpha\mapsto\pi_1 A\}L'')$.
10. Finally, $, \vdash \pi_2 A \Uparrow \{\alpha\mapsto\pi_1 A\}K''$
11. and $, \vdash \pi_2 A : \{\alpha\mapsto\pi_1 A\}K''$.

- Case: Rule 24

$$\frac{\begin{array}{c}, \vdash \Sigma\alpha{:}L'.L'' \\ , \vdash A' : L' \\ , \vdash A'' : \{\alpha\mapsto A'\}L''\end{array}}{, \vdash \langle A', A''\rangle : \Sigma\alpha{:}L'.L''}$$

1. By the inductive hypothesis, $, \vdash A' \Uparrow K'$,
2. $, \vdash A' : K'$,
3. $, \vdash K' \leq S(A' : L')$,
4. $, \vdash A'' \Uparrow K''$,
5. $, \vdash A'' : K''$,
6. and $, \vdash K'' \leq S(A'' : \{\alpha\mapsto A'\}L'')$.
7. Then $, \vdash \langle A', A''\rangle \Uparrow K'{\times}K''$,
8. and $, \vdash \langle A', A''\rangle : K'{\times}K''$.
9. Now $S(\langle A', A''\rangle : \Sigma\alpha{:}L'.L'') = S(\pi_1\langle A', A''\rangle : L'){\times}S(\pi_2\langle A', A''\rangle : \{\alpha\mapsto\pi_1\langle A', A''\rangle\}L'')$.
10. By Proposition 2.1, $, \vdash S(A' : L') \leq S(\pi_1\langle A', A''\rangle : L')$
11. and $, \vdash S(A'' : \{\alpha'\mapsto A'\}L'') \leq S(\pi_2\langle A', A''\rangle : \{\alpha\mapsto\pi_1\langle A', A''\rangle\}L'')$.
12. Therefore, $, \vdash K'{\times}K'' \leq S(\langle A', A''\rangle : \Sigma\alpha{:}L'.L'')$.

- Case: Rule 25

$$\frac{, \vdash A : T}{, \vdash A : S(A)}$$

By the inductive hypothesis, noting that $S(A : S(A)) = S(A)$.

- Case: Rule 27

$$\frac{\begin{array}{c}, \vdash A : \Pi\alpha{:}L'.L''_1 \\ , , \alpha{:}L' \vdash A\alpha : L''\end{array}}{, \vdash A : \Pi\alpha{:}L'.L''}$$

1. By the inductive hypothesis, $, \vdash A \Uparrow K$,
2. $, \vdash A : K$,
3. and $, \vdash K \leq S(A : \Pi\alpha{:}L'.L''_1)$.
4. Now $S(A : \Pi\alpha{:}L'.L''_1) = \Pi\alpha{:}L'.S(A\alpha : L''_1)$
5. so by inversion $K = \Pi\alpha{:}K'.K''$
6. and $, \vdash L' \leq K'$.
7. Also by the inductive hypothesis $, , \alpha{:}L' \vdash A\alpha \Uparrow K''_2$,
8. $, , \alpha{:}L' \vdash A\alpha : K''_2$,
9. and $, , \alpha{:}L' \vdash K''_2 \leq S(A\alpha : L'')$.
10. But since the principal kind synthesis algorithm is deterministic and clearly obeys weakening, we have $K''_2 = \{\alpha\mapsto\alpha\}K'' = K''$.
11. Now $S(A : \Pi\alpha{:}L'.L'') = \Pi\alpha{:}L'.S(A\alpha : L'')$.
12. Therefore $, \vdash \Pi\alpha{:}K'.K'' \leq S(A : \Pi\alpha{:}L'.L'')$.

- Case: Rule 26.

$$\frac{\begin{array}{c}, \vdash \Sigma\alpha{:}L'.L'' \\ , \vdash \pi_1 A : L' \\ , \vdash \pi_2 A : \{\alpha\mapsto\pi_1 A\}L''\end{array}}{, \vdash A : \Sigma\alpha{:}L'.L''}$$

1. First, note that principal kind synthesis never returns a dependent Σ type.
2. By Lemma B.17 and the inductive hypothesis $, \vdash A \Uparrow K' \times K''$ and $, \vdash A : K' \times K''$.
3. Also, $, \vdash \pi_1 A \Uparrow K'$,
4. $, \vdash \pi_1 A : K'$,
5. and $, \vdash K' \leq S(\pi_1 A : L')$.
6. Also, $, \vdash \pi_2 A \Uparrow K''$,
7. $, \vdash \pi_2 A : K''$,
8. and $, \vdash K'' \leq S(\pi_2 A : \{\alpha \mapsto \pi_1 A\} L'')$.
9. Since $S(A : \Sigma\alpha{:}L'.L'') = S(\pi_1 A : L') \times S(\pi_2 A : \{\alpha \mapsto \pi_1 A\} L'')$,
10. $, \vdash K' \times K'' \leq S(A : \Sigma\alpha{:}L'.L'')$.

- Rule 28

$$\frac{, \vdash A : L_2 \qquad , \vdash L_2 \leq L}{, \vdash A : L}$$

The desired result follows from the inductive hypothesis and by Proposition 2.1 to get $, \vdash S(A : L_2) \leq S(A : L)$.

∎

3 An Algorithm for Constructor Equivalence

Following Coquand, we present the equivalence test by defining a set of rules defining algorithmic relations, shown in Figure 4. It is clear that these rules can be translate directly into a deterministic algorithm, since for any goal there is at most one algorithmic rule which can apply. Then decidability of the algorithmic relations corresponds to termination of the algorithm.

Our algorithm is somewhat more involved than that of Coquand because of the context and kind-dependence of equivalence. We divide the algorithmic constructor equivalence rules into a kind-directed part and a structure-directed part, while Coquand needs only structural comparison. Our weak head normalization includes looking for definitions in the context. We have also extended the algorithm in the natural fashion to handle Σ types, pairing, and projection.

Define an *elimination context* to be a series of applications to and projections from "$\diamond$", which we call the context's hole. If E is such a context, then $E[A]$ represents the constructor resulting by replacing the hole in E with A. If a constructor is either of the form $E[\alpha]$ or of the form $E[b_i]$ then we will call this a *path* and denote it by p.

$$\begin{array}{rcl} E & ::= & \diamond \\ & | & E A \\ & | & \pi_1 E \\ & | & \pi_2 E \end{array}$$

The kind extraction relation $, \vdash p \uparrow K$ attempts to determine a kind for a path by taking the kind of the head variable or constant and doing appropriate substitutions and projections. A path is said to *have a definition* if its extracted kind is a singleton kind $S(B)$; in this case we say B is the definition of the path.

The extracted kind is not always the most precise kind. For example, $\alpha{:}T \vdash \alpha \uparrow T$ but the principal type of α in this context would be $S(\alpha)$. We must show that given a well-formed path, kind extraction succeeds and returns a valid kind for this path using induction on the well-formedness proof for the path (with a strengthened induction hypothesis).

Lemma 3.1
If $, \vdash p : K$ *then* $, \vdash p \uparrow L$, $, \vdash p : L$, *and* $, \vdash S(p : L) \leq K$.

Proof: By induction on the proof of the hypothesis.

Kind Extraction

$, \vdash b_i \uparrow T$	
$, \vdash \alpha \uparrow , (\alpha)$	
$, \vdash \pi_1 p \uparrow K_1$	if $, \vdash p \uparrow \Sigma\beta{:}K_1.K_2$
$, \vdash \pi_2 p \uparrow \{\beta{\mapsto}\pi_1 p\}K_2$	if $, \vdash p \uparrow \Sigma\beta{:}K_1.K_2$
$, \vdash pA \uparrow \{\beta{\mapsto}A\}K_2$	if $, \vdash p \uparrow \Pi\beta{:}K_1.K_2$

Weak head reduction

$, \vdash E[(\lambda\alpha{:}K.A)A'] \rightsquigarrow E[\{\alpha{\mapsto}A'\}A]$	
$, \vdash E[\pi_1\langle A_1, A_2\rangle] \rightsquigarrow E[A_1]$	
$, \vdash E[\pi_2\langle A_1, A_2\rangle] \rightsquigarrow E[A_2]$	
$, \vdash E[p] \rightsquigarrow E[B]$	if $, \vdash p \uparrow S(B)$

Weak head normalization

$, \vdash A \Downarrow B$	if $, \vdash A \rightsquigarrow A'$ and $, \vdash A' \Downarrow B$
$, \vdash A \Downarrow A$	otherwise

Algorithmic constructor equivalence

$,_1 \vdash A_1 : T \Leftrightarrow ,_2 \vdash A_2 : T$	if $,_1 \vdash A_1 \Downarrow p_1$, $,_2 \vdash A_2 \Downarrow p_2$, $,_1 \vdash p_1 \uparrow T \leftrightarrow ,_2 \vdash p_2 \uparrow T$
$,_1 \vdash A_1 : S(B_1) \Leftrightarrow ,_2 \vdash A_2 : S(B_2)$	always
$,_1 \vdash A_1 : \Pi\alpha{:}K_1.L_1 \Leftrightarrow ,_2 \vdash A_2 : \Pi\alpha{:}K_2.L_2$	if $,_1, \alpha{:}K_1 \vdash A_1\alpha : L_1 \Leftrightarrow ,_2, \alpha{:}K_2 \vdash A_2\alpha : L_2$
$,_1 \vdash A_1 : \Sigma\alpha{:}K_1.L_1 \Leftrightarrow ,_2 \vdash A_2 : \Sigma\alpha{:}K_2.L_2$	if $,_1 \vdash \pi_1 A_1 : K_1 \Leftrightarrow ,_2 \vdash \pi_1 A_2 : K_2$, and $,_1 \vdash \pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}L_1 \Leftrightarrow ,_2 \vdash \pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_2\}L_2$

Algorithmic path equivalence

$,_1 \vdash b_i \uparrow T \leftrightarrow ,_2 \vdash b_j \uparrow T$	if $i = j$
$,_1 \vdash \alpha \uparrow ,_1(\alpha) \leftrightarrow ,_2 \vdash \alpha \uparrow ,_2(\alpha)$	always
$,_1 \vdash p_1 A_1 \uparrow \{\alpha{\mapsto}A_1\}L_1 \leftrightarrow$ $,_2 \vdash p_2 A_2 \uparrow \{\alpha{\mapsto}A_2\}L_2$	if $,_1 \vdash p_1 \uparrow \Pi\alpha{:}K_1.L_1 \leftrightarrow ,_2 \vdash p_2 \uparrow \Pi\alpha{:}K_2.L_2$, and $,_1 \vdash A_1 : K_1 \Leftrightarrow ,_2 \vdash A_2 : K_2$.
$,_1 \vdash \pi_1 p_1 \uparrow K_1 \leftrightarrow ,_2 \vdash \pi_1 p_2 \uparrow K_2$	if $,_1 \vdash p_1 \uparrow \Sigma\alpha{:}K_1.L_1 \leftrightarrow ,_2 \vdash p_2 \uparrow \Sigma\alpha{:}K_2.L_2$.
$,_1 \vdash \pi_2 p_1 \uparrow \{\alpha{\mapsto}\pi_1 p_1\}L_1 \leftrightarrow$ $,_2 \vdash \pi_2 p_2 \uparrow \{\alpha{\mapsto}\pi_1 p_2\}L_2$	if $,_1 \vdash p_1 \uparrow \Sigma\alpha{:}K_1.L_1 \leftrightarrow ,_2 \vdash p_2 \uparrow \Sigma\alpha{:}K_2.L_2$

Algorithmic kind equivalence

$,_1 \vdash T \Leftrightarrow ,_2 \vdash T$	always
$,_1 \vdash S(A_1) \Leftrightarrow ,_2 \vdash S(A_2)$	if $,_1 \vdash A_1 : T \Leftrightarrow ,_2 \vdash A_2 : T$
$,_1 \vdash \Pi\alpha{:}K_1.L_1 \Leftrightarrow ,_2 \vdash \Pi\alpha{:}K_2.L_2$	if $,_1 \vdash K_1 \Leftrightarrow ,_2 \vdash K_2$ and $,_1, \alpha{:}K_1 \vdash L_1 \Leftrightarrow ,_2, \alpha{:}K_2 \vdash L_2$
$,_1 \vdash \Sigma\alpha{:}K_1.L_1 \Leftrightarrow ,_2 \vdash \Sigma\alpha{:}K_2.L_2$	if $,_1 \vdash K_1 \Leftrightarrow ,_2 \vdash K_2$ and $,_1, \alpha{:}K_1 \vdash L_1 \Leftrightarrow ,_2, \alpha{:}K_2 \vdash L_2$

Figure 4: Algorithmic Relations

- Case: Rule 18. $p = b_i$.
  1. Then $, \vdash b_i \uparrow T$ and $S(b_i : T) = S(b_i)$.
  2. By Rule 18, $, \vdash b_i : T$
  3. and by Rule 9, $, \vdash S(b_i) \leq T$.
- Case: Rule 19. $p = \alpha$.
  1. Then $, \vdash \alpha \uparrow , (\alpha)$.
  2. By Rule 19 $, \vdash \alpha : , (\alpha)$,
  3. and by Proposition 2.1 Part 5, $, \vdash S(\alpha : , (\alpha)) \leq , (\alpha)$.
- Case: Rule 21.

$$\frac{, \vdash p : \Pi\alpha{:}K'.K'' \qquad , \vdash A' : K'}{, \vdash pA' : \{\alpha{\mapsto}A'\}K''}$$

1. By the inductive hypothesis, $, \vdash p \uparrow \Pi\alpha{:}L'.L''$,
2. $, \vdash p : \Pi\alpha{:}L'.L''$, and
3. $, \vdash S(p : \Pi\alpha{:}L'.L'') \leq \Pi\alpha{:}K'.K''$.
4. Then $, \vdash pA' \uparrow \{\alpha{\mapsto}A'\}L''$.
5. Since $S(p : \Pi\alpha{:}L'.L'') = \Pi\alpha{:}L'.S(p\alpha : L'')$,
6. we have by inversion of Rule 12 that $, \vdash K' \leq L'$ and $, , \alpha{:}K' \vdash S(p\alpha : L'') \leq K''$.
7. By subsumption, $, \vdash A' : L'$
8. and hence $, \vdash pA' : \{\alpha{\mapsto}A'\}L''$ by Rule 21.
9. Finally, by Lemma B.4 we have $, \vdash S(pA' : \{\alpha{\mapsto}A'\}L'') \leq \{\alpha{\mapsto}A'\}K''$.

- Case: Rule 22.

$$\frac{, \vdash p : \Sigma\alpha{:}K'.K''}{, \vdash \pi_1 p : K'}$$

1. By the inductive hypothesis, $, \vdash p \uparrow \Sigma\alpha{:}L'.L''$,
2. $, \vdash p : \Sigma\alpha{:}L'.L''$, and
3. $, \vdash S(p : \Sigma\alpha{:}L'.L'') \leq \Sigma\alpha{:}K'.K'$.
4. Then $, \vdash \pi_1 p \uparrow L'$,
5. and by Rule 22, $, \vdash \pi_1 p : L'$.
6. Since $S(p : \Sigma\alpha{:}L'.L'') = S(\pi_1 p : L') \times S(\pi_2 p : \{\alpha{\mapsto}\pi_1 p\}L'')$,
7. by inversion of rule 13 we have $, \vdash S(\pi_1 p : L') \leq K'$.

- Case: Rule 23.

$$\frac{, \vdash p : \Sigma\alpha{:}K'.K''}{, \vdash \pi_2 p : \{\alpha{\mapsto}\pi_1 p\}K'}$$

1. By the inductive hypothesis, $, \vdash p \uparrow \Sigma\alpha{:}L'.L''$,
2. $, \vdash p : \Sigma\alpha{:}L'.L''$, and
3. $, \vdash S(p : \Sigma\alpha{:}L'.L'') \leq \Sigma\alpha{:}K'.K''$.
4. Then $, \vdash \pi_2 p \uparrow \{\alpha{\mapsto}\pi_1 p\}L''$,
5. and $, \vdash \pi_2 p : \{\alpha{\mapsto}\pi_1 p\}L''$ by Rule 23.
6. Since $S(p : \Sigma\alpha{:}L'.L'') = S(\pi_1 p : L') \times S(\pi_2 p : \{\alpha{\mapsto}\pi_1 p\}L'')$,
7. by inversion of Rule 13 $, , \alpha{:}S(\pi_1 p : L') \vdash S(\pi_2 p : \{\alpha{\mapsto}\pi_1 p\}L'') \leq K''$.
8. Then $, \vdash \pi_1 p : S(\pi_1 p' : L')$
9. so by the Substitution Lemma B.4 we have $, \vdash S(\pi_2 p : \{\alpha{\mapsto}\pi_1 p\}L'') \leq \{\alpha{\mapsto}\pi_1 p\}K''$.

- Case: Rule 25

$$\frac{, \vdash p : T}{, \vdash p : S(p)}$$

1. By the inductive hypothesis, $, \vdash p \uparrow L$,
2. $, \vdash p : L$,
3. and $, \vdash S(p : L) \leq T$.
4. Thus L is either T or a singleton, and $S(p : L) = S(p)$.
5. and by reflexivity, $, \vdash S(p) \leq S(p)$.

- Case: Rule 26.

$$\frac{\begin{array}{c}, \vdash \Sigma\alpha{:}K'.K'' \\ , \vdash \pi_1 p : K' \\ , \vdash \pi_2 p : \{\alpha{\mapsto}\pi_1 p\}K''\end{array}}{, \vdash p : \Sigma\alpha{:}K'.K''}$$

1. By Lemma B.17 and the inductive hypothesis, $, \vdash p \uparrow \Sigma\alpha{:}L'.L''$,
2. $, \vdash p : \Sigma\alpha{:}L'.L''$,

3. $, \vdash \pi_1 p \uparrow L'$,
4. $, \vdash \pi_1 p : L'$,
5. $, \vdash S(\pi_1 p : L') \leq K'$,
6. $, \vdash \pi_2 p \uparrow \{\alpha \mapsto \pi_1 p\} L''$,
7. $, \vdash \pi_2 p : \{\alpha \mapsto \pi_1 p\} L''$,
8. and $, \vdash S(\pi_2 p : \{\alpha \mapsto \pi_1 p\} L'') \leq \{\alpha \mapsto \pi_1 p\} K''$.
9. Now to show that $, \vdash S(p : \Sigma\alpha{:}L'.L'') \leq \Sigma\alpha{:}K'.K''$
10. it remains to show that $, \alpha{:}S(\pi_1 p : L') \vdash S(\pi_2 p : \{\alpha \mapsto \pi_1 p\} L'') \leq K''$.
11. But $, \alpha{:}S(\pi_1 p : L') \vdash \{\alpha \mapsto \pi_1 p\} K'' \equiv K''$,
12. so the desired result follows from Line 8 and transitivity.

- Case: Rule 27.

$$\frac{\begin{array}{c}, \vdash p : \Pi\alpha{:}K'.K''_1 \\ , \alpha{:}K' \vdash p\alpha : K''\end{array}}{, \vdash p : \Pi\alpha{:}K'.K''}$$

1. By the inductive hypothesis, $, \vdash p \uparrow \Pi\alpha{:}L'.L''$,
2. $, \vdash p : \Pi\alpha{:}L'.L''$,
3. and $, \vdash (\Pi\alpha{:}L'.S(p\alpha : L'')) \leq \Pi\alpha{:}K'.K''_1$.
4. By inversion, $, \vdash K' \leq L'$.
5. By the inductive hypothesis, and determinacy and weakening of the kind extraction algorithm, $, \alpha{:}K' \vdash p\alpha \uparrow L''$
6. and $, \alpha{:}K' \vdash S(p\alpha : L'') \leq K''$.
7. Therefore, $, \vdash \Pi\alpha{:}L'.S(p\alpha : L'') \leq \Pi\alpha{:}K'.K''$.

- Case: Rule 28.

$$\frac{, \vdash p : K_1 \qquad , \vdash K_1 \leq K_2}{, \vdash p : K_2}$$

1. By the inductive hypothesis, $, \vdash p \uparrow L$,
2. $, \vdash p : L$,
3. and $, \vdash S(p : L) \leq K_1$.
4. By transitivity, $, \vdash S(p : L) \leq K_2$.

■

Corollary 3.2
If $, \vdash E[p] : K$ *and* $, \vdash p \uparrow S(A)$ *then* $, \vdash E[p] \equiv E[A] : K$.

Proof:

1. By Lemma 3.1, $, \vdash E[p] \uparrow L$,
2. $, \vdash E[p] : L$,
3. and $, \vdash S(E[p] : L) \leq K$.
4. By the determinacy of kind extraction, this can be reconciled with $, \vdash p \uparrow S(A)$ only if $E = \diamond$ and $L = S(A)$.
5. Thus by Rule 34, $, \vdash p \equiv A : T$.
6. Now $S(E[p] : L) = S(p)$.
7. By inversion of subkinding, either $K = T$ or $K = S(A')$ with $, \vdash p \equiv A' : T$.
8. In either case, $, \vdash p \equiv A : K$.
9. That is, $, \vdash E[p] \equiv E[A] : K$ as desired.

$\blacksquare$

The weak head reduction relation $, \vdash A \rightsquigarrow B$ contracts the head β-redex of A, if such a redex exists. Otherwise, when the head of A is a path with a definition reduction replaces the head with the definition.

Weak head normalization $, \vdash A \Downarrow B$ repeatedly applies weak head reduction to A until a weak head normal form is found. Weak head reduction and weak head normalization are deterministic, since the head β-redex is always unique if one exists, and a path can have at most one prefix with a definition.

The algorithmic term equivalence relation

$$,_1 \vdash A_1 : K_1 \Leftrightarrow ,_2 \vdash A_2 : K_2$$

is intended to model the declarative equivalence $,_1 \vdash A_1 \equiv A_2 : K_1$, when $\vdash ,_1 \equiv ,_2$ and $,_1 \vdash K_1 \equiv K_2$.

The algorithmic path equivalence relation

$$,_1 \vdash p_1 \uparrow K_1 \leftrightarrow ,_2 \vdash p_2 \uparrow K_2$$

will be shown to implement constructor equality for head normal paths when $\vdash ,_1 \equiv ,_2$. As a notational convenience, this relation explicitly includes the extracted kinds of the two paths being compared.

Lemma 3.3
If $,_1 \vdash A_1 \uparrow K_1 \leftrightarrow ,_2 \vdash A_2 \uparrow K_2$ *then* $,_1 \vdash A_1 \uparrow K_1$ *and* $,_2 \vdash A_2 \uparrow K_2$.

Finally, the algorithmic kind equivalence relation

$$,_1 \vdash K_1 \Leftrightarrow ,_2 \vdash K_2$$

determines whether two kinds are equivalent given $\vdash ,_1 \equiv ,_2$. This easily reduces to checking the equivalence of constructors appearing within singleton kinds.

To prove soundness of this equivalence algorithm, we first prove that weak-head normalization preserves equivalence.

Proposition 3.4
If $, \vdash E[(\lambda\alpha{:}L.A)A'] : K$ *then* $, \vdash E[(\lambda\alpha{:}L.A)A'] \equiv E[\{\alpha{\mapsto}A'\}A] : K$

Proof: By induction on the given derivation.

- Case:
$$\frac{, \vdash \lambda\alpha{:}L'.A : \Pi\alpha{:}K'.K'' \qquad , \vdash A' : K'}{, \vdash (\lambda\alpha{:}L'.A)A' : \{\alpha{\mapsto}A'\}K''}$$
where $E = \diamond$.
  1. Using Proposition B.16 and the correctness of principal kind synthesis we have $, ,\alpha{:}L' \vdash A \Uparrow L''$,
  2. $, ,\alpha{:}L' \vdash A : L''$,
  3. $, \vdash \lambda\alpha{:}L'.A \Uparrow \Pi\alpha{:}L'.L''$,
  4. $, \vdash \lambda\alpha{:}L'.A : \Pi\alpha{:}L'.L''$,
  5. and (using Proposition 2.1) $, \vdash \Pi\alpha{:}L'.L'' \leq \Pi\alpha{:}K'.K''$.
  6. By inversion, $, \vdash K' \leq L'$
  7. and $, ,\alpha{:}K' \vdash L'' \leq K''$.
  8. By subsumption, $, \vdash A' : L'$.
  9. Thus $, \vdash (\lambda\alpha{:}L.A)A' \equiv \{\alpha{\mapsto}A'\}A : \{\alpha{\mapsto}A'\}L''$
  10. By Substitution $, \vdash \{\alpha{\mapsto}A'\}L'' \leq \{\alpha{\mapsto}A'\}K''$.
  11. Therefore by subsumption we have $, \vdash (\lambda\alpha{:}L.A)A' \equiv \{\alpha{\mapsto}A'\}A : \{\alpha{\mapsto}A'\}K''$
- All other cases follow by structural rules and reflexivity of declarative equivalence.

$\blacksquare$

Proposition 3.5

1. *If* $, \vdash E[\pi_1\langle A', A''\rangle] : K$ *then* $, \vdash E[\pi_1\langle A', A''\rangle] \equiv E[A'] : K$.
2. *If* $, \vdash E[\pi_2\langle A', A''\rangle] : K$ *then* $, \vdash E[\pi_2\langle A', A''\rangle] \equiv E[A''] : K$.
3. *If* $, \vdash \langle A', A''\rangle : \Sigma\alpha{:}K'.K''$ *then* $, \vdash A' : K'$ *and* $, \vdash A'' : \{\alpha{\mapsto}A'\}K''$.

Proof:

1. • Case:

$$\frac{, \vdash \langle A', A''\rangle : \Sigma\alpha{:}K'.K''}{, \vdash \pi_1\langle A', A''\rangle : K'}$$

where $E = \diamond$.

(a) Inductively by Part 3, $, \vdash A' : K'$
(b) and $, \vdash A'' : \{\alpha{\mapsto}A'\}K''$.
(c) The desired result follows by Rule 32.

• The remaining cases follow by structural rules and reflexivity.

2. • Case:

$$\frac{, \vdash \langle A', A''\rangle : \Sigma\alpha{:}K'.K''}{, \vdash \pi_2\langle A', A''\rangle : \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K''}$$

where $E = \diamond$.

(a) Inductively by Part 3, $, \vdash A' : K'$
(b) and $, \vdash A'' : \{\alpha{\mapsto}A'\}K''$.
(c) By Rule 33, $, \vdash \pi_2\langle A', A''\rangle : \{\alpha{\mapsto}A'\}K''$.
(d) As in Part 1, $, \vdash E[\pi_1\langle A', A''\rangle] \equiv E[A'] : K$.
(e) So by Lemma B.11 $, \vdash \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K'' \equiv \{\alpha{\mapsto}A'\}K''$.
(f) Thus by subsumption we have $, \vdash \pi_2\langle A', A''\rangle : \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K''$.

• The remaining cases follow by structural rules and reflexivity.

3. • Case:

$$\frac{\begin{array}{c}, \vdash \Sigma\alpha{:}K'.K'' \\ , \vdash A_1 : K' \\ , \vdash A_2 : \{\alpha{\mapsto}K'\}K''\end{array}}{, \vdash \langle A_1, A_2\rangle : \Sigma\alpha{:}K'.K''}$$

Obvious.

• Case:

$$\frac{\begin{array}{c}, \vdash \Sigma\alpha{:}K'.K'' \\ , \vdash \pi_1\langle A', A''\rangle : K' \\ , \vdash \pi_2\langle A', A''\rangle : \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K''\end{array}}{, \vdash \langle A', A''\rangle : \Sigma\alpha{:}K'.K''}$$

(a) Inductively by Part 1, $, \vdash \pi_1\langle A', A''\rangle \equiv A' : K'$.
(b) Inductively by Part 2, $, \vdash \pi_2\langle A', A''\rangle \equiv A'' : \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K''$.
(c) By Lemma B.11, $, \vdash \{\alpha{\mapsto}\pi_1\langle A', A''\rangle\}K'' \equiv \{\alpha{\mapsto}A'\}K''$.
(d) Thus by subsumption and Lemma B.1, $, \vdash A' : K'$
(e) and $, \vdash A'' : \{\alpha{\mapsto}A'\}K''$.

• Case:

$$\frac{\begin{array}{c}, \vdash \langle A', A''\rangle : K_1 \\ , \vdash K_1 \leq \Sigma\alpha{:}K'.K''\end{array}}{, \vdash \langle A', A''\rangle : \Sigma\alpha{:}K'.K''}$$

(a) By inversion, $K_1 = \Sigma\alpha{:}K_1'.K_1''$,
(b) $, \vdash K_1' \leq K'$,
(c) and $, , \alpha{:}K_1' \vdash K_1'' \leq K''$.
(d) By the inductive hypothesis, $, \vdash A' : K_1'$
(e) and $, \vdash A'' : \{\alpha{\mapsto}A'\}K_1''$.

(f) By Lemma B.4, , $\vdash \{\alpha\mapsto A'\}K_1'' \leq \{\alpha\mapsto A'\}K''$.

(g) Then the desired results follow by subsumption.

■

Corollary 3.6
If , $\vdash A : K$ *and* , $\vdash A \Downarrow B$ *then* , $\vdash A \equiv B : K$.

Proof: By transitivity and reflexivity of declarative equivalence, it suffices to show that if , $\vdash A : K$ and , $\vdash A \rightsquigarrow B$ then , $\vdash A \equiv B : K$. But all possibilities for the reduction step are covered by Lemma 3.1, Proposition 3.4, and Proposition 3.5. ■

Theorem 3.7 (Soundness)

1. *If* $\vdash$, $_1 \equiv$, $_2$, , $_1 \vdash K_1 \equiv K_2$, , $_1 \vdash A_1 : K_1$, , $_2 \vdash A_2 : K_2$, *and* , $_1 \vdash A_1 : K_1 \Leftrightarrow$, $_2 \vdash A_2 : K_2$ *then* , $_1 \vdash A_1 \equiv A_2 : K_1$.

2. *If* $\vdash$, $_1 \equiv$, $_2$, , $_1 \vdash p_1 : L_1$, , $_2 \vdash p_2 : L_2$, *and* , $_1 \vdash p_1 \uparrow K_1 \leftrightarrow$, $_2 \vdash p_2 \uparrow K_2$ *then* , $_1 \vdash K_1 \equiv K_2$ *and* , $_1 \vdash p_1 \equiv p_2 : K_1$.

3. *If* $\vdash$, $_1 \equiv$, $_2$, , $_1 \vdash K_1$, , $_2 \vdash K_2$, *and* , $_1 \vdash K_1 \Leftrightarrow$, $_2 \vdash K_2$ *then* , $_1 \vdash K_1 \equiv K_2$.

Proof: Parts 1 and 2 follow by simultaneous induction on the algorithmic judgments and by cases on the last step in the algorithmic derivation. We omit the proof of Part 3, which follows directly by Part 1 and induction.

1.
- Case: , $_1 \vdash A_1 : T \Leftrightarrow$, $_2 \vdash A_2 : T$ because , $_1 \vdash A_1 \Downarrow p_1$, , $_2 \vdash A_2 \Downarrow p_2$, and , $_1 \vdash p_1 \uparrow T \leftrightarrow$, $_2 \vdash p_2 \uparrow T$.
  (a) By Corollary 3.6, , $_1 \vdash A_1 \equiv p_1 : T$
  (b) and , $_2 \vdash A_2 \equiv p_2 : T$.
  (c) By Corolllary B.13 , $_1 \vdash A_2 \equiv p_2 : T$.
  (d) By Lemma B.1, , $_1 \vdash p_1 : T$
  (e) and , $_2 \vdash p_2 : T$.
  (f) By the inductive hypothesis, , $_1 \vdash p_1 \equiv p_2 : T$.
  (g) By symmetry and transitivity of equivalence therefore, , $_1 \vdash A_1 \equiv A_2 : T$.
- Case: , $_1 \vdash A_1 : S(B_1) \Leftrightarrow$, $_2 \vdash A_2 : S(B_2)$.
  (a) By Rule 34, , $_1 \vdash A_1 \equiv B_1 : T$
  (b) and , $_2 \vdash A_2 \equiv B_2 : T$.
  (c) By inversion of Rule 15, , $_1 \vdash B_1 \equiv B_2 : T$.
  (d) By symmetry, transitivity, and Corollary B.13, , $_1 \vdash A_1 \equiv A_2 : T$.
  (e) By Rule 35 , $_1 \vdash A_1 \equiv A_2 : S(A_1)$.
  (f) But , $\vdash S(A_1) \leq S(B_1)$
  (g) so by subsumption , $_1 \vdash A_1 \equiv A_2 : S(B_1)$.
- Case: , $_1 \vdash A_1 : \Pi\alpha{:}K_1.L_1 \Leftrightarrow$, $_2 \vdash A_2 : \Pi\alpha{:}K_2.L_2$ because , $_1, \alpha{:}K_1 \vdash A_1\alpha : L_1 \Leftrightarrow$, $_2, \alpha{:}K_2 \vdash A_2\alpha : L_2$.
  (a) Since $\vdash$, $_1, \alpha{:}K_1 \equiv$, $_2, \alpha{:}K_2$,
  (b) , $_1, \alpha{:}K_1 \vdash A_1\alpha : L_1$,
  (c) , $_2, \alpha{:}K_2 \vdash A_2\alpha : L_2$,
  (d) and , $_1, \alpha{:}K_1 \vdash L_1 \equiv L_2$,
  (e) the inductive hypothesis applies, yielding , $_1, \alpha{:}K_1 \vdash A_1\alpha \equiv A_2\alpha : L_1$.
  (f) Thus by Rule 30, , $_1 \vdash A_1 \equiv A_2 : \Pi\alpha{:}K_1.L_1$.
- , $_1 \vdash A_1 : \Sigma\alpha{:}K_1.L_1 \Leftrightarrow$, $_2 \vdash A_2 : \Sigma\alpha{:}K_2.L_2$ because , $_1 \vdash \pi_1 A_1 : K_1 \Leftrightarrow$, $_2 \vdash \pi_1 A_2 : K_2$, and , $_1 \vdash \pi_2 A_1 : \{\alpha\mapsto\pi_1 A_1\}L_1 \Leftrightarrow$, $_2 \vdash \pi_2 A_2 : \{\alpha\mapsto\pi_1 A_2\}L_2$.
  (a) Since , $_1 \vdash \pi_1 A_1 : K_1$
  (b) , $_2 \vdash \pi_1 A_2 : K_2$,
  (c) and by inversion , $_1 \vdash K_1 \equiv K_2$,
  (d) by the inductive hypothesis we have , $_1 \vdash \pi_1 A_1 \equiv \pi_1 A_2 : K_1$.
  (e) By Lemma B.11, , $_1 \vdash \{\alpha\mapsto\pi_1 A_1\}L_1 \equiv \{\alpha\mapsto\pi_1 A_2\}L_2$.

(f) Then $,_1 \vdash \pi_2 A_1 : \{\alpha\mapsto\pi_1 A_1\}L_1$

(g) and $,_2 \vdash \pi_2 A_2 : \{\alpha\mapsto\pi_1 A_2\}L_2$.

(h) By the inductive hypothesis, $,_1 \vdash \pi_2 A_1 \equiv \pi_2 A_2 : \{\alpha\mapsto\pi_1 A_1\}L_1$.

(i) By Corollary B.13 and Rule 31, $,_1 \vdash A_1 \equiv A_2 : \Sigma\alpha{:}K_1.L_1$.

2. • Case: $,_1 \vdash b_i \uparrow T \leftrightarrow\ ,_2 \vdash b_i \uparrow T$.
   By Lemma B.1, $,_1 \vdash$ ok. Thus by Rule 38, $,_1 \vdash b_i \equiv b_i : T$.

   • Case: $,_1 \vdash \alpha \uparrow\ ,_1(\alpha) \leftrightarrow\ ,_2 \vdash \alpha \uparrow\ ,_2(\alpha)$.
   By Lemma B.1 and Rule 39, $,_1 \vdash \alpha \equiv \alpha :\ ,_1(\alpha)$.

   • Case: $,_1 \vdash p_1 A_1 \uparrow \{\alpha\mapsto A_1\}L_1'' \leftrightarrow\ ,_2 \vdash p_2 A_2 \uparrow \{\alpha\mapsto A_2\}L_2''$ because $,_1 \vdash p_1 \uparrow \Pi\alpha{:}L_1'.L_1'' \leftrightarrow\ ,_2 \vdash p_2 \uparrow \Pi\alpha{:}L_2'.L_2''$ and $,_1 \vdash A_1 : L_1' \Leftrightarrow\ ,_2 \vdash A_2 : L_2'$.

   (a) By Lemma B.17, $,_1 \vdash p_1 : \Pi\alpha{:}K_1'.K_1''$,

   (b) $,_1 \vdash A_1 : K_1'$,

   (c) $,_2 \vdash p_2 : \Pi\alpha{:}K_2'.K_2''$,

   (d) and $,_2 \vdash A_2 : K_2'$.

   (e) By the inductive hypothesis, $,_1 \vdash \Pi\alpha{:}L_1'.L_1'' \equiv \Pi\alpha{:}L_2'.L_2''$.

   (f) and $,_1 \vdash p_1 \equiv p_2 : \Pi\alpha{:}L_1'.L_1''$.

   (g) By Lemma 3.1, $,_1 \vdash S(p_1 : \Pi\alpha{:}L_1'.L_1'') \leq \Pi\alpha{:}K_1'.K_1''$

   (h) and $,_2 \vdash S(p_2 : \Pi\alpha{:}L_2'.L_2'') \leq \Pi\alpha{:}K_2'.K_2''$.

   (i) Thus $,_1 \vdash K_1' \leq L_1'$

   (j) and $,_2 \vdash K_2' \leq L_2'$.

   (k) By subsumption then, $,_1 \vdash A_1 : L_1'$

   (l) and $,_2 \vdash A_2 : L_2'$.

   (m) The induction hypothesis applies, and so $,_1 \vdash A_1 \equiv A_2 : L_1'$.

   (n) Thus $,_1 \vdash p_1 A_1 \equiv p_2 A_2 : \{\alpha\mapsto A_1\}L_1''$

   (o) and by Lemma B.11, $,_1 \vdash \{\alpha\mapsto A_1\}L_1'' \equiv \{\alpha\mapsto A_2\}L_2''$.

   • Case: $,_1 \vdash \pi_1 p_1 \uparrow K_1 \leftrightarrow\ ,_2 \vdash \pi_1 p_2 \uparrow K_2$ because $,_1 \vdash p_1 \uparrow \Sigma\alpha{:}K_1.L_1 \leftrightarrow\ ,_2 \vdash p_2 \uparrow \Sigma\alpha{:}K_2.L_2$

   (a) By Lemma B.17 the inductive hypothesis applies,

   (b) so $,_1 \vdash \Sigma\alpha{:}K_1.L_1 \equiv \Sigma\alpha{:}K_2.L_2$

   (c) and $,_1 \vdash p_1 \equiv p_2 : \Sigma\alpha{:}K_1.L_1$.

   (d) Thus $,_1 \vdash \pi_1 p_1 \equiv \pi_1 p_2 : K_1$

   (e) and by inversion, $,_1 \vdash K_1 \equiv K_2$.

   • Case: $,_1 \vdash \pi_2 p_1 \uparrow \{\alpha\mapsto\pi_1 p_1\}L_1 \leftrightarrow\ ,_2 \vdash \pi_2 p_2 \uparrow \{\alpha\mapsto\pi_1 p_2\}L_2$ because $,_1 \vdash p_1 \uparrow \Sigma\alpha{:}K_1.L_1 \leftrightarrow\ ,_2 \vdash p_2 \uparrow \Sigma\alpha{:}K_2.L_2$.

   (a) By Lemma B.17 the inductive hypothesis applies,

   (b) so $,_1 \vdash \Sigma\alpha{:}K_1.L_1 \equiv \Sigma\alpha{:}K_2.L_2$

   (c) and $,_1 \vdash p_1 \equiv p_2 : \Sigma\alpha{:}K_1.L_1$.

   (d) Thus $,_1 \vdash \pi_2 p_1 \equiv \pi_2 p_2 : \{\alpha\mapsto\pi_1 p_1\}L_1$.

   (e) $,_1 \vdash \pi_1 p_1 \equiv \pi_1 p_2 : K_1$

   (f) So by Lemma B.11, $,_1 \vdash \{\alpha\mapsto\pi_1 p_1\}L_1 \equiv \{\alpha\mapsto\pi_1 p_2\}L_2$

∎

A key aspect of this algorithm is that it can easily be shown to obey symmetry and transitivity properties necessary for the decidability proof. It is for this purpose that the algorithm maintains two contexts and two classifiers. (Section 5 shows that this redundancy can be eliminated in an actual implementation.)

Lemma 3.8 (Algorithmic PER Properties)

1. *If* $\Delta_1 \vdash A_1 : K_1 \Leftrightarrow \Delta_2 \vdash A_2 : K_2$ *then* $\Delta_2 \vdash A_2 : K_2 \Leftrightarrow \Delta_1 \vdash A_1 : K_1$.

2. *If* $\Delta_1 \vdash A_1 : K_1 \Leftrightarrow \Delta_2 \vdash A_2 : K_2$ *and* $\Delta_2 \vdash A_2 : K_2 \Leftrightarrow \Delta_3 \vdash A_3 : K_3$ *then* $\Delta_1 \vdash A_1 : K_1 \Leftrightarrow \Delta_3 \vdash A_3 : K_3$.

3. *If* $\Delta_1 \vdash A_1 \uparrow K_1 \leftrightarrow \Delta_2 \vdash A_2 \uparrow K_2$ *then* $\Delta_2 \vdash A_2 \uparrow K_2 \leftrightarrow \Delta_1 \vdash A_1 \uparrow K_1$.

4. *If* $\Delta_1 \vdash A_1 \uparrow K_1 \leftrightarrow \Delta_2 \vdash A_2 \uparrow K_2$ *and* $\Delta_2 \vdash A_2 \uparrow K_2 \leftrightarrow \Delta_3 \vdash A_3 \uparrow K_3$ *then* $\Delta_1 \vdash A_1 \uparrow K_1 \leftrightarrow \Delta_3 \vdash A_3 \uparrow K_3$.

5. *If* $\Delta_1 \vdash K_1 \Leftrightarrow \Delta_2 \vdash K_2$ *then* $\Delta_2 \vdash K_2 \Leftrightarrow \Delta_1 \vdash K_1$.

6. *If* $\Delta_1 \vdash K_1 \Leftrightarrow \Delta_2 \vdash K_2$ *and* $\Delta_2 \vdash K_2 \Leftrightarrow \Delta_3 \vdash K_3$ *then* $\Delta_1 \vdash K_1 \Leftrightarrow \Delta_3 \vdash K_3$.

Proof Sketch: By induction on execution of the algorithm.

4 Completeness and Termination

To show the completeness and termination for the algorithm we define a collection of Kripke-style logical relations, shown in Figures 5, 6, and 7. The strategy for proving completeness of the algorithm is to define the logical relations, show that logically-related constructors are related by the algorithm, and finally show that provably-equivalent constructors are logically related. Using completeness we can then show the algorithm terminates for all well-formed inputs.

We use the notation Δ to denote a Kripke world. Worlds are restricted to contexts containing no duplicate bound variables; the partial order $\preceq$ on worlds is simply the prefix ordering.

The logical kind validity relation $(\Delta; K)$ **valid** is indexed by the world Δ and is well-defined by induction on the size of kinds. Similarly, the logical constructor validity relation $(\Delta; A; K)$ **valid** is indexed by a Δ and defined by induction on the size of K, which must itself be logically valid.

In addition to validity relations, we have logically-defined binary equivalence relations between (logically valid) types and terms. The unusual part of these relations is that rather than being a binary relation indexed by a world, they are relations between two kinds or constructors which have been determined to be logically valid under *potentially different* worlds. Thus the form of the equivalence of kinds is $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$ and the form of the equivalence on constructors is $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$. With this modification, the logical relations are otherwise defined in a reasonably familiar manner. At the base and singleton kinds we impose the algorithmic equivalence as the definition of the logical relation. At higher kinds we use a Kripke-style logical relations interpretation of Π and Σ.

With these definitions in hand we construct some derived relations. The relation $(\Delta_1; K_1 \leq L_1)$ **is** $(\Delta_2; K_2 \leq L_2)$ is defined to satisfy the following "subsumption-like" behavior:

$$\frac{\begin{array}{c}(\Delta_1; A_1; K_1) \ \textbf{is} \ (\Delta_2; A_2; K_2) \\ (\Delta_1; K_1 \leq L_1) \ \textbf{is} \ (\Delta_2; K_2 \leq L_2)\end{array}}{(\Delta_1; A_1; L_1) \ \textbf{is} \ (\Delta_2; A_2; L_2)}$$

Finally, we have validity and equivalence relations on environments (substitutions mapping variables to constructors) which are defined by pointwise validity and pointwise equivalence.

We first give some basic properties of the algorithm and logical relations.

Lemma 4.1 (Weakening)

1. *If* $, , , '' \vdash A \rightsquigarrow B$ *and* $\mathrm{dom}(, ') \cap \mathrm{dom}(, , , '') = \emptyset$ *then* $, , , ', , '' \vdash A \rightsquigarrow B$

2. *If* $, , , '' \vdash A \Downarrow p$ *and* $\mathrm{dom}(, ') \cap \mathrm{dom}(, , , '') = \emptyset$ *then* $, , , ', , '' \vdash A \Downarrow p$.

3. *If* $, , , '' \vdash A \uparrow K$ *and* $\mathrm{dom}(, ') \cap \mathrm{dom}(, , , '') = \emptyset$ *then* $, , , ', , '' \vdash A \uparrow K$.

4. *If* $, _1, , ''_1 \vdash A_1 : K_1 \Leftrightarrow , _2, , ''_2 \vdash A_2 : K_2$, $\mathrm{dom}(, '_1) \cap \mathrm{dom}(, _1, , ''_1) = \emptyset$, *and* $\mathrm{dom}(, '_2) \cap \mathrm{dom}(, _2, , ''_2) = \emptyset$ *then* $, _1, , '_1, , ''_1 \vdash A_1 : K_1 \Leftrightarrow , _2, , '_2, , ''_2 \vdash A_2 : K_2$.

5. *If* $, _1, , ''_1 \vdash A_1 \uparrow K_1 \leftrightarrow , _2, , ''_2 \vdash A_2 \uparrow K_2$, $\mathrm{dom}(, '_1) \cap \mathrm{dom}(, _1, , ''_1) = \emptyset$, *and* $\mathrm{dom}(, '_2) \cap \mathrm{dom}(, _2, , ''_2) = \emptyset$ *then* $, _1, , '_1, , ''_1 \vdash A_1 \uparrow K_1 \leftrightarrow , _2, , '_2, , ''_2 \vdash A_2 \uparrow K_2$.

- $(\Delta_1; K_1)$ **valid** iff
  1.
     - $K_1 = T$
     - Or, $K_1 = S(A_1)$ and $(\Delta_1; A_1; T)$ **valid**
     - Or, $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $(\Delta_1; K_1')$ **valid** and $\forall \Delta_1' \succeq \Delta_1, \Delta_1'' \succeq \Delta_1$ if $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_1''; A_2; K_1')$ then $(\Delta_1'; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_1''; \{\alpha{\mapsto}A_2\}K_1'')$
     - Or, $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $(\Delta_1; K_1')$ **valid** and $\forall \Delta_1' \succeq \Delta_1, \Delta_1'' \succeq \Delta_1$ if $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_1''; A_2; K_1')$ then $(\Delta_1'; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_1''; \{\alpha{\mapsto}A_2\}K_1'')$
- $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$ iff
  1. $(\Delta_1; K_1)$ **valid** and $(\Delta_2; K_2)$ **valid**.
  2. And,
     - $K_1 = T$ and $K_2 = T$
     - Or, $K_1 = S(A_1)$ and $K_2 = S(A_2)$ and $(\Delta_1; A_1; T)$ **is** $(\Delta_2; A_2; T)$
     - Or, $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$ and $(\Delta_1; K_1')$ **is** $(\Delta_2; K_2')$ and $\forall \Delta_1' \succeq \Delta_1, \Delta_2' \succeq \Delta_2$ if $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_2'; A_2; K_2')$ then $(\Delta_1'; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_2'; \{\alpha{\mapsto}A_2\}K_2'')$
     - Or, $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$ and $(\Delta_1; K_1')$ **is** $(\Delta_2; K_2')$ and $\forall \Delta_1' \succeq \Delta_1, \Delta_2' \succeq \Delta_2$ if $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_2'; A_2; K_2')$ then $(\Delta_1'; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_2'; \{\alpha{\mapsto}A_2\}K_2'')$
- $(\Delta_1; K_1 \leq L_1)$ **is** $(\Delta_2; K_2 \leq L_2)$ iff
  1. $\forall \Delta_1' \succeq \Delta_1, \Delta_2' \succeq \Delta_2$ if $(\Delta_1'; A_1; K_1)$ **is** $(\Delta_2'; A_2; K_2)$ then $(\Delta_1'; A_1; L_1)$ **is** $(\Delta_2'; A_2; L_2)$.

Figure 5: Logical Relations on Kinds

- $(\Delta; A; K_1)$ **valid** iff
  1. $(\Delta; K_1)$ **valid**
  2. And,
     - $K_1 = T$ and $\Delta \vdash A : T \Leftrightarrow \Delta \vdash A : T$.
     - Or, $K_1 = S(B)$ and $(\Delta; A; T)$ **is** $(\Delta; B; T)$.
     - Or, $K_1 = \Pi\alpha{:}K.L$, and $\forall \Delta' \succeq \Delta, \Delta'' \succeq \Delta$ if $(\Delta'; B'; K)$ **is** $(\Delta''; B''; K)$ then $(\Delta'; AB'; \{\alpha{\mapsto}B'\}L)$ **is** $(\Delta''; AB''; \{\alpha{\mapsto}B''\}L)$.
     - Or, $K_1 = \Sigma\alpha{:}K.L$, $(\Delta; \pi_1 A; K)$ **valid** and $(\Delta; \pi_2 A; \{\alpha{\mapsto}\pi_1 A\}L)$ **valid**
- $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$ iff
  1. $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$
  2. And, $(\Delta_1; A_1; K_1)$ **valid** and $(\Delta_2; A_2; K_2)$ **valid**
  3. And,
     - $K_1 = K_2 = T$ and $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
     - Or, $K_1 = S(B_1)$, $K_2 = S(B_2)$, and $(\Delta_1; A_1; T)$ **is** $(\Delta_2; A_2; T)$
     - Or, $K_1 = \Pi\alpha{:}K_1'.K_1''$, $K_2 = \Pi\alpha{:}K_2'.K_2''$, and $\forall \Delta_1' \succeq \Delta_1, \Delta_2' \succeq \Delta_2$ if $(\Delta_1'; B_1; K_1')$ **is** $(\Delta_2'; B_2; K_2')$ then $(\Delta_1'; A_1 B_1; \{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2'; A_2 B_2; \{\alpha{\mapsto}B_2\}K_2'')$.
     - Or, $K_1 = \Sigma\alpha{:}K_1'.K_1''$, $K_2 = \Sigma\alpha{:}K_2'.K_2''$, $(\Delta_1; \pi_1 A_1; K_1')$ **is** $(\Delta_2; \pi_1 A_2; K_2')$ and $(\Delta_1; \pi_2 A_1; \{\alpha{\mapsto}\pi_1 A_1\}K_1'')$ **is** $(\Delta_2; \pi_2 A_2; \{\alpha{\mapsto}\pi_1 A_2\}K_2'')$

Figure 6: Logical Relations on Constructors

- $(\Delta; \gamma; \,,\,)$ **valid** iff
  1. $\forall \alpha \in \mathrm{dom}(\,,\,).\ (\Delta; \gamma\alpha; \gamma(\,,\,(\alpha)))$ **valid**.
- $(\Delta_1; \gamma_1; \,,\,_1)$ **is** $(\Delta_2; \gamma_2; \,,\,_2)$ iff
  1. $(\Delta_1; \gamma_1; \,,\,_1)$ **valid** and $(\Delta_2; \gamma_2; \,,\,_2)$ **valid**
  2. And, $\forall \alpha \in \mathrm{dom}(\,,\,_1) = \mathrm{dom}(\,,\,_2)$. $(\Delta_1; \gamma_1\alpha; \gamma_1(\,,\,_1\alpha))$ **is** $(\Delta_2; \gamma_2\alpha; \gamma_2(\,,\,_2\alpha))$.

Figure 7: Logical Relations on Substitutions

6. *If* $\,,\,_1, \,,\,''_1 \vdash K_1 \Leftrightarrow \,,\,_2, \,,\,''_2 \vdash K_2$, $\mathrm{dom}(\,,\,'_1) \cap \mathrm{dom}(\,,\,_1, \,,\,''_1) = \emptyset$, *and* $\mathrm{dom}(\,,\,'_2) \cap \mathrm{dom}(\,,\,_2, \,,\,''_2) = \emptyset$ *then* $\,,\,_1, \,,\,'_1, \,,\,''_1 \vdash K_1 \Leftrightarrow \,,\,_2, \,,\,'_2, \,,\,''_2 \vdash K_2$.

Lemma 4.2 (Monotonicity)

1. *If* $(\Delta_1; K_1)$ **valid** *and* $\Delta'_1 \succeq \Delta_1$ *then* $(\Delta'_1; K_1)$ **valid**.
2. *If* $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$, $\Delta'_1 \succeq \Delta_1$, *and* $\Delta'_2 \succeq \Delta_2$ *then* $(\Delta'_1; K_1)$ **is** $(\Delta'_2; K_2)$.
3. *If* $(\Delta_1; K_1 \le L_1)$ **is** $(\Delta_2; K_2 \le L_2)$, $\Delta'_1 \succeq \Delta_1$, *and* $\Delta'_2 \succeq \Delta_2$ *then* $(\Delta'_1; K_1 \le L_1)$ **is** $(\Delta'_2; K_2 \le L_2)$.
4. *If* $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$, $\Delta'_1 \succeq \Delta_1$, *and* $\Delta'_2 \succeq \Delta_2$ *then* $(\Delta'_1; A_1; K_1)$ **is** $(\Delta'_2; A_2; K_2)$.
5. *If* $(\Delta_1; A_1; K_1)$ **valid** *and* $\Delta'_1 \succeq \Delta_1$ *then* $(\Delta'_1; A_1; K_1)$ **valid**.
6. *If* $(\Delta_1; \gamma_1; \,,\,_1)$ **is** $(\Delta_2; \gamma_2; \,,\,_2)$, $\Delta'_1 \succeq \Delta_1$, *and* $\Delta'_2 \succeq \Delta_2$ *then* $(\Delta'_1; \gamma_1; \,,\,_1)$ **is** $(\Delta'_2; \gamma_2; \,,\,_2)$

We next give a technical lemma which shows that logical equivalence of kinds is enough to get logical subkinding.

Lemma 4.3
If $(\Delta_1; L_1)$ **is** $(\Delta_2; L_2)$, $(\Delta_1; K_1)$ **is** $(\Delta_1; L_1)$, *and* $(\Delta_2; K_2)$ **is** $(\Delta_2; L_2)$ *then* $(\Delta_1; K_1 \le L_1)$ **is** $(\Delta_2; K_2 \le L_2)$.

Proof: Assume $(\Delta_1; L_1)$ **is** $(\Delta_2; L_2)$, $(\Delta_1; K_1)$ **is** $(\Delta_1; L_1)$, and $(\Delta_2; K_2)$ **is** $(\Delta_2; L_2)$.
Let $(\Delta'_1, \Delta'_2) \succeq (\Delta_1, \Delta_2)$ and assume $(\Delta'_1; A_1; K_1)$ **is** $(\Delta'_2; A_2; K_2)$. Then $(\Delta'_1; K_1)$ **is** $(\Delta'_2; K_2)$.

- Case $K_1 = K_2 = L_1 = L_2 = T$. $(\Delta'_1; A_1; T)$ **is** $(\Delta'_2; A_2; T)$ by assumption.
- Case $K_1 = S(B_1)$, $K_2 = S(B_2)$, $L_1 = S(C_1)$, and $L_2 = S(C_2)$.
  1. By monotonicity, $\Delta'_1 \vdash B_1 : T \Leftrightarrow \Delta'_1 \vdash C_1 : T$
  2. and $\Delta'_2 \vdash B_2 : T \Leftrightarrow \Delta'_2 \vdash C_2 : T$.
  3. Similarly, $\Delta'_1 \vdash A_1 : T \Leftrightarrow \Delta'_1 \vdash B_1 : T$,
  4. $\Delta'_2 \vdash A_2 : T \Leftrightarrow \Delta'_2 \vdash B_2 : T$, and
  5. and $\Delta'_1 \vdash A_1 : T \Leftrightarrow \Delta'_2 \vdash A_2 : T$.
  6. Thus by Lemma 3.8, $\Delta'_1 \vdash A_1 : T \Leftrightarrow \Delta'_1 \vdash C_1 : T$
  7. and $\Delta'_2 \vdash A_2 : T \Leftrightarrow \Delta'_2 \vdash C_2 : T$.
  8. Therefore $(\Delta'_1; A_1; S(C_1))$ **valid**,
  9. $(\Delta'_2; A_2; S(C_2))$ **valid**,
  10. and $(\Delta'_1; A_1; S(C_1))$ **is** $(\Delta'_2; A_2; S(C_2))$.
- Case: $K_1 = \Pi\alpha{:}K'_1.K''_1$, $K_2 = \Pi\alpha{:}K'_2.K''_2$, $L_1 = \Pi\alpha{:}L'_1.L''_1$, and $L_2 = \Pi\alpha{:}L'_2.L''_2$.
  1. Let $(\Delta''_1, \Delta''_2) \succeq (\Delta'_1, \Delta'_2)$ and assume $(\Delta''_1; B_1; L'_1)$ **is** $(\Delta''_2; B_2; L'_2)$.
  2. By monotonicity, $(\Delta''_1; K'_1)$ **is** $(\Delta''_2; K'_2)$,

3. $(\Delta_1''; L_1')$ **is** $(\Delta_2''; L_2')$,
4. $(\Delta_1''; K_1')$ **is** $(\Delta_1''; L_1')$, and
5. $(\Delta_2''; K_2')$ **is** $(\Delta_2''; L_2')$.
6. By the inductive hypothesis, $(\Delta_1''; L_1' \leq K_1')$ **is** $(\Delta_2''; L_2' \leq K_2')$, $(\Delta_1''; L_1' \leq K_1')$ **is** $(\Delta_1''; L_1' \leq L_1')$, and $(\Delta_2''; L_2' \leq K_2')$ **is** $(\Delta_2''; L_2' \leq L_2')$.
7. Thus $(\Delta_1''; B_1; K_1')$ **is** $(\Delta_2''; B_2; K_2')$.
8. Since $(\Delta_1''; B_1; L_1')$ **is** $(\Delta_1''; B_1; L_1')$ and $(\Delta_2''; B_2; L_2')$ **is** $(\Delta_2''; B_2; L_2')$,
9. we have $(\Delta_1''; B_1; K_1')$ **is** $(\Delta_1''; B_1; L_1')$,
10. and $(\Delta_2''; B_2; K_2')$ **is** $(\Delta_2''; B_2; L_2')$.
11. So, $(\Delta_1''; A_1 B_1; \{\alpha \mapsto B_1\} K_1'')$ **is** $(\Delta_2''; A_2 B_2; \{\alpha \mapsto B_2\} K_2'')$,
12. $(\Delta_1''; \{\alpha \mapsto B_1\} K_1'')$ **is** $(\Delta_1''; \{\alpha \mapsto B_1\} L_1'')$,
13. $(\Delta_1''; \{\alpha \mapsto B_1\} L_1'')$ **is** $(\Delta_2''; \{\alpha \mapsto B_2\} L_2'')$,
14. and $(\Delta_2''; \{\alpha \mapsto B_2\} K_2'')$ **is** $(\Delta_2''; \{\alpha \mapsto B_2\} L_2'')$.
15. By the inductive hypothesis, $(\Delta_1''; \{\alpha \mapsto B_1\} K_1'' \leq \{\alpha \mapsto B_1\} L_1'')$ **is** $(\Delta_2''; \{\alpha \mapsto B_2\} K_2'' \leq \{\alpha \mapsto B_2\} L_2'')$.
16. Thus $(\Delta_1''; A_1 B_1; \{\alpha \mapsto B_1\} L_1'')$ **is** $(\Delta_2''; A_2 B_2; \{\alpha \mapsto B_2\} L_2'')$.
17. Therefore $(\Delta_1'; A_1; \Pi\alpha{:}L_1'.L_1'')$ **is** $(\Delta_2'; A_2; \Pi\alpha{:}L_2'.L_2'')$.

- Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$, $K_2 = \Sigma\alpha{:}K_2'.K_2''$, $L_1 = \Sigma\alpha{:}L_1'.L_1''$, and $L_2 = \Sigma\alpha{:}L_2'.L_2''$.

1. $(\Delta_1'; \pi_1 A_1; K_1')$ **is** $(\Delta_2'; \pi_1 A_2; K_2')$.
2. Also, $(\Delta_1'; K_1')$ **is** $(\Delta_2'; K_2')$,
3. $(\Delta_1'; L_1')$ **is** $(\Delta_2'; L_2')$,
4. $(\Delta_1'; K_1')$ **is** $(\Delta_1'; L_1')$,
5. and $(\Delta_2'; K_2')$ **is** $(\Delta_2'; L_2')$.
6. By the inductive hypothesis, $(\Delta_1'; K_1' \leq L_1')$ **is** $(\Delta_2'; K_2' \leq L_2')$,
7. so $(\Delta_1'; \pi_1 A_1; L_1')$ **is** $(\Delta_2'; \pi_1 A_2; L_2')$.
8. By similar considerations, $(\Delta_1'; \{\alpha \mapsto \pi_1 A_1\} K_1'')$ **is** $(\Delta_1'; \{\alpha \mapsto \pi_1 A_1\} L_1'')$,
9. $(\Delta_2'; \{\alpha \mapsto \pi_2 A_2\} K_2'')$ **is** $(\Delta_2'; \{\alpha \mapsto \pi_1 A_2\} L_1'')$,
10. and $(\Delta_1'; \{\alpha \mapsto \pi_1 A_1\} L_1'')$ **is** $(\Delta_2'; \{\alpha \mapsto \pi_1 A_2\} L_2'')$.
11. By the inductive hypothesis,
$(\Delta_1'; \{\alpha \mapsto \pi_1 A_1\} K_1'' \leq \{\alpha \mapsto \pi_1 A_1\} L_1'')$ **is** $(\Delta_2'; \{\alpha \mapsto \pi_1 A_2\} K_2'' \leq \{\alpha \mapsto \pi_1 A_2\} L_2'')$.
12. Since $(\Delta_1'; \pi_2 A_1; \{\alpha \mapsto \pi_1 A_1\} K_1'')$ **is** $(\Delta_2'; \pi_2 A_2; \{\alpha \mapsto \pi_1 A_2\} K_2'')$,
13. we have $(\Delta_1'; \pi_2 A_1; \{\alpha \mapsto \pi_1 A_1\} L_1'')$ **is** $(\Delta_2'; \pi_2 A_2; \{\alpha \mapsto \pi_1 A_2\} L_2'')$.
14. Therefore $(\Delta_1'; A_1; \Sigma\alpha{:}L_1'.L_1'')$ **is** $(\Delta_2'; A_2; \Sigma\alpha{:}L_2'.L_2'')$.

∎

An easy corollary of this lemma may be visualized as the following rule:

$$\frac{\begin{array}{ccc} (\Delta_1; A_1; K_1) & \textbf{is} & (\Delta_2; A_2; K_2) \\ (\Delta_1; K_1) & \textbf{is} & (\Delta_2; K_2) \\ \textbf{is} & & \textbf{is} \\ (\Delta_1; L_1) & \textbf{is} & (\Delta_2; L_2) \end{array}}{(\Delta_1; A_1; L_1) \quad \textbf{is} \quad (\Delta_2; A_2; L_2)}$$

The logical relations obey reflexivity, symmetry, and transitivity properties. The logical relations were carefully defined so that the following property holds:

Lemma 4.4 (Reflexivity)
1. $(\Delta; K)$ **valid** *if and only if* $(\Delta; K)$ **is** $(\Delta; K)$.

2. $(\Delta; A; K)$ **valid** *if and only if* $(\Delta; A; K)$ **is** $(\Delta; A; K)$.

3. $(\Delta; \gamma; ,\ )$ **valid** *if and only if* $(\Delta; \gamma; ,\ )$ **is** $(\Delta; \gamma; ,\ )$.

Proof: The "if" direction is immediate from the definitions of the logical relations, so we only show the "only if" direction.

1. By induction on the size of K. Assume $(\Delta; K)$ **valid**.

   - Case: $K = T$. Follows by definition of $(\Delta; T)$ **is** $(\Delta; T)$.
   - Case: $K = S(B)$.
     (a) $(\Delta; B; T)$ **valid**.
     (b) $\Delta \vdash B : T \Leftrightarrow \Delta \vdash B : T$.
     (c) Then $(\Delta; B; T)$ **valid**
     (d) and $(\Delta; B; T)$ **is** $(\Delta; B; T)$.
     (e) Therefore $(\Delta; S(B))$ **is** $(\Delta; S(B))$.
   - Case: $K = \Pi\alpha{:}K'.K''$.
     (a) By $(\Delta; \Pi\alpha{:}K'.K'')$ **valid** we have $(\Delta; K')$ **valid**.
     (b) By the inductive hypothesis, $(\Delta; K')$ **is** $(\Delta; K')$.
     (c) Let $(\Delta', \Delta'') \succeq (\Delta, \Delta)$
     (d) and assume $(\Delta'; A_1; K')$ **is** $(\Delta''; A_2; K')$.
     (e) By $(\Delta; \Pi\alpha{:}K'.K'')$ **valid** we have $(\Delta'; \{\alpha \mapsto A_1\}K'')$ **is** $(\Delta''; \{\alpha \mapsto A_2\}K'')$.
     (f) Therefore $(\Delta; \Pi\alpha{:}K'.K'')$ **is** $(\Delta; \Pi\alpha{:}K'.K'')$.
   - Case: $K = \Sigma\alpha{:}K'.K''$.
     Same proof as for Π case.

2. By induction on the size of A. Assume $(\Delta; A; K)$ **valid**. Then $(\Delta; K)$ **valid** so that by Part 1, $(\Delta; K)$ **is** $(\Delta; K)$.

   - Case: $K = T$.
     (a) $(\Delta; A; T)$ **valid** implies $\Delta \vdash A : T \Leftrightarrow \Delta \vdash A : T$.
     (b) Therefore, $(\Delta; A; T)$ **is** $(\Delta; A; T)$.
   - Case: $K = S(B)$.
     (a) $(\Delta; A; S(B))$ **valid** implies $\Delta \vdash A : T \Leftrightarrow \Delta \vdash B : T$.
     (b) By Lemma 3.8, $\Delta \vdash A : T \Leftrightarrow \Delta \vdash A : T$,
     (c) so $(\Delta; A; T)$ **valid**
     (d) and $(\Delta; A; T)$ **is** $(\Delta; A; T)$.
     (e) Therefore $(\Delta; A; S(B))$ **is** $(\Delta; A; S(B))$.
   - Case: $K = \Pi\alpha{:}K'.K''$.
     (a) Let $(\Delta', \Delta'') \succeq (\Delta, \Delta)$
     (b) and assume $(\Delta'; B_1; K')$ **is** $(\Delta''; B_2; K')$.
     (c) Then $(\Delta'; AB_1; \{\alpha \mapsto B_1\}K'')$ **is** $(\Delta''; AB_2; \{\alpha \mapsto B_2\}K'')$.
     (d) Therefore $(\Delta; A; \Pi\alpha{:}K'.K'')$ **is** $(\Delta; A; \Pi\alpha{:}K'.K'')$.
   - Case: $K = \Sigma\alpha{:}K'.K''$.
     (a) Then $(\Delta; \pi_1 A; K')$ **valid**
     (b) and $(\Delta; \pi_2 A; \{\alpha \mapsto \pi_1 A\}K'')$ **valid**.
     (c) By the inductive hypothesis, $(\Delta; \pi_1 A; K')$ **is** $(\Delta; \pi_1 A; K')$
     (d) and $(\Delta; \pi_2 A; \{\alpha \mapsto \pi_1 A\}K'')$ **is** $(\Delta; \pi_2 A; \{\alpha \mapsto \pi_1 A\}K'')$.
     (e) Therefore $(\Delta; A; \Sigma\alpha{:}K'.K'')$ **is** $(\Delta; A; \Sigma\alpha{:}K'.K'')$.

3. (a) Assume $(\Delta; \gamma; ,\ )$ **valid**.
   (b) Let $x \in \mathrm{dom}(,\ )$ be given.

(c) Then $(\Delta;\gamma x;\gamma(,x))$ **valid**.

(d) By Lemma 4.4, $(\Delta;\gamma x;\gamma(,x))$ **is** $(\Delta;\gamma x;\gamma(,x))$.

(e) Therefore $(\Delta;\gamma;,)$ **is** $(\Delta;\gamma;,)$.

∎

Symmetry is straightforward and exactly analogous to the symmetry properties of the algorithmic relations.

Lemma 4.5 (Symmetry)

1. *If* $(\Delta_1;K_1)$ **is** $(\Delta_2;K_2)$ *then* $(\Delta_2;K_2)$ **is** $(\Delta_1;K_1)$
2. *If* $(\Delta_1;A_1;K_1)$ **is** $(\Delta_2;A_2;K_2)$ *then* $(\Delta_2;A_2;K_2)$ **is** $(\Delta_1;A_1;K_1)$.
3. *If* $(\Delta_1;\gamma_1;,_1)$ **is** $(\Delta_2;\gamma_2;,_2)$ *then* $(\Delta_2;\gamma_2;,_2)$ **is** $(\Delta_1;\gamma_1;,_1)$.

Proof:

1. Assume $(\Delta_1;K_1)$ **is** $(\Delta_2;K_2)$. Then $(\Delta_1;K_1)$ **valid** and $(\Delta_2;K_2)$ **valid**.
   - Case: $K_1 = K_2 = T$. Trivial.
   - Case: $K_1 = S(A_1)$, $K_2 = S(A_2)$.
     (a) $(\Delta_1;A_1;T)$ **is** $(\Delta_2;A_2;T)$.
     (b) Inductively by Part 2, $(\Delta_2;A_2;T)$ **is** $(\Delta_1;A_1;T)$.
     (c) Therefore $(\Delta_2;S(A_2))$ **is** $(\Delta_1;S(A_1))$.
   - Case: $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$.
     (a) $(\Delta_1;K_1')$ **is** $(\Delta_2;K_2')$ by $(\Delta_1;K_1)$ **is** $(\Delta_2;K_2)$.
     (b) By induction, $(\Delta_2;K_2')$ **is** $(\Delta_1;K_1')$.
     (c) Let $(\Delta_2',\Delta_1') \succeq (\Delta_2,\Delta_1)$ and assume $(\Delta_2';A_2;K_2')$ **is** $(\Delta_1';A_1;K_1')$.
     (d) Inductively by Part 2, $(\Delta_1';A_1;K_1')$ **is** $(\Delta_2';A_2;K_2')$.
     (e) By $(\Delta_1;K_1)$ **is** $(\Delta_2;K_2)$ again, $(\Delta_1';\{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_2';\{\alpha{\mapsto}A_2\}K_2'')$
     (f) By the inductive hypothesis again, $(\Delta_2';\{\alpha{\mapsto}A_2\}K_2'')$ **is** $(\Delta_1';\{\alpha{\mapsto}A_1\}K_1'')$.
     (g) Therefore, $(\Delta_2;\Pi\alpha{:}K_2'.K_2'')$ **is** $(\Delta_1;\Pi\alpha{:}K_1'.K_1'')$.
   - Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$. Same proof as for Π types.
2. Assume $(\Delta_1;A_1;K_1)$ **is** $(\Delta_2;A_2;K_2)$. Then $(\Delta_1;K_1)$ **is** $(\Delta_2;K_2)$, $(\Delta_1;A_1;K_1)$ **valid**, and $(\Delta_2;A_2;K_2)$ **valid**.
   By Part 1, $(\Delta_2;K_2)$ **is** $(\Delta_1;K_1)$.
   - Case $K_1 = K_2 = T$.
     (a) $\Delta_1 \vdash A_1 : K_1 \Leftrightarrow \Delta_2 \vdash A_2 : K_2$
     (b) By Lemma 3.8, $\Delta_2 \vdash A_2 : K_2 \Leftrightarrow \Delta_1 \vdash A_1 : K_1$.
     (c) Therefore $(\Delta_2;A_2;T)$ **is** $(\Delta_1;A_1;T)$.
   - Case $K_1 = S(B_1)$ and $K_2 = S(B_2)$.
     (a) $(\Delta_1;A_1;T)$ **is** $(\Delta_2;A_2;T)$.
     (b) By the inductive hypothesis, $(\Delta_2;A_2;T)$ **is** $(\Delta_1;A_1;T)$.
     (c) Therefore $(\Delta_2;A_2;S(B_1))$ **is** $(\Delta_1;A_1;S(B_2))$.
   - Case $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$.
     (a) Let $(\Delta_2',\Delta_1') \succeq (\Delta_2,\Delta_1)$
     (b) and assume $(\Delta_2';B_2;K_2')$ **is** $(\Delta_1';B_1;K_1')$.
     (c) By the inductive hypothesis, $(\Delta_1';B_1;K_1')$ **is** $(\Delta_2';B_2;K_2')$.
     (d) Thus $(\Delta_1';A_1B_1;\{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2';A_2B_2;\{\alpha{\mapsto}B_2\}K_2'')$.
     (e) By the inductive hypothesis, $(\Delta_2';A_2B_2;\{\alpha{\mapsto}B_2\}K_2'')$ **is** $(\Delta_1';A_1B_1;\{\alpha{\mapsto}B_1\}K_1'')$.
     (f) Therefore $(\Delta_2;A_2;\Pi\alpha{:}K_2'.K_2'')$ **is** $(\Delta_1;A_1;\Pi\alpha{:}K_1'.K_1'')$.

- Case $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$.
  (a) Then $(\Delta_1; \pi_1 A_1; K_1')$ **is** $(\Delta_2; \pi_1 A_2; K_2')$
  (b) and $(\Delta_1; \pi_2 A_1; \{\alpha{\mapsto}\pi_1 A_1\}K_1'')$ **is** $(\Delta_2; \pi_2 A_2; \{\alpha{\mapsto}\pi_1 A_2\}K_2'')$.
  (c) By the inductive hypothesis, $(\Delta_2; \pi_1 A_2; K_2')$ **is** $(\Delta_1; \pi_1 A_1; K_1')$
  (d) and $(\Delta_2; \pi_2 A_2; \{\alpha{\mapsto}\pi_1 A_2\}K_2'')$ **is** $(\Delta_1; \pi_2 A_1; \{\alpha{\mapsto}\pi_1 A_1\}K_1'')$.
  (e) Therefore $(\Delta_2; A_2; \Sigma\alpha{:}K_2'.K_2'')$ **is** $(\Delta_1; A_1; \Sigma\alpha{:}K_1'.K_1'')$.

3. (a) Assume $(\Delta_1; \gamma_1; ,_1)$ **is** $(\Delta_2; \gamma_2; ,_2)$. Then $(\Delta_1; \gamma_1; ,_1)$ **valid** and $(\Delta_2; \gamma_2; ,_2)$ **valid**.
   (b) Let $x \in \text{dom}(,_2)$ be given.
   (c) Then $x \in \text{dom}(,_1)$.
   (d) Then $(\Delta_1; \gamma_1\alpha; \gamma_1(,_1\alpha))$ **is** $(\Delta_2; \gamma_2\alpha; \gamma_2(,_2\alpha))$.
   (e) By Part 2, $(\Delta_2; \gamma_2\alpha; \gamma_2(,_2\alpha))$ **is** $(\Delta_1; \gamma_1\alpha; \gamma_1(,_1\alpha))$.
   (f) Therefore $(\Delta_2; \gamma_2; ,_2)$ **is** $(\Delta_1; \gamma_1; ,_1)$.

∎

In contrast, the logical relation cannot be easily shown to obey the same transitivity property as the algorithm; it does hold at the base kind but does not lift to function kinds. We therefore prove a slightly weaker property, which is nevertheless what we need for the remainder of the proof. The key difference is that the transitivity property for the algorithm involves three contexts/worlds whereas the following lemma only involves two.

Lemma 4.6 (Transitivity)
1. *If* $(\Delta_1; K_1)$ **is** $(\Delta_1; L_1)$ *and* $(\Delta_1; L_1)$ **is** $(\Delta_2; K_2)$ *then* $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$.
2. *If* $(\Delta_1; A_1; K_1)$ **is** $(\Delta_1; B_1; L_1)$ *and* $(\Delta_1; B_1; L_1)$ **is** $(\Delta_2; A_2; K_2)$ *then* $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$.

Proof:
1. Assume $(\Delta_1; K_1)$ **is** $(\Delta_1; L_1)$ and $(\Delta_1; L_1)$ **is** $(\Delta_2; K_2)$. First, $(\Delta_1; K_1)$ **valid** and $(\Delta_2; K_2)$ **valid**.
   - Case: $K_1 = L_1 = K_2 = T$.
     $(\Delta_1; T)$ **is** $(\Delta_2; T)$ always.
   - Case: $K_1 = S(A_1)$, $L_1 = S(B_1)$, and $K_2 = S(A_2)$.
     (a) Then $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_1 \vdash B_1 : T$
     (b) and $\Delta_1 \vdash B_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
     (c) By Lemma 3.8, $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
     (d) Therefore $(\Delta_1; S(A_1))$ **is** $(\Delta_2; S(A_2))$.
   - Case: $K_1 = \Pi\alpha{:}K_1'.K_1''$, $L_1 = \Pi\alpha{:}L_1'.L_1''$, and $K_2 = \Pi\alpha{:}K_2'.K_2''$.
     (a) $(\Delta_1; K_1')$ **is** $(\Delta_1; L_1')$ and $(\Delta_1; L_1')$ **is** $(\Delta_2; K_2')$.
     (b) By induction, $(\Delta_1; K_1')$ **is** $(\Delta_2; K_2')$.
     (c) Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$
     (d) and assume $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_2'; A_2; K_2')$.
     (e) By Lemma 4.4, $(\Delta_1; K_1')$ **is** $(\Delta_1; K_1')$.
     (f) By monotonicity and Lemma 4.3, $(\Delta_1'; K_1' \le K_1')$ **is** $(\Delta_1'; K_1' \le L_1')$.
     (g) Since $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_1'; A_1; K_1')$,
     (h) we have $(\Delta_1'; A_1; K_1')$ **is** $(\Delta_1'; A_1; L_1')$.
     (i) Thus $(\Delta_1'; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_1; \{\alpha{\mapsto}A_1\}L_1'')$.
     (j) Similarly, $(\Delta_1'; K_1' \le L_1')$ **is** $(\Delta_2'; K_2' \le K_2')$.
     (k) Then $(\Delta_1'; A_1; L_1')$ **is** $(\Delta_2'; A_2; K_2')$.
     (l) So, $(\Delta_1'; \{\alpha{\mapsto}A_1\}L_1'')$ **is** $(\Delta_2'; \{\alpha{\mapsto}A_2\}K_2'')$.
     (m) By induction, $(\Delta_1; \{\alpha{\mapsto}A_1\}K_1'')$ **is** $(\Delta_2; \{\alpha{\mapsto}A_2\}K_2'')$.
     (n) Therefore $(\Delta_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; \Pi\alpha{:}K_2'.K_2'')$.

- Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$, $L_1 = \Sigma\alpha{:}L_1'.L_1''$, and $K_2 = \Sigma\alpha{:}K_2'.K_2''$.
  Same proof as for Π types.

2. Assume $(\Delta_1; A_1; K_1)$ **is** $(\Delta_1; B_1; L_1)$ and $(\Delta_1; B_1; L_1)$ **is** $(\Delta_2; A_2; K_2)$. Then $(\Delta_1; A_1; K_1)$ **valid**, $(\Delta_2; A_2; K_2)$ **valid**, $(\Delta_1; K_1)$ **is** $(\Delta_1; L_1)$, and $(\Delta_1; L_1)$ **is** $(\Delta_2; K_2)$. By Part 1, $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$.

   - Case: $K_1 = L_1 = K_2 = T$.
     (a) $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_1 \vdash B_1 : T$
     (b) and $\Delta_1 \vdash B_1 : T \Leftrightarrow \Delta_2 \vdash A_1 : T$.
     (c) By Lemma 3.8, $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
     (d) Therefore $(\Delta_1; A_1; T)$ **is** $(\Delta_2; A_2; T)$.
   - Case: $K_1 = S(A_1')$, $L_1 = S(B_1')$, and $K_2 = S(A_2')$.
     (a) $(\Delta_1; A_1; T)$ **is** $(\Delta_1; B_1; T)$
     (b) and $(\Delta_1; B_1; T)$ **is** $(\Delta_2; A_2; T)$.
     (c) By the inductive hypothesis, $(\Delta_1; A_1; T)$ **is** $(\Delta_2; A_2; T)$.
     (d) Therefore $(\Delta_1; A_1; S(A_1'))$ **is** $(\Delta_2; A_2; S(A_2'))$.
   - Case: $K_1 = \Pi\alpha{:}K_1'.K_1''$, $L_1 = \Pi\alpha{:}L_1'.L_1''$, and $K_2 = \Pi\alpha{:}K_2'.K_2''$.
     (a) Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$
     (b) and assume $(\Delta_1'; A_1'; K_1')$ **is** $(\Delta_2'; A_2'; K_2')$.
     (c) Then by monotonicity $(\Delta_1'; K_1')$ **is** $(\Delta_1'; L_1')$ and $(\Delta_1'; L_1')$ **is** $(\Delta_2'; K_2')$.
     (d) By Lemma 4.3, $(\Delta_1'; K_1' \leq K_1')$ **is** $(\Delta_1'; K_1' \leq L_1')$.
     (e) By Part 2, $(\Delta_1'; A_1'; K_1')$ **is** $(\Delta_1'; A_1'; K_1')$,
     (f) so $(\Delta_1'; A_1'; K_1')$ **is** $(\Delta_1'; A_1'; L_1')$.
     (g) Thus $(\Delta_1'; A_1A_1'; \{\alpha\mapsto A_1'\}K_1'')$ **is** $(\Delta_1'; B_1A_1'; \{\alpha\mapsto A_1'\}L_1'')$.
     (h) Similarly, $(\Delta_1'; K_1' \leq L_1')$ **is** $(\Delta_2'; K_2' \leq K_2')$,
     (i) so $(\Delta_1'; A_1'; L_1')$ **is** $(\Delta_2'; A_2'; K_2')$.
     (j) Thus, $(\Delta_1'; B_1A_1'; \{\alpha\mapsto A_1'\}L_1'')$ **is** $(\Delta_2'; A_2A_2'; \{\alpha\mapsto A_2'\}K_2'')$.
     (k) By the inductive hypothesis, $(\Delta_1'; A_1A_1'; \{\alpha\mapsto A_1'\}K_1'')$ **is** $(\Delta_2'; A_2A_2'; \{\alpha\mapsto A_2'\}K_2'')$.
     (l) Therefore, $(\Delta_1; A_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2; \Pi\alpha{:}K_2'.K_2'')$.
   - Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$, $L_1 = \Sigma\alpha{:}L_1'.L_1''$, and $K_2 = \Sigma\alpha{:}K_2'.K_2''$.
     (a) $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_1; \pi_1B_1; L_1')$
     (b) and $(\Delta_1; \pi_1B_1; L_1')$ **is** $(\Delta_2; \pi_1A_2; K_2')$.
     (c) By the inductive hypothesis, $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_2; \pi_1A_2; K_2')$.
     (d) Similarly, $(\Delta_1; \pi_2A_1; \{\alpha\mapsto\pi_1A_1\}K_1'')$ **is** $(\Delta_1; \pi_2B_1; \{\alpha\mapsto\pi_1B_1\}L_1'')$
     (e) and $(\Delta_1; \pi_2B_1; \{\alpha\mapsto\pi_1B_1\}L_1'')$ **is** $(\Delta_2; \pi_2A_2; \{\alpha\mapsto\pi_1A_2\}K_2'')$.
     (f) By the inductive hypothesis, $(\Delta_1; \pi_2A_1; \{\alpha\mapsto\pi_1A_1\}K_1'')$ **is** $(\Delta_2; \pi_2A_2; \{\alpha\mapsto\pi_1A_2\}K_2'')$.
     (g) Therefore, $(\Delta_1; A_1; \Sigma\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2; \Sigma\alpha{:}K_2'.K_2'')$.

∎

Because of this restricted formulation, we cannot use symmetry and transitivity to derive properties such as "if $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$ then $(\Delta_1; K_1)$ **is** $(\Delta_1; K_1)$". An important purpose of the validity predicates is to make sure that this property does in fact hold (by building it into the definition of the equivalence logical relations).

Next we show that logical relations are closed under head expansion and reduction. Define $\Gamma \vdash A_1 \simeq A_2$ to mean that A_1 and A_2 have a common weak head reduct. The following lemma then follows by induction on the size of kinds.

Lemma 4.7 (Weak Head Closure)

1. *If* $\Gamma \vdash A \rightsquigarrow B$ *then* $\Gamma \vdash E[A] \rightsquigarrow E[B]$
2. *If* $\Gamma \vdash A_1 \simeq A_2$ *then* $\Gamma \vdash E[A_1] \simeq E[A_2]$.

3. *If* $(\Delta; A; K)$ **valid** $\Delta \vdash A' \simeq A$, *then* $(\Delta; A'; K)$ **valid**.

4. *If* $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$, $\Delta_1 \vdash A_1' \simeq A_1$, *and* $\Delta_2 \vdash A_2' \simeq A_2$ *then* $(\Delta_1; A_1'; K_1)$ **is** $(\Delta_2; A_2'; K_2)$.

Proof:

1. Obvious by definition of $\Gamma \vdash A \rightsquigarrow B$.
2. By repeated application of Part 1.
3. By induction on the size of K. Assume $(\Delta; A; K)$ **valid** and $\Delta \vdash A' \simeq A$. Note that $(\Delta; K)$ **valid**.

   - Case: $K = T$.
     (a) $\Delta \vdash A : T \Leftrightarrow \Delta \vdash A : T$.
     (b) By the definition of the algorithm and determinacy of weak head reduction, $\Delta \vdash A' : T \Leftrightarrow \Delta \vdash A' : T$.
     (c) Therefore $(\Delta; A'; T)$ **valid**.
   - Case: $K = S(B)$
     (a) Then $\Delta \vdash A : T \Leftrightarrow \Delta \vdash B : T$
     (b) so by the definition of the algorithm and determinacy of weak head reduction $\Delta \vdash A' : T \Leftrightarrow \Delta \vdash B : T$
     (c) which yields $(\Delta; A'; S(B))$ **valid**
   - Case: $K = \Pi\alpha{:}K'.K''$.
     (a) Let $(\Delta', \Delta'') \succeq (\Delta, \Delta)$
     (b) and assume that $(\Delta'; B_1; K')$ **is** $(\Delta''; B_2; K')$.
     (c) Then $(\Delta'; AB_1; \{\alpha{\mapsto}B_1\}K'')$ **is** $(\Delta''; AB_2; \{\alpha{\mapsto}B_2\}K'')$,
     (d) By Part 2 and an obvious context weakening property, $\Delta' \vdash AB_1 \simeq A'B_1$
     (e) and $\Delta'' \vdash AB_2 \simeq A'B_2$.
     (f) By the inductive hypothesis, $(\Delta'; A'B_1; \{\alpha{\mapsto}B_1\}K'')$ **is** $(\Delta''; A'B_2; \{\alpha{\mapsto}B_2\}K'')$.
     (g) Therefore, $(\Delta; A'; \Pi\alpha{:}K'.K'')$ **valid**.
   - Case: $K = \Sigma\alpha{:}K'.K''$.
     (a) Then $(\Delta; \pi_1 A; K')$ **valid**
     (b) and by Part 2, $\Delta \vdash \pi_1 A' \simeq \pi_1 A$.
     (c) By the inductive hypothesis, $(\Delta_1; \pi_1 A_1'; K_1')$ **valid**.
     (d) and inductively by Part 4, $(\Delta; \pi_1 A; K')$ **is** $(\Delta; \pi_1 A'; K')$.
     (e) Similarly, $(\Delta_1; \pi_2 A; \{\alpha{\mapsto}\pi_1 A\}K'')$ **valid**,
     (f) and $\Delta \vdash \pi_2 A' \simeq \pi_2 A$,
     (g) so by the inductive hypothesis again, $(\Delta; \pi_2 A'; \{\alpha{\mapsto}\pi_1 A\}K'')$ **valid**.
     (h) But $(\Delta; \{\alpha{\mapsto}\pi_1 A\}K'')$ **is** $(\Delta; \{\alpha{\mapsto}\pi_1 A'\}K'')$,
     (i) so by Reflexivity and Lemma 4.3, $(\Delta; \{\alpha{\mapsto}\pi_1 A\}K'' \leq \{\alpha{\mapsto}\pi_1 A'\}K'')$ **is** $(\Delta; \{\alpha{\mapsto}\pi_1 A\}K'' \leq \{\alpha{\mapsto}\pi_1 A'\}K'')$.
     (j) so by Reflexivity $(\Delta; \pi_2 A'; \{\alpha{\mapsto}\pi_1 A'\}K'')$ **valid**.
     (k) Therefore, $(\Delta; A'; \Sigma\alpha{:}K'.K'')$ **valid**.

4. By induction on the size of K_1.
   Assume $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$, $\Delta_1 \vdash A_1' \simeq A_1$, and $\Delta_2 \vdash A_2' \simeq A_2$. First, note that $(\Delta_1; A_1; K_1)$ **valid**, $(\Delta_2; A_2; K_2)$ **valid**, and $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$. By the argument in Part 3, $(\Delta_1; A_1'; K_1)$ **valid** and $(\Delta_2; A_2'; K_2)$ **valid**.

   - Case: $K_1 = K_2 = T$.
     (a) $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
     (b) By the definition of the algorithm, $\Delta_1 \vdash A_1' : T \Leftrightarrow \Delta_2 \vdash A_2' : T$.
     (c) Therefore $(\Delta_1; A_1'; T)$ **is** $(\Delta_2; A_2'; T)$.
   - Case: $K_1 = S(B_1)$ and $K_2 = S(B_2)$.

(a) Then $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$
(b) so $\Delta_1 \vdash A_1' : T \Leftrightarrow \Delta_2 \vdash A_2' : T$
(c) which yields $(\Delta_1; A_1'; S(B_1))$ **is** $(\Delta_2; A_2'; S(B_2))$.

- Case: $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$.

(a) Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$
(b) and assume that $(\Delta_1'; B_1; K_1')$ **is** $(\Delta_2'; B_2; K_2')$.
(c) Then $(\Delta_1'; A_1B_1; \{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2'; A_2B_2; \{\alpha{\mapsto}B_2\}K_2'')$,
(d) By Part 2 and an obvious weakening property, $\Delta_1' \vdash A_1B_1 \simeq A_1'B_1$
(e) and $\Delta_2' \vdash A_2B_2 \simeq A_2'B_2$.
(f) By the inductive hypothesis $(\Delta_1'; A_1'B_1; \{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2'; A_2'B_2; \{\alpha{\mapsto}B_2\}K_2'')$.
(g) Therefore, $(\Delta_1; A_1'; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2'; \Pi\alpha{:}K_2'.K_2'')$.

- Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$.

(a) Then $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_2; \pi_1A_2; K_2')$,
(b) $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_1; \pi_1A_1; K_1')$,
(c) $(\Delta_2; \pi_1A_2; K_2')$ **is** $(\Delta_2; \pi_1A_2; K_2')$,
(d) and by Part 2, $\Delta_1 \vdash \pi_1A_1' \simeq \pi_1A_1$,
(e) and $\Delta_2 \vdash \pi_1A_2' \simeq \pi_1A_2$.
(f) By the inductive hypothesis, $(\Delta_1; \pi_1A_1'; K_1')$ **is** $(\Delta_2; \pi_1A_2'; K_2')$,
(g) $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_1; \pi_1A_1'; K_1')$,
(h) and $(\Delta_2; \pi_1A_2; K_2')$ **is** $(\Delta_2; \pi_1A_2'; K_2')$.
(i) Similarly, $(\Delta_1; \pi_2A_1; \{\alpha{\mapsto}\pi_1A_1\}K_1'')$ **is** $(\Delta_2; \pi_2A_2; \{\alpha{\mapsto}\pi_1A_2\}K_2'')$,
(j) $\Delta_1 \vdash \pi_2A_1' \simeq \pi_2A_1$,
(k) and $\Delta_2 \vdash \pi_2A_2' \simeq \pi_2A_2$.
(l) By the inductive hypothesis again, $(\Delta_1; \pi_2A_1'; \{\alpha{\mapsto}\pi_1A_1\}K_1'')$ **is** $(\Delta_2; \pi_2A_2'; \{\alpha{\mapsto}\pi_1A_2\}K_2'')$.
(m) But $(\Delta_1; K_1)$ **is** $(\Delta_1; K_1)$ and $(\Delta_2; K_2)$ **is** $(\Delta_2; K_2)$,
(n) so $(\Delta_1; \{\alpha{\mapsto}\pi_1A_1\}K_1'')$ **is** $(\Delta_1; \{\alpha{\mapsto}\pi_1A_1'\}K_1'')$,
(o) $(\Delta_2; \{\alpha{\mapsto}\pi_1A_2\}K_2'')$ **is** $(\Delta_2; \{\alpha{\mapsto}\pi_1A_2'\}K_2'')$,
(p) and $(\Delta_1; \{\alpha{\mapsto}\pi_1A_1'\}K_1'')$ **is** $(\Delta_2; \{\alpha{\mapsto}\pi_1A_2'\}K_2'')$.
(q) By Lemma 4.3, $(\Delta_1; \{\alpha{\mapsto}\pi_1A_1\}K_1'' \leq \{\alpha{\mapsto}\pi_1A_1'\}K_1'')$ **is** $(\Delta_2; \{\alpha{\mapsto}\pi_1A_1\}K_2'' \leq \{\alpha{\mapsto}\pi_1A_1'\}K_2'')$.
(r) so $(\Delta_1; \pi_2A_1'; \{\alpha{\mapsto}\pi_1A_1'\}K_1'')$ **is** $(\Delta_2; \pi_2A_2'; \{\alpha{\mapsto}\pi_1A_2'\}K_2'')$.
(s) Therefore, $(\Delta_1; A_1'; \Sigma\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2'; \Sigma\alpha{:}K_2'.K_2'')$.

∎

Following all this preliminary work, we can now show by induction on the size of kinds that equivalence under the logical relations implies equivalence under the algorithm. This requires a stronger induction hypothesis: that under suitable conditions variables (and more generally paths) are logically valid or logically related.

Lemma 4.8 (Main Lemma)

1. *If* $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$ *then* $\Delta_1 \vdash K_1 \Leftrightarrow \Delta_2 \vdash K_2$.

2. *If* $(\Delta_1; A_1; K_1)$ **is** $(\Delta_2; A_2; K_2)$ *then* $\Delta_1 \vdash A_1 : K_1 \Leftrightarrow \Delta_2 \vdash A_2 : K_2$.

3. *If* $(\Delta; K)$ **valid**, $\Delta \vdash p \uparrow K \leftrightarrow \Delta \vdash p \uparrow K$, *then* $(\Delta; p; K)$ **valid**.

4. *If* $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$ *and* $\Delta_1 \vdash p_1 \uparrow K_1 \leftrightarrow \Delta_2 \vdash p_2 \uparrow K_2$ *then* $(\Delta_1; p_1; K_1)$ **is** $(\Delta_2; p_2; K_2)$.

Proof: By induction on the size of the kinds involved.
For Part 4, note that in all cases $\Delta_1 \vdash p_1 \uparrow K_1 \leftrightarrow \Delta_1 \vdash p_1 \uparrow K_1$ and $\Delta_2 \vdash p_2 \uparrow K_2 \leftrightarrow \Delta_2 \vdash p_2 \uparrow K_2$ by symmetry and transitivity of the algorithm, $(\Delta_1; K_1)$ **valid**, and $(\Delta_2; K_2)$ **valid**. Hence by Part 3, $(\Delta_1; p_1; K_1)$ **valid** and $(\Delta_2; p_2; K_2)$ **valid**.

- Case: $K = K_1 = K_2 = T$.
  1. $\Delta_1 \vdash T \Leftrightarrow \Delta_2 \vdash T$ by the definition of the algorithm.
  2. (a) Assume $(\Delta_1; A_1; T)$ **is** $(\Delta_2; A_2; T)$.
     (b) By the definition of this relation, $\Delta_1 \vdash A_1 : T \Leftrightarrow \Delta_2 \vdash A_2 : T$.
  3. (a) Assume $(\Delta; T)$ **valid** and
     (b) $\Delta \vdash p \uparrow T \leftrightarrow \Delta \vdash p \uparrow T$.
     (c) By Lemma 3.3, $\Delta \vdash p \uparrow T$.
     (d) Then $\Delta \vdash p \Downarrow p$.
     (e) so $\Delta \vdash p : T \Leftrightarrow \Delta \vdash p : T$.
     (f) Therefore $(\Delta; p; T)$ **valid**.
  4. (a) Assume $\Delta_1 \vdash p_1 \uparrow T \leftrightarrow \Delta_2 \vdash p_2 \uparrow T$
     (b) and $(\Delta_1; T)$ **is** $(\Delta_2; T)$.
     (c) By Lemma 3.3, $\Delta_1 \vdash p_1 \uparrow T$ and $\Delta_2 \vdash p_2 \uparrow T$.
     (d) Thus $\Delta_1 \vdash p_1 \Downarrow p_1$ and $\Delta_2 \vdash p_2 \Downarrow p_2$.
     (e) so $\Delta_1 \vdash p_1 : T \Leftrightarrow \Delta_2 \vdash p_2 : T$.
     (f) Therefore $(\Delta_1; p_1; T)$ **is** $(\Delta_2; p_2; T)$.
- Case: $K = S(B)$, $K_1 = S(B_1)$, and $K_2 = S(B_2)$.
  1. (a) Assume $(\Delta_1; K_1)$ **is** $(\Delta_2; K_2)$.
     (b) Then by definition $(\Delta_1; B_1; T)$ **is** $(\Delta_2; B_2; T)$,
     (c) so $\Delta_1 \vdash B_1 : T \Leftrightarrow \Delta_2 \vdash B_2 : T$.
     (d) Therefore, $\Delta_1 \vdash S(B_1) \Leftrightarrow \Delta_2 \vdash S(B_2)$.
  2. (a) By definition, $\Delta_1 \vdash A_1 : S(B_1) \Leftrightarrow \Delta_2 \vdash A_2 : S(B_2)$ always.
  3. (a) Assume $(\Delta; S(B))$ **valid**,
     (b) and $\Delta \vdash p \uparrow S(B) \leftrightarrow \Delta \vdash p \uparrow S(B)$.
     (c) By Lemma 3.3, $\Delta \vdash p \uparrow S(B)$.
     (d) Then $\Delta \vdash p \rightsquigarrow B$ so $\Delta \vdash p \simeq B$.
     (e) By $(\Delta; S(B))$ **valid**, $\Delta \vdash B : T \Leftrightarrow \Delta \vdash B : T$.
     (f) By the definition of the algorithm, $\Delta \vdash p : T \Leftrightarrow \Delta \vdash B : T$.
     (g) Therefore $(\Delta; p; S(B))$ **valid**.
  4. (a) Assume $(\Delta_1; S(B_1))$ **is** $(\Delta_2; S(B_2))$,
     (b) and $\Delta_1 \vdash p_1 \uparrow S(B_1) \leftrightarrow \Delta_2 \vdash p_2 \uparrow S(B_1)$.
     (c) By definition of the logical relations, $\Delta_1 \vdash B_1 : T \Leftrightarrow \Delta_2 \vdash B_2 : T$.
     (d) By Lemma 3.3, $\Delta_1 \vdash p_1 \uparrow S(B_1)$ and $\Delta_2 \vdash p_2 \uparrow S(B_2)$.
     (e) That is, $\Delta_1 \vdash p_1 \rightsquigarrow B_1$ and $\Delta_2 \vdash p_2 \rightsquigarrow B_1$.
     (f) Hence $\Delta_1 \vdash p_1 : T \Leftrightarrow \Delta_2 \vdash p_2 : T$.
     (g) Therefore $(\Delta_1; p_1; S(B_1))$ **is** $(\Delta_2; p_2; S(B_1))$.
- Case: $K = \Pi\alpha{:}K'.K''$, $K_1 = \Pi\alpha{:}K_1'.K_1''$, and $K_2 = \Pi\alpha{:}K_2'.K_2''$.
  1. (a) Assume $(\Delta_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; \Pi\alpha{:}K_2'.K_2'')$.
     (b) Then $(\Delta_1; K_1')$ **is** $(\Delta_2; K_2')$.
     (c) By the inductive hypothesis we have $\Delta_1 \vdash K_1' \Leftrightarrow \Delta_2 \vdash K_2'$.
     (d) Now $\Delta_1, \alpha{:}K_1' \vdash \alpha \uparrow K_1' \leftrightarrow \Delta_2, \alpha{:}K_2' \vdash \alpha \uparrow K_2'$.
     (e) Inductively by Part 4, $(\Delta_1, \alpha{:}K_1'; \alpha; K_1')$ **is** $(\Delta_2, \alpha{:}K_2'; \alpha; K_2')$.
     (f) Thus $(\Delta_1, \alpha{:}K_1'; K_1'')$ **is** $(\Delta_2, \alpha{:}K_2'; K_2'')$
     (g) By the inductive hypothesis, $\Delta_1, \alpha{:}K_1' \vdash K_1'' \Leftrightarrow \Delta_2, \alpha{:}K_2' \vdash K_2''$.
     (h) Therefore $\Delta_1 \vdash \Pi\alpha{:}K_1'.K_1'' \Leftrightarrow \Delta_2 \vdash \Pi\alpha{:}K_2'.K_2''$.
  2. (a) Assume $(\Delta_1; A_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2; \Pi\alpha{:}K_2'.K_2'')$.
     (b) Then $(\Delta_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; \Pi\alpha{:}K_2'.K_2'')$

   (c) so as above, inductively by Part 4 we have $(\Delta_1, \alpha{:}K_1'; \alpha; K_1')$ **is** $(\Delta_2, \alpha{:}K_2'; \alpha; K_2')$.
   (d) Then $(\Delta_1, \alpha{:}K_1'; A_1\alpha; K_1'')$ **is** $(\Delta_2, \alpha{:}K_2'; A_2\alpha; K_2'')$.
   (e) By the inductive hypothesis again, $\Delta_1, \alpha{:}K_1' \vdash A_1\alpha : K_1'' \Leftrightarrow \Delta_2, \alpha{:}K_2' \vdash A_2\alpha : K_2''$.
   (f) Therefore $\Delta_1 \vdash A_1 : \Pi\alpha{:}K_1'.K_1'' \Leftrightarrow \Delta_2 \vdash A_2 : \Pi\alpha{:}K_1'.K_1''$.

3. (a) Assume $(\Delta; K)$ **valid**
   (b) and $\Delta \vdash p \uparrow K \leftrightarrow \Delta \vdash p \uparrow K$.
   (c) Let $(\Delta', \Delta'') \succeq (\Delta, \Delta)$
   (d) and assume $(\Delta'; B'; K')$ **is** $(\Delta''; B''; K')$.
   (e) Inductively by Part 2, $\Delta' \vdash B' : K' \Leftrightarrow \Delta'' \vdash B'' : K'$.
   (f) Thus using Weakening, $\Delta' \vdash pB' \uparrow \{\alpha{\mapsto}B'\}K'' \leftrightarrow \Delta'' \vdash pB'' \uparrow \{\alpha{\mapsto}B''\}K''$.
   (g) By $(\Delta; K)$ **valid**, $(\Delta'; \{\alpha{\mapsto}B'\}K'')$ **is** $(\Delta''; \{\alpha{\mapsto}B''\}K'')$.
   (h) Inductively by Part 4, $(\Delta'; pB'; \{\alpha{\mapsto}B'\}K'')$ **is** $(\Delta''; pB''; \{\alpha{\mapsto}B''\}K'')$.
   (i) Therefore $(\Delta; p; \Pi\alpha{:}K'.K'')$ **valid**.

4. (a) Assume $(\Delta_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; \Pi\alpha{:}K_2'.K_2'')$,
   (b) and $\Delta_1 \vdash p_1 \uparrow \Pi\alpha{:}K_1'.K_1'' \leftrightarrow \Delta_2 \vdash p_2 \uparrow \Pi\alpha{:}K_2'.K_2''$.
   (c) Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$ and assume that $(\Delta_1'; B_1; K_1')$ **is** $(\Delta_2'; B_2; K_2')$.
   (d) Then $(\Delta_1'; \{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2'; \{\alpha{\mapsto}B_2\}K_2'')$.
   (e) Inductively by Part 2, $\Delta_1' \vdash B_1 : K_1' \Leftrightarrow \Delta_2' \vdash B_2 : K_2'$,
   (f) and by Weakening, $\Delta_1' \vdash p_1 \uparrow \Pi\alpha{:}K_1'.K_1'' \leftrightarrow \Delta_2' \vdash p_2 \uparrow \Pi\alpha{:}K_2'.K_2''$,
   (g) so we have $\Delta_1' \vdash p_1B_1 \uparrow \{\alpha{\mapsto}B_1\}K_1'' \leftrightarrow \Delta_2' \vdash p_2B_2 \uparrow \{\alpha{\mapsto}B_2\}K_2''$.
   (h) By the inductive hypothesis, $(\Delta_1'; p_1B_1; \{\alpha{\mapsto}B_1\}K_1'')$ **is** $(\Delta_2'; p_2B_2; \{\alpha{\mapsto}B_2\}K_2'')$.
   (i) Therefore $(\Delta_1; p_1; \Pi\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; p_2; \Pi\alpha{:}K_2'.K_2'')$.

- Case: $K = \Sigma\alpha{:}K'.K''$, $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$.

1. The corresponding argument for the Π case also applies here.
2. (a) Assume $(\Delta_1; A_1; \Sigma\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; A_2; \Sigma\alpha{:}K_2'.K_2'')$.
   (b) Then $(\Delta_1; \pi_1A_1; K_1')$ **is** $(\Delta_2; \pi_1A_2; K_2')$.
   (c) and $(\Delta_1; \pi_2A_1; \{\alpha{\mapsto}\pi_1A_1\}K_1'')$ **is** $(\Delta_2; \pi_2A_2; \{\alpha{\mapsto}\pi_1A_2\}K_2'')$.
   (d) By the inductive hypothesis, $\Delta_1 \vdash \pi_1A_1 : K_1' \Leftrightarrow \Delta_2 \vdash \pi_1A_2 : K_2'$
   (e) and $\Delta_1 \vdash \pi_2A_1 : \{\alpha{\mapsto}\pi_1A_1\}K_1'' \Leftrightarrow \Delta_2 \vdash \pi_2A_2 : \{\alpha{\mapsto}\pi_1A_2\}K_2''$.
   (f) Therefore $\Delta_1 \vdash A_1 : \Sigma\alpha{:}K_1'.K_1'' \Leftrightarrow \Delta_2 \vdash A_2 : \Sigma\alpha{:}K_2'.K_2''$.
3. (a) Assume $(\Delta; K)$ **valid**,
   (b) and $\Delta \vdash p \uparrow K \leftrightarrow \Delta \vdash p \uparrow K$.
   (c) By definition of the algorithm, $\Delta \vdash \pi_1p \uparrow K' \leftrightarrow \Delta \vdash \pi_1p \uparrow K'$
   (d) and $\Delta \vdash \pi_2p \uparrow \{\alpha{\mapsto}\pi_1p\}K'' \leftrightarrow \Delta \vdash \pi_2p \uparrow \{\alpha{\mapsto}\pi_1p\}K''$.
   (e) By the induction hypothesis, $(\Delta; \pi_1p; K')$ **valid**.
   (f) By Lemma 4.4, $(\Delta; \pi_1p; K')$ **is** $(\Delta; \pi_1p; K')$.
   (g) By $(\Delta; K)$ **valid**, $(\Delta; \{\alpha{\mapsto}\pi_1p\}K'')$ **is** $(\Delta; \{\alpha{\mapsto}\pi_1p\}K'')$.
   (h) Thus $(\Delta; \{\alpha{\mapsto}\pi_1p\}K'')$ **valid**.
   (i) By the induction hypothesis again, $(\Delta; \pi_2p; \{\alpha{\mapsto}\pi_1p\}K'')$ **valid**.
   (j) Therefore, $(\Delta; p; \Sigma\alpha{:}K'.K'')$ **valid**.
4. (a) Assume $(\Delta_1; \Sigma\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; \Sigma\alpha{:}K_2'.K_2'')$,
   (b) and $\Delta_1 \vdash p_1 \uparrow \Sigma\alpha{:}K_1'.K_1'' \leftrightarrow \Delta_2 \vdash p_2 \uparrow \Sigma\alpha{:}K_2'.K_2''$.
   (c) Then $\Delta_1 \vdash \pi_1p_1 \uparrow K_1' \leftrightarrow \Delta_2 \vdash \pi_1p_2 \uparrow K_2'$
   (d) and $\Delta_1 \vdash \pi_2p_1 \uparrow \{\alpha{\mapsto}\pi_1p_1\}K_1'' \leftrightarrow \Delta_2 \vdash \pi_2p_2 \uparrow \{\alpha{\mapsto}\pi_1p_2\}K_2''$.
   (e) The inductive hypothesis applies, yielding $(\Delta_1; \pi_1p_1; K_1')$ **is** $(\Delta_2; \pi_1p_2; K_2')$
   (f) and $(\Delta_1; \pi_2p_1; \{\alpha{\mapsto}\pi_1p_1\}K_1'')$ **is** $(\Delta_2; \pi_2p_2; \{\alpha{\mapsto}\pi_1p_2\}K_2'')$.
   (g) Therefore $(\Delta_1; p_1; \Sigma\alpha{:}K_1'.K_1'')$ **is** $(\Delta_2; p_2; \Sigma\alpha{:}K_2'.K_2'')$.

$\blacksquare$

Finally we come to the Fundamental Theorem of Logical Relations, which relates provable equivalence of two constructors to the logical relations The statement of the theorem is strengthened to involve related substitutions of constructors for variables within constructors and kinds.

Theorem 4.9 (Fundamental Theorem)

1. *If* , $\vdash K$ *and* $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$ *then* $(\Delta_1;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 K)$.
2. *If* , $\vdash K_1 \leq K_2$ *and* $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$ *then* $(\Delta_1;\gamma_1 K_1 \leq \gamma_1 K_2)$ **is** $(\Delta_2;\gamma_2 K_1 \leq \gamma_2 K_2)$, $(\Delta_1;\gamma_1 K_1)$ **is** $(\Delta_2;\gamma_2 K_1)$, *and* $(\Delta_1;\gamma_1 K_2)$ **is** $(\Delta_2;\gamma_2 K_2)$.
3. *If* , $\vdash K_1 \equiv K_2$ *and* $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$ *then* $(\Delta_1;\gamma_1 K_1)$ **is** $(\Delta_2;\gamma_2 K_2)$, $(\Delta_1;\gamma_1 K_1)$ **is** $(\Delta_2;\gamma_2 K_1)$, *and* $(\Delta_1;\gamma_1 K_2)$ **is** $(\Delta_2;\gamma_2 K_2)$.
4. *If* , $\vdash A : K$ *and* $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$ *then* $(\Delta_1;\gamma_1 A;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A;\gamma_2 K)$.
5. *If* , $\vdash A_1 \equiv A_2 : K$ *and* $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$ *then* $(\Delta_1;\gamma_1 A_1;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A_1;\gamma_2 K)$, $(\Delta_1;\gamma_1 A_1;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A_2;\gamma_2 K)$, *and* $(\Delta_1;\gamma_1 A_2;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A_2;\gamma_2 K)$.

Proof: By simultaneous induction on the hypothesized derivation.
In all cases, $(\Delta_1;\gamma_1; ,\ )$ **is** $(\Delta_1;\gamma_1; ,\ )$ and $(\Delta_2;\gamma_2; ,\ )$ **is** $(\Delta_2;\gamma_2; ,\ )$.

Kind Well-formedness Rules: , $\vdash K$.

- Case: Rule 5.
  1. $\gamma_1 T = \gamma_2 T = T$.
  2. $(\Delta_1;T)$ **is** $(\Delta_2;T)$.
- Case: Rule 6.
  1. By the inductive hypothesis, $(\Delta_1;\gamma_1 A;T)$ **is** $(\Delta_2;\gamma_2 A;T)$.
  2. Therefore $(\Delta_1;S(\gamma_1 A))$ **is** $(\Delta_2;S(\gamma_2 A))$.
- Case: Rule 7.
  1. By Lemma B.1, there is a strict subderivation , $,\alpha{:}K' \vdash$ ok
  2. and by inversion a strict subderivation , $\vdash K'$.
  3. By the inductive hypothesis, $(\Delta_1;\gamma_1 K')$ **is** $(\Delta_2;\gamma_2 K')$.
  4. Let $(\Delta_1',\Delta_2') \succeq (\Delta_1,\Delta_2)$
  5. and assume that $(\Delta_1';A_1;\gamma_1 K')$ **is** $(\Delta_2';A_2;\gamma_2 K')$.
  6. Then by monotonicity $(\Delta_1';\gamma_1[\alpha{\mapsto}A_1]; ,\ ,\alpha{:}K')$ **is** $(\Delta_2';\gamma_2[\alpha{\mapsto}A_2]; ,\ ,\alpha{:}K')$.
  7. By the inductive hypothesis, $(\Delta_1';(\gamma_1[\alpha{\mapsto}A_1])K'')$ **is** $(\Delta_2';(\gamma_2[\alpha{\mapsto}A_2])K'')$.
  8. That is, $(\Delta_1';\{\alpha{\mapsto}A_1\}((\gamma_1[\alpha{\mapsto}\alpha])K''))$ **is** $(\Delta_2';\{\alpha{\mapsto}A_2\}((\gamma_2[\alpha{\mapsto}\alpha])K''))$.
  9. Therefore, $(\Delta_1;\gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2;\gamma_2(\Pi\alpha{:}K'.K''))$.
- Case: Rule 8. Just like previous case.

Subkinding Rules: , $\vdash K_1 \leq K_2$.
Let $(\Delta_1',\Delta_2') \succeq (\Delta_1,\Delta_2)$ and assume $(\Delta_1';B_1;\gamma_1 K_1)$ **is** $(\Delta_2';B_2;\gamma_2 K_1)$.

- Case: Rule 9. By assumption, $(\Delta_1';B_1;T)$ **is** $(\Delta_2';B_2;T)$.
- Also, $(\Delta_1;T)$ **is** $(\Delta_2;T)$
- and, by the same argument as for Rule 6, $(\Delta_1;S(\gamma_1 K))$ **is** $(\Delta_2;S(\gamma_2 K))$.
- Case: Rule 10. Trivial, since $\gamma_1 T = \gamma_2 T = T$ and $(\Delta_1;T)$ **is** $(\Delta_2;T)$.
- Case: Rule 11.

1. By the inductive hypothesis we have $(\Delta_1'; \gamma_1 A_1; T)$ **is** $(\Delta_2'; \gamma_2 A_1; T)$
2. and $(\Delta_1'; \gamma_1 A_2; T)$ **is** $(\Delta_2'; \gamma_2 A_2; T)$.
3. Thus $(\Delta_1; S(\gamma_1 A_1))$ **is** $(\Delta_2; S(\gamma_2 A_1))$
4. and $(\Delta_1; S(\gamma_1 A_2))$ **is** $(\Delta_2; S(\gamma_2 A_2))$.

5. By the inductive hypothesis we have $(\Delta_1'; \gamma_1 A_1; T)$ **is** $(\Delta_2'; \gamma_2 A_2; T)$,
6. $(\Delta_1'; \gamma_1 A_1; T)$ **is** $(\Delta_1'; \gamma_1 A_2; T)$,
7. and $(\Delta_2'; \gamma_2 A_1; T)$ **is** $(\Delta_2'; \gamma_2 A_2; T)$.
8. Thus $(\Delta_1'; S(\gamma_1 A_1))$ **is** $(\Delta_2'; S(\gamma_2 A_2))$,
9. $(\Delta_1'; S(\gamma_1 A_1))$ **is** $(\Delta_1'; S(\gamma_1 A_2))$,
10. and $(\Delta_2'; S(\gamma_2 A_1))$ **is** $(\Delta_2'; S(\gamma_2 A_2))$.
11. By Symmetry and Transitivity, $(\Delta_1'; S(\gamma_1 A_2))$ **is** $(\Delta_2'; S(\gamma_2 A_2))$,
12. so by Lemma 4.3, $(\Delta_1'; S(\gamma_1 A_1) \leq S(\gamma_1 A_2))$ **is** $(\Delta_2'; S(\gamma_2 A_1) \leq S(\gamma_2 A_2))$.
13. Therefore $(\Delta_1'; B_1; S(\gamma_1 A_2))$ **is** $(\Delta_2'; B_2; S(\gamma_2 A_2))$.

- Case: Rule 12.

1. By the inductive hypothesis, $(\Delta_1; \gamma_1(\Pi\alpha{:}K_1'.K_1''))$ **is** $(\Delta_2; \gamma_2(\Pi\alpha{:}K_1'.K_1''))$.
2. For the same reasons as for Rule 7, $(\Delta_1; \gamma_1(\Pi\alpha{:}K_1'.K_1''))$ **is** $(\Delta_2; \gamma_2(\Pi\alpha{:}K_1'.K_1''))$.

3. Let $\Delta_1'', \Delta_1''' \succeq \Delta_1'$
4. and assume $(\Delta_1''; B_1'; \gamma_1 K_2')$ **is** $(\Delta_1'''; B_1''; \gamma_1 K_2')$.
5. By monotonicity and the inductive hypothesis, $(\Delta_1''; \gamma_1 K_2' \leq \gamma_1 K_1')$ **is** $(\Delta_1'''; \gamma_1 K_2' \leq \gamma_1 K_1')$.
6. Thus $(\Delta_1''; B_1'; \gamma_1 K_1')$ **is** $(\Delta_1'''; B_1''; \gamma_1 K_1')$.
7. Now by reflexivity and monotonicity, $(\Delta_1''; B_1; \gamma_1 K_1)$ **is** $(\Delta_1'''; B_1; \gamma_1 K_1)$.
8. Thus $(\Delta_1''; B_1 B_1'; (\gamma_1[\alpha\mapsto B_1'])K_1'')$ **is** $(\Delta_1'''; B_1 B_1''; (\gamma_1[\alpha\mapsto B_1''])K_1'')$.
9. Now $(\Delta_1''; \gamma_1[\alpha\mapsto B_1']; , , \alpha{:}K_2')$ **is** $(\Delta_1'''; \gamma_1[\alpha\mapsto B_1'']; , , \alpha{:}K_2')$.
10. By the inductive hypothesis again,
    $(\Delta_1''; (\gamma_1[\alpha\mapsto B_1'])K_1'' \leq (\gamma_1[\alpha\mapsto B_1'])K_2'')$ **is** $(\Delta_1'''; (\gamma_1[\alpha\mapsto B_1''])K_1'' \leq (\gamma_1[\alpha\mapsto B_1''])K_2'')$,
11. so $(\Delta_1''; B_1 B_1'; (\gamma_1[\alpha\mapsto B_1'])K_2'')$ **is** $(\Delta_1'''; B_1 B_1''; (\gamma_1[\alpha\mapsto B_1''])K_2'')$.
12. Note that $(\Delta_1''; (\gamma_1[\alpha\mapsto B_1'])K_2'')$ **is** $(\Delta_1'''; (\gamma_1[\alpha\mapsto B_1''])K_2'')$.
13. Therefore, $(\Delta_1'; B_1; \gamma_1(\Pi\alpha{:}K_2'.K_2''))$ **valid**.

14. An analogous argument shows that $(\Delta_2'; B_2; \gamma_2(\Pi\alpha{:}K_2'.K_2''))$ **valid**.

15. Let $(\Delta_1'', \Delta_2'') \succeq (\Delta_1', \Delta_2')$
16. and assume $(\Delta_1''; B_1'; \gamma_1 K_2')$ **is** $(\Delta_2''; B_2'; \gamma_2 K_2')$.
17. By the inductive hypothesis, $(\Delta_1'; \gamma_1 K_2' \leq \gamma_1 K_1')$ **is** $(\Delta_2'; \gamma_2 K_2' \leq \gamma_2 K_1')$.
18. so $(\Delta_1''; B_1'; \gamma_1 K_1')$ **is** $(\Delta_2''; B_2'; \gamma_2 K_1')$
19. and $(\Delta_1''; B_1 B_1'; (\gamma_1[\alpha\mapsto B_1'])K_1'')$ **is** $(\Delta_2''; B_2 B_2'; (\gamma_2[\alpha\mapsto B_2'])K_1'')$.
20. By monotonicity, $(\Delta_1''; \gamma_1[\alpha\mapsto B_1']; , , \alpha{:}K_2')$ **is** $(\Delta_2''; \gamma_2[\alpha\mapsto B_2']; , , \alpha{:}K_2')$.
21. By the inductive hypothesis again,
    $(\Delta_1''; (\gamma_1[\alpha\mapsto B_1'])K_1'' \leq (\gamma_1[\alpha\mapsto B_1'])K_2'')$ **is** $(\Delta_2''; (\gamma_2[\alpha\mapsto B_2'])K_1'' \leq (\gamma_2[\alpha\mapsto B_2'])K_2'')$,
22. so $(\Delta_1''; B_1 B_1'; (\gamma_1[\alpha\mapsto B_1'])K_2'')$ **is** $(\Delta_2''; B_2 B_2'; (\gamma_2[\alpha\mapsto B_2'])K_2'')$.
23. Thus $(\Delta_1'; B_1; \gamma_1(\Pi\alpha{:}K_2'.K_2''))$ **is** $(\Delta_2'; B_2; \gamma_2(\Pi\alpha{:}K_2'.K_2''))$.

- Rule 13.

1. By the inductive hypothesis, $(\Delta_1; \gamma_1(\Sigma\alpha{:}K_2'.K_2''))$ **is** $(\Delta_2; \gamma_2(\Sigma\alpha{:}K_2'.K_2''))$.

2. For the same reasons as for Rule 7, $(\Delta_1;\gamma_1(\Sigma\alpha{:}K_1'.K_1''))\ \mathsf{is}\ (\Delta_2;\gamma_2(\Sigma\alpha{:}K_1'.K_1''))$.

3. $(\Delta_1';\pi_1 B_1;\gamma_1 K_1')\,\mathbf{valid}$.
4. By the inductive hypothesis, $(\Delta_1';\gamma_1 K_1' \leq \gamma_1 K_2')\ \mathsf{is}\ (\Delta_1';\gamma_1 K_1' \leq \gamma_1 K_2')$.
5. Thus by reflexivity, $(\Delta_1';\pi_1 B_1;\gamma_1 K_2')\,\mathbf{valid}$.
6. Now $(\Delta_1';\gamma_1[\alpha\mapsto\pi_1 B_1];,\ ,\alpha{:}K_1')\ \mathsf{is}\ (\Delta_1';\gamma_1[\alpha\mapsto\pi_1 B_1];,\ ,\alpha{:}K_1')$
7. so by the inductive hypothesis,
   $(\Delta_1';(\gamma_1[\alpha\mapsto\pi_1 B_1])K_1'' \leq (\gamma_1[\alpha\mapsto\pi_1 B_1])K_2'')\ \mathsf{is}\ (\Delta_1';(\gamma_1[\alpha\mapsto\pi_1 B_1])K_1'' \leq (\gamma_1[\alpha\mapsto\pi_1 B_1])K_2'')$.
8. Since $(\Delta_1';\pi_2 B_1;(\gamma_1[\alpha\mapsto\pi_1 B_1])K_1'')\,\mathbf{valid}$,
9. Using reflexivity, $(\Delta_1';\pi_2 B_1;(\gamma_1[\alpha\mapsto\pi_1 B_1])K_2'')\,\mathbf{valid}$.
10. Therefore, $(\Delta_1';B_1;\gamma_1(\Sigma\alpha{:}K_2'.K_2''))\,\mathbf{valid}$.

11. An analogous argument shows that $(\Delta_2';B_2;\gamma_2(\Sigma\alpha{:}K_2'.K_2''))\,\mathbf{valid}$.

12. $(\Delta_1';\pi_1 B_1;\gamma_1 K_1')\ \mathsf{is}\ (\Delta_2';\pi_1 B_2;\gamma_2 K_1')$.
13. By the inductive hypothesis, $(\Delta_1';\gamma_1 K_1' \leq \gamma_1 K_2')\ \mathsf{is}\ (\Delta_2';\gamma_2 K_1' \leq \gamma_2 K_2')$.
14. $(\Delta_1';\pi_1 B_1;\gamma_1 K_2')\ \mathsf{is}\ (\Delta_2';\pi_1 B_2;\gamma_2 K_2')$.
15. Now $(\Delta_1';\gamma_1[\alpha\mapsto\pi_1 B_1];,\ ,\alpha{:}K_1')\ \mathsf{is}\ (\Delta_2';\gamma_2[\alpha\mapsto\pi_1 B_2];,\ ,\alpha{:}K_1')$
16. so by the inductive hypothesis,
    $(\Delta_1';(\gamma_1[\alpha\mapsto\pi_1 B_1])K_1'' \leq (\gamma_1[\alpha\mapsto\pi_1 B_1])K_2'')\ \mathsf{is}\ (\Delta_2';(\gamma_2[\alpha\mapsto\pi_1 B_2])K_1'' \leq (\gamma_2[\alpha\mapsto\pi_1 B_2])K_2'')$.
17. Since $(\Delta_1';\pi_2 B_1;(\gamma_1[\alpha\mapsto\pi_1 B_1])K_1'')\ \mathsf{is}\ (\Delta_2';\pi_2 B_2;(\gamma_2[\alpha\mapsto\pi_1 B_2])K_1'')$,
18. $(\Delta_1';\pi_2 B_1;(\gamma_1[\alpha\mapsto\pi_1 B_1])K_2'')\ \mathsf{is}\ (\Delta_2';\pi_2 B_2;(\gamma_2[\alpha\mapsto\pi_1 B_2])K_2'')$.
19. Therefore, $(\Delta_1';B_1;\gamma_1(\Sigma\alpha{:}K_2'.K_2''))\ \mathsf{is}\ (\Delta_2';B_2;\gamma_2(\Sigma\alpha{:}K_2'.K_2''))$.

Kind Equivalence Rules: $,\ \vdash K_1 \equiv K_2$.
It suffices to prove that if $,\ \vdash K_1 \equiv K_2$ and $(\Delta_1;\gamma_1;,\ )\ \mathsf{is}\ (\Delta_2;\gamma_2;,\ )$ then $(\Delta_1;\gamma_1 K_1)\ \mathsf{is}\ (\Delta_2;\gamma_2 K_2)$, because we can apply this to get $(\Delta_2;\gamma_2 K_1)\ \mathsf{is}\ (\Delta_2;\gamma_2 K_2)$, so $(\Delta_1;\gamma_1 K_1)\ \mathsf{is}\ (\Delta_2;\gamma_2 K_1)$ follows by Symmetry and Transitivity. A similar argument yields $(\Delta_1;\gamma_1 K_2)\ \mathsf{is}\ (\Delta_2;\gamma_2 K_2)$.

- Rule 14. Trivial.
- Rule 15.
  1. By the inductive hypothesis, $(\Delta_1;\gamma_1 A_1;T)\ \mathsf{is}\ (\Delta_2;\gamma_2 A_2;T)$.
  2. Therefore, $(\Delta_1;S(\gamma_1 A_1))\ \mathsf{is}\ (\Delta_2;S(\gamma_2 A_2))$.
- Rule 16.
  1. By the inductive hypothesis, $(\Delta_1;\gamma_1 K_1')\ \mathsf{is}\ (\Delta_2;\gamma_2 K_2')$.
  2. Let $(\Delta_1',\Delta_2') \succeq (\Delta_1,\Delta_2)$
  3. and assume $(\Delta_1';A_1;\gamma_1 K_1')\ \mathsf{is}\ (\Delta_2';A_2;\gamma_2 K_2')$.
  4. By the inductive hypothesis, $(\Delta_1';\gamma_1 K_1')\ \mathsf{is}\ (\Delta_2';\gamma_2 K_2')$.
  5. $(\Delta_1';\gamma_1 K_1')\ \mathsf{is}\ (\Delta_1';\gamma_1 K_2')$,
  6. and $(\Delta_2';\gamma_2 K_1')\ \mathsf{is}\ (\Delta_2';\gamma_2 K_2')$.
  7. By Symmetry and Transitivity, $(\Delta_2';\gamma_2 K_2')\ \mathsf{is}\ (\Delta_1';\gamma_1 K_1')$,
  8. $(\Delta_1';\gamma_1 K_1')\ \mathsf{is}\ (\Delta_1';\gamma_2 K_1')$
  9. and by Reflexivity $(\Delta_1';\gamma_1 K_1')\ \mathsf{is}\ (\Delta_1';\gamma_1 K_1')$.
  10. By Lemma 4.3, $(\Delta_1';\gamma_1 K_1' \leq \gamma_1 K_1')\ \mathsf{is}\ (\Delta_1';\gamma_2 K_2' \leq \gamma_2 K_1')$,
  11. so $(\Delta_1';A_1;\gamma_1 K_1')\ \mathsf{is}\ (\Delta_2';A_2;\gamma_2 K_1')$.
  12. By monotonicity, then, $(\Delta_1';\gamma_1[\alpha\mapsto A_1];,\ ,\alpha{:}K_1')\ \mathsf{is}\ (\Delta_2';\gamma_2[\alpha\mapsto A_2];,\ ,\alpha{:}K_1')$.
  13. By the inductive hypothesis again, $(\Delta_1';(\gamma_1[\alpha\mapsto A_1])K_1'')\ \mathsf{is}\ (\Delta_2';(\gamma_2[\alpha\mapsto A_2])K_2'')$.

14. Therefore $(\Delta_1; \gamma_1(\Pi\alpha{:}K_1'.K_1''))$ **is** $(\Delta_2; \gamma_2(\Pi\alpha{:}K_2'.K_2''))$.

- Rule 17. Same proof as for previous case.

Constructor Validity Rules: $, \vdash A : K$.

- Case: Rule 18.
  1. $\gamma_1 b_i = \gamma_2 b_i = b$ and $\gamma_1 T = \gamma_2 T = T$.
  2. $\Delta_1 \vdash b_i : T \Leftrightarrow \Delta_2 \vdash b_i : T$,
  3. $\Delta_1 \vdash b_i : T \Leftrightarrow \Delta_1 \vdash b_i : T$,
  4. and $\Delta_2 \vdash b_i : T \Leftrightarrow \Delta_2 \vdash b_i : T$.
  5. Thus $(\Delta_1; b_i; T)$ **is** $(\Delta_2; b_i; T)$.
- Case: Rule 19.
  By the assumption on γ_1 and γ_2, $(\Delta_1; \gamma_1 x; \gamma_1(, x))$ **is** $(\Delta_2; \gamma_2 x; \gamma_2(, x))$.
- Case: Rule 20.
  1. By Lemma B.1 there is a strict subderivation $, \vdash K'$.
  2. By the inductive hypothesis, $(\Delta_1; \gamma_1 K')$ **is** $(\Delta_2; \gamma_2 K')$.
  3. Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$ and assume $(\Delta_1'; B_1; \gamma_1 K')$ **is** $(\Delta_2'; B_2; \gamma_2 K')$.
  4. Using monotonicity, $(\Delta_1'; \gamma_1[\alpha{\mapsto}B_1];, ,\alpha{:}K')$ **is** $(\Delta_2'; \gamma_2[\alpha{\mapsto}B_2];, ,\alpha{:}K')$.
  5. By the inductive hypothesis, $(\Delta_1'; (\gamma_1[\alpha{\mapsto}B_1])A; (\gamma_1[\alpha{\mapsto}B_1])K'')$ **is** $(\Delta_2'; (\gamma_2[\alpha{\mapsto}B_2])A; (\gamma_2[\alpha{\mapsto}B_2])K'')$.
  6. Now $\Delta_1 \vdash (\gamma_1[\alpha{\mapsto}B_1])A \simeq (\gamma_1(\lambda\alpha{:}K'.A))B_1$
  7. and $\Delta_2 \vdash (\gamma_2[\alpha{\mapsto}B_2])A \simeq (\gamma_2(\lambda\alpha{:}K'.A))B_2$.
  8. By Lemma 4.7, $(\Delta_1'; (\gamma_1(\lambda\alpha{:}K'.A))B_1; (\gamma_1[\alpha{\mapsto}B_1])K'')$ **is** $(\Delta_2'; (\gamma_2(\lambda\alpha{:}K'.A))B_2; (\gamma_2[\alpha{\mapsto}B_2])K'')$.
  9. Similar arguments analogous to lines 3–8 (and reflexivity) show that $(\Delta_1; \gamma_1(\lambda\alpha{:}K'.A); \gamma_1(\Pi\alpha{:}K'.K''))$ **valid**
  10. and $(\Delta_2; \gamma_2(\lambda\alpha{:}K'.A); \gamma_2(\Pi\alpha{:}K'.K''))$ **valid**.
  11. Therefore $(\Delta_1; \gamma_1(\lambda\alpha{:}K'.A); \gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2(\lambda\alpha{:}K'.A); \gamma_2(\Pi\alpha{:}K'.K''))$.
- Case: Rule 21
  1. By the inductive hypothesis $(\Delta_1; \gamma_1 A; \gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2 A; \gamma_2(\Pi\alpha{:}K'.K''))$
  2. and $(\Delta_1; \gamma_1 A'; \gamma_1 K')$ **is** $(\Delta_2; \gamma_2 A'; \gamma_2 K')$.
  3. Therefore, $(\Delta_1; \gamma_1(AA'); \gamma_1(\{\alpha{\mapsto}A'\}K''))$ **is** $(\Delta_2; \gamma_2(AA'); \gamma_2(\{\alpha{\mapsto}A'\}K''))$.
- Case: Rule 22.
  1. By the inductive hypothesis, $(\Delta_1; \gamma_1 A; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2 A; \gamma_2(\Sigma\alpha{:}K'.K''))$.
  2. Therefore $(\Delta_1; \pi_1\gamma_1 A; \gamma_1 K')$ **is** $(\Delta_2; \pi_1\gamma_2 A; \gamma_2 K')$.
- Case: Rule 23.
  1. By the inductive hypothesis, $(\Delta_1; \gamma_1 A; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2 A; \gamma_2(\Sigma\alpha{:}K'.K''))$.
  2. Therefore $(\Delta_1; \pi_2\gamma_1 A; \gamma_1(\{\alpha{\mapsto}\pi_1 A\}K''))$ **is** $(\Delta_2; \pi_2\gamma_2 A; \gamma_2(\{\alpha{\mapsto}\pi_1 A\}K''))$.
- Case: Rule 24.
  1. By the inductive hypothesis, $(\Delta_1; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2(\Sigma\alpha{:}K'.K''))$.
  2. By the inductive hypothesis and reflexivity, $(\Delta_1; \gamma_1 A_1; \gamma_1 K')$ **valid**
  3. and $(\Delta_1; \gamma_1 A_2; (\gamma_1[\alpha{\mapsto}\gamma_1 A_1])K'')$ **valid**.
  4. Now $\Delta_1 \vdash \gamma_1 A_1 \simeq \pi_1\langle\gamma_1 A_1, \gamma_1 A_2\rangle$
  5. and $\Delta_1 \vdash \gamma_1 A_2 \simeq \pi_2\langle\gamma_1 A_1, \gamma_1 A_2\rangle$.

6. by Lemma 4.7 we have $(\Delta_1; \pi_1\langle\gamma_1 A_1, \gamma_1 A_2\rangle; \gamma_1 K')$ **valid**,
7. $(\Delta_1; \pi_2\langle\gamma_1 A_1, \gamma_1 A_2\rangle; (\gamma_1[\alpha\mapsto\gamma_1 A_1])K'')$ **valid**.
8. and $(\Delta_1; \pi_1\langle\gamma_1 A_1, \gamma_1 A_2\rangle; \gamma_1 K')$ **is** $(\Delta_1; \gamma_1 A_1; \gamma_1 K')$.
9. Then $(\Delta_1; (\gamma_1[\alpha\mapsto\gamma_1 A_1])K'')$ **is** $(\Delta_1; (\gamma_1[\alpha\mapsto\pi_1\langle\gamma_1 A_1, \gamma_1 A_2\rangle])K'')$.
10. Using Lemma 4.3, $(\Delta_1; \pi_2\langle\gamma_1 A_1, \gamma_1 A_2\rangle; (\gamma_1[\alpha\mapsto\pi_1\langle\gamma_1 A_1, \gamma_1 A_2\rangle])K'')$ **valid**.
11. Therefore, $(\Delta_1; \langle\gamma_1 A_1, \gamma_1 A_2\rangle; \gamma_1(\Sigma\alpha{:}K'.K''))$ **valid**

12. A very similar argument shows that $(\Delta_2; \langle\gamma_2 A_1, \gamma_2 A_2\rangle; \gamma_2(\Sigma\alpha{:}K'.K''))$ **valid**
13. and an analogous argument shows that
$(\Delta_1; \langle\gamma_1 A_1, \gamma_1 A_2\rangle; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \langle\gamma_2 A_1, \gamma_2 A_2\rangle; \gamma_2(\Sigma\alpha{:}K'.K''))$.

- Case: Rule 25
  1. By the inductive hypothesis, $(\Delta_1; \gamma_1 A; T)$ **is** $(\Delta_2; \gamma_2 A; T)$.
  2. As in the case for Rule 6, $(\Delta_1; S(\gamma_1 A))$ **is** $(\Delta_2; S(\gamma_2 A))$.
  3. Thus $(\Delta_1; \gamma_1 A; S(\gamma_1 A))$ **valid**,
  4. $(\Delta_2; \gamma_2 A; S(\gamma_2 A))$ **valid**,
  5. and $(\Delta_1; \gamma_1 A; S(\gamma_1 A))$ **is** $(\Delta_2; \gamma_2 A; S(\gamma_2 A))$.
- Case: Rule 26.
  1. By the inductive hypothesis, $(\Delta_1; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2(\Sigma\alpha{:}K'.K''))$,
  2. $(\Delta_1; \pi_1(\gamma_1 A); \gamma_1 K')$ **is** $(\Delta_2; \pi_1(\gamma_2 A); \gamma_2 K')$,
  3. and $(\Delta_1; \pi_2(\gamma_1 A); \gamma_1(\{\alpha\mapsto\pi_1 A\}K''))$ **is** $(\Delta_2; \pi_2(\gamma_2 A); \gamma_2(\{\alpha\mapsto\pi_1 A\}K''))$.
  4. Thus $(\Delta_1; \gamma_1 A; \gamma_1(\Sigma\alpha{:}K'.K''))$ **valid**,
  5. $(\Delta_2; \gamma_2 A; \gamma_2(\Sigma\alpha{:}K'.K''))$ **valid**,
  6. and therefore $(\Delta_1; \gamma_1 A; \gamma_1(\Sigma\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2 A; \gamma_2(\Sigma\alpha{:}K'.K''))$,
- Case: Rule 27
  1. By Lemma B.1 and the inductive hypothesis, $(\Delta_1; \gamma_1 K')$ **is** $(\Delta_2; \gamma_2 K')$.
  2. Let $\Delta_1', \Delta_1'' \succeq \Delta_1$
  3. and assume $(\Delta_1'; B_1'; \gamma_1 K')$ **is** $(\Delta_1''; B_1''; \gamma_1 K')$.
  4. By monotonicity, $(\Delta_1'; \gamma_1[\alpha\mapsto B_1'];\,,\,,\alpha{:}K')$ **is** $(\Delta_1''; \gamma_1[\alpha\mapsto B_1''];\,,\,,\alpha{:}K')$.
  5. By the inductive hypothesis,
  $(\Delta_1'; (\gamma_1[\alpha\mapsto B_1'])(A\alpha); (\gamma_1[\alpha\mapsto B_1'])K'')$ **is** $(\Delta_1''; (\gamma_2[\alpha\mapsto B_1''])(A\alpha); (\gamma_2[\alpha\mapsto B_1''])K'')$.
  6. That is, $(\Delta_1'; (\gamma_1 A)B_1'; (\gamma_1[\alpha\mapsto B_1'])K'')$ **is** $(\Delta_1''; (\gamma_2 A)B_1''; (\gamma_2[\alpha\mapsto B_1''])K'')$.
  7. Therefore, $(\Delta_1; \gamma_1(\Pi\alpha{:}K'.K''))$ **valid**
  8. and $(\Delta_1; \gamma_1 A; \gamma_1(\Pi\alpha{:}K'.K''))$ **valid**.

  9. A similar proof shows that $(\Delta_2; \gamma_2 A; \gamma_2(\Pi\alpha{:}K'.K''))$ **valid**.

  10. Let $(\Delta_1', \Delta_2') \succeq (\Delta_1, \Delta_2)$
  11. and assume $(\Delta_1'; B_1; \gamma_1 K')$ **is** $(\Delta_2'; B_2; \gamma_2 K')$.
  12. By monotonicity, $(\Delta_1'; \gamma_1[\alpha\mapsto B_1];\,,\,,\alpha{:}K')$ **is** $(\Delta_2'; \gamma_2[\alpha\mapsto B_2];\,,\,,\alpha{:}K')$.
  13. By the inductive hypothesis,
  $(\Delta_1'; (\gamma_1[\alpha\mapsto B_1])(A\alpha); (\gamma_1[\alpha\mapsto B_1])K'')$ **is** $(\Delta_2'; (\gamma_2[\alpha\mapsto B_2])(A\alpha); (\gamma_2[\alpha\mapsto B_2])K'')$.
  14. That is, $(\Delta_1'; (\gamma_1 A)B_1; (\gamma_1[\alpha\mapsto B_1])K'')$ **is** $(\Delta_2'; (\gamma_2 A)B_2; (\gamma_2[\alpha\mapsto B_2])K'')$.
  15. Therefore, $(\Delta_1; \gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2(\Pi\alpha{:}K'.K''))$
  16. and $(\Delta_1; \gamma_1 A; \gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2 A; \gamma_2(\Pi\alpha{:}K'.K''))$.

- Case: Rule 28
    1. By the inductive hypothesis, $(\Delta_1; \gamma_1 A; \gamma_1 K_1)$ **is** $(\Delta_1; \gamma_2 A; \gamma_2 K_1)$
    2. and $(\Delta_1; \gamma_1 K_1 \leq \gamma_1 K_2)$ **is** $(\Delta_2; \gamma_2 K_1 \leq \gamma_2 K_2)$.
    3. Therefore, $(\Delta_1; \gamma_1 A; \gamma_1 K_2)$ **is** $(\Delta_1; \gamma_2 A; \gamma_2 K_2)$

Constructor Equivalence Rules: , $\vdash A_1 \equiv A_2 : K$.
It suffices to prove that if , $\vdash A_1 \equiv A_2 : K$ and $(\Delta_1; \gamma_1; ,\ )$ **is** $(\Delta_2; \gamma_2; ,\ )$ then $(\Delta_1; \gamma_1 A_1; \gamma_1 K)$ **is** $(\Delta_2; \gamma_2 A_2; \gamma_2 K)$, because it follows that $(\Delta_2; \gamma_2 A_1; \gamma_2 K_1)$ **is** $(\Delta_2; \gamma_2 A_2; \gamma_2 K_2)$, so $(\Delta_1; \gamma_1 A_1; \gamma_1 K)$ **is** $(\Delta_2; \gamma_2 A_2; \gamma_2 K)$ by Symmetry and Transitivity. A similar argument yields $(\Delta_1; \gamma_1 A_2; \gamma_1 K)$ **is** $(\Delta_2; \gamma_2 A_2; \gamma_2 K)$.

- Case: Rule 29.
    1. By the arguments for Rule 40,
       $(\Delta_1; \gamma_1(\lambda\alpha{:}K'.A_1); \gamma_1(\Pi\alpha{:}K'.K''))$ **is** $(\Delta_2; \gamma_2(\lambda\alpha{:}K'.A_2); \gamma_2(\Pi\alpha{:}K'.K''))$.
    2. By the inductive hypothesis, $(\Delta_1; \gamma_1 A_1'; \gamma_1 K')$ **is** $(\Delta_2; \gamma_2 A_2'; \gamma_2 K')$.
    3. Therefore, $(\Delta_1; \gamma_1((\lambda\alpha{:}K'.A_1)A_1'); \gamma_1(\{\alpha \mapsto A_1'\}K''))$ **is** $(\Delta_2; \gamma_2((\lambda\alpha{:}K'.A_2)A_2'); \gamma_2(\{\alpha \mapsto A_2'\}K''))$.
    4. Similarly $(\Delta_1; \gamma_1((\lambda\alpha{:}K'.A_1)A_1'); \gamma_1(\{\alpha \mapsto A_1'\}K''))$ **is** $(\Delta_2; \gamma_2((\lambda\alpha{:}K'.A_1)A_1'); \gamma_2(\{\alpha \mapsto A_1'\}K''))$.
    5. But $\Delta_2 \vdash \gamma_2((\lambda\alpha{:}K'.A_2)A_2') \simeq \gamma_2(\{\alpha \mapsto A_2'\}A_2)$.
    6. Thus by Lemma 4.7,
       $(\Delta_1; \gamma_1((\lambda\alpha{:}K'.A_1)A_1'); \gamma_1(\{\alpha \mapsto A_1'\}K''))$ **is** $(\Delta_2; \gamma_2(\{\alpha \mapsto A_2'\}A_2); \gamma_2(\{\alpha \mapsto A_2'\}K''))$.
    7. Then since $(\Delta_2; \gamma_2 A_1'; \gamma_2 K')$ **is** $(\Delta_2; \gamma_2 A_2'; \gamma_2 K')$
    8. we have $(\Delta_2; \gamma_2(\{\alpha \mapsto A_1'\}K''))$ **is** $(\Delta_2; \gamma_2(\{\alpha \mapsto A_2'\}K''))$.
    9. By Lemma 4.3, $(\Delta_1; \gamma_1((\lambda\alpha{:}K'.A_1)A_1'); \gamma_1(\{\alpha \mapsto A_1'\}K''))$ **is** $(\Delta_2; \gamma_2(\{\alpha \mapsto A_2'\}A_2); \gamma_2(\{\alpha \mapsto A_1'\}K''))$.
- Case: Rule 30.
  Exact analog to the proof of Rule 27.
- Case: Rule 31.
  This proof is analogous to the proof for Rule 26 except that due to the assymmetry of the rule's last premise we must note that $(\Delta_1; \gamma_2(\{\alpha \mapsto \pi_1 A_1\}K''))$ **is** $(\Delta_2; \gamma_2(\{\alpha \mapsto \pi_1 A_2\}K''))$ and use Lemma 4.3.
- Case: Rule 32.
    1. By an argument as in the proof of Rule 24,
       $(\Delta_1; \gamma_1\langle A_1, A_2\rangle; \gamma_1(K_1 \times K_2))$ **is** $(\Delta_2; \gamma_2\langle A_1, A_2\rangle; \gamma_2(K_1 \times K_2))$.
    2. Thus $(\Delta_1; \gamma_1(\pi_1\langle A_1, A_2\rangle); \gamma_1 K_1)$ **is** $(\Delta_2; \gamma_2(\pi_1\langle A_1, A_2\rangle); \gamma_2 K_1)$.
    3. By the inductive hypothesis, $(\Delta_1; \gamma_1 A_1'; \gamma_1 K_1)$ **is** $(\Delta_2; \gamma_2 A_1'; \gamma_2 K_1)$.
    4. Now $(\Delta_1; \gamma_1\langle A_1, A_2\rangle; \gamma_1(K_1 \times K_2))$ **is** $(\Delta_2; \gamma_2\langle A_1', A_2\rangle; \gamma_2(K_1 \times K_2))$.
    5. and $\Delta_2 \vdash \gamma_2 \pi_1\langle A_1', A_2\rangle \simeq \gamma_2 A_1'$.
    6. By Lemma 4.7, $(\Delta_1; \gamma_1 \pi_1\langle A_1, A_2\rangle; \gamma_1 K_1)$ **is** $(\Delta_2; \gamma_2 A_1'; \gamma_2 K_1)$.
- Case: Rule 33.
  Same argument as previous case.
- Case: Rule 34.
    1. By the inductive hypothesis, $(\Delta_1; \gamma_1 A; S(\gamma_1 B))$ **is** $(\Delta_2; \gamma_2 A; S(\gamma_2 B))$.
    2. Thus $\Delta_1 \vdash \gamma_1 A : T \Leftrightarrow \Delta_2 \vdash \gamma_2 A : T$,
    3. $\Delta_1 \vdash \gamma_1 B : T \Leftrightarrow \Delta_2 \vdash \gamma_2 B : T$,
    4. and $\Delta_2 \vdash \gamma_2 A : T \Leftrightarrow \Delta_2 \vdash \gamma_2 B : T$.
    5. By transitivity, $\Delta_1 \vdash \gamma_1 A : T \Leftrightarrow \Delta_2 \vdash \gamma_2 B : T$.
    6. Therefore $(\Delta_1; \gamma_1 A; T)$ **is** $(\Delta_2; \gamma_2 A; T)$,
    7. $(\Delta_1; \gamma_1 B; T)$ **is** $(\Delta_2; \gamma_2 B; T)$,

8. and $(\Delta_1;\gamma_1 A;T)$ **is** $(\Delta_2;\gamma_2 B;T)$.

- Case: Rule 35.
  By the inductive hypothesis and the definitions of the relations.
- Case: Rule 36.
  By the inductive hypothesis and Lemma 4.5.
- Case: Rule 37.
  1. By the inductive hypothesis, $(\Delta_1;\gamma_1 A;\gamma_1 K)$ **is** $(\Delta_1;\gamma_1 A';\gamma_1 K)$
  2. and $(\Delta_1;\gamma_1 A';\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A'';\gamma_2 K)$.
  3. By Lemma 4.6, $(\Delta_1;\gamma_1 A;\gamma_1 K)$ **is** $(\Delta_2;\gamma_2 A'';\gamma_2 K)$.
- Case: Rule 38.
  By the definition of the algorithm and the logical relations.
- Case: Rule 39.
  By the assumption regarding γ_1 and γ_2.
- Case: Rule 40.
  Analogous to the proof for rule 20.
- Case: Rule 41.
  1. Using the inductive hypothesis, $(\Delta_1;\gamma_1(AA_1);\gamma_1(\{\alpha\mapsto A_1\}K_2))$ **is** $(\Delta_2;\gamma_2(A'A'_1);\gamma_2(\{\alpha\mapsto A'_1\}K_2))$.
  2. Therefore by Lemma 4.3, $(\Delta_1;\gamma_1(AA_1);\gamma_1(\{\alpha\mapsto A_1\}K_2))$ **is** $(\Delta_2;\gamma_2(A'A'_1);\gamma_2(\{\alpha\mapsto A_1\}K_2))$.
- Case: Rule 42.
  Analogous to the proof for Rule 22.
- Case: Rule 43.
  Analogous to proofs for Rule 23 and Rule 41, except that the assymmetry of the conclusion requires a use of Lemma 4.3.
- Case: Rule 44.
  Analogous to proof for Rule 24 except that the assymmetry of the rule's last premise requires a use of Lemma 4.3.
- Case: Rule 45.
  By the inductive hypothesis and the definition of the logical relations.

∎

A straightforward proof by induction on well-formed contexts shows that the identity substitution is related to itself:

Lemma 4.10
If , $\vdash$ *ok then for all* $\beta \in \mathrm{dom}(,\ )$ *we have* $(,\ ;\beta;,\ \beta)$ **is** $(,\ ;\beta;,\ \beta)$. *That is,* $(,\ ;\mathrm{id};,\ )$ **is** $(,\ ;\mathrm{id};,\ )$ *where* id *is the identity function.*

Proof: By induction on the proof of , $\vdash$ ok.

- Case: Empty context. Vacuous.
- Case: $,\ ,\alpha{:}K$.
  1. By Lemma B.1, , $\vdash K$, and , $\vdash$ ok.
  2. Also, $\alpha \notin \mathrm{dom}(,\ )$.
  3. By the inductive hypothesis, $(,\ ;\beta;,\ \beta)$ **is** $(,\ ;\beta;,\ \beta)$ for all $\beta \in \mathrm{dom}(,\ )$.
  4. By monotonicity, $(,\ ,\alpha{:}K;\beta;((,\ ,\alpha{:}K)\beta))$ **is** $(,\ ,\alpha{:}K;\beta;((,\ ,\alpha{:}K)\beta))$ for all $\beta \in \mathrm{dom}(,\ )$.
  5. By Theorem 4.9, $(,\ ;K)$ **is** $(,\ ;K)$
  6. and by monotonicity $(,\ ,\alpha{:}K;K)$ **is** $(,\ ,\alpha{:}K;K)$

7. Now $, ,\alpha{:}K \vdash \alpha \uparrow K \leftrightarrow , ,\alpha{:}K \vdash \alpha \uparrow K$,
8. so by Lemma 4.8, $(, ,\alpha{:}K;\alpha;K)$ **is** $(, ,\alpha{:}K;\alpha;K)$.

∎

This yields our completeness result for the algorithm:

Corollary 4.11 (Completeness)

1. *If* $, \vdash K_1 \equiv K_2$ *then* $(, ;K_1)$ **is** $(, ;K_2)$.
2. *If* $, \vdash A_1 \equiv A_2 : K$ *then* $(, ;A_1;K)$ **is** $(, ;A_2;K)$.
3. *If* $, \vdash K_1 \equiv K_2$ *then* $, \vdash K_1 \Leftrightarrow , \vdash K_2$.
4. *If* $, \vdash A_1 \equiv A_2 : K$ *then* $, \vdash A_1 : K \Leftrightarrow , \vdash A_2 : K$.

Proof:

1,2 By Lemma 4.10, we can apply the Fundamental Theorem with γ_1 and γ_2 being identity substitutions.

3,4 Follows directly from parts 1 and 2 and the Main Lemma.

∎

Lemma 4.12

1. *If* $,_1 \vdash A_1 \uparrow K_1 \leftrightarrow ,_1 \vdash A_1 \uparrow K_1$ *and* $,_2 \vdash A_2 \uparrow K_2 \leftrightarrow ,_2 \vdash A_2 \uparrow K_2$ *then* $,_1 \vdash A_1 \uparrow K_1 \leftrightarrow ,_2 \vdash A_2 \uparrow K_2$ *is decidable.*
2. *If* $,_1 \vdash A_1 : K_1 \Leftrightarrow ,_1 \vdash A_1 : K_1$ *and* $,_2 \vdash A_2 : K_2 \Leftrightarrow ,_2 \vdash A_2 : K_2$ *then* $,_1 \vdash A_1 : K_1 \Leftrightarrow ,_2 \vdash A_2 : K_2$ *is decidable.*
3. *If* $,_1 \vdash K_1 \Leftrightarrow ,_1 \vdash K_1$ *and* $,_2 \vdash K_2 \Leftrightarrow ,_2 \vdash K_2$ *then* $,_1 \vdash K_1 \Leftrightarrow ,_2 \vdash K_2$ *is decidable.*

Proof Sketch: By induction on the proof of the first assumption.
Roughly speaking, the algorithm does independent expansion of the two terms and compares the results. If we know that the expansion process terminates for the two terms individually, then the simultaneous expand-and-compare of both terms will also terminate (possibly earlier if the terms are inequivalent).

Corollary 4.13 (Decidability)

1. *If* $, \vdash A_1 : K$ *and* $, \vdash A_2 : K$ *then* $, \vdash A_1 : K \Leftrightarrow , \vdash A_2 : K$ *is decidable.*
2. *If* $, \vdash K_1$ *and* $, \vdash K_2$ *then* $, \vdash K_1 \Leftrightarrow , \vdash K_2$ *is decidable.*

Proof: By Corollary 4.11, comparison of each well-formed type or term with itself is decidable. by Lemma 4.12, therefore, the comparison of the two types or two terms is decidable. ∎

We conclude this section with an application of completeness.

Proposition 4.14 (Consistency)
Let b_1 *and* b_2 *be two distinct constants of kind* T. *Then the judgment "*$\vdash b_1 \equiv b_2 : T$*" is not provable.*

This inequivalence (and the inequivalence of $\lambda\alpha{:}T.\alpha$ and $\lambda\alpha{:}T.b_i$ at kind $T{\to}T$ mentioned in Section 2.2) is obvious for algorithmic equivalence, which by completeness transfers to inequivalence in the declarative system.

In proving soundness of the TILT compiler's intermediate language, these sorts of consistency properties are essential. The argument that, for example, the only closed values of type `int` are the integers would fail if the type `int` were provably to another base type.

Kind Extraction

$, \vdash b_i \uparrow T$	
$, \vdash \alpha \uparrow , (\alpha)$	
$, \vdash \pi_1 p \uparrow K_1$	if $, \vdash p \uparrow \Sigma\beta{:}K_1.K_2$
$, \vdash \pi_2 p \uparrow \{\beta{\mapsto}\pi_1 p\}K_2$	if $, \vdash p \uparrow \Sigma\beta{:}K_1.K_2$
$, \vdash pA \uparrow \{\beta{\mapsto}A\}K_2$	if $, \vdash p \uparrow \Pi\beta{:}K_1.K_2$

Weak head reduction

$, \vdash E[(\lambda\alpha{:}K.A)A'] \leadsto E[\{\alpha{\mapsto}A'\}A]$	
$, \vdash E[\pi_1\langle A_1, A_2\rangle] \leadsto E[A_1]$	
$, \vdash E[\pi_2\langle A_1, A_2\rangle] \leadsto E[A_2]$	
$, \vdash p \leadsto B$	if $, \vdash p \uparrow S(B)$

Weak head normalization

$, \vdash A \Downarrow B$	if $, \vdash A \leadsto A'$ and $, \vdash A' \Downarrow B$
$, \vdash B \Downarrow B$	otherwise

Algorithmic constructor equivalence

$, \vdash A_1 \Leftrightarrow A_2 : T$	if $, \vdash A_1 \Downarrow p_1$, $, \vdash A_2 \Downarrow p_2$, and $, \vdash p_1 \leftrightarrow p_2 \uparrow T$
$, \vdash A_1 \Leftrightarrow A_2 : S(B)$	always
$, \vdash A_1 \Leftrightarrow A_2 : \Pi\alpha{:}K'.K''$	if $, , \alpha{:}K' \vdash A_1\alpha \Leftrightarrow A_2\alpha : K''$
$, \vdash A_1 \Leftrightarrow A_2 : \Sigma\alpha{:}K'.K''$	if $, \vdash \pi_1 A_1 \Leftrightarrow \pi_1 A_2 : K'$ and $, \vdash \pi_2 A_1 \Leftrightarrow \pi_2 A_2 : \{\alpha{\mapsto}\pi_1 A_1\}K''$

Algorithmic path equivalence

$, \vdash b_i \leftrightarrow b_j \uparrow T$	if $i = j$
$, \vdash \alpha \leftrightarrow \alpha \uparrow , (\alpha)$	
$, \vdash p_1 A_1 \leftrightarrow p_2 A_2 \uparrow \{\alpha{\mapsto}A_1\}K_2$	if $, \vdash p_1 \leftrightarrow p_2 \uparrow \Pi\alpha{:}K_1.K_2$ andalso $, \vdash A_1 \Leftrightarrow A_2 : K_1$
$, \vdash \pi_1 p_1 \leftrightarrow \pi_1 p_2 \uparrow K_1$	if $, \vdash p_1 \leftrightarrow p_2 \uparrow \Sigma\alpha{:}K_1.K_2$
$, \vdash \pi_2 p_1 \leftrightarrow \pi_2 p_2 \uparrow \{\alpha{\mapsto}\pi_1 p_1\}K_2$	if $, \vdash p_1 \leftrightarrow p_2 \uparrow \Sigma\alpha{:}K_1.K_2$

Algorithmic kind equivalence

$, \vdash T \Leftrightarrow T$	always
$, \vdash S(A_1) \Leftrightarrow S(A_2)$	if $, \vdash A_1 \Leftrightarrow A_2 : T$
$, \vdash \Pi\alpha{:}K_1.L_1 \Leftrightarrow \Pi\alpha{:}K_2.L_2$	if $, \vdash K_1 \Leftrightarrow K_2$ and $, , \alpha{:}K_1 \vdash L_1 \Leftrightarrow L_2$
$, \vdash \Sigma\alpha{:}K_1.L_1 \Leftrightarrow \Sigma\alpha{:}K_2.L_2$	if $, \vdash K_1 \Leftrightarrow K_2$ and $, , \alpha{:}K_1 \vdash L_1 \Leftrightarrow L_2$

Figure 8: A Simplified Algorithm

5 A Simpler Algorithm

We have shown that constructor equivalence is decidable by presenting a sound, complete and terminating algorithm. However, as an implementation it inefficiently maintains two typing contexts and two classifying kinds. We would prefer an algorithm more like the declarative rules for equivalence, having only a single typing context and a single classifier. The revised algorithmic relations are shown in Figure 8.

The definition of this simplified algorithm is asymmetric because of essentially arbitrary choices between two provably equivalent kinds for the classifier or the typing context. Because we cannot prove directly that this simplified algorithm satisfies any symmetry or transitivity properties, we cannot simply use the same proof strategy. However, we can show the simplification is complete with respect to the previous algorithm, from which the remaining correctness properties follow easily.

One other small simplification is that in weak head reduction we need not worry about a path having a proper prefix with a definition; for well-formed constructors this can never occur. (See the proof of Corollary 3.2.)

We first define a "size" metric on derivations in the six-place algorithmic system. This metric measures the size of the derivation ignoring head reduction or head normalization steps; equivalently, we can define the metric as the number of term or path equivalence rules used in the derivation. Since every judgment has at most one derivation in the six-place system, we can refer unambiguously to the size of a provable algorithmic judgment.

The important properties of this metric are summarized in the following two lemmas.

Lemma 5.1

1. *If* $, _1 \vdash A_1 : K_1 \Leftrightarrow \, , _2 \vdash A_2 : K_2$ *and* $, _1 \vdash A_1 : K_1 \Leftrightarrow \, , _3 \vdash A_3 : K_3$ *then the two derivations have equal sizes.*

2. *If* $, _1 \vdash A_1 \uparrow K_1 \leftrightarrow \, , _2 \vdash A_2 \uparrow K_2$ *and* $, _1 \vdash A_1 \uparrow K_1 \leftrightarrow \, , _3 \vdash A_3 \uparrow K_3$ *then the two derivations have equal sizes.*

Proof: [By induction on the hypothesized derivations]

- Assume $, _1 \vdash A_1 : T \Leftrightarrow \, , _2 \vdash A_2 : T$ and $, _1 \vdash A_1 : T \Leftrightarrow \, , _3 \vdash A_3 : T$. Then $, _1 \vdash A_1 \Downarrow p_1$, $, _2 \vdash A_2 \Downarrow p_2$, $, _3 \vdash A_3 \Downarrow p_3$, $, _1 \vdash p_1 \uparrow T \leftrightarrow \, , _2 \vdash p_2 \uparrow T$, and $, _1 \vdash p_1 \uparrow T \leftrightarrow \, , _3 \vdash p_3 \uparrow T$. By the inductive hypothesis, these last two algorithmic judgments have equal sizes, so the original equivalences have equal sizes (greater by one).
- Assume $, _1 \vdash A_1 : S(B_1) \Leftrightarrow \, , _2 \vdash A_2 : S(B_2)$ and $, _1 \vdash A_1 : S(B_1) \Leftrightarrow \, , _3 \vdash A_3 : S(B_3)$. Then the derivations both have a size of one.
- Assume $, _1 \vdash A_1 : \Pi\alpha{:}A_1'.A_1'' \Leftrightarrow \, , _2 \vdash A_2 : \Pi\alpha{:}A_2'.A_2''$ and $, _1 \vdash A_1 : \Pi\alpha{:}A_1'.A_1'' \Leftrightarrow \, , _3 \vdash A_3 : \Pi\alpha{:}A_3'.A_3''$. Then $, _1, \alpha{:}K_1' \vdash A_1\,\alpha : K_1'' \Leftrightarrow \, , _2, \alpha{:}K_2' \vdash A_2\,\alpha : K_2''$ and $, _1, \alpha{:}K_1' \vdash A_1\,\alpha : K_1'' \Leftrightarrow \, , _3, \alpha{:}K_2' \vdash A_3\,\alpha : K_3''$. By the inductive hypothesis these derivations have equal sizes and hence the original equivalence judgments have equal sizes (greater by one).
- Assume $, _1 \vdash A_1 : \Sigma\alpha{:}A_1'.A_1'' \Leftrightarrow \, , _2 \vdash A_2 : \Sigma\alpha{:}A_2'.A_2''$ and $, _1 \vdash A_1 : \Sigma\alpha{:}A_1'.A_1'' \Leftrightarrow \, , _3 \vdash A_3 : \Sigma\alpha{:}A_3'.A_3''$. Then $, _1 \vdash \pi_1 A_1 : K_1' \Leftrightarrow \, , _2 \vdash \pi_1 A_2 : K_2'$, $, _1 \vdash \pi_1 A_1 : K_1' \Leftrightarrow \, , _3 \vdash \pi_1 A_3 : K_3'$,
  $, _1 \vdash \pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K_1'' \Leftrightarrow \, , _2 \vdash \pi_1 A_2 : \{\alpha{\mapsto}\pi_1 A_2\}K_2''$, and
  $, _1 \vdash \pi_2 A_1 : \{\alpha{\mapsto}\pi_1 A_1\}K_1'' \Leftrightarrow \, , _3 \vdash \pi_1 A_3 : \{\alpha{\mapsto}\pi_1 A_3\}K_3''$. By the inductive hypothesis twice, both pairs of judgments contain two derivations with equal sizes.
- Assume $, _1 \vdash b_i \uparrow T \leftrightarrow \, , _2 \vdash b_i \uparrow T$ and $, _1 \vdash b_i \uparrow T \leftrightarrow \, , _3 \vdash b_i \uparrow T$. Both derivations have size one.
- Assume $, _1 \vdash \alpha \uparrow \, , _1(\alpha) \leftrightarrow \, , _2 \vdash \alpha \uparrow \, , _2(\alpha)$ and $, _1 \vdash \alpha \uparrow \, , _1(\alpha) \leftrightarrow \, , _3 \vdash \alpha \uparrow \, , _3(\alpha)$. Both derivations have size one.
- The remaining three cases follow directly by the inductive hypothesis.

∎

Lemma 5.2

1. *If* $, _1 \vdash A_1 : K_1 \Leftrightarrow \, , _2 \vdash A_2 : K_2$ *then the derivation* $, _2 \vdash A_2 : K_2 \Leftrightarrow \, , _1 \vdash A_1 : K_1$ *has the same size.*

2. *If* $, _1 \vdash A_1 \uparrow K_1 \leftrightarrow \, , _2 \vdash A_2 \uparrow K_2$ *then the derivation* $, _2 \vdash A_2 \uparrow K_2 \leftrightarrow \, , _1 \vdash A_1 \uparrow K_1$ *has the same size.*

Proof Sketch: The two derivations are essentially mirror-images of each other, and hence use the same number of rules of each kind.

Using the metric, we can show the completeness of the four-place algorithm with respect to the six-place algorithm.

Lemma 5.3

1. *If* $\vdash \, , _1 \equiv \, , _2$, $, _1 \vdash K_1 \equiv K_2$, $, _1 \vdash A_1 : K_1$, $, _2 \vdash A_2 : K_2$, *and* $, _1 \vdash A_1 : K_1 \Leftrightarrow \, , _2 \vdash A_2 : K_2$ *then* $, _1 \vdash A_1 \Leftrightarrow A_2 : K_1$.

2. *If* $\vdash \, , _1 \equiv \, , _2$, $, _1 \vdash K_1 \equiv K_2$, $, _1 \vdash A_1 : K_1$, $, _2 \vdash A_2 : K_2$, *and* $, _1 \vdash A_1 \uparrow K_1 \leftrightarrow \, , _2 \vdash A_2 \uparrow K_2$ *then* $, _1 \vdash A_1 \leftrightarrow A_2 \uparrow K_1$.

Proof: [By induction on the size of the hypothesized algorithmic derivation.]
Assume $\vdash \, , _1 \equiv \, , _2$, $, _1 \vdash K_1 \equiv K_2$, $, _1 \vdash A_1 : K_1$, and $, _2 \vdash A_2 : K_2$.

- Case: $, _1 \vdash A_1 : T \Leftrightarrow , _2 \vdash A_2 : T$ because $, _1 \vdash A_1 \Downarrow p_1$, $, _2 \vdash A_2 \Downarrow p_2$, and $, _1 \vdash p_1 \uparrow T \leftrightarrow , _2 \vdash p_2 \uparrow T$.
  Now by the soundness and completeness of the six-place algorithm we have $, _1 \vdash A_1 : T \Leftrightarrow , _1 \vdash A_2 : T$, where $, _1 \vdash A_2 \Downarrow p_2'$ and $, _1 \vdash p_1 \uparrow T \leftrightarrow , _1 \vdash p_2' \uparrow T$.
  By Lemma 5.1, the sizes of the two proofs of algorithmic path equivalence have equal sizes. Since this size is less than the size of the original algorithmic judgment (by one), we may apply the inductive hypothesis to get $, _1 \vdash p_1 \leftrightarrow p_2' \uparrow T$. Therefore, $, _1 \vdash A_1 \Leftrightarrow A_2 : T$.
- The remaining cases are all either trivial or follow directly from the inductive hypothesis.

∎

Corollary 5.4 (Completeness)
If $, \vdash A_1 \equiv A_2 : K$ *then* $, \vdash A_1 \Leftrightarrow A_2 : K$.

Proof: Assume $, \vdash A_1 \equiv A_2 : K$. By the completeness of the six-place algorithm, $, \vdash A_1 : K \Leftrightarrow , \vdash A_2 : K$. Then $, \vdash A_1 \Leftrightarrow A_2 : K$ by Lemma 5.3. ∎

Theorem 5.5 (Soundness)
1. *If* $, \vdash A_1 : K$, $, \vdash A_2 : K$, *and* $, \vdash A_1 \Leftrightarrow A_2 : K$ *then* $, \vdash A_1 \equiv A_2 : K$.

2. *If* $, \vdash p_1 : K_1$, $, \vdash p_2 : K_2$, *and* $, \vdash p_1 \leftrightarrow p_2 \uparrow K$ *then* $, \vdash p_1 \equiv p_2 : K$.

Proof Sketch: By induction on the hypothesized derivations, exactly analogous with the soundness proof for the six-place algorithm.

Lemma 5.6
1. *If* $, \vdash A_1 \leftrightarrow A_1 \uparrow K$ *and* $, \vdash A_2 \leftrightarrow A_2 \uparrow K$ *then* $, \vdash A_1 \leftrightarrow A_2 \uparrow K$ *is decidable.*

2. *If* $, \vdash A_1 \Leftrightarrow A_1 : K$ *and* $, \vdash A_2 \Leftrightarrow A_2 : K$ *then* $, \vdash A_1 \Leftrightarrow A_2 : K$ *is decidable.*

3. *If* $, \vdash K_1 \Leftrightarrow K_1$ *and* $, \vdash K_2 \Leftrightarrow K_2$ *then* $, \vdash K_1 \Leftrightarrow K_2$ *is decidable.*

Proof: Essentially the same proof as in the original algorithm. ∎

Theorem 5.7 (Decidability)
1. *If* $, \vdash A_1 : K$ *and* $, \vdash A_2 : K$ *then* $, \vdash A_1 \Leftrightarrow A_2 : K$ *is decidable.*

2. *If* $, \vdash K_1$ *and* $, \vdash K_2$ *then* $, \vdash K_1 \Leftrightarrow K_2$ *is decidable.*

Proof: Follows from reflexivity of constructor and kind equivalence, Completeness, and Lemma 5.6. ∎

6 Related Work

Our proof was inspired by that of Coquand [3], but because the equivalence considered there was not context-sensitive in any way our algorithm and proof are substantially different. Because of the validity logical relations and the form of the symmetry and transitivity properties for logical equivalence, our initial attempts to use more traditional Kripke logical relations (with worlds being pairs of contexts) were unsuccessful.

Other researchers have considered lambda calculi with more interesting equivalences. Lillibridge [10] considered a language in which equivalence depends on the typing context. He eliminates the context-sensitivity by tagging each path with its enclosing typing context, and then gives a rewriting strategy for this tagged system. Curien and Ghelli [5] gave a proof of decidability of term equivalence in $F_{\leq}$ with $\beta\eta$-reduction and a Top type. Because their Top type is both terminal and maximal, equivalence depends on both the typing context and the type at which terms are compared. They eliminate context-sensitivity by inserting explicit coercions to mark uses of subsumption and then give a rewriting strategy for the calculus with coercions. Both Lillibridge's and Curien and Ghelli's approaches require an extra step to transfer decidability results from this system without context-sensitivity back to the original systems.

Severi and Poll [15] study confluence and normalization of $\beta\delta$-reduction for a pure type system with definitions (let bindings), where δ is the replacement of an occurrence of a variable with its definition. This calculus contains no notion of partial definitions and no subtyping.

David Aspinall [1] studied a calculus $\lambda_{\leq\{\}}$ with singleton types and β-equivalence. Labelled singletons are primitive notions in this system; in the absence of η-equivalence the encoding of Section 2.3 does not work. He conjectured that equivalence in this system was decidable. Karl Crary [4] studied an extension of $\lambda_{\leq}^{\Pi\Sigma S}$ with subtyping and power kinds and also conjectured that typechecking was decidable.

7 Conclusion and Future Work

We have confirmed that $\beta\eta$-equivalence for well-formed constructors is decidable in the presence of singleton kinds by providing a sound, complete, and terminating algorithm. This algorithm — with minor extensions such as stopping early when constructors are found to be α-equivalent — is used by the internal typechecker of the TILT compiler.

Although the pattern of our logical relations proof is fairly standard, our formulation — in particular, the equivalence relation involving two constructors, two kinds, and two worlds — appears novel, as is the extension to subkinding and singleton kinds.

We believe that our proof should generalize well to extensions of $\lambda_{\leq}^{\Pi\Sigma S}$ such as subtyping and power kinds like those found in Crary's work. The technique may be applicable to other calculi, especially those with context-sensitive equivalence.

We are currently investigating the addition of singleton *types* to the TILT compiler. These seem a promising formalized vehicle for expressing the information needed by cross-module inlining [2, 16] and modeling the structure sharing feature of original Standard ML.

8 Acknowledgements

We would like to thank Lars Birkedal for his suggestion that the form of the logical relations should mirror the form of the algorithmic relations and Karl Crary for his detailed critique of our proofs. We also thank Perry Cheng, Mark Lillibridge, Leaf Petersen, Frank Pfenning, John Reynolds, Jon Riecke, and Rick Statman for helpful discussions.

References

[1] David Aspinall. Subtyping with Singleton Types. In *Proc. Computer Science Logic (CSL'94)*, 1995. In Springer LNCS 933.

[2] Matthias Blume and Andrew W. Appel. Lambda-Splitting: A Higher-Order Approach to Cross-Module Optimizations. In *Proc. 1997 International Conference on Functional Programming (ICFP '97)*, pages 112–124, June 1997.

[3] Thierry Coquand. An Algorithm for Testing Conversion in Type Theory. In Gérard Huet and G. Plotkin, editors, *Logical frameworks*, pages 255–277. Cambridge University Press, 1991.

[4] Karl F. Crary. *Type-Theoretic Methodology for Practical Programming Languages*. PhD thesis, Department of Computer Science, Cornell University, 1998.

[5] Pierre-Louis Curien and Giorgio Ghelli. Decidability and Confluence of $\beta\eta\mathrm{top}_{\leq}$ Reduction in $F_{\leq}$. *Information and Computation*, 1/2:57–114, 1994.

[6] Robert Harper and Mark Lillibridge. A Type-Theoretic Approach to Higher-Order Modules with Sharing. In *Proc. 21st Symposium on Principles of Programming Languages*, pages 123–137, 1994.

[7] Robert Harper, John C. Mitchell, and Eugenio Moggi. Higher-order Modules and the Phase Distinction. In *17th Symposium on Principles of Programming Languages*, pages 341–354, 1990.

[8] Xavier Leroy. Manifest types, modules, and separate compilation. In *Proc. 21st Symposium on Principles of Programming Languages*, pages 109–122, 1994.

[9] Xavier Leroy. Applicative Functors and Fully Transparent Higher-Order Modules. In *Proc. 22nd Symposium on Principles of Programming Languages*, pages 142–153, 1995.

[10] Mark Lillibridge. *Translucent Sums: A Foundation for Higher-Order Module Systems.* PhD thesis, School of Computer Science, Carnegie Mellon University, 1997. Available as CMU Technical Report CMU-CS-97-122.

[11] Yasuhiko Minamide, Greg Morrisett, and Robert Harper. Typed Closure Conversion. In *Proc. 23rd Symposium on Princples of Programming Languages*, pages 271–283, 1996.

[12] Greg Morrisett. *Compiling with Types.* PhD thesis, School of Computer Science, Carnegie Mellon University, 1995. Available as CMU Technical Report CMU-CS-95-226.

[13] Greg Morrisett, David Walker, Karl Crary, and Neal Glew. From System F to Typed Assembly Language. Technical Report TR97-1651, Department of Computer Science, Cornell University, 1997.

[14] George C. Necula. Proof-Carrying Code. In *24th Symposium on Principles of Programming Languages*, pages 106–119. ACM Press, 1997.

[15] Paula Severi and Eric Poll. Pure Type Systems with definitions. In *Logical Foundations of Computer Science '94*, number 813 in LNCS, 1994.

[16] Zhong Shao. Typed Cross-Module Compilation. In *Proc. 1998 ACM SIGPLAN International Conference on Functional Programming (ICFP '98)*, pages 141–152, September 1998.

[17] David Tarditi, Greg Morrisett, Perry Cheng, Chris Stone, Robert Harper, and Peter Lee. TIL: A Type-Directed Optimizing Compiler for ML. In *Proc. ACM SIGPLAN '96 Conference on Programming Language Design and Implementation*, pages 181–192, 1996.

A Rules for $\lambda_{\leq}^{\Pi\Sigma S}$

Well-Formed Context $\boxed{, \vdash \mathbf{ok}}$

$$\frac{}{\bullet \vdash \mathrm{ok}} \qquad (1)$$

$$\frac{, \vdash K \qquad \alpha \notin \mathrm{dom}(,)}{, , \alpha{:}K \vdash \mathrm{ok}} \qquad (2)$$

Context Equivalence $\boxed{\vdash , _1 \equiv , _2}$

$$\frac{}{\vdash \bullet \equiv \bullet} \qquad (3)$$

$$\frac{\vdash , _1 \equiv , _2 \qquad , _1 \vdash K_1 \equiv K_2 \qquad \alpha \notin \mathrm{dom}(, _1)}{\vdash , _1, \alpha{:}K_1 \equiv , _2, \alpha{:}K_2} \qquad (4)$$

Well-Formed Kind $\boxed{, \vdash K}$

$$\frac{, \vdash \mathrm{ok}}{, \vdash T} \qquad (5)$$

$$\frac{, \vdash A : T}{, \vdash S(A)} \qquad (6)$$

$$\frac{, , \alpha{:}K' \vdash K''}{, \vdash \Pi\alpha{:}K'.K''} \qquad (7)$$

$$\frac{, , \alpha{:}K' \vdash K''}{, \vdash \Sigma\alpha{:}K'.K''} \qquad (8)$$

Subkinding $\boxed{, \vdash K \leq K'}$

$$\frac{, \vdash A : T}{, \vdash S(A) \leq T} \qquad (9)$$

$$\frac{, \vdash \mathrm{ok}}{, \vdash T \leq T} \qquad (10)$$

$$\frac{, \vdash A_1 \equiv A_2 : T}{, \vdash S(A_1) \leq S(A_2)} \qquad (11)$$

$$\frac{\begin{array}{c} , \vdash \Pi\alpha{:}K_1'.K_1'' \\ , \vdash K_2' \leq K_1' \qquad , , \alpha{:}K_2' \vdash K_1'' \leq K_2'' \end{array}}{, \vdash \Pi\alpha{:}K_1'.K_1'' \leq \Pi\alpha{:}K_2'.K_2''} \qquad (12)$$

$$\frac{\begin{array}{c} , \vdash \Sigma\alpha{:}K_2'.K_2'' \\ , \vdash K_1' \leq K_2' \qquad , , \alpha{:}K_1' \vdash K_1'' \leq K_2'' \end{array}}{, \vdash \Sigma\alpha{:}K_1'.K_1'' \leq \Sigma\alpha{:}K_2'.K_2''} \qquad (13)$$

Kind Equivalence $\boxed{, \vdash K_1 \equiv K_2}$

$$\frac{, \vdash \mathrm{ok}}{, \vdash T \equiv T} \tag{14}$$

$$\frac{, \vdash A_1 \equiv A_2 : T}{, \vdash S(A_1) \equiv S(A_2)} \tag{15}$$

$$\frac{, \vdash K_2' \equiv K_1' \qquad , , \alpha{:}K_1' \vdash K_1'' \equiv K_2''}{, \vdash \Pi\alpha{:}K_1'.K_1'' \equiv \Pi\alpha{:}K_2'.K_2''} \tag{16}$$

$$\frac{, \vdash K_1' \equiv K_2' \qquad , , \alpha{:}K_1' \vdash K_1'' \equiv K_2''}{, \vdash \Sigma\alpha{:}K_1'.K_1'' \equiv \Sigma\alpha{:}K_2'.K_2''} \tag{17}$$

Well-Formed Constructor $\boxed{, \vdash A : K}$

$$\frac{, \vdash \mathrm{ok}}{, \vdash b_i : T} \tag{18}$$

$$\frac{, \vdash \mathrm{ok}}{, \vdash \alpha : , (\alpha)} \tag{19}$$

$$\frac{, , \alpha{:}K' \vdash A : K''}{, \vdash \lambda\alpha{:}K'.A : \Pi\alpha{:}K'.K''} \tag{20}$$

$$\frac{, \vdash A : \Pi\alpha{:}K'.K'' \qquad , \vdash A' : K'}{, \vdash AA' : \{\alpha{\mapsto}A'\}K''} \tag{21}$$

$$\frac{, \vdash A : \Sigma\alpha{:}K'.K''}{, \vdash \pi_1 A : K'} \tag{22}$$

$$\frac{, \vdash A : \Sigma\alpha{:}K'.K''}{, \vdash \pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K'} \tag{23}$$

$$\frac{\begin{array}{c} , \vdash \Sigma\alpha{:}K'.K'' \\ , \vdash A_1 : K' \\ , \vdash A_2 : \{\alpha{\mapsto}A_1\}K'' \end{array}}{, \vdash \langle A_1, A_2\rangle : \Sigma\alpha{:}K'.K''} \tag{24}$$

$$\frac{, \vdash A : T}{, \vdash A : S(A)} \tag{25}$$

$$\frac{\begin{array}{c} , \vdash \Sigma\alpha{:}K'.K'' \\ , \vdash \pi_1 A : K' \\ , \vdash \pi_2 A : \{\alpha{\mapsto}\pi_1 A\}K'' \end{array}}{, \vdash A : \Sigma\alpha{:}K'.K''} \tag{26}$$

$$\frac{\begin{array}{c} , \vdash A : \Pi\alpha{:}K'.K_1'' \\ , , \alpha{:}K' \vdash A\alpha : K'' \end{array}}{, \vdash A : \Pi\alpha{:}K'.K''} \tag{27}$$

$$\frac{, \vdash A : K_1 \qquad , \vdash K_1 \leq K_2}{, \vdash A : K_2} \tag{28}$$

Constructor Equivalence $\boxed{,\ \vdash A \equiv A' : K}$

$$\frac{,\ ,\alpha{:}K' \vdash A_1 \equiv A_2 : K'' \qquad ,\ \vdash A_1' \equiv A_2' : K'}{,\ \vdash (\lambda\alpha{:}K'.A_1)A_1' \equiv \{\alpha{\mapsto}A_2'\}A_2 : \{\alpha{\mapsto}A_1'\}K''} \tag{29}$$

$$\frac{\begin{array}{c},\ \vdash A_1 : \Pi\alpha{:}K'.K_1'' \\ ,\ \vdash A_2 : \Pi\alpha{:}K'.K_2'' \\ ,\ ,\alpha{:}K' \vdash A_1\alpha \equiv A_2\alpha : K''\end{array}}{,\ \vdash A_1 \equiv A_2 : \Pi\alpha{:}K'.K''} \tag{30}$$

$$\frac{\begin{array}{c},\ \vdash \Sigma\alpha{:}K'.K'' \\ ,\ \vdash \pi_1A_1 \equiv \pi_1A_2 : K' \\ ,\ \vdash \pi_2A_1 \equiv \pi_2A_2 : \{\alpha{\mapsto}\pi_1A_1\}K''\end{array}}{,\ \vdash A_1 \equiv A_2 : \Sigma\alpha{:}K'.K''} \tag{31}$$

$$\frac{,\ \vdash A_1 \equiv A_1' : K_1 \qquad ,\ \vdash A_2 : K_2}{,\ \vdash \pi_1\langle A_1, A_2\rangle \equiv A_1' : K_1} \tag{32}$$

$$\frac{,\ \vdash A_1 : K_1 \qquad ,\ \vdash A_2 \equiv A_2' : K_2}{,\ \vdash \pi_2\langle A_1, A_2\rangle \equiv A_2' : K_2} \tag{33}$$

$$\frac{,\ \vdash A : S(B)}{,\ \vdash A \equiv B : T} \tag{34}$$

$$\frac{,\ \vdash A \equiv B : T}{,\ \vdash A \equiv B : S(A)} \tag{35}$$

$$\frac{,\ \vdash A' \equiv A : K}{,\ \vdash A \equiv A' : K} \tag{36}$$

$$\frac{,\ \vdash A \equiv A' : K \qquad ,\ \vdash A' \equiv A'' : K}{,\ \vdash A \equiv A'' : K} \tag{37}$$

$$\frac{,\ \vdash \text{ok}}{,\ \vdash b_i \equiv b_i : T} \tag{38}$$

$$\frac{,\ \vdash \text{ok}}{,\ \vdash \alpha \equiv \alpha : ,\ (\alpha)} \tag{39}$$

$$\frac{,\ \vdash K_1' \equiv K_2' \qquad ,\ ,\alpha{:}K_1' \vdash A_1 \equiv A_2 : K''}{,\ \vdash \lambda\alpha{:}K_1'.A_1 \equiv \lambda\alpha{:}K_2'.A_2 : \Pi\alpha{:}K'.K''} \tag{40}$$

$$\frac{,\ \vdash A \equiv A' : \Pi\alpha{:}K_1.K_2 \qquad ,\ \vdash A_1 \equiv A_1' : K_1}{,\ \vdash AA_1 \equiv A'A_1' : \{\alpha{\mapsto}A_1\}K_2} \tag{41}$$

$$\frac{,\ \vdash A_1 \equiv A_2 : \Sigma\alpha{:}K'.K''}{,\ \vdash \pi_1A_1 \equiv \pi_1A_2 : K'} \tag{42}$$

$$\frac{,\ \vdash A_1 \equiv A_2 : \Sigma\alpha{:}K'.K''}{,\ \vdash \pi_2A_1 \equiv \pi_2A_2 : \{\alpha{\mapsto}\pi_1A_1\}K''} \tag{43}$$

$$\frac{\begin{array}{c},\ \vdash \Sigma\alpha{:}K'.K'' \\ ,\ \vdash A_1' \equiv A_2' : K' \\ ,\ \vdash A_1'' \equiv A_2'' : \{\alpha{\mapsto}A_1'\}K''\end{array}}{,\ \vdash \langle A_1', A_1''\rangle \equiv \langle A_2', A_2''\rangle : \Sigma\alpha{:}K'.K''} \tag{44}$$

$$\frac{,\ \vdash A_1 \equiv A_2 : K \qquad ,\ \vdash K \le K'}{,\ \vdash A_1 \equiv A_2 : K'} \tag{45}$$

- K_1 is K_2 $[\Delta]$ iff
  1. $\Delta \vdash$ ok
  2. And,
     - $K_1 = T$ and $K_2 = T$
     - Or, $K_1 = S(A_1)$ and $K_2 = S(A_2)$ and A_1 is A_2 in T $[\Delta]$.
     - Or, $K_1 = \Pi\alpha{:}K_1'.K_1''$ and $K_2 = \Pi\alpha{:}K_2'.K_2''$ and K_1' is K_2' $[\Delta]$ and $\forall\Delta' \succeq \Delta$ if A_1 is A_2 in K_1' $[\Delta']$ then $\{\alpha\mapsto A_1\}K_1''$ is $\{\alpha\mapsto A_2\}K_2''$ $[\Delta']$
     - Or, $K_1 = \Sigma\alpha{:}K_1'.K_1''$ and $K_2 = \Sigma\alpha{:}K_2'.K_2''$ and K_1' is K_2' $[\Delta]$ and $\forall\Delta' \succeq \Delta$ if A_1 is A_2 in K_1' $[\Delta']$ then $\{\alpha\mapsto A_1\}K_1''$ is $\{\alpha\mapsto A_2\}K_2''$ $[\Delta']$
- A_1 is A_2 in K $[\Delta]$ iff
  1. $\Delta \vdash A_1 : K$
  2. And, $\Delta \vdash A_2 : K$
  3. And, $\Delta \vdash A_1 \equiv A_2 : K$
- γ_1 is γ_2 in , $[\Delta]$ iff
  1. $\Delta \vdash$ ok
  2. And, $\forall\alpha \in \mathrm{dom}(,).\ \gamma_1(,\alpha)$ is $\gamma_2(,\alpha)$ $[\Delta]$
  3. And, $\forall\alpha \in \mathrm{dom}(,).\ \gamma_1\alpha$ is $\gamma_2\alpha$ in $\gamma_1(,\alpha)$ $[\Delta]$.

Figure 9: Logical Relations for Declarative Properties

B Declarative Properties of $\lambda_{\leq}^{\Pi\Sigma S}$

To prove many of the important properties of the declarative system, we use a Kripke logical relations argument with a more standard form than that used in the main paper to prove completeness. The definition of the relations is shown in Figure 9. As in the main paper, a Kripke world Δ is a context, and worlds are ordered by the prefix ordering.

The logical relations in Figure 9 are *not* used outside this section, and should not be confused with the logical relations of Section 4. (It seems possible that the two logical relations arguments could be combined into one, as in Coquand's work, but we have decided to keep them separate for this presentation.)

Lemma B.1

1. *If* $, \vdash \mathcal{J}$ *then there is a subderivation* $, \vdash$ *ok.*
2. *If* $,_1, \alpha{:}K, ,_2 \vdash \mathcal{J}$ *then there is a subderivation* $,_1 \vdash K$.

Proof: By induction on derivations. ∎

Lemma B.2 (Reflexivity)

1. *If* $, \vdash K$ *then* $, \vdash K \equiv K$.
2. *If* $, \vdash K$ *then* $, \vdash K \leq K$.
3. *If* $, \vdash A : K$ *then* $, \vdash A \equiv A : K$.

Lemma B.3 (Weakening 1)

1. *If* $,_1, ,_3 \vdash \mathcal{J}$ *and* $,_1, ,_2, ,_3 \vdash$ *ok then* $,_1, ,_2, ,_3 \vdash \mathcal{J}$.
2. *If* $,_1, \alpha{:}K_2, ,_2 \vdash \mathcal{J}$, $,_1 \vdash K_1 \leq K_2$, *and* $,_1 \vdash K_1$ *then* $,_1, \alpha{:}K_1, ,_2 \vdash \mathcal{J}$.

Lemma B.4 (Substitution)

1. *If* $, \vdash \mathcal{J}$*,* $\Delta \vdash$ *ok, and* $(\forall\alpha \in \mathrm{dom}(,).\Delta \vdash \gamma\alpha : \gamma(,(\alpha)))$ *then* $\Delta \vdash \gamma(\mathcal{J})$*.*

2. *If* $,_1, \alpha{:}K, ,_2 \vdash \mathcal{J}$ *and* $,_1 \vdash A : K$ *then* $,_1, \{\alpha{\mapsto}A\}, _2 \vdash \{\alpha{\mapsto}A\}\mathcal{J}$

Proof:

1. By induction on derivations.
2. By Part 1.

∎

Lemma B.5
The logical relations in Figure 9 are monotone (preserved under world extension.)

Lemma B.6
If K_1 is K_2 $[\Delta]$ *then* $\Delta \vdash K_1 \equiv K_2$*,* $\Delta \vdash K_1 \leq K_2$*,* $\Delta \vdash K_2 \leq K_1$*,* $\Delta \vdash K_1$*, and* $\Delta \vdash K_2$*,*

Proof: [By induction on the size of kinds]

- Case: T is T $[\Delta]$.
  Follows by $\Delta \vdash$ ok.
- Case: $S(A_1)$ is $S(A_2)$ $[\Delta]$.
  1. Then A_1 is A_2 in T $[\Delta]$,
  2. so $\Delta \vdash A_1 \equiv A_2 : T$, $\Delta \vdash A_1 : T$, and $\Delta \vdash A_2 : T$.
  3. The desired results follow.
- Case: $\Pi\alpha{:}K_1'.K_1''$ is $\Pi\alpha{:}K_2'.K_2''$ $[\Delta]$.
  1. K_1' is K_2' $[\Delta]$,
  2. so by the inductive hypothesis $\Delta \vdash K_1' \equiv K_2'$, $\Delta \vdash K_1' \leq K_2'$, $\Delta \vdash K_2' \leq K_1'$, $\Delta \vdash K_1'$, and $\Delta \vdash K_2'$.
  3. Then $\Delta, \alpha{:}K_1' \vdash$ ok,
  4. so $\Delta, \alpha{:}K_1' \vdash \alpha \equiv \alpha : K_1'$ and $\Delta, \alpha{:}K_1' \vdash \alpha : K_1'$.
  5. Thus α is α in K_1' $[\Delta, \alpha{:}K_1']$
  6. and K_1'' is K_2'' $[\Delta, \alpha{:}K_1']$.
  7. By the inductive hypothesis, $\Delta, \alpha{:}K_1' \vdash K_1'' \equiv K_2''$, $\Delta, \alpha{:}K_1' \vdash K_1'' \leq K_2''$, $\Delta, \alpha{:}K_1' \vdash K_2'' \leq K_1''$, $\Delta, \alpha{:}K_1' \vdash K_1''$, and $\Delta, \alpha{:}K_1' \vdash K_2''$.
  8. Thus $\Delta \vdash \Pi\alpha{:}K_1'.K_1'' \equiv \Pi\alpha{:}K_2'.K_2''$, $\Delta \vdash \Pi\alpha{:}K_1'.K_1''$, and $\Delta \vdash \Pi\alpha{:}K_2'.K_2'' \leq \Pi\alpha{:}K_1'.K_1''$.
  9. By Weakening, $\Delta, \alpha{:}K_2' \vdash K_1'' \leq K_2''$ and $\Delta, \alpha{:}K_2' \vdash K_2''$.
  10. Therefore $\Delta \vdash \Pi\alpha{:}K_1'.K_1'' \leq \Pi\alpha{:}K_2'.K_2''$ and $\Delta \vdash \Pi\alpha{:}K_2'.K_2''$.
- Case: $\Sigma\alpha{:}K_1'.K_1''$ is $\Sigma\alpha{:}K_2'.K_2''$ $[\Delta]$.
  Essentially the same argument as in the Π case.

∎

Corollary B.7
If A_1 is A_2 in K_1 $[\Delta]$ *and* K_1 is K_2 $[\Delta]$ *then* A_1 is A_2 in K_2 $[\Delta]$.

Lemma B.8

1. *If* A_1 is A_2 in K $[\Delta]$ *then* A_2 is A_1 in K $[\Delta]$.

2. *If* A_1 is A_2 in K $[\Delta]$ *and* A_2 is A_3 in K $[\Delta]$ *then* A_1 is A_3 in K $[\Delta]$.

3. *If* K_1 is K_2 $[\Delta]$ *then* K_2 is K_1 $[\Delta]$.

4. *If* K_1 is K_2 $[\Delta]$ *and* K_2 is K_3 $[\Delta]$ *then* K_1 is K_3 $[\Delta]$.

5. *If* γ_1 is γ_2 in $,$ $[\Delta]$ *then* γ_2 is γ_1 in $,$ $[\Delta]$.

6. *If* γ_1 is γ_2 in , $[\Delta]$ *and* γ_2 is γ_3 in , $[\Delta]$ *then* γ_1 is γ_3 in , $[\Delta]$.

Proof:

1. By the symmetry rule for constructor equivalence.
2. By the transitivity rule for constructor equivalence.
3. By induction on the sizes of kinds.
   - Case: T is T $[\Delta]$. Trivial.
   - Case: $S(A_1)$ is $S(A_2)$ $[\Delta]$. Follows by Part 1.
   - Case: $\Pi\alpha{:}K_1'.K_1''$ is $\Pi\alpha{:}K_2'.K_2''$ $[\Delta]$.
     (a) K_1' is K_2' $[\Delta]$, so by the inductive hypothesis K_2' is K_1' $[\Delta]$.
     (b) Let $\Delta' \succeq \Delta$ be given and assume A_2 is A_1 in K_2' $[\Delta']$.
     (c) By Part 1, A_1 is A_2 in K_2' $[\Delta']$.
     (d) By Corollary B.7, A_1 is A_2 in K_1' $[\Delta']$.
     (e) Then $\{\alpha{\mapsto}A_1\}K_1''$ is $\{\alpha{\mapsto}A_2\}K_2''$ $[\Delta']$.
     (f) By the inductive hypothesis, $\{\alpha{\mapsto}A_2\}K_2''$ is $\{\alpha{\mapsto}A_1\}K_1''$ $[\Delta']$.
     (g) Therefore, $\Pi\alpha{:}K_2'.K_2''$ is $\Pi\alpha{:}K_1'.K_1''$ $[\Delta]$.
   - Case: $\Sigma\alpha{:}K_1'.K_1''$ is $\Sigma\alpha{:}K_2'.K_2''$ $[\Delta]$. Same as previous case.
4. By induction on the sizes of types.
   - Case: $K_1 = K_2 = K_3 = T$. Trivial.
   - Case: $K_1 = S(A_1)$, $K_2 = S(A_2)$, and $K_3 = S(A_3)$. Follows by Part 2.
   - Case: $K_1 = \Pi\alpha{:}K_1'.K_1''$, $K_2 = \Pi\alpha{:}K_2'.K_2''$, and $K_3 = \Pi\alpha{:}K_3'.K_3''$.
     (a) K_1' is K_2' $[\Delta]$ and K_2' is K_3' $[\Delta]$,
     (b) so by the inductive hypothesis K_1' is K_3' $[\Delta]$.
     (c) Let $\Delta' \succeq \Delta$ and assume A_1 is A_3 in K_1' $[\Delta']$.
     (d) By Parts 1 and 2, A_1 is A_1 in K_1' $[\Delta']$.
     (e) By Corollary B.7, A_1 is A_3 in K_2' $[\Delta']$.
     (f) Thus $\{\alpha{\mapsto}A_1\}K_1''$ is $\{\alpha{\mapsto}A_1\}K_2''$ $[\Delta]$
     (g) and $\{\alpha{\mapsto}A_1\}K_2''$ is $\{\alpha{\mapsto}A_3\}K_3''$ $[\Delta]$.
     (h) By the inductive hypothesis, $\{\alpha{\mapsto}A_1\}K_1''$ is $\{\alpha{\mapsto}A_3\}K_3''$ $[\Delta]$.
     (i) Therefore, K_1 is K_3 $[\Delta]$.
   - Case: $K_1 = \Sigma\alpha{:}K_1'.K_1''$, $K_2 = \Sigma\alpha{:}K_2'.K_2''$, and $K_3 = \Sigma\alpha{:}K_3'.K_3''$. Same proof as in the Π case.
5. By Parts 1 and 3 and Corollary B.7.
6. By Parts 2, 4, 5, and Corollary B.7.

∎

Theorem B.9

1. *If* γ_1 is γ_2 in , $[\Delta]$ *and* , $\vdash A_1 \equiv A_2 : K$ *then* $\gamma_1 A_1$ is $\gamma_2 A_2$ in $\gamma_1 K$ $[\Delta]$ *and* $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
2. *If* γ_1 is γ_2 in , $[\Delta]$ *and* , $\vdash A : K$ *then* $\gamma_1 A$ is $\gamma_2 A$ in $\gamma_1 K$ $[\Delta]$ *and* $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
3. *If* γ_1 is γ_2 in , $[\Delta]$ *and* , $\vdash K$ *then* $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
4. *If* γ_1 is γ_2 in , $[\Delta]$ *and* , $\vdash K_1 \leq K_2$ *then* $\gamma_1 K_1$ is $\gamma_2 K_1$ $[\Delta]$, $\gamma_1 K_2$ is $\gamma_2 K_2$ $[\Delta]$, *and* $\Delta \vdash \gamma_1 K_1 \leq \gamma_2 K_2$.
5. *If* γ_1 is γ_2 in , $[\Delta]$ *and* , $\vdash K_1 \equiv K_2$ *then* $\gamma_1 K_1$ is $\gamma_2 K_2$ $[\Delta]$.

Proof:

- Case: Rule 5. T is T $[\Delta]$ because $\Delta \vdash \mathrm{ok}$.
- Case: Rule 6.
  1. By the inductive hypothesis, $\gamma_1 A$ is $\gamma_2 A$ in T $[\Delta]$.

2. Therefore, $S(\gamma_1 A)$ is $S(\gamma_2 A)$ $[\Delta]$.

- Case: Rule 7.
  1. By Lemma B.1 there exists a strict subderivation $\Gamma \vdash K'$.
  2. By the inductive hypothesis, $\gamma_1 K'$ is $\gamma_2 K'$ $[\Delta]$.
  3. Let $\Delta' \succeq \Delta$ and assume A_1 is A_2 in $\gamma_1 K'$ $[\Delta']$.
  4. By monotonicity, $\gamma_1[\alpha\mapsto A_1]$ is $\gamma_2[\alpha\mapsto A_2]$ in $\Gamma, \alpha{:}K'$ $[\Delta']$.
  5. By the inductive hypothesis, $(\gamma_1[\alpha\mapsto A_1])K''$ is $(\gamma_2[\alpha\mapsto A_2])K''$ $[\Delta']$.
  6. That is, $\{\alpha\mapsto A_1\}(\gamma_1[\alpha\mapsto\alpha]K'')$ is $\{\alpha\mapsto A_2\}(\gamma_2[\alpha\mapsto\alpha]K'')$ $[\Delta']$.
  7. Therefore, $\gamma_1(\Pi\alpha{:}K'.K'')$ is $\gamma_2(\Pi\alpha{:}K'.K'')$ $[\Delta]$.
- Case: Rule 8. Same argument as for previous rule.
- Case: Subkinding and kind equivalence rules. Straightforward.
- Case: Constructor validity rules. Essentially the same as reflexive instances of the constructor equivalence rules.
- Case: Rule 29.
  1. As in Rule 40, $\Delta \vdash \gamma_1(\lambda\alpha{:}K'.A_1) : \gamma_1(\Pi\alpha{:}K'.K'')$
  2. and $\gamma_1(\Pi\alpha{:}K'.K'')$ is $\gamma_2(\Pi\alpha{:}K'.K'')$ $[\Delta]$.
  3. By the inductive hypothesis, $\Delta \vdash \gamma_1 A'_1 : \gamma_1 K'$.
  4. Thus $\Delta \vdash \gamma_1((\lambda\alpha{:}K'.A_1)A'_1) : \gamma_1(\{\alpha\mapsto A'_1\}K'')$.

  5. By the inductive hypothesis, $\gamma_1 A'_1$ is $\gamma_2 A'_2$ in $\gamma_1 K'$ $[\Delta]$.
  6. Thus $\gamma_1[\alpha\mapsto\gamma_1 A'_1]$ is $\gamma_2[\alpha\mapsto\gamma_2 A'_2]$ in $\Gamma, \alpha{:}K'$ $[\Delta]$.
  7. By the inductive hypothesis, $\Delta \vdash (\gamma_2[\alpha\mapsto\gamma_2 A'_2])A_2 : (\gamma_1[\alpha\mapsto\gamma_1 A'_1])K''$.

  8. $\gamma_1[\alpha\mapsto\alpha]$ is $\gamma_2[\alpha\mapsto\alpha]$ in $\Gamma, \alpha{:}K'$ $[\Delta, \alpha{:}\gamma_1 K']$.
  9. By the inductive hypothesis, $\gamma_1[\alpha\mapsto\alpha]A_1$ is $\gamma_2[\alpha\mapsto\alpha]A_2$ in $\gamma_1[\alpha\mapsto\alpha]K''$ $[\Delta, \alpha{:}\gamma_1 K']$.
  10. Thus $\Delta \vdash \gamma_1((\lambda\alpha{:}K'.A_1)A'_1) \equiv \gamma_2(\{\alpha\mapsto A'_2\}A_2) : \gamma_1(\{\alpha\mapsto A'_1\}K'')$.

  11. Finally, $\gamma_1 A'_1$ is $\gamma_2 A'_2$ in $\gamma_1 K'$ $[\Delta]$
  12. so $\gamma_1(\{\alpha\mapsto A'_1\}K'')$ is $\gamma_2(\{\alpha\mapsto A'_2\}K'')$ $[\Delta]$.
- Case: Rule 30.
  1. As in the argument for Rule 7, $\gamma_1(\Pi\alpha{:}K'.K'')$ is $\gamma_2(\Pi\alpha{:}K'.K'')$ $[\Delta]$.
  2. In particular, by Lemma B.1 there is a strict subderivation $\Gamma \vdash K'$.
  3. By the inductive hypothesis $\gamma_1 K'$ is $\gamma_2 K'$ $[\Delta]$.
  4. Also using the inductive hypothesis and Lemma B.6, $\Delta \vdash \gamma_1 A_1 : \gamma_1(\Pi\alpha{:}K'.K''_1)$
  5. and $\Delta \vdash \gamma_2 A_1 : \gamma_1(\Pi\alpha{:}K'.K''_1)$.

  6. By Lemma B.6, $\Delta \vdash \gamma_1 K'$, so $\Delta, \alpha{:}\gamma_1 K' \vdash \text{ok}$.
  7. By monotonicity, $\gamma_1[\alpha\mapsto\alpha]$ is $\gamma_2[\alpha\mapsto\alpha]$ in $\Gamma, \alpha{:}K'$ $[\Delta, \alpha{:}\gamma_1 K']$.
  8. By the inductive hypothesis, $(\gamma_1 K_1)\alpha$ is $(\gamma_2 K_2)\alpha$ in $\gamma_1[\alpha\mapsto\alpha]K''$ $[\Delta, \alpha{:}\gamma_1 K']$
  9. Thus $\Delta, \alpha{:}\gamma_1 K' \vdash (\gamma_1 A_1)\alpha \equiv (\gamma_2 A_2)\alpha : \gamma_1[\alpha\mapsto\alpha]K''$,
  10. $\Delta, \alpha{:}\gamma_1 K' \vdash (\gamma_1 A_1)\alpha : \gamma_1[\alpha\mapsto\alpha]K''$,
  11. and $\Delta, \alpha{:}\gamma_1 K' \vdash (\gamma_2 A_2)\alpha : \gamma_1[\alpha\mapsto\alpha]K''$.
  12. Thus $\Delta \vdash \gamma_1 A_1 : \gamma_1(\Pi\alpha{:}K'.K'')$,
  13. $\Delta \vdash \gamma_2 A_2 : \gamma_1(\Pi\alpha{:}K'.K'')$,
  14. and $\Delta \vdash \gamma_1 A_1 \equiv \gamma_2 A_2 : \gamma_1(\Pi\alpha{:}K'.K'')$.
  15. Therefore $\gamma_1 A_1$ is $\gamma_2 A_2$ in $\gamma_1(\Pi\alpha{:}K'.K'')$ $[\Delta]$

- Case: Rule 31.
  1. By the inductive hypothesis, $\gamma_1(\Sigma\alpha{:}K'.K'')$ is $\gamma_2(\Sigma\alpha{:}K'.K'')$ $[\Delta]$,
  2. $\pi_1(\gamma_1 A_1)$ is $\pi_1(\gamma_2 A_2)$ in $\gamma_1 K'$ $[\Delta]$,
  3. and $\pi_2(\gamma_1 A_1)$ is $\pi_2(\gamma_2 A_2)$ in $\gamma_1(\{\alpha{\mapsto}\pi_1 A_1\}K'')$ $[\Delta]$.
  4. Thus $\Delta \vdash \gamma_1 A_1 \equiv \gamma_2 A_2 : \gamma_1(\Sigma\alpha{:}K'.K'')$,
  5. and $\Delta \vdash \gamma_1 A_1 : \gamma_1(\Sigma\alpha{:}K'.K'')$.
  6. Also, $\gamma_1(\{\alpha{\mapsto}\pi_1 A_1\}K'')$ is $\gamma_2(\{\alpha{\mapsto}\pi_1 A_2\}K'')$ $[\Delta]$,
  7. so by subsumption $\Delta \vdash \gamma_2(\pi_2 A_2) : \gamma_2(\{\alpha{\mapsto}\pi_1 A_2\}K'')$.
  8. Similarly, $\gamma_1 K'$ is $\gamma_2 K'$ $[\Delta]$
  9. so by subsumption $\Delta \vdash \gamma_2(\pi_1 A_2) : \gamma_2 K' \Delta$.
  10. Then $\Delta \vdash \gamma_2 A_2 : \gamma_2(\Sigma\alpha{:}K'.K'')$.
  11. and by subsumption, $\Delta \vdash \gamma_2 A_2 : \gamma_1(\Sigma\alpha{:}K'.K'')$.
- Case: Rule 32.
  1. By the inductive hypotheses, $\gamma_1 A_1$ is $\gamma_2 A_1'$ in $\gamma_1 K_1$ $[\Delta]$,
  2. $\gamma_1 K_1$ is $\gamma_2 K_1$ $[\Delta]$,
  3. and $\gamma_1 A_2$ is $\gamma_2 A_2$ in $\gamma_1 K_2$ $[\Delta]$.
  4. Thus $\Delta \vdash \gamma_1 A_1 \equiv \gamma_2 A_1' : \gamma_1 K_1$,
  5. $\Delta \vdash \gamma_1 A_1 : \gamma_1 K_1$,
  6. $\Delta \vdash \gamma_2 A_1' : \gamma_1 K_1$,
  7. and $\Delta \vdash \gamma_1 A_2 : \gamma_1 K_2$.
  8. Then $\Delta \vdash \gamma_1\langle A_1, A_2\rangle : \gamma_1(K_1{\times}K_2)$,
  9. so $\Delta \vdash \gamma_1(\pi_1\langle A_1, A_2\rangle) : \gamma_1 K_1$.
  10. Also, $\Delta \vdash \gamma_1(\pi_1\langle A_1, A_2\rangle) \equiv \gamma_2 A_2 : \gamma_1 K_1$,
  11. so $\gamma_1(\pi_1\langle A_1, A_2\rangle)$ is $\gamma_2 A_2$ in $\gamma_1 K_1$ $[\Delta]$.
- Case: Rule 33. Similar proof as in previous case.
- Case: Rule 34.
  1. By the inductive hypothesis $\Delta \vdash \gamma_1 A \equiv \gamma_2 A : S(\gamma_1 B)$,
  2. $\Delta \vdash \gamma_1 A : S(\gamma_1 B)$,
  3. $\Delta \vdash \gamma_2 A : S(\gamma_1 B)$,
  4. and $S(\gamma_1 B)$ is $S(\gamma_2 B)$ $[\Delta]$.
  5. Thus $\Delta \vdash \gamma_1 B \equiv \gamma_2 B : T$
  6. and $\Delta \vdash \gamma_2 B : T$.
  7. $\Delta \vdash \gamma_1 A \equiv \gamma_1 B : T$.
  8. By transitivity, $\Delta \vdash \gamma_1 A \equiv \gamma_2 B : T$.
  9. By subsumption $\Delta \vdash \gamma_1 A : T$.
  10. Finally, T is T $[\Delta]$.
- Case: Rule 35.
  1. By the inductive hypothesis $\gamma_1 A$ is $\gamma_2 B$ in T $[\Delta]$
  2. so $\Delta \vdash \gamma_1 A \equiv \gamma_2 B : T$,
  3. $\Delta \vdash \gamma_1 A : T$,
  4. and $\Delta \vdash \gamma_2 B : T$.
  5. Then $\Delta \vdash \gamma_1 A \equiv \gamma_2 B : S(\gamma_1 A)$,
  6. $\Delta \vdash \gamma_1 A : S(\gamma_1 A)$,

7. and $\Delta \vdash \gamma_2 B : S(\gamma_2 B)$.
8. But $\Delta \vdash S(\gamma_2 B) \leq S(\gamma_1 A)$
9. so by subsumption $\Delta \vdash \gamma_2 B : S(\gamma_1 A)$.
10. Finally, by the IH and transitivity and symmetry, $\gamma_1 A$ is $\gamma_2 A$ in T $[\Delta]$
11. so $S(\gamma_1 A)$ is $S(\gamma_2 A)$ $[\Delta]$.

- Case: Rule 36.
  1. γ_2 is γ_1 in , $[\Delta]$.
  2. By the inductive hypothesis, $\gamma_2 A'$ is $\gamma_1 A$ in $\gamma_2 K$ $[\Delta]$
  3. and $\gamma_2 K$ is $\gamma_1 K$ $[\Delta]$.
  4. Thus $\Delta \vdash \gamma_2 K \equiv \gamma_1 K$.
  5. By Corollary B.7 and symmetry, $\gamma_1 A$ is $\gamma_2 A'$ in $\gamma_1 K$ $[\Delta]$.
  6. By symmetry, $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
- Case: Rule 37.
  1. γ_1 is γ_1 in , $[\Delta]$.
  2. By the inductive hypothesis, $\gamma_1 A$ is $\gamma_1 A'$ in $\gamma_1 K$ $[\Delta]$,
  3. $\gamma_1 A'$ is $\gamma_2 A''$ in $\gamma_1 K$ $[\Delta]$,
  4. and $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
  5. By transitivity, $\gamma_1 A$ is $\gamma_2 A''$ in $\gamma_1 K$ $[\Delta]$.
- Case: Rule 38.
  1. Since $\Delta \vdash \mathrm{ok}$, we have $\Delta \vdash b \equiv b : T$ and $\Delta \vdash b : T$.
  2. Also, T is T $[\Delta]$.
- Case: Rule 39. By assumption.
- Case: Rule 40.
  1. By the inductive hypothesis, $\gamma_1 K'$ is $\gamma_2 K'$ $[\Delta]$.
  2. As in the proof for Rule 7, we have $\gamma_1(\Pi\alpha{:}K'.K'')$ is $\gamma_2(\Pi\alpha{:}K'.K'')$ $[\Delta]$.
  3. Now $\gamma_1[\alpha\mapsto\alpha]$ is $\gamma_2[\alpha\mapsto\alpha]$ in , $,\alpha{:}K'$ $[\Delta,\alpha{:}\gamma_1 K']$.
  4. By the inductive hypothesis, $(\gamma_1[\alpha\mapsto\alpha])A_1$ is $(\gamma_2[\alpha\mapsto\alpha])A_2$ in $(\gamma_1[\alpha\mapsto\alpha])K''$ $[\Delta,\alpha{:}\gamma_1 K']$.
  5. so $\Delta,\alpha{:}\gamma_1 K' \vdash (\gamma_1[\alpha\mapsto\alpha])A_1 \equiv (\gamma_2[\alpha\mapsto\alpha])A_2 : (\gamma_1[\alpha\mapsto\alpha])K''$,
  6. and $\Delta,\alpha{:}\gamma_1 K' \vdash (\gamma_1[\alpha\mapsto\alpha])A_1 : (\gamma_1[\alpha\mapsto\alpha])K''$.
  7. Thus $\Delta \vdash \lambda\alpha{:}\gamma_1 K'.(\gamma_1[\alpha\mapsto\alpha])A_1 \equiv \lambda\alpha{:}\gamma_2 K'.(\gamma_2[\alpha\mapsto\alpha])A_2 : \gamma_1(\Pi\alpha{:}K'.K'')$
  8. and $\Delta \vdash \lambda\alpha{:}\gamma_1 K'.(\gamma_1[\alpha\mapsto\alpha])A_1 : \gamma_1(\Pi\alpha{:}K'.K'')$.
  9. Similarly, $\gamma_2[\alpha\mapsto\alpha]$ is $\gamma_1[\alpha\mapsto\alpha]$ in , $,\alpha{:}K'$ $[\Delta,\alpha{:}\gamma_2 K']$
  10. So by the inductive hypothesis $\Delta,\alpha{:}\gamma_2 K' \vdash \gamma_2[\alpha\mapsto\alpha]A_2 : \gamma_2[\alpha\mapsto\alpha]K''$.
  11. Then $\Delta \vdash \lambda\alpha{:}\gamma_2 K'.(\gamma_2[\alpha\mapsto\alpha])A_2 : \gamma_2(\Pi\alpha{:}K'.K'')$
  12. and by subsumption $\Delta \vdash \lambda\alpha{:}\gamma_2 K'.(\gamma_2[\alpha\mapsto\alpha])A_2 : \gamma_1(\Pi\alpha{:}K'.K'')$.
- Case: Rule 41.
  1. By the inductive hypothesis $\gamma_1 A$ is $\gamma_2 A'$ in $\gamma_1(\Pi\alpha{:}K_1.K_2)$ $[\Delta]$,
  2. $\gamma_1 A_1$ is $\gamma_2 A_1'$ in $\gamma_1 K_1$ $[\Delta]$,
  3. and $\gamma_1(\Pi\alpha{:}K_1.K_2)$ is $\gamma_2(\Pi\alpha{:}K_1.K_2)$ $[\Delta]$.
  4. Thus $\Delta \vdash \gamma_1(AA_1) \equiv \gamma_2(A'A_1') : \gamma_1(\{\alpha\mapsto A_1\}K_2)$.
  5. $\Delta \vdash \gamma_1(AA_1) : \gamma_1(\{\alpha\mapsto A_1\}K_2)$,
  6. $\Delta \vdash \gamma_2(A'A_1') : \{\alpha\mapsto\gamma_2 A_1'\}(\gamma_1[\alpha\mapsto\alpha]K_2)$.

7. But $\gamma_2 A'_1$ is $\gamma_2 A'_1$ in $\gamma_2 K_1$ $[\Delta]$
8. so $\{\alpha{\mapsto}\gamma_2 A'_1\}(\gamma_1[\alpha{\mapsto}\alpha]K_2)$ is $\gamma_2(\{\alpha{\mapsto}A'_1\}K_2)$ $[\Delta]$,
9. and by subsumption $\Delta \vdash \gamma_2(A'A'_1) : \gamma_1(\{\alpha{\mapsto}A_1\}K_2)$.
10. Finally, $\gamma_1(\{\alpha{\mapsto}A_1\}K_2)$ is $\gamma_2(\{\alpha{\mapsto}A_1\}K_2)$ $[\Delta]$.

- Rule 42.
  1. By the inductive hypothesis, $\gamma_1 A_1$ is $\gamma_2 A_2$ in $\gamma_1(\Sigma\alpha{:}K'.K'')$ $[\Delta]$
  2. and $\gamma_1(\Sigma\alpha{:}K'.K'')$ is $\gamma_2(\Sigma\alpha{:}K'.K'')$ $[\Delta]$.
  3. Thus $\Delta \vdash \gamma_1(\pi_1 A_1) \equiv \gamma_2(\pi_1 A_2) : \gamma_1 K'$,
  4. $\Delta \vdash \gamma_1(\pi_1 A_1) : \gamma_1 K'$,
  5. $\Delta \vdash \gamma_2(\pi_1 A_2) : \gamma_1 K'$,
  6. and $\gamma_1 K'$ is $\gamma_2 K'$ $[\Delta]$.
- Rule 43
  1. As in the previous case, $\gamma_1(\Sigma\alpha{:}K'.K'')$ is $\gamma_2(\Sigma\alpha{:}K'.K'')$ $[\Delta]$,
  2. $\gamma_1 A_1$ is $\gamma_2 A_2$ in $\gamma_1(\Sigma\alpha{:}K'.K'')$ $[\Delta]$,
  3. and $\gamma_1(\pi_1 A_1)$ is $\gamma_2(\pi_1 A_2)$ in $\gamma_1 K'$ $[\Delta]$.
  4. and $\gamma_1(\Sigma\alpha{:}K'.K'')$ is $\gamma_2(\Sigma\alpha{:}K'.K'')$ $[\Delta]$.
  5. Also, $\Delta \vdash \gamma_1(\pi_2 A_1) \equiv \gamma_2(\pi_2 A_2) : \gamma_1(\{\alpha{\mapsto}\pi_1 A_1\}K')$,
  6. $\Delta \vdash \gamma_1(\pi_2 A_1) : \gamma_1(\{\alpha{\mapsto}\pi_1 A_1\}K'')$,
  7. and $\Delta \vdash \gamma_2(\pi_2 A_2) : \{\alpha{\mapsto}\gamma_2(\pi_1 A_1)\}(\gamma_1[\alpha{\mapsto}\alpha]K'')$.
  8. But $\gamma_1(\Sigma\alpha{:}K'.K'')$ is $\gamma_1(\Sigma\alpha{:}K'.K'')$ $[\Delta]$,
  9. so $\{\alpha{\mapsto}\gamma_1(\pi_1 A_1)\}(\gamma_1[\alpha{\mapsto}\alpha]A_2)$ is $\{\alpha{\mapsto}\gamma_2(\pi_1 A_1)\}(\gamma_1[\alpha{\mapsto}\alpha]A_2)$ $[\Delta]$.
  10. By subsumption, then, $\Delta \vdash \gamma_2(\pi_2 A_2) : \gamma_2(\{\alpha{\mapsto}\pi_1 A_1\}K'')$
- Case: Rule 44. Follows easily by the inductive hypotheses.
- Case: Rule 45. Follows easily by the inductive hypotheses.

∎

Lemma B.10
If $, \vdash$ *ok,* $\Delta \vdash$ *ok, and* $(\forall\alpha \in \mathrm{dom}(,).\Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,\alpha))$ *then* γ_1 is γ_2 in $,$ $[\Delta]$.

Proof: By induction on $, \vdash$ ok.

- Case: Rule 1. Follows by $\Delta \vdash$ ok; the other conditions are vacuously true.
- Case: Rule 2.
  1. By Lemma B.1 there is a strict subderivation $, \vdash$ ok.
  2. By the inductive hypothesis, γ_1 is γ_2 in $,$ $[\Delta]$.
  3. By Theorem B.9, $\gamma_1 K$ is $\gamma_2 K$ $[\Delta]$.
  4. Therefore, γ_1 is γ_2 in $, , \alpha{:}K$ $[\Delta]$.

∎

Corollary B.11 (Functionality)

1. *If* $, \vdash K$ *and* $\Delta \vdash$ *ok and* $(\forall\alpha \in \mathrm{dom}(,).\ \Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,(\alpha)))$ *then* $\Delta \vdash \gamma_1 K \equiv \gamma_2 K$.
2. *If* $, \vdash K_1 \equiv K_2$ *and* $\Delta \vdash$ *ok and* $(\forall\alpha \in \mathrm{dom}(,).\ \Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,(\alpha)))$ *then* $\Delta \vdash \gamma_1 K_1 \equiv \gamma_2 K_2$.
3. *If* $, \vdash K_1 \le K_2$ *and* $\Delta \vdash$ *ok and* $(\forall\alpha \in \mathrm{dom}(,).\ \Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,(\alpha)))$ *then* $\Delta \vdash \gamma_1 K_1 \le \gamma_2 K_2$.
4. *If* $, \vdash A : K$ *and* $\Delta \vdash$ *ok and* $(\forall\alpha \in \mathrm{dom}(,).\ \Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,(\alpha)))$ *then* $\Delta \vdash \gamma_1 A \equiv \gamma_2 A : \gamma_1 K$.

5. *If* $, \vdash A_1 \equiv A_2 : K$ *and* $\Delta \vdash ok$ *and* $(\forall \alpha \in \mathrm{dom}(,).\ \Delta \vdash \gamma_1\alpha \equiv \gamma_2\alpha : \gamma_1(,(\alpha)))$ *then* $\Delta \vdash \gamma_1 A_1 \equiv \gamma_2 A_2 : \gamma_1 K$.

Corollary B.12 (Validity)

1. *If* $, \vdash A_1 \equiv A_2 : K$ *then* $, \vdash A_1 : K$, $, \vdash A_2 : K$, *and* $, \vdash K$.
2. *If* $, \vdash A : K$ *then* $, \vdash K$.
3. *If* $, \vdash K_1 \leq K_2$ *then* $, \vdash K_1$ *and* $, \vdash K_2$.
4. *If* $, \vdash K_1 \equiv K_2$ *then* $, \vdash K_1$ *and* $, \vdash K_2$.

Corollary B.13 (Weakening 2)

1. *If* $,_1, \alpha{:}K_2, ,_2 \vdash \mathcal{J}$ *and* $,_1 \vdash K_1 \leq K_2$ *then* $,_1, \alpha{:}K_1, ,_2 \vdash \mathcal{J}$.
2. *If* $, \vdash \mathcal{J}$ *and* $\vdash , \equiv ,'$ *then* $,' \vdash \mathcal{J}$.

Corollary B.14
Kind equivalence is symmetric, transitive, and reflexive on well-formed types, while subkinding is transitive and reflexive on well-formed kinds.

Corollary B.15
If $, \vdash K_1 \equiv K_2$ *then* $, \vdash K_1 \leq K_2$ *and* $, \vdash K_2 \leq K_1$.

Proposition B.16
If $, \vdash \lambda\alpha{:}K'.A : L$ *then* $,, \alpha{:}K' \vdash A : K''$.

Proof: By induction on derivations. For proofs ending with Rule 20 the desired result is given directly; for Rules 27 and 28, the result follows directly by the inductive hypothesis. ∎

Lemma B.17

1. *If* $, \vdash E[A] : L$ *then there is a subderivation of the form* $, \vdash A : K$.
2. *If* $, \vdash E[AA'] : L$ *then there exists a kind* $\Pi\alpha{:}K'.K''$ *such that* $, \vdash A : \Pi\alpha{:}K'.K''$ *and* $, \vdash A' : K'$.

Proof:

1. By induction on typing derivations. If $E = \diamond$ then the result follows trivially; otherwise, the result follows by the inductive hypothesis.
2. By induction on typing derivations. If $E = \diamond$ and the proof concludes with a use of the application rule then the result follows by inversion; in all other cases, the result follows by the inductive hypothesis.

∎