The Coherence of Languages with Intersection Types

John C. Reynolds*
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA
john.reynolds@cs.cmu.edu

Abstract

When a programming language has a sufficiently rich type structure, there can be more than one proof of the same typing judgement; potentially this can lead to semantic ambiguity since the semantics of a typed language is a function of such proofs. When no such ambiguity arises, we say that the language is *coherent*. In this paper we prove the coherence of a class of lambda-calculus-based languages that use the intersection type discipline, including both a purely functional programming language and the Algol-like programming language Forsythe.

1 Introduction

As the type structure of programming languages has become richer, the problem of avoiding semantic ambiguity has become more difficult. In [5], this problem was resolved for first-order languages with implicit conversions (i.e. subtypes) and generic operators. In the present paper, we consider the more general case of higher-order languages with implicit conversions and intersection types [2]. For such languages, we will find that the usual kind of semantic equations give meaning, not to phrases of the language itself, but rather to proofs of typing judgements about such phrases. Thus to insure against semantic ambiguity, we must prove that distinct proofs of the same typing judgement possess the same meaning. Following [1] (which treats a closely

*This research was sponsored in part by National Science Foundation Grant CCR-8922109 and in part by the Avionics Lab, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, OH 45433-6543 under Contract F33615-90-C-1465, Arpa Order No. 7597. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

related problem for a language with implicit conversions and polymorphism), we call this property *coherence*.

The kind of language for which we will prove coherence is a lambda calculus extended to include object manipulation, fixed points, and conditionals. More precisely, our results hold for a class of such languages obtained by varying such entities as primitive types, constants, and semantic categories, subject to appropriate constraints.

Two particular cases of importance are a purely functional language (albeit without recursively defined types), and a syntactically desugared version of the Algol-like language Forsythe [7]. Certain aspects of the general treatment, notably the inclusion of operations for object construction, field selection, merging, and conditional branching, are specifically motivated by the desire to encompass Forsythe.

Our results depend upon some basic properties of intersection types (also called conjunctive types) that are proved in [6] and summarized in the following section.

2 Types and Their Semantics

We consider the set of types defined by the abstract grammar

$$\theta ::= \rho \mid \theta \to \theta \mid \iota\!:\!\theta \mid \mathbf{ns} \mid \theta \,\&\, \theta$$

where the metavariable ι ranges over identifiers, and the metavariable ρ ranges over primitive types. Here $\to$ is the usual operator for constructing types of procedures (or, in an applicative language, functions), & is the binary intersection operator, and **ns** (called ω in [2]) denotes the intersection of zero types. The construction $\iota\!:\!\theta$ is meant to denote a type that is isomorphic to θ, but (in an intuitive sense) disjoint from θ. (In programming terminology, one can regard $\iota\!:\!\theta$ as the type of objects or records with a single field named ι ranging over the type θ.)

We write $\theta \le \theta'$ to indicate that θ is a subtype of θ'. The subtype relation $\le$ is the least preorder on the set of types such that

$$\theta \le \mathbf{ns} \qquad \theta_1 \,\&\, \theta_2 \le \theta_1 \qquad \theta_1 \,\&\, \theta_2 \le \theta_2$$

$$\text{If } \theta \le \theta_1 \text{ and } \theta \le \theta_2 \text{ then } \theta \le \theta_1 \,\&\, \theta_2 \,.$$

$$\text{If } \rho \le_{\text{prim}} \rho' \text{ then } \rho \le \rho' \,. \qquad \text{If } \theta \le \theta' \text{ then } \iota\!:\!\theta \le \iota\!:\!\theta' \,.$$

$$\text{If } \theta_1' \le \theta_1 \text{ and } \theta_2 \le \theta_2' \text{ then } \theta_1 \to \theta_2 \le \theta_1' \to \theta_2' \,.$$

$$\iota\!:\!\theta_1 \,\&\, \iota\!:\!\theta_2 \le \iota\!:\!(\theta_1 \,\&\, \theta_2)$$

$$(\theta \to \theta_1) \,\&\, (\theta \to \theta_2) \le \theta \to (\theta_1 \,\&\, \theta_2)$$

$$\mathbf{ns} \le \iota\!:\!\mathbf{ns} \qquad \mathbf{ns} \le \theta \to \mathbf{ns}$$

where $\le_{\text{prim}}$ is a given subtype preorder for primitive types. We write $\theta \simeq \theta'$ when θ and θ' are equivalent, i.e. when $\theta \le \theta'$ and $\theta' \le \theta$.

The first four of the above relationships establish that **ns** is a greatest type and that $\theta_1 \,\&\, \theta_2$ is a greatest lower bound of θ_1 and θ_2. Note that we say "a" rather than "the"; since we have a preorder rather than a partial order, greatest types and greatest lower bounds are unique only to within equivalence.

The specific choice of the primitive types and their subtype relation is not relevant to the coherence property. However, we will impose two requirements: First, the preordered set of primitive types must be pairwise bounded complete, i.e. every pair of primitive types with an upper bound must have a least upper bound. Second (so that we can introduce conditionals), there must be a particular primitive type **bool**.

Under the first of these assumptions, it is shown in [6] that

Proposition 1 *If $\theta_1 \,\&\, \theta_2 \leq \theta'$ then there are θ'_1 and θ'_2 such that $\theta_1 \leq \theta'_1$, $\theta_2 \leq \theta'_2$, and $\theta' \simeq \theta'_1 \,\&\, \theta'_2$.*

Proposition 2 *Every pair of types has a least upper bound. In particular, there is a function $\sqcup$ mapping pairs of types into types such that $\theta_1 \sqcup \theta_2$ is a least upper bound of θ_1 and θ_2, and*

$$\theta \sqcup \mathbf{ns} \simeq \mathbf{ns}$$

$$\theta_1 \sqcup (\theta_2 \,\&\, \theta_3) \simeq (\theta_1 \sqcup \theta_2) \,\&\, (\theta_1 \sqcup \theta_3)$$

$$\rho \sqcup \iota\colon\theta \simeq \mathbf{ns}$$

$$\rho \sqcup (\theta_1 \to \theta_2) \simeq \mathbf{ns}$$

$$\iota\colon\theta_1 \sqcup (\theta_2 \to \theta_3) \simeq \mathbf{ns}$$

$$\rho_1 \sqcup \rho_2 \simeq \rho_1 \sqcup_{\text{prim}} \rho_2 \text{ when } \rho_1 \sqcup_{\text{prim}} \rho_2 \text{ exists}$$

$$\rho_1 \sqcup \rho_2 \simeq \mathbf{ns} \text{ when } \rho_1 \sqcup_{\text{prim}} \rho_2 \text{ does not exist}$$

$$\iota\colon\theta_1 \sqcup \iota\colon\theta_2 \simeq \iota\colon(\theta_1 \sqcup \theta_2)$$

$$\iota_1\colon\theta_1 \sqcup \iota_2\colon\theta_2 \simeq \mathbf{ns} \text{ when } \iota_1 \neq \iota_2$$

$$(\theta_1 \to \theta'_1) \sqcup (\theta_2 \to \theta'_2) \simeq (\theta_1 \,\&\, \theta_2) \to (\theta'_1 \sqcup \theta'_2)\ .$$

(As with greatest lower bounds, least upper bounds are unique only to within an equivalence. Note, by the way, that the function $\sqcup$, unlike &, is not a type constructor.)

In proving coherence, we will assume that the semantics of our language is based upon a category $\mathcal{K}$ with the following properties:

1. The category $\mathcal{K}$ is Cartesian closed.
2. There is a subcategory $\mathcal{S}$ of $\mathcal{K}$ such that every finite diagram in $\mathcal{S}$ has a limit in $\mathcal{S}$ that is also a limit in $\mathcal{K}$. (In particular, we assume that $\mathcal{S}$ and $\mathcal{K}$ have the same distinguished product.)

3. There is a functor $\underset{\mathcal{S}}{\Longrightarrow}$ from $\mathcal{S}^{\mathrm{op}} \times \mathcal{S}$ to $\mathcal{S}$ that is a restriction of the exponentiation functor $\underset{\mathcal{K}}{\Longrightarrow}$ from $\mathcal{K}^{\mathrm{op}} \times \mathcal{K}$ to $\mathcal{K}$. (This does not imply that $\mathcal{S}$ is Cartesian closed, since it need not contain application and abstraction morphisms.) For each object k of $\mathcal{S}$, the functor $k \underset{\mathcal{S}}{\Longrightarrow} -$ preserves limits in $\mathcal{S}$.

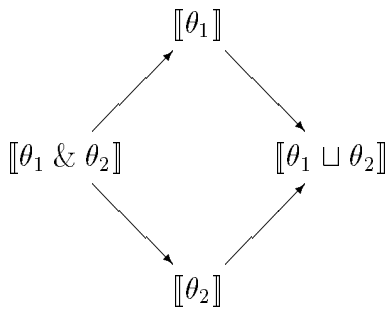
4. There is a functor $[\![-]\!]_{\mathrm{prim}}$ from the preordered set of primitive types (viewed as a category) to $\mathcal{S}$ that assigns meanings to the primitive types ρ and implicit conversions to the morphisms $\rho \leq_{\mathrm{prim}} \rho'$ of the primitive subtype relation.

For a purely functional language, for example, $\mathcal{K}$ is the category CPO whose objects are complete partial orders with least elements and whose morphisms are continuous functions, and $\mathcal{S}$ is the subcategory whose morphisms are strict. For Forsythe, on the other hand, both $\mathcal{K}$ and $\mathcal{S}$ are subcategories of the functor category CPO^{Σ} used by F. Oles to describe the semantics of Algol-like languages [3, 4]. Specifically, $\mathcal{K}$ is the full subcategory whose objects are functors that map morphisms into strict functions, $\mathcal{S}$ is the further subcategory whose morphisms are natural transformations that map objects into strict functions, and Σ is an appropriate category of "store shapes".

In [6], it is shown from the above general properties of $\mathcal{K}$ that one can define a semantic functor $[\![-]\!]$ characterized as follows:

Proposition 3 *There is a "semantic" functor $[\![-]\!]$ from the preordered set of types (viewed as a category) to $\mathcal{S}$ that satisfies*

1. *$[\![-]\!]$ is an extension of $[\![-]\!]_{\mathrm{prim}}$.*
2. *$[\![\theta_1 \to \theta_2]\!] = [\![\theta_1]\!] \underset{\mathcal{S}}{\Longrightarrow} [\![\theta_2]\!]$ and $[\![\theta_1 \to \theta_2 \leq \theta_1' \to \theta_2']\!] = [\![\theta_1' \leq \theta_1]\!] \underset{\mathcal{S}}{\Longrightarrow} [\![\theta_2 \leq \theta_2']\!]$.*
3. *$[\![\iota\!:\!\theta]\!] = [\![\theta]\!]$ and $[\![\iota\!:\!\theta \leq \iota\!:\!\theta']\!] = [\![\theta \leq \theta']\!]$.*
4. *$[\![\mathbf{ns}]\!]$ is a terminal object.*
5. *$[\![\theta_1 \,\&\, \theta_2]\!]$, along with the implicit conversions $[\![\theta_1 \,\&\, \theta_2 \leq \theta_1]\!]$ and $[\![\theta_1 \,\&\, \theta_2 \leq \theta_2]\!]$, is a pullback of $[\![\theta_1]\!]$, $[\![\theta_2]\!]$, and $[\![\theta_1 \sqcup \theta_2]\!]$:*



(As illustrated by this diagram, we adopt the convention that an unlabeled arrow from $[\![\theta]\!]$ to $[\![\theta']\!]$ denotes the implicit conversion $[\![\theta \leq \theta']\!]$. Later, we will also use this convention for products of implicit conversions.)

Actually, the functor constructed in [6] only gives isomorphisms, such as $[\![\theta_1 \to \theta_2]\!] \simeq [\![\theta_1]\!] \underset{\mathcal{S}}{\Longrightarrow} [\![\theta_2]\!]$ and $[\![\iota{:}\,\theta]\!] \simeq [\![\theta]\!]$. Without any loss of generality, however, we can make the present exposition far more concise by assuming that these isomorphisms are equalities.

Notice that, because of the last part of the above proposition, the semantics of types is not compositional in the conventional sense, since the meaning $[\![\theta_1 \,\&\, \theta_2]\!]$ of $\theta_1 \,\&\, \theta_2$ depends, not only upon the meanings of θ_1 and θ_2, but also upon the meaning of $\theta_1 \sqcup \theta_2$ and the implicit conversions among these meanings.

A *type assignment* π is a mapping from a finite set of identifiers into types; the appropriate preordering is that $\pi \leq \pi'$ whenever $\operatorname{dom}\pi' \subseteq \operatorname{dom}\pi$ and, for all $\iota \in \operatorname{dom}\pi'$, $\pi\iota \leq \pi'\iota$. We write $[\![-]\!]^*$ for the functor from the preordered set of type assignments (viewed as a category) to $\mathcal{S}$, such that

$$[\![\pi]\!]^* \stackrel{\text{def}}{=} \prod_{\iota \in \operatorname{dom}\pi}^{\mathcal{S}} [\![\pi\iota]\!] \,,$$

and $[\![\pi \leq \pi']\!]^*$ is the unique morphism making the diagram

$$\begin{array}{ccc} \displaystyle\prod_{\iota \in \operatorname{dom}\pi}^{\mathcal{S}} [\![\pi\iota]\!] & \xrightarrow{\ \mathsf{p}\ } & [\![\pi\iota]\!] \\ {\scriptstyle [\![\pi \,\leq\, \pi']\!]^*}\downarrow & & \downarrow \\ \displaystyle\prod_{\iota \in \operatorname{dom}\pi'}^{\mathcal{S}} [\![\pi'\iota]\!] & \xrightarrow{\ \mathsf{p}'\ } & [\![\pi'\iota]\!] \end{array}$$

(where p and p' denote projection morphisms) commute in $\mathcal{S}$ for every $\iota \in \operatorname{dom}\pi'$.

3 Phrases and their Semantics

We write $\pi \vdash p : \theta$ for the judgement that a phrase p has type θ under the type assignment π, and $\mathcal{P}_{\pi\theta}$ for the set of phrases p for which $\pi \vdash p : \theta$ is a valid typing judgement. The valid typing judgements of our language are those that are provable from a certain set of inference rules. At the outset, it is important to note that some of these rules, such as the rule for subtypes:

$$\frac{\pi \vdash p : \theta_1}{\pi \vdash p : \theta} \quad \text{when } \theta_1 \leq \theta$$

are not syntax-directed, since they allow one to infer a judgement about a phrase from a previous judgement about the *same* phrase.

The semantics of phrases (or more precisely of typing judgements, since a phrase satisfying several judgements will have several meanings) is given by a family of semantic functions

$$\mu_{\pi\theta} \in \mathcal{P}_{\pi\theta} \to ([\![\pi]\!]^* \underset{\mathcal{K}}{\longrightarrow} [\![\theta]\!]) \,.$$

Conventionally, one would expect to define these functions by a collection of semantic equations, one for each "reason" that a phrase belongs to one of the $\mathcal{P}_{\pi\theta}$, i.e. for each inference rule. But as soon as one tries to write a semantic equation for a non-syntax-directed rule, such as the intuitively obvious rule

$$\mu_{\pi\theta}[\,p] = \mu_{\pi\theta_1}[\,p]\,;[\![\theta_1 \leq \theta]\!] \qquad \text{when } \pi \vdash p : \theta_1 \text{ and } \theta_1 \leq \theta$$

for subtypes, it becomes clear that something is awry: Since conventional semantic equations are cases of a definition by induction on the size of phrases, they should not define the meaning of a phrase in terms of another meaning of the same phrase. (Note, by the way, that we use a semicolon to denote composition in diagrammatic order.)

In fact, instead of defining the meaning of phrases, the semantic equations for our language define the meaning of proofs that such phrases satisfy typing judgements. They actually define a function μ (distinguished from $\mu_{\pi\theta}$ by the absence of subscripts) that maps each proof P of a judgement $\pi \vdash p : \theta$ into a morphism in $[\![\pi]\!]^* \xrightarrow[\mathcal{K}]{} [\![\theta]\!]$. The definition is by induction on the size of the proof, with a case analysis on the rule used at the main step of the proof.

For each inference rule

$$\frac{J_1 \quad J_2 \quad \ldots \quad J_n}{J}$$

the corresponding semantic equation gives the meaning of the proofs that use the rule at their main step, as a function of the meanings of the subproofs of the judgements that are premisses of the rule. In stating the semantic equation, we will write P for the proof of J whose meaning is being defined and P_i for the subproof of the premiss J_i. For example, the semantic equation for the subtype rule is

$$\mu[P] = \mu[P_1]\,;[\![\theta_1 \leq \theta]\!]\,.$$

The property of coherence bridges the gap between the function μ that is actually defined by the semantic equations and the functions $\mu_{\pi\theta}$ that give meanings to typing judgements independently of their proofs. We say that μ is *coherent* iff $\mu[P_1] = \mu[P_2]$ whenever P_1 and P_2 both prove the same typing judgement. This is exactly the property needed to make the obvious definition

$$\mu_{\pi\theta}[\,p] \stackrel{\text{def}}{=} \mu[P] \text{ for any } P \text{ proving } \pi \vdash p : \theta$$

unambiguous.

4 Inference Rules and Semantic Equations

In this section we give the inference rules and semantic equations for the class of languages for which we will prove coherence. The first three of these rules are the only ones that are not syntax-directed.

- Subtypes

$$\frac{\pi \vdash p : \theta_1}{\pi \vdash p : \theta} \quad \text{when } \theta_1 \leq \theta \qquad\qquad \mu[P] = \mu[P_1] \,;\, [\![\theta_1 \leq \theta]\!] \,.$$

- Nonsense

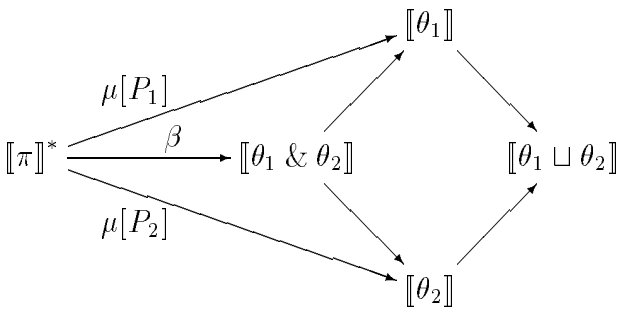
$$\frac{}{\pi \vdash p : \mathbf{ns}} \qquad\qquad \mu[P] = u_\pi \,,$$

where u_π is the unique morphism from $[\![\pi]\!]^*$ to the terminal element $[\![\mathbf{ns}]\!]$.

- Intersections

$$\frac{\pi \vdash p : \theta_1 \qquad \pi \vdash p : \theta_2}{\pi \vdash p : \theta_1 \,\&\, \theta_2} \qquad\qquad \mu[P] = \beta \,,$$

where β is the unique mediating morphism into the pullback:



This semantic equation only makes sense if the perimeter of the above diagram commutes. In the next section, as part of the proof of coherence, we will establish that this condition is always met.

- Identifiers

$$\frac{}{\pi \vdash \iota : \pi(\iota)} \quad \text{when } \iota \in \operatorname{dom} \pi \qquad\qquad \mu[P] = \mathsf{p} \,,$$

where p is the projection morphism from $[\![\pi]\!]^* = \prod^{\mathcal{S}}_{\iota \in \operatorname{dom} \pi} [\![\pi\iota]\!]$ to $[\![\pi\iota]\!]$.

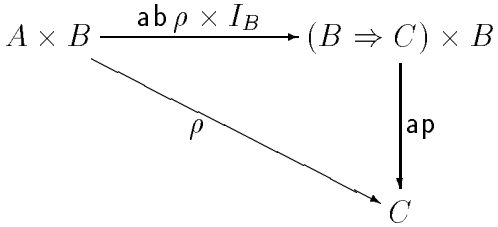
- Constants

$$\frac{}{\pi \vdash c : \theta_c} \qquad\qquad \mu[P] = u_\pi \,;\, \mu_c \,.$$

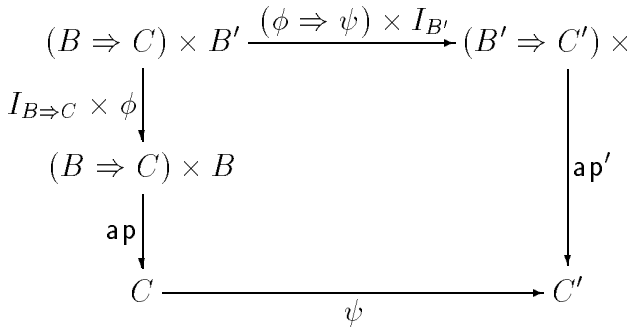
Here θ_c is the type of the constant c, μ_c is its meaning, which must be a morphism from the terminal element $[\![\mathbf{ns}]\!]$ to $[\![\theta_c]\!]$, and u_π is the unique morphism from $[\![\pi]\!]^*$ to the terminal element.

We will not specify any particular constants. It should be noted, however, that when θ_c is an intersection of procedural types, e.g. $(\theta_1 \to \theta_1') \,\&\, (\theta_2 \to \theta_2')$, the requirement that μ_c must be a morphism from $[\![\mathbf{ns}]\!]$ to $[\![\theta_c]\!]$ is a nontrivial constraint that has the same effect as the naturality constraints on generic operators in [5].

The semantics of the next two rules relies upon the Cartesian closed nature of $\mathcal{K}$. For all objects B and C there is an *exponentiation* object $B \Rightarrow C$ and an *application* morphism $\mathsf{ap} \in (B \Rightarrow C) \times B \to C$ such that, for all objects A and morphisms $\rho \in A \times B \to C$, there is exactly one *abstraction* morphism $\mathsf{ab}\,\rho \in A \to (B \Rightarrow C)$ making



commute in $\mathcal{K}$. Moreover, the exponentiation operator $\Rightarrow$ extends to a functor from $\mathcal{K}^{\mathrm{op}} \times \mathcal{K}$ to $\mathcal{K}$ with the following action on morphisms: if $\phi \in B' \to B$ and $\psi \in C \to C'$ then $\phi \Rightarrow \psi$ is the unique morphism in $(B \Rightarrow C) \to (B' \Rightarrow C')$ making



commute in $\mathcal{K}$. Note that the subcategory $\mathcal{S}$ is assumed to be closed under products and exponentiations of both objects and morphisms.

- Applications

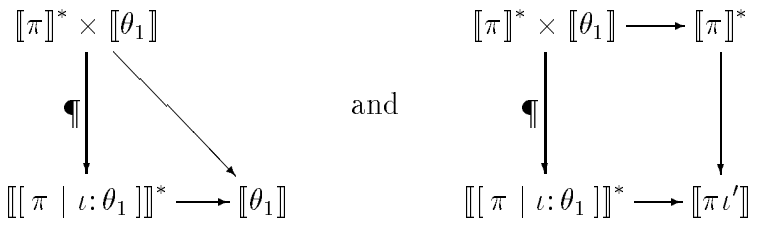
$$\frac{\pi \vdash p_1 : \theta_1 \to \theta \qquad \pi \vdash p_2 : \theta_1}{\pi \vdash p_1\, p_2 : \theta} \qquad\qquad \mu[P] = \langle \mu[P_1], \mu[P_2] \rangle \,;\, \mathsf{ap}\,,$$

where $\langle \mu[P_1], \mu[P_2] \rangle$ is the mediating morphism from the object $[\![\pi]\!]^*$ to the product $([\![\theta_1]\!] \underset{\mathcal{K}}{\Longrightarrow} [\![\theta]\!]) \times [\![\theta_1]\!]$, and ap is the application morphism from this product to $[\![\theta]\!]$.

- Abstractions

$$\frac{[\,\pi \mid \iota{:}\theta_1\,] \vdash p_1 : \theta_2}{\pi \vdash \lambda\iota.\; p_1 : \theta_1 \to \theta_2} \qquad\qquad \mu[P] = \mathsf{ab}(\P \,;\, \mu[P_1])\,.$$

Here $[\,\pi \mid \iota{:}\theta_1\,]$ denotes the type assignment with domain $\mathrm{dom}\,\pi \cup \{\iota\}$ that maps ι into θ_1 and all other ι' in its domain into $\pi\iota'$, while $\P$ denotes the unique morphism making



commute, the latter for all $\iota' \in \operatorname{dom}\pi - \{\iota\}$. (The unlabeled arrows in these diagrams denote the evident projection morphisms.)

Notice that abstractions do not contain type information, so that our inference rules describe an implicitly typed language. Of course, in practice one must use an explicitly typed language, since there is no type-inference decision procedure for intersection types [2]. But explicit type information is not needed to establish coherence; indeed the property of coherence is stronger when formulated for the implicitly typed case, since it implies that changing explicit type information cannot change semantics.

The next two cases deal with operations for constructing objects and extracting their fields (which are central to the object-oriented capabilities of Forsythe). Intuitively, $\iota \equiv p$ constructs an object with a single field whose name is ι and whose value is p, while $p.\iota$ extracts the value of the field named ι from the object p. In the actual semantics, however, $[\![\iota{:}\theta]\!] = [\![\theta]\!]$, i.e. single-field objects are identical to their field values, so that the semantic equations are trivial.

- Object Construction

$$\frac{\pi \vdash p_1 : \theta_1}{\pi \vdash (\iota \equiv p_1) : (\iota{:}\theta_1)} \qquad \mu[P] = \mu[P_1]\,.$$

- Field Selection

$$\frac{\pi \vdash p_1 : (\iota{:}\theta)}{\pi \vdash p_1.\iota : \theta} \qquad \mu[P] = \mu[P_1]\,.$$

The next four rules describe the Forsythe "merging" operation. The essential idea is that a phrase p_1, p_2 should take on the meanings of both p_1 and p_2. However, to avoid ambiguity, the phrase p_2 is required to be either a lambda expression or an object constructor, whose meaning "overwrites" the corresponding type of meaning of p_1. (The object-constructor case provides a limited kind of object concatenation that is much simpler than, for example, the approach of [8].)

- Merging

$$\frac{\pi \vdash \lambda\iota.\ p_2 : \theta_1 \to \theta_2}{\pi \vdash (p_1, \lambda\iota.\ p_2) : \theta_1 \to \theta_2} \qquad \mu[P] = \mu[P_1]$$

$$\frac{\pi \vdash p_1 : \theta}{\pi \vdash (p_1, \lambda\iota.\ p_2) : \theta} \quad \text{when } \theta \text{ is a primitive type or has the form } \iota{:}\theta_1 \qquad \mu[P] = \mu[P_1]$$

$$\frac{\pi \vdash \iota \equiv p_2 : (\iota{:}\theta_1)}{\pi \vdash (p_1, \iota \equiv p_2) : (\iota{:}\theta_1)} \qquad \mu[P] = \mu[P_1]$$

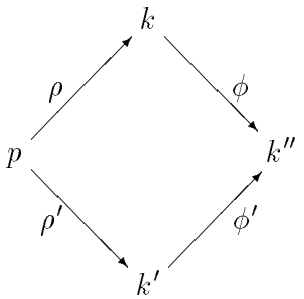
$$\frac{\pi \vdash p_1 : \theta}{\pi \vdash (p_1, \iota \equiv p_2) : \theta} \quad \text{when } \theta \text{ is a primitive type or has the form } \theta_1 \to \theta_2 \text{ or } \iota'{:}\theta_1 \text{ where } \iota' \neq \iota \qquad \mu[P] = \mu[P_1]\,.$$

Finally, we give rules for fixed-point and conditional constructions. In these cases we cannot give an explicit semantics without specifying a particular semantic category; for example, the semantics are different for Forsythe than for a purely functional langauge. So we state the semantic equations in terms of unspecified entities Y_k and cd_θ for which we can give sufficient conditions to ensure coherence.

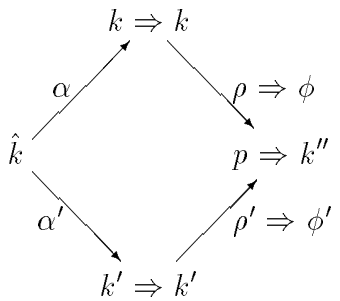
- Fixed Points

$$\frac{\pi \vdash p_1 : \theta \to \theta}{\pi \vdash \mathbf{rec}\, p_1 : \theta} \qquad \mu[P] = \mu[P_1]\,;\mathsf{Y}_{[\![\theta]\!]}\,.$$

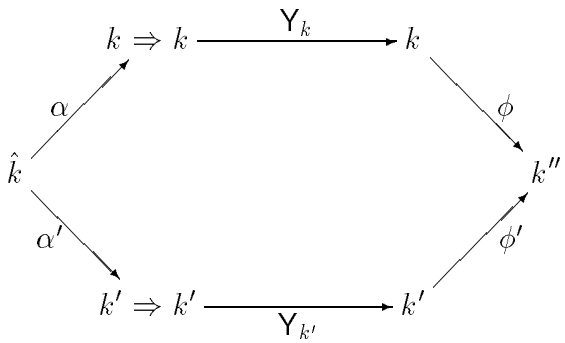
Here $\mathsf{Y}_k \in (k \Rightarrow k) \to k$ is an object-indexed family of morphisms in $\mathcal{K}$ that satisfies the following condition: If p, ρ, ρ' is a pullback in $\mathcal{S}$ of



and if the diagram



commutes in $\mathcal{K}$ then



commutes in $\mathcal{K}$.

- Conditionals

$$\frac{\pi \vdash p_1 : \mathbf{bool} \qquad \pi \vdash p_2 : \theta \qquad \pi \vdash p_3 : \theta}{\pi \vdash \mathbf{if}\, p_1\, \mathbf{then}\, p_2\, \mathbf{else}\, p_3 : \theta} \qquad \mu[P] = \langle \mu[P_1], \mu[P_2], \mu[P_3] \rangle ; \mathsf{cd}_\theta\,,$$

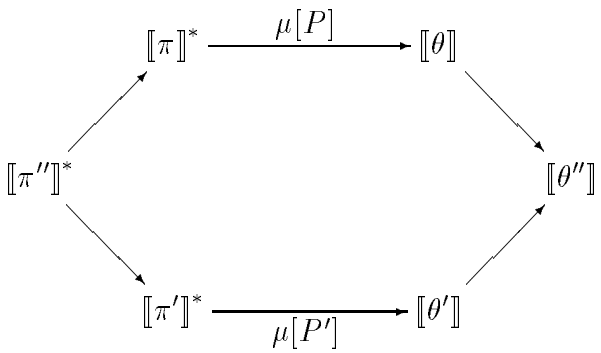
where $\langle \mu[P_1], \mu[P_2], \mu[P_3] \rangle$ is the mediating morphism from $[\![\pi]\!]^*$ to $[\![\mathbf{bool}]\!] \times [\![\theta]\!] \times [\![\theta]\!]$. Here cd_θ is a type-indexed family of morphisms in $\mathcal{K}$ that is required to be a natural transformation from the functor $[\![\mathbf{bool}]\!] \times [\![-]\!] \times [\![-]\!]$ to the functor $[\![-]\!]$ (regarded as functors from the preordered set of types to $\mathcal{K}$). (See Appendix.)

The reader who is familiar with [7] will notice that we have omitted a number of inference rules for various constructions in Forsythe. However, in an implicitly typed version of this language these constructions can all be defined as syntactic sugar, i.e. as abbreviations for phrases in the more basic language described above (with the inclusion of extra constants, such as *add*, that do not actually occur in Forsythe).

5 A Proof of Coherence

We will now prove that the inference rules and semantic equations in the previous section define a semantic function μ that is coherent. In fact, we will prove a stronger property of μ:

Proposition 4 *Suppose P proves $\pi \vdash p : \theta$, P' proves $\pi' \vdash p : \theta'$, $\pi'' \leq \pi$, $\pi'' \leq \pi'$, $\theta \leq \theta''$, and $\theta' \leq \theta''$. Then*



commutes in $\mathcal{K}$.

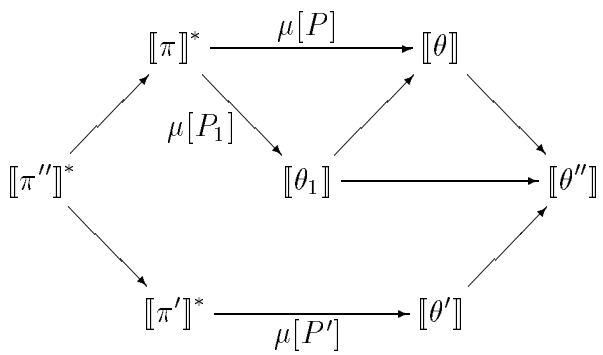
The special case where $\pi = \pi' = \pi''$ and $\theta = \theta' = \theta''$ gives the coherence property. (Also, the special case where $\pi = \pi' = \pi''$ and $\theta'' = \theta \sqcup \theta'$ will be needed to show that the function μ is well-defined in the case of intersections.)

Proof: The proof is by induction on the sum of the sizes of the proofs P and P'.

(1a) If the main step of P is

$$\frac{\pi \vdash p : \theta_1}{\pi \vdash p : \theta} \quad \text{when } \theta_1 \leq \theta$$

then $\mu[P] = \mu[P_1]\,;[\![\theta_1 \leq \theta]\!]$. In the diagram



the inner hexagon commutes by the induction hypothesis, the leftmost triangle by the semantic equation, and the rightmost triangle since $[\![-]\!]$ is a functor. Thus the outer hexagon commutes.

(1b) If the main step of P' is the rule for subtypes, then an argument symmetric to that of (1a) applies.

(2a) If the main step of P is

$$\frac{}{\pi \vdash p : \mathbf{ns}}$$

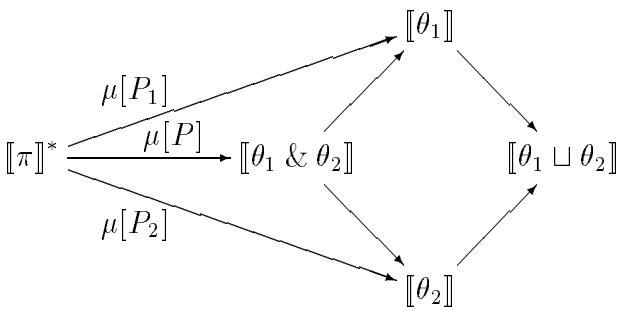
then $\mu[P]$ is the unique morphism from $[\![\pi]\!]^*$ to $[\![\mathbf{ns}]\!]$. Then θ is $\mathbf{ns}$, so that $\theta'' \simeq \mathbf{ns}$, and since $[\![-]\!]$ is a functor, $[\![\theta'']\!]$ is isomorphic to $[\![\mathbf{ns}]\!]$, which is a terminal object. Thus $[\![\theta'']\!]$ is also terminal, so that the hexagon commutes.

(2b) If the main step of P' is the rule for nonsense, then an argument symmetric to that of (2a) applies.

(3a) If the main step of P is

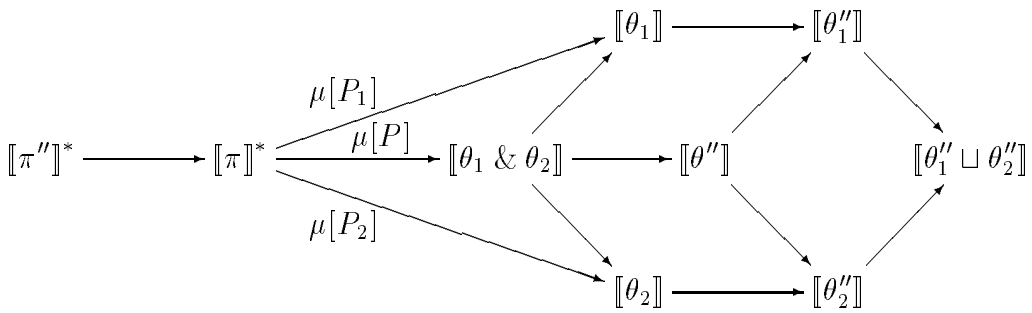
$$\frac{\pi \vdash p : \theta_1 \qquad \pi \vdash p : \theta_2}{\pi \vdash p : \theta_1 \,\&\, \theta_2}$$

then $\mu[P]$ is the unique mediating morphism into the pullback:



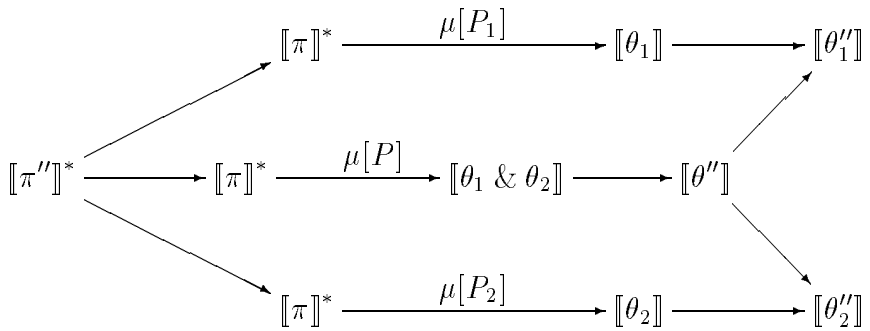
This semantic equation only makes sense if the perimeter of the diagram commutes, but this condition is insured by the induction hypothesis, applied to P_1 and P_2.

Since $\theta = \theta_1 \,\&\, \theta_2 \leq \theta''$, Proposition 1 shows that there are θ_1'' and θ_2'' such that $\theta_1 \leq \theta_1''$, $\theta_2 \leq \theta_2''$, and $\theta'' \simeq \theta_1'' \,\&\, \theta_2''$. Thus everything in

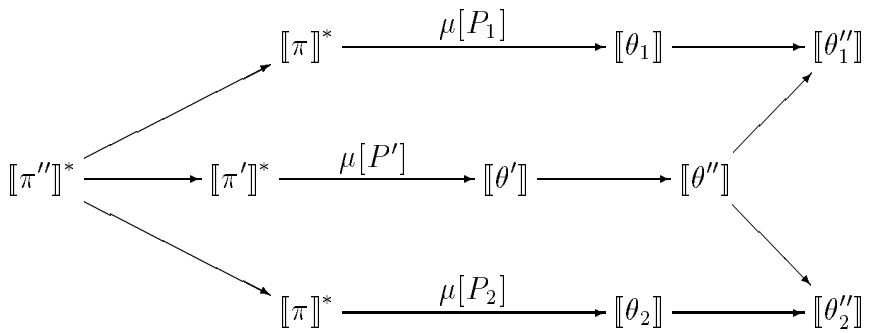


commutes, and $[\![\theta'']\!]$ is a pullback of $[\![\theta_1'']\!]$, $[\![\theta_2'']\!]$, and $[\![\theta_1'' \sqcup \theta_2'']\!]$.

It follows that the outside paths in



are a cone for this pullback and, since the inner polygons commute, the middle path is a mediating morphism. But, by two applications of the induction hypothesis, to P_1 and P' and to P_2 and P' respectively, the inner polygons in



also commute, so that the middle path here is also a mediating morphism. Then the equality of the mediating morphisms gives the desired result.

(3b) If the main step of P' is the rule for intersections, then an argument symmetric to that of (3a) applies.

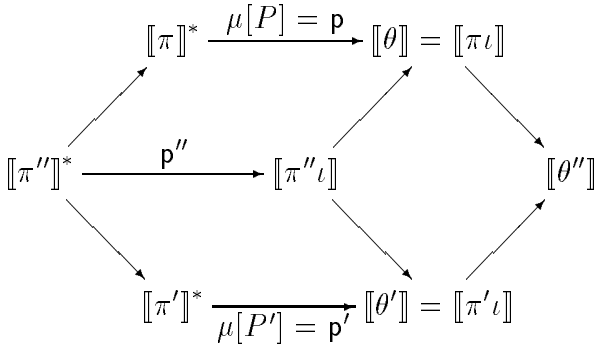
At this stage, we have considered all of the cases where the main step of P or P' is a non-syntax-directed rule. In the remaining cases, since both P and P' prove judgements about the same phrase p, their main steps must be syntax-directed rules for the same syntactic construction. Except in the case of the merging constructions,

there is exactly one rule, which must be used as the main step of both proofs, for each construction.

(4) If p is an identifier ι, then the main steps of P and P' must be

$$\frac{}{\pi \vdash \iota : \pi(\iota)} \qquad \text{and} \qquad \frac{}{\pi' \vdash \iota : \pi'(\iota)}$$

where ι belongs to both $\operatorname{dom}\pi$ and $\operatorname{dom}\pi'$, and $\mu[P]$ and $\mu[P']$ must be projection morphisms. Then the definition of $[\![\pi'' \leq \pi]\!]^*$ and $[\![\pi'' \leq \pi']\!]^*$ insures that the two parallelograms on the left of

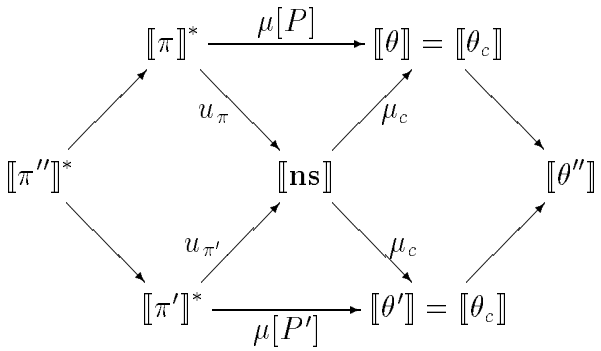


commute, while the functoriality of $[\![-]\!]$ insures that the diamond on the right commutes.

(5) If p is a constant c, then the main steps of P and P' must be

$$\frac{}{\pi \vdash c : \theta_c} \qquad \text{and} \qquad \frac{}{\pi' \vdash c : \theta_c}$$

with $\mu[P] = u_\pi \,;\mu_c$ and $\mu[P'] = u'_\pi \,;\mu_c$. Then the triangles in

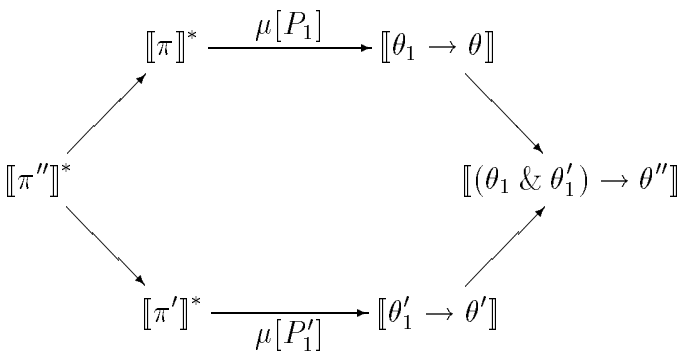


commute, the left diamond commutes since $[\![\mathbf{ns}]\!]$ is terminal, and the right diamond commutes trivially.

(6) If p is an application $p_1\, p_2$, then the main steps of P and P' must be

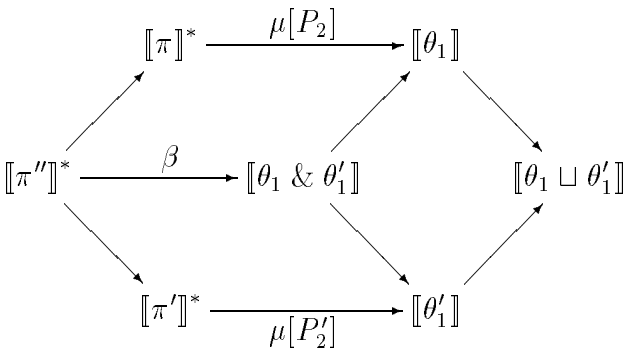
$$\frac{\pi \vdash p_1 : \theta_1 \to \theta \qquad \pi \vdash p_2 : \theta_1}{\pi \vdash p_1\, p_2 : \theta} \qquad \text{and} \qquad \frac{\pi' \vdash p_1 : \theta'_1 \to \theta' \qquad \pi' \vdash p_2 : \theta'_1}{\pi' \vdash p_1\, p_2 : \theta'}$$

with $\mu[P] = \langle \mu[P_1], \mu[P_2] \rangle \,;\mathsf{ap}$ and $\mu[P'] = \langle \mu[P'_1], \mu[P'_2] \rangle \,;\mathsf{ap}'$. Since both $\theta_1 \to \theta$ and $\theta'_1 \to \theta'$ are subtypes of $(\theta_1 \,\&\, \theta'_1) \to \theta''$, the application of the induction hypothesis to P_1 and P'_1 shows that



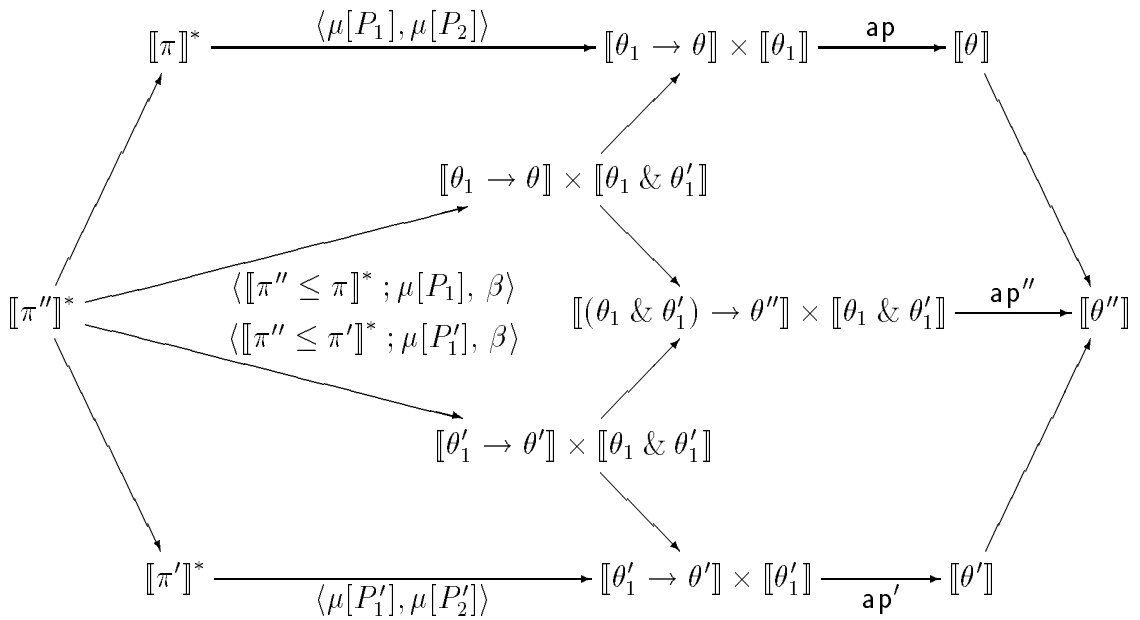
(1)

commutes. On the other hand, since θ_1 and θ'_1 are both subtypes of $\theta_1 \sqcup \theta'_1$, the application of the induction hypothesis to P_2 and P'_2 shows that the perimeter of



(2)

commutes, so that the definition of intersection as a pullback establishes that there is a unique morphism β making the two parallelograms commute. Now consider the diagram



From the commutativity of Diagram 1 and elementary properties of products, the central quadrilateral commutes. From the commutativity of the parallelograms in Diagram 2 and elementary properties of products, the quadrilaterals at the upper and lower left commute. Finally, since Proposition 3 gives

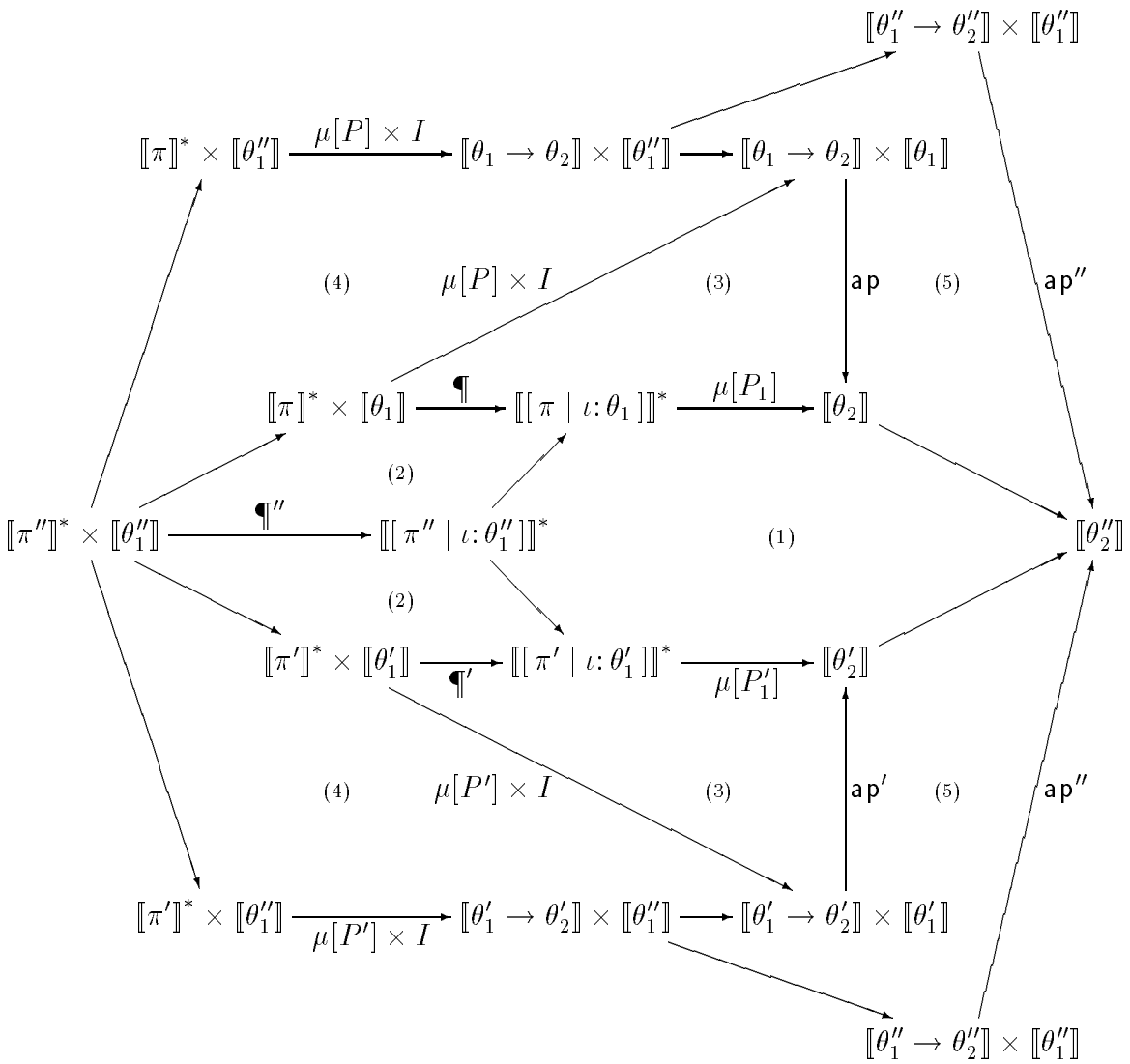
$$[\![\theta_1 \to \theta \le (\theta_1 \,\&\, \theta_1') \to \theta'']\!] = [\![(\theta_1 \,\&\, \theta_1') \le \theta_1]\!] \Rightarrow [\![\theta \le \theta'']\!]$$

(and similarly for $\theta_1' \to \theta'$), the definition of $\Rightarrow$ on morphisms implies that the pentagons at the upper (and lower) right commute. The desired result follows from the semantic equations.

(7) If p is an abstraction $\lambda\iota.\ p_1$, then the main steps of P and P' must be

$$\frac{[\,\pi \mid \iota{:}\theta_1\,] \vdash p_1 : \theta_2}{\pi \vdash \lambda\iota.\ p_1 : \theta_1 \to \theta_2} \qquad \text{and} \qquad \frac{[\,\pi' \mid \iota{:}\theta_1'\,] \vdash p_1 : \theta_2'}{\pi' \vdash \lambda\iota.\ p_1 : \theta_1' \to \theta_2'}$$

with $\mu[P] = \mathsf{ab}(\P\,; \mu[P_1])$ and $\mu[P'] = \mathsf{ab}(\P'\,; \mu[P_1'])$. Let $\theta_1'' = \theta_1 \,\&\, \theta_1'$ and $\theta_2'' = \theta_2 \sqcup \theta_2'$, and consider the diagram



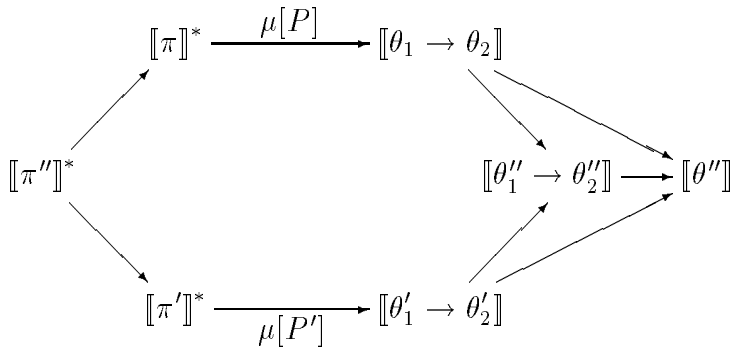
The commutativity of (1) is established by the induction hypothesis, that of (2) by the definition of ¶ and properties of products, that of (3) by the semantic equation and the property of Cartesian closure, that of (4) by properties of products, and that of (5) by Proposition 3 and the definition of $\Rightarrow$ on morphisms.

From the commutativity of the perimeter, using properties of products again, we get

$$\Big(([\![\pi'' \leq \pi]\!]^* \,;\, \mu[P] \,;\, [\![\theta_1 \to \theta_2 \leq \theta_1'' \to \theta_2'']\!]) \times I_{[\![\theta_1'']\!]}\Big) \,;\, \mathsf{ap}'' =$$

$$\Big(([\![\pi'' \leq \pi']\!]^* \,;\, \mu[P'] \,;\, [\![\theta_1' \to \theta_2' \leq \theta_1'' \to \theta_2'']\!]) \times I_{[\![\theta_1'']\!]}\Big) \,;\, \mathsf{ap}'' \,.$$

Then the uniqueness of abstraction morphisms establishes the commutativity of the inner hexagon in

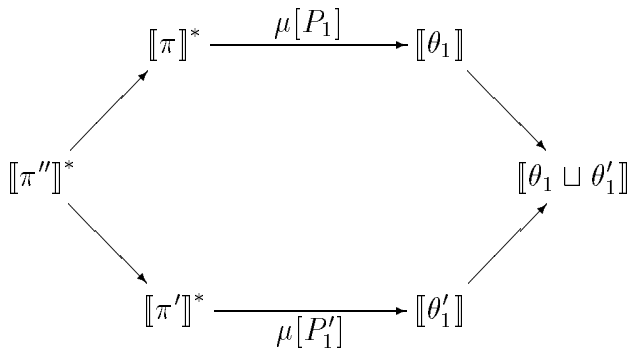


The rest of the diagram is justified by the fact that $\theta_1'' \to \theta_2'' = (\theta_1 \,\&\, \theta_1') \to (\theta_2 \sqcup \theta_2')$ is a least upper bound of $\theta = \theta_1 \to \theta_2$ and $\theta' = \theta_1' \to \theta_2'$, and thus a subtype of θ''.

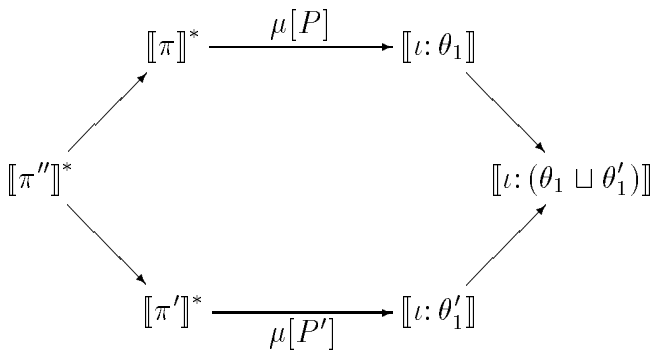
(8) If p is an object constructor $\iota \equiv p_1$, then the main steps of P and P' must be

$$\frac{\pi \vdash p_1 : \theta_1}{\pi \vdash (\iota \equiv p_1) : (\iota{:}\,\theta_1)} \qquad \text{and} \qquad \frac{\pi' \vdash p_1 : \theta_1'}{\pi' \vdash (\iota \equiv p_1) : (\iota{:}\,\theta_1')}$$

with $\mu[P] = \mu[P_1]$ and $\mu[P'] = \mu[P_1']$. The induction hypothesis gives the commutativity of



so that, by the semantic equation and Proposition 3,



commutes. Then, as in the previous case, the desired result follows from the fact that $\iota\colon(\theta_1 \sqcup \theta_1')$ is a least upper bound of $\theta = \iota\colon\theta_1$ and $\theta' = \iota\colon\theta_1'$, and thus a subtype of θ''.

(9) If p is a selection operation $p_1.\iota$, then the main steps of P and P' must be

$$\frac{\pi \vdash p_1 : (\iota\colon\theta)}{\pi \vdash p_1.\iota : \theta} \quad \text{and} \quad \frac{\pi' \vdash p_1 : (\iota\colon\theta')}{\pi' \vdash p_1.\iota : \theta'}$$

with $\mu[P] = \mu[P_1]$ and $\mu[P'] = \mu[P_1']$. The argument here is essentially the same as in the previous case.

(10) If p is a merging operation $p_1, \lambda\iota.\ p_2$, the the main steps of P and P' must be

$$\frac{\pi \vdash \lambda\iota.\ p_2 : \theta_1 \to \theta_2}{\pi \vdash (p_1, \lambda\iota.\ p_2) : \theta_1 \to \theta_2} \quad \text{or} \quad \frac{\pi \vdash p_1 : \theta}{\pi \vdash (p_1, \lambda\iota.\ p_2) : \theta} \quad \begin{array}{l}\text{when } \theta \text{ is a primitive}\\ \text{type or has the form } \iota\colon\theta_1\end{array}$$

and

$$\frac{\pi' \vdash \lambda\iota.\ p_2 : \theta_1' \to \theta_2'}{\pi' \vdash (p_1, \lambda\iota.\ p_2) : \theta_1' \to \theta_2'} \quad \text{or} \quad \frac{\pi' \vdash p_1 : \theta'}{\pi' \vdash (p_1, \lambda\iota.\ p_2) : \theta'} \quad \begin{array}{l}\text{when } \theta' \text{ is a primitive}\\ \text{type or has the form } \iota\colon\theta_1'\end{array}$$

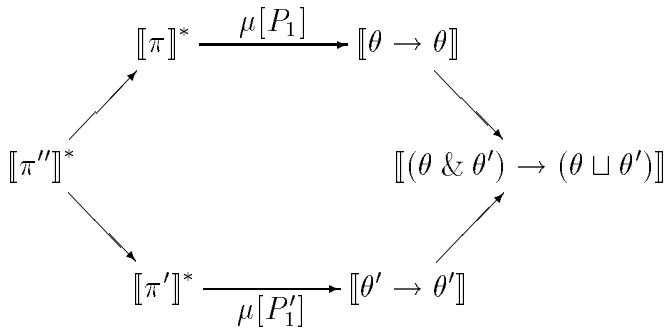
with $\mu[P] = \mu[P_1]$ and $\mu[P'] = \mu[P_1']$. If the main steps of P and P' are both rules on the left or both rules on the right, then the induction hypothesis gives the desired result immediately. On the other hand, if one main step is the rule on the left and the other is the rule on the right then, by Proposition 2, **ns** is a least upper bound of θ and θ', so that $\theta'' \simeq \mathbf{ns}$. Then the desired result follows since $[\![\theta'']\!]$ is terminal.

(11) If p is a merging operation $p_1, \iota \equiv p_2$, then the argument is similar to the previous case.

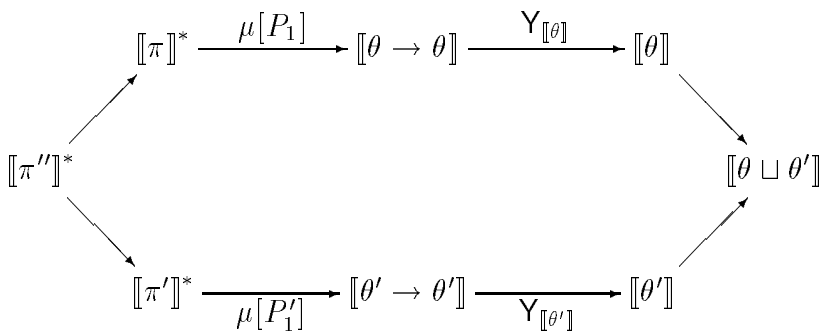
(12) If p is a fixed-point construction $\mathbf{rec}\, p_1$, then the main steps of P and P' must be

$$\frac{\pi \vdash p_1 : \theta \to \theta}{\pi \vdash \mathbf{rec}\, p_1 : \theta} \quad \text{and} \quad \frac{\pi' \vdash p_1 : \theta' \to \theta'}{\pi' \vdash \mathbf{rec}\, p_1 : \theta'}$$

with $\mu[P] = \mu[P_1]\,;\mathsf{Y}_{[\![\theta]\!]}$ and $\mu[P'] = \mu[P'_1]\,;\mathsf{Y}_{[\![\theta']\!]}$. Since $[\![\theta \,\&\, \theta']\!]$ is a pullback of $[\![\theta]\!]$, $[\![\theta']\!]$, and $[\![\theta \sqcup \theta']\!]$, and since by the induction hypothesis



commutes, the condition satisfied by the Y_k implies the commutativity of

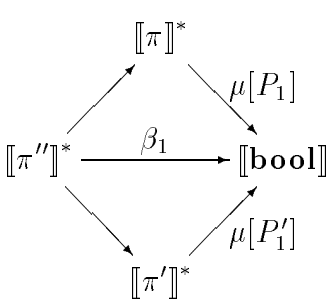


Then the desired result follows from the semantic equations and $\theta \sqcup \theta' \leq \theta''$.

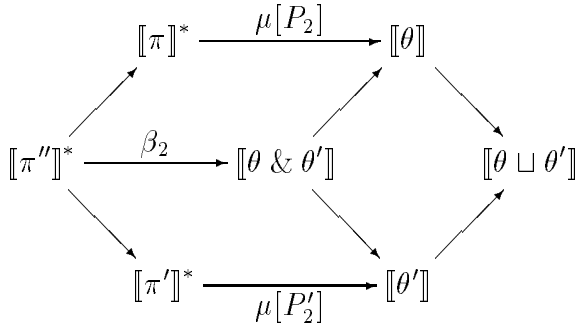
(13) If p is a conditional construction $\mathbf{if}\, p_1\, \mathbf{then}\, p_2\, \mathbf{else}\, p_3$, then the main steps of P and P' must be

$$\frac{\pi \vdash p_1 : \mathbf{bool} \qquad \pi \vdash p_2 : \theta \qquad \pi \vdash p_3 : \theta}{\pi \vdash \mathbf{if}\, p_1\, \mathbf{then}\, p_2\, \mathbf{else}\, p_3 : \theta} \qquad \frac{\pi' \vdash p_1 : \mathbf{bool} \qquad \pi' \vdash p_2 : \theta' \qquad \pi' \vdash p_3 : \theta'}{\pi' \vdash \mathbf{if}\, p_1\, \mathbf{then}\, p_2\, \mathbf{else}\, p_3 : \theta'}$$

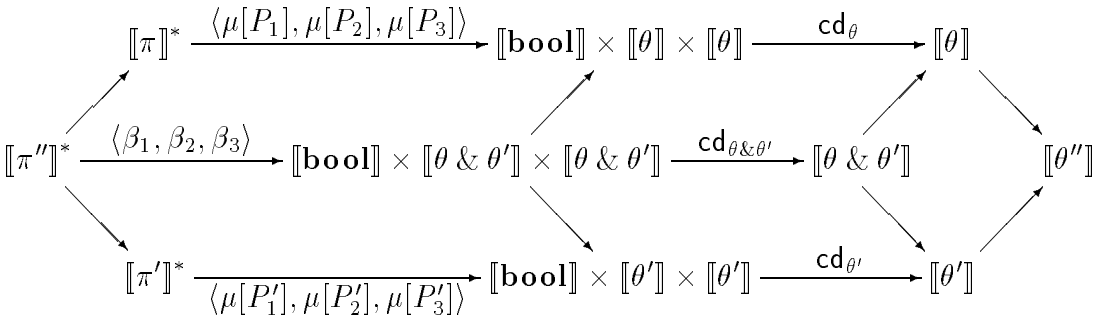
with $\mu[P] = \langle \mu[P_1], \mu[P_2], \mu[P_3] \rangle\,;\mathsf{cd}_\theta$ and $\mu[P'] = \langle \mu[P'_1], \mu[P'_2], \mu[P'_3] \rangle\,;\mathsf{cd}_{\theta'}$. By the induction hypothesis for each of the three premisses, the perimeters of



and



and a similar diagram for P_3 and P'_3 must commute. Thus, trivially in the first case and by the definition of $\theta \& \theta'$ as a pullback in the remaining cases, there are morphisms β_1, β_2, and β_3 that make the interior triangles and parallelograms commute. Then the lefthand parallelograms in



commute, and the rest of the diagram commutes since cd is natural and $[\![-]\!]$ is a functor. The desired result follows from the semantic equations. (*End of Proof*)

Acknowledgement

The author wishes to thank Andre Scedrov for convincing him of the importance of coherence.

References

[1] V. Breazu-Tannen, T. Coquand, C. A. Gunter, and A. Scedrov. Inheritance as implicit coercion. *Information and Computation*, 93(1):172–222, July 1991.

[2] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Functional characters of solvable terms. *Z. Math. Logik Grundlagen Math.*, 27:45–58, 1981.

[3] F. J. Oles. *A Category-Theoretic Approach to the Semantics of Programming Languages*. Ph. D. dissertation, Syracuse University, August 1982.

[4] F. J. Oles. Type algebras, functor categories, and block structure. In M. Nivat and J. C. Reynolds, editors, *Algebraic Methods in Semantics*, pages 543–573. Cambridge Univ. Press, Cambridge, England, 1985.

[5] J. C. Reynolds. Using category theory to design implicit conversions and generic operators. In N. D. Jones, editor, *Semantics-Directed Compiler Generation*, volume 94 of *Lect. Notes in Computer Sci.*, pages 211–258, Berlin, 1980. Springer-Verlag.

[6] J. C. Reynolds. Conjunctive types and Algol-like languages (abstract of invited lecture). In *Proc. Symp. on Logic in Computer Science*, page 119, 1987. Full text in preparation.

[7] J. C. Reynolds. Preliminary design of the programming language Forsythe. Report CMU-CS-88-159, Carnegie Mellon University, June 21, 1988.

[8] M. Wand. Type inference for record concatenation and multiple inheritance. In *Proc. Fourth Annual Symp. on Logic in Computer Science*, pages 92–97, 1989.

A Conditionals in Forsythe

To complete the definition of Forsythe and show that it is coherent, we would have to specify the categories $\mathcal{K}$ and $\mathcal{S}$, the morphism families Y_k and cd_θ, and the meaning of various constants, and demonstrate that these entities satisfy the assumptions stated in this paper. For the most part, such a development is beyond the scope of our presentation. In this appendix, however, we consider one aspect that is closely related to the rest of the paper and does not involve the details of functor-category semantics.

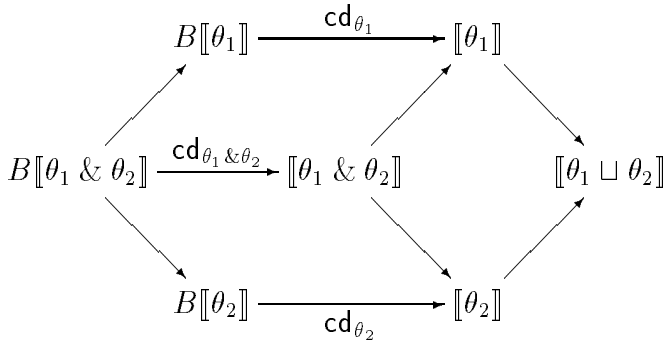
In Forsythe, the conditional construction is first defined for primitive types and then defined by induction for all types. Here we show that if the initial definition for primitives is a natural transformation then the inductively defined extension is a natural transformation.

Let B be the functor from the preordered set of types to $\mathcal{K}$ such that $B(-) = [\![\mathbf{bool}]\!] \times (-) \times (-)$, and assume that cd is a natural transformation from $B[\![J(-)]\!]$ to $[\![J(-)]\!]$, where J is the injection from primitive types to types. Then we extend $\mathsf{cd}_\theta \in B[\![\theta]\!] \to [\![\theta]\!]$ to all types θ as follows:

- $\mathsf{cd}_{\theta_1 \to \theta_2} = \mathsf{ab}\Big(\Psi_{[\![\theta_1 \to \theta_2]\!],[\![\theta_1]\!]}\,;\, B(\mathsf{ap})\,;\, \mathsf{cd}_{\theta_2}\Big)$,

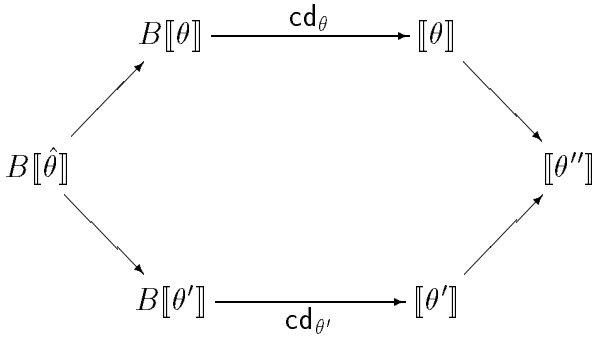
where $\Psi_{k,k'}$ is the evident morphism from the product $([\![\mathbf{bool}]\!] \times k \times k) \times k'$ to the product $[\![\mathbf{bool}]\!] \times (k \times k') \times (k \times k')$ that "duplicates its second argument".

- $\mathsf{cd}_{\iota:\theta} = \mathsf{cd}_\theta$.
- $\mathsf{cd}_{\mathbf{ns}}$ is the unique morphism from $B[\![\mathbf{ns}]\!]$ to $[\![\mathbf{ns}]\!]$.
- $\mathsf{cd}_{\theta_1 \,\&\, \theta_2}$ is the unique morphism into the pullback:



The final case of the definition only makes sense if the perimeter of the diagram commutes, but the following proposition will establish that this condition is always met.

Proposition 5 *Suppose $\hat{\theta}$ is a lower bound, and θ'' is an upper bound, of θ and θ'. Then*



commutes in $\mathcal{K}$.

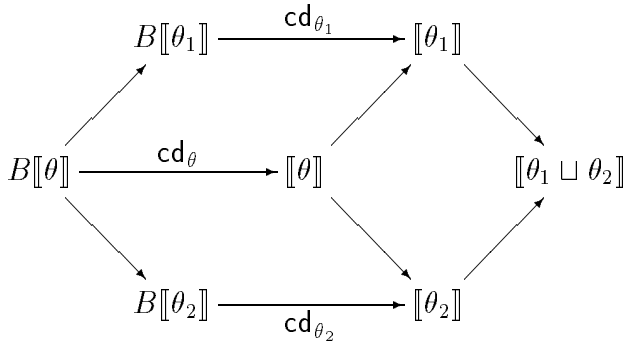
Notice that this proposition, in the special case where $\hat{\theta} = \theta \leq \theta' = \theta''$, implies the naturality of cd.

Proof: The proof is by induction on the sum of the sizes of the types θ and θ'.

(1a) If $\theta = \mathbf{ns}$ then $\theta'' \simeq \mathbf{ns}$, so that $[\![\theta'']\!]$ is terminal, and the hexagon commutes.

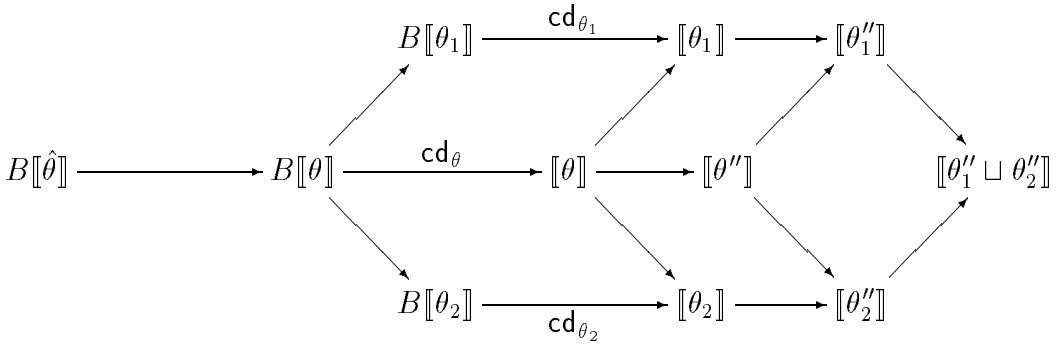
(1b) If $\theta' = \mathbf{ns}$ then an argument symmetric to (1a) applies.

(2a) If $\theta = \theta_1 \,\&\, \theta_2$ then cd_θ is the unique mediating morphism into the pullback:



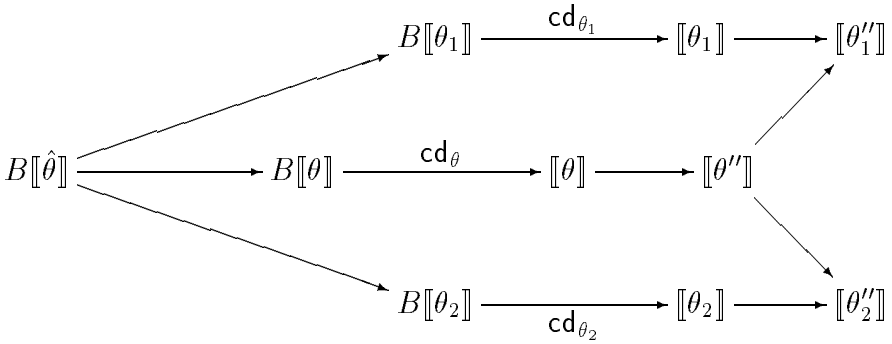
This only makes sense if the perimeter of this diagram commutes, but this condition is insured by the induction hypothesis, applied to θ_1 and θ_2.

Since $\theta = \theta_1 \,\&\, \theta_2 \leq \theta''$, Proposition 1 shows that there are θ_1'' and θ_2'' such that $\theta_1 \leq \theta_1''$, $\theta_2 \leq \theta_2''$, and $\theta'' \simeq \theta_1'' \,\&\, \theta_2''$. Thus everything in

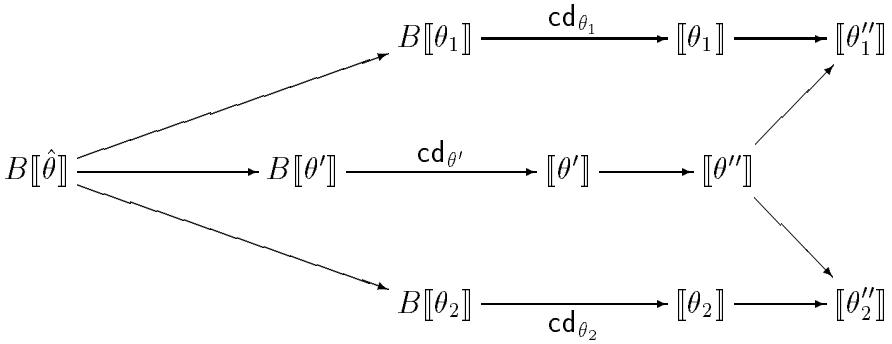


commutes, and $[\![\theta'']\!]$ is a pullback of $[\![\theta''_1]\!]$, $[\![\theta''_2]\!]$, and $[\![\theta''_1 \sqcup \theta''_2]\!]$.

It follows that the outside paths in



are a cone for this pullback and, since the inner polygons commute, the middle path is a mediating morphism. But, by two applications of the induction hypothesis, to θ_1 and θ' and to θ_2 and θ' respectively, the inner polygons in

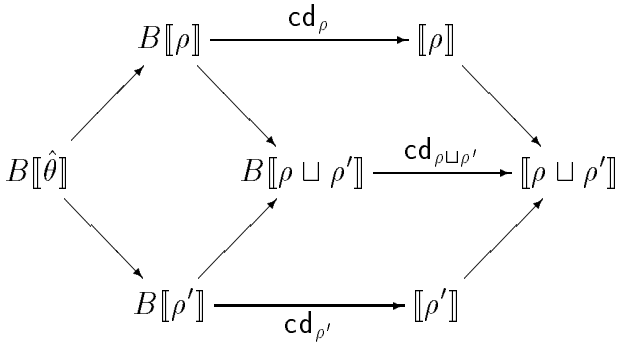


also commute, so that the middle path here is also a mediating morphism. Then the equality of the mediating morphisms gives the desired result.

(2b) If $\theta' = \theta_1 \,\&\, \theta_2$ then an argument symmetric to (2a) applies.

(3) In the remaining cases, Proposition 2 shows that **ns** is a least upper bound of θ and θ' unless θ and θ' are both primitive types and have a primitive least upper bound, or both procedural types, or both object types with the same field name. But when **ns** is a least upper bound, $[\![\theta'']\!]$ must be terminal.

(4) If $\theta = \rho$ and $\theta' = \rho'$ are primitive types with a least upper bound, then the naturality of cd for primitive types gives the commutativity of

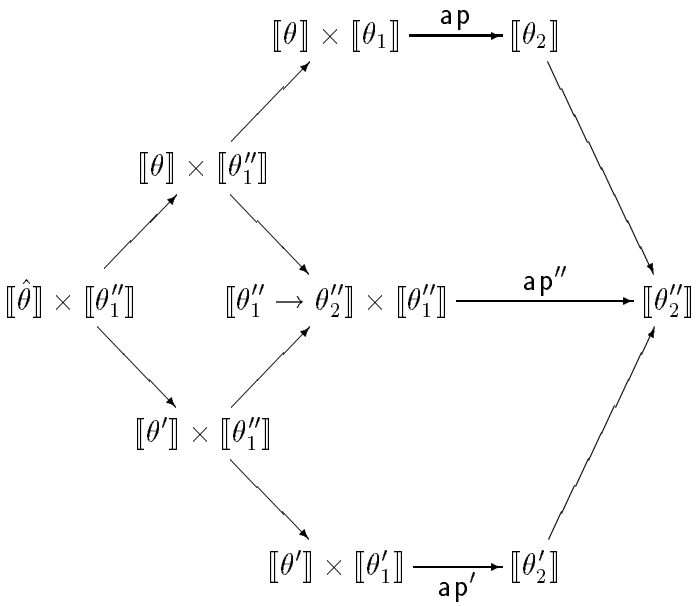


and the desired result follows since $\rho \sqcup \rho' = \theta \sqcup \theta' \leq \theta''$.

(5) If $\theta = \theta_1 \to \theta_2$ and $\theta' = \theta'_1 \to \theta'_2$ then

$$\mathsf{cd}_\theta = \mathsf{ab}\Big(\Psi_{[\![\theta]\!],[\![\theta_1]\!]}\,;B(\mathsf{ap})\,;\mathsf{cd}_{\theta_2}\Big) \qquad \text{and} \qquad \mathsf{cd}_{\theta'} = \mathsf{ab}\Big(\Psi_{[\![\theta']\!],[\![\theta'_1]\!]}\,;B(\mathsf{ap}')\,;\mathsf{cd}_{\theta'_2}\Big)\,.$$

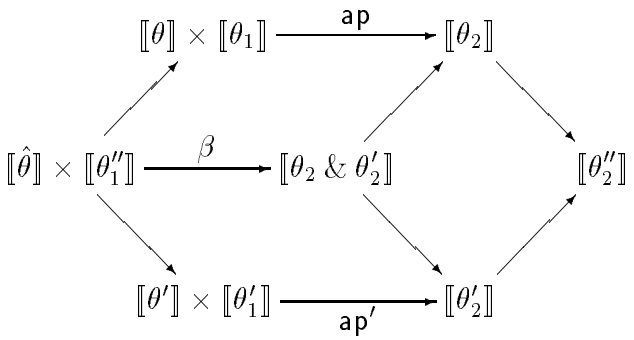
Let $\theta''_1 = \theta_1 \,\&\, \theta'_1$ and $\theta''_2 = \theta_2 \sqcup \theta'_2$, and consider the diagram



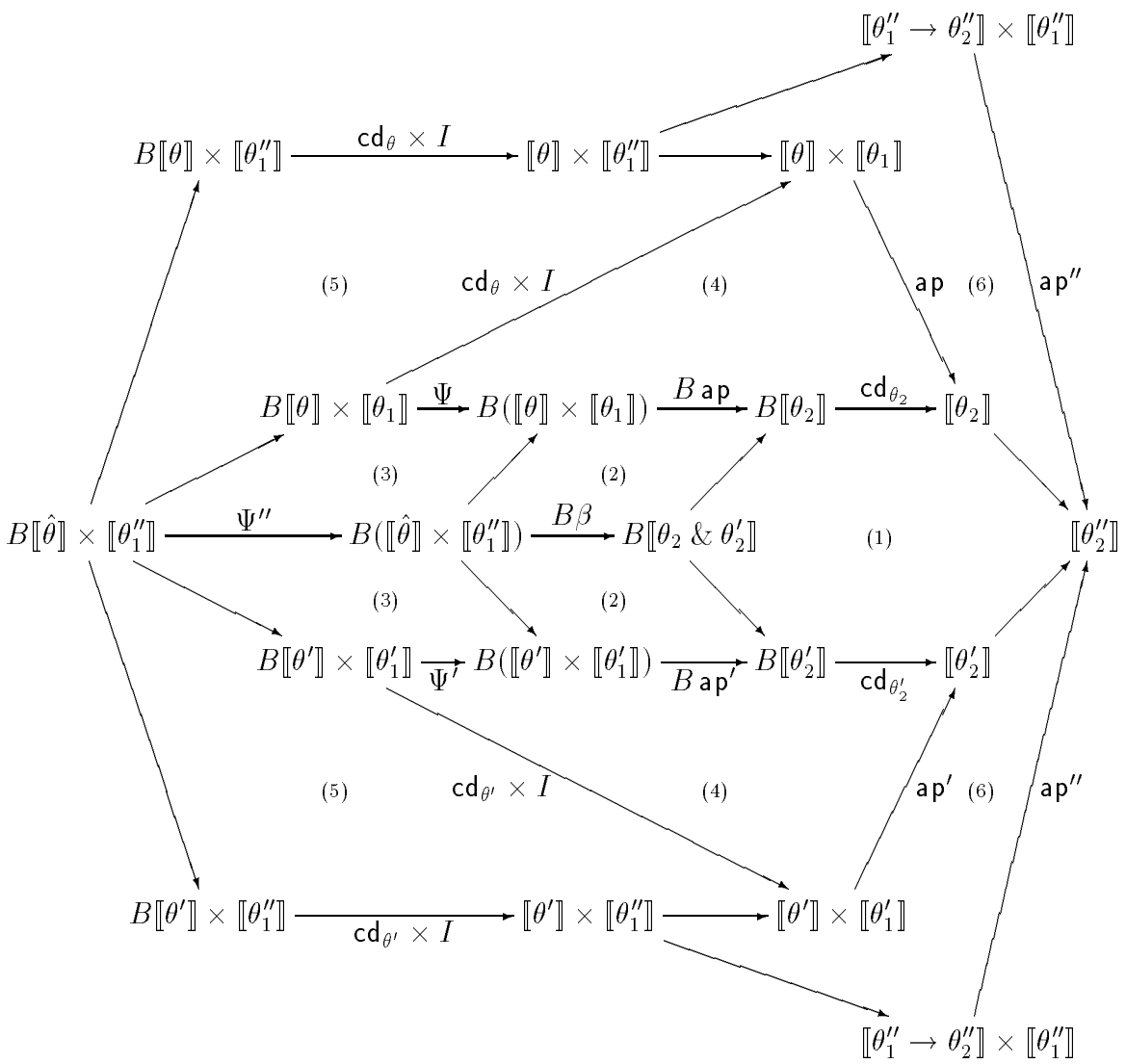
Since

$$[\![\theta \leq \theta''_1 \to \theta''_2]\!] = [\![\theta_1 \to \theta_2 \leq \theta''_1 \to \theta''_2]\!] = [\![\theta''_1 \leq \theta_1]\!] \Rightarrow [\![\theta_2 \leq \theta''_2]\!]\,,$$

(and similarly for θ'), the definition of $\Rightarrow$ on morphisms implies that the upper (and lower) pentagons commute. Thus the perimeter commutes, and by the definition of $\theta_2 \,\&\, \theta'_2$ as a pullback, there is a unique mediating morphism β that makes the two parallelograms in



commute. Now consider the diagram



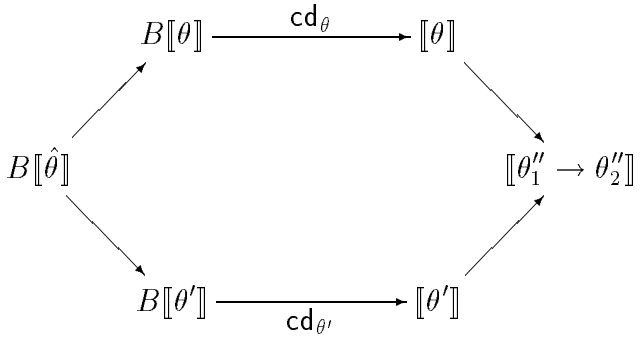
The commutativity of (1) is established by the induction hypothesis, that of (2) by applying B to the previous diagram, that of (3) by properties of products that insure that Ψ is natural, that of (4) by the definition of cd_θ and the property of Cartesian closure, that of (5) by properties of products, and that of (6) by Proposition 3 and the definition of $\Rightarrow$ on morphisms.

From the commutivity of the perimeter, using properties of products again, we get

$$\Big((B[\![\hat{\theta} \le \theta]\!] \,;\, \mathsf{cd}_\theta \,;\, [\![\theta \le \theta_1'' \to \theta_2'']\!]) \times I_{[\![\theta_1'']\!]}\Big) \,;\, \mathsf{ap}'' =$$

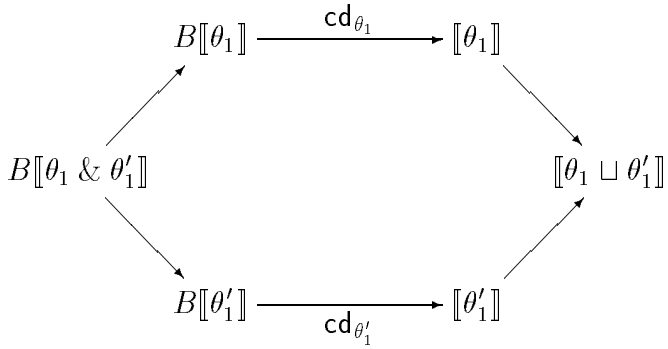
$$\Big((B[\![\hat{\theta} \le \theta']\!] \,;\, \mathsf{cd}_{\theta'} \,;\, [\![\theta' \le \theta_1'' \to \theta_2'']\!]) \times I_{[\![\theta_1'']\!]}\Big) \,;\, \mathsf{ap}'' \,.$$

Then the uniqueness of abstraction morphisms establishes the commutativity of

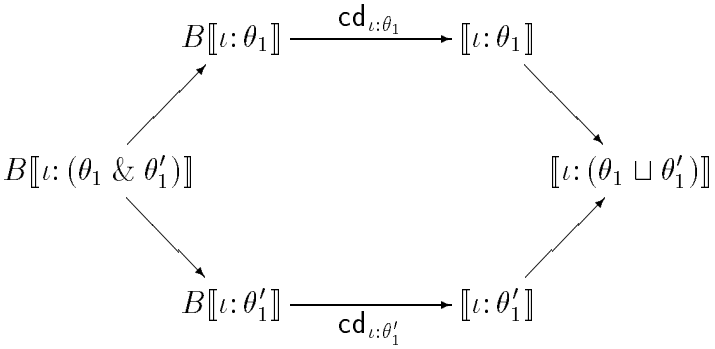


and the desired result follows since $\theta_1'' \to \theta_2''$ is a least upper bound of θ and θ', and thus a subtype of θ''.

(6) If $\theta = \iota{:}\theta_1$ and $\theta' = \iota{:}\theta_1'$ then the induction hypothesis gives the commutativity of



Then since $\mathsf{cd}_{\iota:\theta_1} = \mathsf{cd}_{\theta_1}$ and $[\![\iota{:}\theta_1]\!] = [\![\theta_1]\!]$, and similarly for θ_1', $\theta_1 \,\&\, \theta_1'$, $\theta_1 \sqcup \theta_1'$,



commutes. The desired result follows since $\hat{\theta} \leq \iota{:}\theta_1 \& \iota{:}\theta_1' \leq \iota{:}(\theta_1 \& \theta_1')$ and $\iota{:}(\theta_1 \sqcup \theta_1') \simeq \iota{:}\theta_1 \sqcup \iota{:}\theta_1' \leq \theta''$. *(End of Proof)*