

Checking Algorithms for Pure Type Systems^{***}

L.S. van Benthem Jutting¹, J. McKinna² and R. Pollack²

¹ Faculty of Mathematics and Computer Science, University of Nijmegen,
Toernooiveld 1, 6525 ED Nijmegen, The Netherlands

² Laboratory for Foundations of Computer Science, University of Edinburgh,
The King's Buildings, Edinburgh, EH9 3JZ, Scotland
jhm@dcs.ed.ac.uk, rap@dcs.ed.ac.uk

1 Introduction

This work is motivated by the problem of finding reasonable algorithms for typechecking Pure Type Systems [Bar91] (PTS). There are several implementations of formal systems that are either PTS or closely related to PTS. For example, LEGO [LP92] implements the Pure Calculus of Constructions (PCC) [CH88], the Extended Calculus of Constructions [Luo90] and the Edinburgh Logical Framework (LF) [HHP87]. ELF [Pfe89] implements LF; CONSTRUCTOR [Hel91] implements arbitrary PTS with finite set of sorts. Are these implementations actually correct? Of course, we may enumerate all derivations of a given PTS, and Jutting [vBJ93] has shown that a large class of normalizing PTS have decidable typechecking by computing the normal forms of types, but such techniques are obviously not usable in practice. Algorithms in the literature for particular type systems, such as Huet's Constructive Engine [Hue89], do not obviously extend even to such tame classes as the normalizing and functional PTS.

In the rest of this section we briefly review the definition and well-known theory of PTS, outline the basic approach to checking algorithms, and analyse the difficulty in using this approach for checking PTS. In section 1.5, we outline the plan for the rest of this paper.

1.1 Pure Type Systems

A *Pure Type System* is a quadruple $\mathfrak{S} = \{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$, where

- \mathcal{S} is the set of *sorts*; elements of \mathcal{S} will be denoted by s, s_0, s_1, \dots ,
- \mathcal{V} is the set of *variables*; elements of \mathcal{V} will be denoted by x, y, z ,
- $\mathcal{S} \cap \mathcal{V} = \emptyset$,
- $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{S}$ is the set of *axioms* which we assume to be nonempty,
- $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{S}$ is the set of *Π -rules*.

We usually assume we are discussing some specific PTS, $\mathfrak{S} = \{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$, which may be assumed to have special properties in later sections. Let us also assume \mathcal{S} is denumerable, and \mathcal{A} and \mathcal{R} are decidable relations, although these assumptions are only used in discussing algorithmic properties of inductive presentations of relations, not the relationships between different presentations.

Definition 1 Pseudoterms. The set T of *pseudoterms* of \mathfrak{S} is the smallest set satisfying

$$\underline{\mathcal{S} \cup \mathcal{V} \subseteq T},$$

* This work was supported by the ESPRIT Basic Research Actions on Logical Frameworks and Types for Proofs and Programs, and by grants from the British Science and Engineering Research Council.

** A version of this paper appears in *Types for Proofs and Programs: International Workshop TYPES'93, Nijmegen, May 1993, Selected Papers*, LNCS 806

If $a \in \mathbb{T}$ and $b \in \mathbb{T}$ then $ab \in \mathbb{T}$,
 If $A \in \mathbb{T}$, $B \in \mathbb{T}$ and $x \in \mathcal{V}$ then $(\Pi x:A.B) \in \mathbb{T}$,
 If $A \in \mathbb{T}$, $b \in \mathbb{T}$ and $x \in \mathcal{V}$ then $(\lambda x:A.b) \in \mathbb{T}$.

Elements of \mathbb{T} will be denoted by $a, b, c, \dots, A, B, C, \dots$. The notions of free and bound variables are defined as usual, with $\text{FV}(a)$ denoting the set of free variables of a . We consider equality between terms to be equivalence modulo α -conversion and this equivalence will be denoted by $=$.

Convention 2 Variables. In this presentation we are informal about variables, and omit side conditions such as “ x is a fresh variable”. Much of this paper has been formalized in the LEGO system in a presentation with explicit variable names [MP93], but thinking in terms of de Bruijn nameless variables will clarify many questions that might arise about variables in the following.

Reduction and conversion We denote the substitution of a for x in b by $b[x := a]$, and write \rightarrow for one step β -reduction, \twoheadrightarrow for β -reduction and \simeq for β -convertibility.

From section 4 onward we extend our concept of reduction by allowing for contractions of the form

$$(\Pi x:A.B) a \rightarrow_{\pi} B[x := a]$$

i.e. for application of a product to an argument. Intuitively a π -redex $(\Pi x:A.B) a$ denotes a coordinate axis in the product $\Pi x:A.B$. Such considerations first appear in the pioneering work of the AUTOMATH group [vD80], and allows a presentation of the basic typing relation free of any direct appeal to substitution, which in contemporary presentations of type theory such as [Bar92] appears explicitly in the rule for typing an application (rule **App** below). We then have the new notion of $\beta\pi$ -reduction, \twoheadrightarrow_{π} , and $\beta\pi$ -conversion, \simeq_{π} , generated by the elementary β - and π -contractions. The well known properties of substitution and reduction, e.g. the Church-Rosser property, extend to \twoheadrightarrow_{π} and \simeq_{π} . These properties will be used freely in the sequel.

Contexts A *context*, Γ , is a sequence of assignments $x:A$. If

$$\Gamma = x_1:A_1, x_2:A_2, \dots, x_n:A_n \quad (n \geq 0)$$

we write $x_i:A_i \in \Gamma$ for $1 \leq i \leq n$, and $\text{dom}(\Gamma)$ for the set $\{x_1, x_2, \dots, x_n\}$. The free variables of Γ are defined by $\text{FV}(\Gamma) = \bigcup_i \text{FV}(A_i)$. The empty context is written \emptyset , and the set of all contexts is \mathbb{C} .

Inclusion between contexts is defined by

$$\Gamma_1 \sqsubseteq \Gamma_2 \stackrel{\Delta}{=} \forall x, A [x:A \in \Gamma_1 \Rightarrow x:A \in \Gamma_2]$$

The notion of (one step) β -reduction is easily extended to contexts:

$$A \rightarrow B \Rightarrow \Gamma_1, x:A, \Gamma_2 \rightarrow \Gamma_1, x:B, \Gamma_2.$$

1.2 Correctness

We define a relation, \vdash , called *correctness* as follows.

Definition 3 Correctness. The relation $\vdash \subseteq \mathbf{C} \times \mathbf{T} \times \mathbf{T}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l}
\text{Srt} \quad \emptyset \vdash s_1 : s_2 \qquad \langle s_1, s_2 \rangle \in \mathcal{A} \\
\\
\text{Var} \quad \frac{\Gamma \vdash A : s}{\Gamma, x:A \vdash x : A} \\
\\
\text{Wk} \quad \frac{\Gamma \vdash b : B \quad \Gamma \vdash A : s}{\Gamma, x:A \vdash b : B} \qquad b \in \mathcal{S} \cup \mathcal{V} \\
\\
\text{Pi} \quad \frac{\Gamma \vdash A : s_1 \quad \Gamma, x:A \vdash B : s_2}{\Gamma \vdash \Pi x:A. B : s_3} \qquad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{Lda} \quad \frac{\Gamma \vdash A : s_1 \quad \Gamma, x:A \vdash b : B \quad \Gamma, x:A \vdash B : s_2}{\Gamma \vdash \lambda x:A. b : \Pi x:A. B} \qquad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{App} \quad \frac{\Gamma \vdash a : \Pi x:B. A \quad \Gamma \vdash b : B}{\Gamma \vdash ab : A[x := b]} \\
\\
\text{Cnv} \quad \frac{\Gamma \vdash a : A \quad \Gamma \vdash B : s}{\Gamma \vdash a : B} \qquad A \simeq B
\end{array}$$

The cognoscenti will notice that the rule **Wk** in the definition above is not the usual one, as b is restricted to variables and sorts. It is easy to see from the lemmas below that the relation defined above is equivalent to the usual **PTS** system. In fact the presentation above gives a better development of the basic metatheory, because the generation lemma does not depend on weakening. The following properties of **PTS** can be proved along the lines of [GN91] or [Bar92]. Most proofs are by induction on the structure of \vdash derivations.

Lemma 4 Free Variables.

$$\begin{array}{l}
\text{If} \quad \Gamma \vdash a : A \\
\text{then} \quad FV(a) \cup FV(A) \subseteq \text{dom}(\Gamma).
\end{array}$$

Lemma 5 Start. Suppose $\Gamma \vdash a : A$.

- i* $\langle s_1, s_2 \rangle \in \mathcal{A} \Leftrightarrow \Gamma \vdash s_1 : s_2$
- ii* If $x:B \in \Gamma$ then $\Gamma \vdash x : B$

Lemma 6 Generation.

- i* If $\Gamma \vdash s : A$
then $\exists s_0 [A \simeq s_0 \text{ and } \langle s, s_0 \rangle \in \mathcal{A}]$.
- ii* If $\Gamma \vdash x : A$
then $\exists A_0 [A \simeq A_0 \text{ and } x:A_0 \in \Gamma]$.
- iii* If $\Gamma \vdash \Pi x:C. D : A$
then $\exists s_1, s_2, s_3 [A \simeq s_3, \langle s_1, s_2, s_3 \rangle \in \mathcal{R}, \Gamma \vdash C : s_1 \text{ and } \Gamma, x:C \vdash D : s_2]$.

- iv* If $\Gamma \vdash \lambda x:C.d : A$
then $\exists s_1, s_2, s_3, D [A \simeq \Pi x:C.D, \langle s_1, s_2, s_3 \rangle \in \mathcal{R},$
 $\Gamma \vdash C : s_1, \Gamma, x:C \vdash d : D \text{ and } \Gamma, x:C \vdash D : s_2]$.
- v* If $\Gamma \vdash c d : A$
then $\exists x, C, D [A \simeq C[x := d], \Gamma \vdash c : \Pi x:D.C \text{ and } \Gamma \vdash d : D]$.

Lemma 7 Weakening.

- If $\Gamma_1 \sqsubseteq \Gamma_2, \Gamma_1 \vdash a : A \text{ and } \Gamma_2 \vdash b : B$
then $\Gamma_2 \vdash a : A$.

Lemma 8 Substitution.

- If $\Gamma_1, x:A, \Gamma_2 \vdash b : B \text{ and } \Gamma_1 \vdash a : A$
then $\Gamma_1, \Gamma_2[x := a] \vdash b[x := a] : B[x := a]$.

Lemma 9 Correctness of Types.

- If $\Gamma \vdash a : A$
then either $A \in \mathcal{S}$ or $\exists s \in \mathcal{S} [\Gamma \vdash A : s]$.

Lemma 10 Closure under β -reduction, Subject Reduction.

- If $\Gamma \vdash a : A, \Gamma \rightarrow \Gamma_0, a \rightarrow a_0 \text{ and } A \rightarrow A_0$
then $\Gamma_0 \vdash a_0 : A_0$.

1.3 Syntax Directed Systems

It is our purpose to describe algorithms, which for given Γ, a and A construct a derivation for $\Gamma \vdash a : A$. It is known that this problem is undecidable for some PTS, e.g. $\lambda\star$, so a semi-algorithm is all we can hope for. There is a trivial such semi-algorithm by enumerating all possible derivations in turn. However we want an *efficient* semi-algorithm, and we want to know something about when the problem is decidable. In [vBJ93] it is shown that if a PTS is normalizing (i.e. all well-typed terms have a normal form) and has a finite set of sorts, \mathcal{S} , then it is decidable, but the given algorithm computes the normal form of types, so while much better than the trivial semi-algorithm above, it is still infeasible.

Consider the rules for correctness, definition 3. There is one rule for deriving the type of a sort, **Srt**; one rule for the type of a variable, **Var**; etc. for **Pi**, **Lda** and **App**. Thus it is natural to construct a derivation of $\Gamma \vdash a : A$ by looking at the shape of a : if a is a sort, use **Srt**, etc. This is not quite right, because **Srt** and **Var** also specify the shape of the context in their conclusion; e.g. **Srt** only works on the empty context. But this is what rule **Wk** is for; it is applicable exactly when we want to use **Srt** or **Var**, but cannot because the context is of the wrong shape. (This is one reason we prefer our restricted weakening rule to the general one in [Bar92].) This improved plan, to build a derivation $\Gamma \vdash a : A$ guided by the shape of Γ and a (which we call the *subject* of the judgement) still has one flaw: the rule **Cnv** may be used at any point in a derivation without changing the shape of the subject. Putting it the other way around, you cannot decide when to use **Cnv** in a derivation by looking at the shape of the subject. A system which does not suffer from such a drawback will be called *syntax directed*. The idea to use a syntax directed presentation for type checking is found in [Mar72] and [Hue89]. We define this notion somewhat informally.

Definition 11 syntax directed.

A set of rules for a relation \vdash is called *nearly syntax directed* if for every Γ, a there is at most one rule with a conclusion $\Gamma \vdash a : A$.

A set of rules for a relation \vdash is called *syntax directed* if for every Γ, a there is at most one rule with a conclusion $\Gamma \vdash a : A$ and this rule (if present) produces exactly one type A for a .

The relation defined by a syntax directed set of rules is necessarily the graph of a partial function $\Gamma, a \mapsto A$. Whether or not the rules allow us to decide this relation depends on the side conditions of the rules, that is, those “premisses” that are not the relation being defined. Similarly, how efficiently we can compute the partial function defined by the rules depends on how efficiently we can compute the side conditions.

Let us assume we have a syntax directed set of rules for a relation equivalent to the correctness relation, \vdash . Given an algorithm to compute the corresponding partial function $\Gamma, a \mapsto A$, which we may call a *type-synthesis* algorithm, how are we to solve our original problem of *typechecking*, that is, given Γ, a and A to construct a derivation of $\Gamma \vdash a : A$? We use our algorithm to compute some type A_0 for a , and some type B for A . Provided that B reduces to some sort s (otherwise A was not a possible type), it now suffices to test A and A_0 for convertibility, and then appeal to the **Cnv** rule.

In order to produce a syntax directed system that is equivalent to the correctness relation, \vdash , we consider how the rule **Cnv** is used in derivations, and are led to propose a nearly syntax directed system of rules defining a relation \vdash_{nsd} . In order to define it, we first introduce some notation.

Notation 12. We write $\Gamma \vdash_{nsd} a \twoheadrightarrow A$ for $\Gamma \vdash_{nsd} a : A_0$ and $A_0 \twoheadrightarrow A$. Similar notations eliding the names of intermediate terms will be used in the rest of the paper.

Definition 13 \vdash_{nsd} . The relation $\vdash_{nsd} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l}
\text{Srt-nsd} \quad \circlearrowleft \vdash_{nsd} s_1 : s_2 \qquad \langle s_1, s_2 \rangle \in \mathcal{A} \\
\\
\text{Var-nsd} \quad \frac{\Gamma \vdash_{nsd} A \twoheadrightarrow s}{\Gamma, x:A \vdash_{nsd} x : A} \\
\\
\text{Wk-nsd} \quad \frac{\Gamma \vdash_{nsd} b : B \quad \Gamma \vdash_{nsd} A \twoheadrightarrow s}{\Gamma, x:A \vdash_{nsd} b : B} \qquad b \in \mathcal{S} \cup \mathcal{V} \\
\\
\text{Pi-nsd} \quad \frac{\Gamma \vdash_{nsd} A \twoheadrightarrow s_1 \quad \Gamma, x:A \vdash_{nsd} B \twoheadrightarrow s_2}{\Gamma \vdash_{nsd} \Pi x:A. B : s_3} \qquad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{Lda-nsd} \quad \frac{\Gamma \vdash_{nsd} A \twoheadrightarrow s_1 \quad \Gamma, x:A \vdash_{nsd} b \twoheadrightarrow B \quad \Gamma, x:A \vdash_{nsd} B \twoheadrightarrow s_2}{\Gamma \vdash_{nsd} \lambda x:A. b : \Pi x:A. B} \qquad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{App-nsd} \quad \frac{\Gamma \vdash_{nsd} a \twoheadrightarrow \Pi x:B. A \quad \Gamma \vdash_{nsd} b \twoheadrightarrow B}{\Gamma \vdash_{nsd} a b : A[x := b]}
\end{array}$$

For this system we can prove soundness.

Lemma 14 Soundness of \vdash_{nsd} .

If $\Gamma \vdash_{nsd} a : A$ then $\Gamma \vdash a : A$.

Proof. By induction on $\Gamma \vdash_{nsd} a : A$, using correctness of types (lemma 9) and closure (lemma 10) for \vdash . \square

Conversely we would like to have

Completeness of \vdash_{nsd} : If $\Gamma \vdash a : A$ then $\exists A_0 \in \mathbb{T} [A \simeq A_0 \text{ and } \Gamma \vdash_{nsd} a : A_0]$.

However we have not been able to prove this. In fact the rule **Lda-nsd** turns out to be an obstacle, both in proving this property directly and of proving some form of closure or correctness of types for \vdash_{nsd} .

1.4 Expansion Postponement

In order to analyse the situation, split **Cnv** into two rules, one for expansion and one for reduction, getting a new system with correctness relation \vdash_{er} .

Definition 15 \vdash_{er} . The relation $\vdash_{er} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the ordinary PTS-rules, but having instead of **Cnv** the following two rules.

$$\text{Red-er} \quad \frac{\Gamma \vdash_{er} a : A}{\Gamma \vdash_{er} a : B} \quad A \rightarrow B$$

$$\text{Exp-er} \quad \frac{\Gamma \vdash_{er} a : A \quad \Gamma \vdash_{er} B : s}{\Gamma \vdash_{er} a : B} \quad B \rightarrow A$$

Lemma 16 Equivalence of \vdash and \vdash_{er} . $\Gamma \vdash a : A \Leftrightarrow \Gamma \vdash_{er} a : A$

Proof. We treat the two cases.

\Rightarrow By induction on $\Gamma \vdash a : A$. The interesting case is the use of **Cnv**: $\Gamma \vdash a : B$ as a consequence of $\Gamma \vdash a : A$, $\Gamma \vdash B : s$ and $A \simeq B$. The induction hypothesis gives $\Gamma \vdash_{er} a : A$ and $\Gamma \vdash_{er} B : s$. By Church-Rosser A and B have a common reduct C , so $\Gamma \vdash_{er} a : C$ by **Red-er** and $\Gamma \vdash_{er} a : B$ by **Exp-er**.

\Leftarrow By induction on $\Gamma \vdash_{er} a : A$ using closure for \vdash in the case **Red-er**. \square

In the spirit of our program to remove non-syntax-directed rules, we ask whether **Exp-er** has any use in derivations (the question was proposed in this form by Henk Barendregt), i.e. we consider a new system, \vdash_r , which does not have the expansion rule.

Definition 17 \vdash_r . The relation $\vdash_r \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the ordinary PTS-rules, but having instead of the rule **Cnv** the following rule.

$$\text{Red-r} \quad \frac{\Gamma \vdash_r a : A}{\Gamma \vdash_r a : B} \quad A \rightarrow B$$

Observe that \vdash_r is equivalent to \vdash_{nsd}

Lemma 18 equivalence of \vdash_r and \vdash_{nsd} .

- i* If $\Gamma \vdash_{nsd} a : A$ then $\Gamma \vdash_r a : A$
- ii* If $\Gamma \vdash_r a : A$ then $\exists A_0 [A_0 \rightarrow A \text{ and } \Gamma \vdash_{nsd} a : A_0]$

The proof is straightforward; in fact we constructed \vdash_{nsd} from \vdash_r by moving uses of **Red-r** to the end of derivations, i.e by permuting **Red-r** downward wherever possible through the premisses of other rules.

Clearly \vdash_r is contained in \vdash , i.e. \vdash_r is sound for \vdash . We would also like to prove \vdash_r is complete for \vdash in the following sense

Expansion Postponement: If $\Gamma \vdash a : A$ then $\exists A_0 [A \rightarrow A_0 \text{ and } \Gamma \vdash_r a : A_0]$.

All attempts to prove this have failed to date. The reason is that no form of a closure lemma has been proved for \vdash_r and this is due to the failure of inductive proofs on the third premiss of the rule **Lda-r**. (This is the crux of the difficulty in proving \vdash_{nsd} complete for \vdash .) Informally we might say that the difficulty is that B moves from right of the colon in the second premiss, where it is an “output” of the subderivation $\Gamma, x:A \vdash_r b : B$, to left of the colon in the third premiss, where it is an “input” to the subderivation $\Gamma, x:A \vdash_r B : s_2$. We don’t have enough structural information on outputs to reason about them as inputs. In proving closure for \vdash we use (the expansion part of) **Cnv** to adjust the shape of outputs, but this is not available in \vdash_r . Closure is a very delicate property of **PTS**. At this moment the authors feel that they cannot honestly call expansion postponement for arbitrary **PTS** a conjecture.

Expansion postponement is an *intensional* concept. It concerns a set of derivation rules for some relation, rather than the relation itself. See remark 3.1 for a frustrating illustration of this point.

We now have, somewhat inexactly,

$$\vdash_{nsd} = \vdash_r \subseteq \vdash_{er} = \vdash$$

Expansion postponement is the property that $\vdash_r \supseteq \vdash_{er}$. Thus, if we assume that expansion postponement holds for some specific **PTS**, then \vdash_{nsd} is a nearly syntax directed presentation of the \vdash relation for that **PTS**.

Remark. Since we cannot now prove \vdash_r has subject reduction or predicate reduction, it is interesting to define a relation \vdash_R , similar to \vdash_r , having the same rules as \vdash except for **Cnv**, which is replaced by

$$\text{Red-R} \quad \frac{\Gamma \vdash_R a : A \quad \Gamma \vdash_R B : s}{\Gamma \vdash_R a : B} \quad A \rightarrow B$$

There is an expansion postponement problem for \vdash_R , which we also cannot prove or disprove. Clearly $\vdash_R \subseteq \vdash_r$, so \vdash_R is “even worse” than \vdash_r . However \vdash_R is easily seen to have the correctness-of-types property, which we cannot prove for \vdash_r . In fact, we can show that correctness-of-types for \vdash_r implies r-expansion-postponement for all *functional* **PTS** (functional **PTS** are defined in section 2.2), but this has not helped in the expansion postponement problem for \vdash_R .

1.5 Plan for this paper

There are two obstructions to deriving syntax directed presentations of arbitrary **PTS**, namely the possible non-determinism of the side conditions, and, more seriously, the third premiss of the **Lda** rule. The purpose of this premiss is to guarantee correctness of the type derived for a lambda term using the **Lda** rule, that is, to make Correctness of Types (lemma 9) true. In this paper we consider two different approaches for removing this troublesome premiss while preserving some form of equivalence with \vdash . We begin in section 2 with further introductory and motivating observations.

Our first approach, in section 3, is to consider a subclass of PTS, called *semi-full*, which have the property that the first two premisses of the **Lda** rule imply the third premiss. This allows a nearly syntax directed presentation for this class, that fails to be syntax directed only because of non-functionality of the relations \mathcal{A} and \mathcal{R} . We introduce a technique of *schematization* to make this presentation syntax directed, and to use it as a typechecking algorithm. This section is a warm-up for the use of schematization in later sections.

Special subclasses of PTS such as semi-full are so redundant that the third premiss of the **Lda** rule is unnecessary. Even for the general PTS this premiss is more restrictive than necessary. In the rest of the paper our second, more general approach, looks more deeply at the presentation of \vdash in order to take advantage of this redundancy. In section 4 we discuss two closely connected typing relations \vdash_o, \vdash_{tp} which are more liberal than \vdash . Alone, the system \vdash_{tp} is too weak to have any reasonable properties (for example, closure obviously fails), but corollaries 54 and 55 show that for terms already known to be well-typed, \vdash_{tp} suffices for correct typechecking. As it stands, this last statement is somewhat inexact, but for now we postpone the precise account of the delicate interaction between \vdash, \vdash_o and \vdash_{tp} .

These liberal relations are used in section 5 to give a nearly syntax directed presentation of the general PTS, in particular replacing the troublesome third premiss of the **Lda** rule with an appeal to \vdash_{tp} . This presentation allows arguments about PTS that we do not know how to carry out in the standard presentation of \vdash given above; for example we prove strengthening for arbitrary PTS. Sections 4 and 5 are the core of the new ideas in this paper, and can be read on their own by a knowledgeable reader.

For non-functional PTS, the **Lda** rule retains an essential non-determinism, as terms may acquire more types by reduction (see example 1 in section 2.2 below), but for functional PTS the nearly syntax directed presentation of section 5 can easily be made syntax directed, as explained in section 6, and can then be seen as an efficient semi-decision procedure for all functional PTS, and as an efficient decision procedure for the decidable ones.

In section 7 we give a syntax directed system that is related to the general PTS, and discuss how to make it into an efficient semi-decision procedure.

A machine-checked presentation From an early draft of this paper, the second and third authors worked to give a completely machine-checked presentation in the LEGO system of all the results of this paper. We have achieved this objective [MP94], except for considerations of schematic terms and judgments (see subsection 3.2 below). For example, the strengthening theorem (63) is formally checked. Further extensions of this work, for example to systems with cumulativity rules for the sorts, may be found in the the third author's forthcoming Ph.D. thesis [Pol94].

2 Preliminary Observations

This section is still introductory. We address several points that are used in future sections, but whose discussion there would be lost in the fray of other matters.

2.1 Making the Application Rule Syntax Directed

Remembering that, lacking a proof of Expansion Postponement, \vdash_{nsd} is not necessarily complete for \vdash , we nonetheless examine the **App-nsd** rule in more detail, as motivation for following sections.

$$\text{App-nsd} \quad \frac{\Gamma \vdash_{nsd} a :=> \Pi x:B.A \quad \Gamma \vdash_{nsd} b :=> B}{\Gamma \vdash_{nsd} a b : A[x := b]}$$

It fails to be syntax directed because we haven't specified when to stop reducing in the left premiss. In particular A , which appears to the right of the colon in the conclusion, is not determined. The intention is to check that a has some functional type (i.e. some Π -type), and that the type of b matches the domain of a 's functional type. For this it is sufficient to do weak-head reduction (denoted by \rightarrow^{wh}) on the type of a , since every Π -type is a weak-head normal form. We are led to an alternative, syntax directed rule

$$\frac{\Gamma \vdash_{nsd} a : \rightarrow^{wh} \Pi x:B.A \quad \Gamma \vdash_{nsd} b : B'}{\Gamma \vdash_{nsd} ab : A[x := b]} \quad B \simeq B'$$

in which the type of ab is uniquely determined by the type of a . Rules of this shape will be used in several following sections.

2.2 Functional Pure Type Systems

We digress to define a well behaved class of PTS that occur commonly in practice.

Definition 19 Functional PTS. A PTS is called *functional* iff

- $\langle s_1, s \rangle \in \mathcal{A}$ and $\langle s_1, s' \rangle \in \mathcal{A}$ implies $s = s'$, and
- $\langle s_1, s_2, s \rangle \in \mathcal{R}$ and $\langle s_1, s_2, s' \rangle \in \mathcal{R}$ implies $s = s'$.

For a functional PTS \mathcal{A} and \mathcal{R} are the graphs of partial functions from \mathcal{S} to \mathcal{S} and from $\mathcal{S} \times \mathcal{S}$ to \mathcal{S} respectively, but we do not necessarily have procedures to compute these functions. It is well known that for functional PTS we have uniqueness of types (cf. [GN91] or [Bar92]).

Lemma 20 Uniqueness of Types. *For functional PTS*

$$\begin{array}{l} \text{If} \quad \Gamma \vdash a : A_0 \text{ and } \Gamma \vdash a : A_1 \\ \text{then} \quad A_0 \simeq A_1. \end{array}$$

Subject Expansion By the Subject Reduction Lemma (lemma 10) we know that a term does not lose any types by reduction. Might a well-typed term *gain* types by reduction? For functional PTS this cannot happen.

Lemma 21 Subject Expansion. *For PTS with Uniqueness of Types*

$$\begin{array}{l} \text{If} \quad \Gamma \vdash a : A, b \rightarrow a, \text{ and } \Gamma \vdash b : B \\ \text{then} \quad \Gamma \vdash b : A \end{array}$$

The condition $\Gamma \vdash b : B$ in this lemma, that b has some type, is necessary to rule out untypable expansions.

Proof. By closure (lemma 10) $\Gamma \vdash a : B$. By uniqueness of types $A \simeq B$. By correctness of types (lemma 9) $A \in \mathcal{S}$ or $\Gamma \vdash A : s$ for some s . In the first case, $B \rightarrow A$, and we are done by closure. In the second case we are done by the rule **Cnv**. \square

The general PTS does not have the subject expansion property:

Example 1. Consider the non-functional PTS

$$\begin{array}{l} \mathcal{S} = \{\star, \Delta, \nabla, \square\} \\ \mathcal{A} = \{\langle \star, \Delta \rangle, \langle \star, \nabla \rangle, \langle \Delta, \square \rangle\} \\ \mathcal{R} = \{\langle \square, \square, \square \rangle\} \end{array}$$

Let $a = (\lambda x: \Delta .x) \star$. We have $a \rightarrow \star$, with $\circlearrowleft \vdash \star : \Delta$ and $\circlearrowleft \vdash \star : \nabla$. Consider the types of a in the empty context. First notice that $\circlearrowleft \vdash a : \Delta$

$$\frac{\frac{\frac{\circlearrowleft \vdash \Delta : \square \quad \circlearrowleft \vdash \Delta : \square \quad \circlearrowleft \vdash \Delta : \square}{x:\Delta \vdash x : \Delta} \quad \frac{\circlearrowleft \vdash \Delta : \square}{x:\Delta \vdash \Delta : \square} \quad \langle \square, \square, \square \rangle \in \mathcal{R}}{\circlearrowleft \vdash \lambda x: \Delta .x : \Pi x: \Delta .\Delta} \quad \circlearrowleft \vdash \star : \Delta}{\circlearrowleft \vdash (\lambda x: \Delta .x) \star : \Delta}$$

Now we claim that $\circlearrowleft \vdash a : \nabla$ is *not* derivable, so the well-typed term a gains types by reduction to \star . To prove this claim we use the generation lemma (lemma 6). In general, if $\circlearrowleft \vdash a : X$ for some X , then there exist Y, Z , with $\circlearrowleft \vdash \lambda x: \Delta .x : \Pi x:Y.Z$ and $X \simeq Z[x := \star]$. By the generation lemma again, $x:\Delta \vdash x : W, x:\Delta \vdash W : s$, and $Z \simeq W$. By the generation lemma again, $W \simeq \Delta$, hence $\circlearrowleft \nvdash a : \nabla$. Notice also that this argument never appeals to “there is no rule of a certain shape”; even if we expand \mathcal{R} to $\mathcal{S} \times \mathcal{S} \times \mathcal{S}$, every type of a converts with Δ .

Examining this example more closely, observe that a variable can have only one type (up to conversion), while a sort can have several types in non-functional PTS (see [vBJ93] for a deep analysis of this point). In the example, we use reduction to replace a variable by a sort, increasing the types derivable for a term. However, this is not the only way that subject expansion can fail: consider the term $b = (\lambda x: \Delta .\star) \star$. We leave it to the reader to check that again $b \rightarrow \star$, $\circlearrowleft \vdash b : \Delta$ but $\circlearrowleft \nvdash b : \nabla$. In this case the reason is that $\circlearrowleft \vdash \Delta : \square$, but ∇ has no type at all.

Must this process of obtaining more types by reduction eventually stop or are there (non-functional) PTS with infinitely many sorts, and (non normalizing) terms in these PTS that keep acquiring more and more types by reduction?

As an aside, notice that this PTS is strongly normalizing. It can be mapped into three levels of a strongly normalizing predicative type hierarchy (for definiteness we mention ECC [Luo90], but much weaker systems will do) by the PTS-morphism ([Geu, Geu93])

$$\star \mapsto \text{Type}(0) \quad \Delta \mapsto \text{Type}(1) \quad \nabla \mapsto \text{Type}(1) \quad \square \mapsto \text{Type}(2).$$

Since this mapping preserves sorts, axioms and rules, any well-typed term in the system above has a well-typed image, and therefore strongly normalizes. Further, since \mathcal{S} is finite, this system has decidable typechecking [vBJ93]. \square

It is quite difficult to think of a PTS of independent interest that is not functional!

2.3 Side Conditions and Decidability of Typechecking

Remember that we assumed \mathcal{A} and \mathcal{R} are decidable. Clearly, however, the set

$$\mathcal{A}_\bullet \triangleq \{ s \in \mathcal{S} \mid \exists s_0 \in \mathcal{S} [\langle s, s_0 \rangle \in \mathcal{A}] \}$$

may not be decidable³. (This is true even for functional PTS.) Furthermore, if \mathcal{A}_\bullet is not decidable, then the PTS does not have decidable typechecking, because

$$x:s \vdash x : s \quad \Leftrightarrow \quad s \in \mathcal{A}_\bullet.$$

Similarly if the relation

$$\mathcal{R}_\bullet \triangleq \{ \langle s_1, s_2 \rangle \in \mathcal{S} \times \mathcal{S} \mid \exists s_3 \in \mathcal{S} [\langle s_1, s_2, s_3 \rangle \in \mathcal{R}] \}$$

³ The set of *topsorts*, $\mathcal{A}^\bullet \triangleq \mathcal{S} \setminus \mathcal{A}_\bullet$, is interesting in its own right; see [Ber90] or [Pol94].

is not decidable, then the PTS may not have decidable typechecking, but since some Π -rules may not be usable the situation is not as clear.

In a functional PTS, \mathcal{A} (respectively \mathcal{R}) is the graph of a partial function. If \mathcal{A}_\bullet (respectively \mathcal{R}_\bullet) is decidable, then we can compute that function: use decidability of \mathcal{A}_\bullet (respectively \mathcal{R}_\bullet) to decide if the function is defined on particular input, and if so use decidability of \mathcal{A} (respectively \mathcal{R}) to search the denumerable set \mathcal{S} for the function value.

As \mathcal{A} and \mathcal{R} are arbitrary given relations, there is not much interesting to say about these side conditions.

2.4 An Optimization: Valid Contexts

There is one easy optimization of the definition of PTS which will shorten derivations, and therefore also the checking algorithms which could produce such derivations. In the premisses of the rules **Wk**, **Pi**, **Lda**, **App** and **Cnv** the context Γ occurs more than once, and in order to construct a complete derivation its validity must be checked in each subderivation in which it appears. It is much more efficient to assume that we start with a valid context, and only check that when rules extend the context (i.e. the right premiss of **Pi** and the two right premisses of **Lda**) they maintain validity. This is also more in keeping with the implementations which are actually used, where we work in a “current context” of mathematical assumptions. (In practice, the context also contains definitions, but in this paper we will not discuss definitions.)

We formalize this optimization in the following definition. $\Gamma \vdash_{vtyp} a : A$ should be interpreted as “ a has type A relative to the (possibly incorrect) context Γ ” and $\Gamma \vdash_{vcxt}$ as “ Γ is a valid (or correct) context”.

Definition 22 \vdash_{vtyp} and \vdash_{vcxt} . The relation $\vdash_{vtyp} \subseteq \mathbf{C} \times \mathbf{T} \times \mathbf{T}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l}
\text{Srt-vtyp} \quad \Gamma \vdash_{vtyp} s_1 : s_2 \qquad \langle s_1, s_2 \rangle \in \mathcal{A} \\
\text{Var-vtyp} \quad \Gamma \vdash_{vtyp} x : A \qquad x:A \in \Gamma \\
\text{Pi-vtyp} \quad \frac{\Gamma \vdash_{vtyp} A : s_1 \quad \Gamma, x:A \vdash_{vtyp} B : s_2}{\Gamma \vdash_{vtyp} \Pi x:A. B : s_3} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad x \notin \text{dom}(\Gamma) \\
\text{Lda-vtyp} \quad \frac{\Gamma \vdash_{vtyp} A : s_1 \quad \Gamma, x:A \vdash_{vtyp} b : B \quad \Gamma, x:A \vdash_{vtyp} B : s_2}{\Gamma \vdash_{vtyp} \lambda x:A. b : \Pi x:A. B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad x \notin \text{dom}(\Gamma) \\
\text{App-vtyp} \quad \frac{\Gamma \vdash_{vtyp} a : \Pi x:B. A \quad \Gamma \vdash_{vtyp} b : B}{\Gamma \vdash_{vtyp} a b : A[x := b]} \\
\text{Cnv-vtyp} \quad \frac{\Gamma \vdash_{vtyp} a : A \quad \Gamma \vdash_{vtyp} B : s}{\Gamma \vdash_{vtyp} a : B} \qquad A \simeq B
\end{array}$$

The predicate \vdash_{vcxt} is the smallest predicate on \mathbf{C} satisfying the following rules.

$$\begin{array}{l}
\text{Nil-vcxt} \quad \emptyset \vdash_{vcxt} \\
\text{Cons-vcxt} \quad \frac{\Gamma \vdash_{vcxt} \quad \Gamma \vdash_{vtyp} A : s}{\Gamma, x:A \vdash_{vcxt}} \quad x \notin \text{dom}(\Gamma)
\end{array}$$

Lemma 23 *Equivalence of \vdash and \vdash_{vtyp} .*

$$\Gamma \vdash a : A \quad \Leftrightarrow \quad \Gamma \vdash_{vctx} \text{ and } \Gamma \vdash_{vtyp} a : A$$

In order to prove this, first observe that weakening is trivial for \vdash_{vtyp} . Also it is easy to derive generation properties (compare lemma 6) for sorts and variables.

Lemma 24 *Weakening and Generation for \vdash_{vtyp} .*

- i* If $\Gamma_1 \vdash_{vtyp} a : A$ and $\Gamma_1 \sqsubseteq \Gamma_2$
then $\Gamma_2 \vdash_{vtyp} a : A$
- ii* If $\Gamma \vdash_{vtyp} x : A$
then $x:A_0 \in \Gamma$, where either $A = A_0$ or $[\Gamma \vdash_{vtyp} A : s \text{ and } A \simeq A_0]$
- iii* If $\Gamma \vdash_{vtyp} s : A$
then $\langle s, s_0 \rangle \in \mathcal{A}$ and either $A = s_0$ or $[\Gamma \vdash_{vtyp} A : s_1 \text{ and } A \simeq s_0]$

Proof of lemma 23.

\Rightarrow By induction on $\Gamma \vdash a : A$, using 24 for the case of **Wk**.

\Leftarrow By induction on the lexicographic order, first the sum of the lengths of the derivations of $\Gamma \vdash_{vctx}$ and $\Gamma \vdash_{vtyp} a : A$, second the length of the derivation of $\Gamma \vdash_{vtyp} a : A$. \square

In what follows we consider many different rule systems related to **PTS**. These systems could be based on the optimized presentation \vdash_{vtyp} , or could be optimized by this technique as they come up. In fact, for simplicity, we will start from the basic **PTS**, \vdash , and will not mention this optimization again, leaving its application to the reader.

3 Semi-full Pure Type Systems

A **PTS** is called *full* iff for all $s_1, s_2 \in \mathcal{S}$ there exists $s_3 \in \mathcal{S}$ with $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. In full **PTS** the troublesome third premiss of the **Lda**-rule can be omitted. Let us focus on the **Lda**-rule:

$$\text{Lda} \quad \frac{\Gamma \vdash A : s_1 \quad \Gamma, x:A \vdash b : B \quad \Gamma, x:A \vdash B : s_2}{\Gamma \vdash \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

The purpose of premisses $\Gamma \vdash A : s_1$ and $\Gamma, x:A \vdash B : s_2$ is to guarantee that the type $\Pi x:A.B$ will be well-formed. But we know from the premiss $\Gamma, x:A \vdash b : B$ that $\Gamma \vdash A : s_A$ for some s_A , and by correctness of types we have either $B \in \mathcal{S}$ or $\Gamma, x:A \vdash B : s_B$ for some s_B . In this latter case, for full **PTS**, we conclude there exists s with $\langle s_A, s_B, s \rangle \in \mathcal{R}$, so $\Pi x:A.B$ is well formed, and it seems sound to replace the right premiss of the **Lda**-rule by the requirement that $B \notin \mathcal{A}^\bullet$ or rather, making a positive statement, that $B \in \mathcal{S}$ implies $\langle B, s_B \rangle \in \mathcal{A}$. We can generalize this idea somewhat beyond full **PTS**.

Definition 25 **Semi-Full**. A **PTS** is *semi-full* iff for all $s_1 \in \mathcal{S}$

$$\exists s_2, s_3 [\langle s_1, s_2, s_3 \rangle \in \mathcal{R}] \quad \Rightarrow \quad \forall s_2 \exists s_3 [\langle s_1, s_2, s_3 \rangle \in \mathcal{R}].$$

While the Pure Calculus of Constructions, $\lambda P\omega$, and various extensions with type universes are full (e.g. **ECC** [Luo90]), the Edinburgh Logical Framework, λP , is semi-full. In [HHP92] it is shown that λP is decidable by giving an algorithm which computes the normal form of types; a very expensive computation in practice. We will show that the algorithm which is actually used in **LEGO** is both sound and complete. Notice that $\lambda P\omega$ and λP are the only semi-full systems in the λ -cube.

so the middle of the three vertical relations remains an inequality. The rightmost equality on the bottom row is by straightforward induction: this is what we gain by removing the right premiss of the lambda rule. As you see, we cannot yet conclude that every \vdash -derivation can be mimicked up to conversion by a \vdash_r -derivation, or by a \vdash_R -derivation, even when restricted to semi-full PTS.

As **Lda-sf** does not have the troublesome third premiss, it is now straightforward to define a nearly syntax directed system, which is sound and complete for \vdash_{sf} .

Definition 28 \vdash_{sfnsd} . The relation $\vdash_{sfnsd} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l}
\text{Srt-sfnsd} \quad \bigcirc \vdash_{sfnsd} s_1 : s_2 \quad \langle s_1, s_2 \rangle \in \mathcal{A} \\
\\
\text{Var-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} A : \Rightarrow s}{\Gamma, x:A \vdash_{sfnsd} x : A} \\
\\
\text{Wk-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} b : B \quad \Gamma \vdash_{sfnsd} A : \Rightarrow s}{\Gamma, x:A \vdash_{sfnsd} b : B} \quad b \in \mathcal{S} \cup \mathcal{V} \\
\\
\text{Pi-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} A : \Rightarrow s_1 \quad \Gamma, x:A \vdash_{sfnsd} B : \Rightarrow s_2}{\Gamma \vdash_{sfnsd} \Pi x:A.B : s_3} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{Lda1-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} A : \Rightarrow s_1 \quad \Gamma, x:A \vdash_{sfnsd} b : B}{\Gamma \vdash_{sfnsd} \lambda x:A.b : \Pi x:A.B} \quad \begin{array}{l} \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\ B \notin \mathcal{S} \end{array} \\
\\
\text{Lda2-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} A : \Rightarrow s_1 \quad \Gamma, x:A \vdash_{sfnsd} b : s_b}{\Gamma \vdash_{sfnsd} \lambda x:A.b : \Pi x:A.s_b} \quad \begin{array}{l} \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\ \langle s_b, s_4 \rangle \in \mathcal{A} \end{array} \\
\\
\text{App-sfnsd} \quad \frac{\Gamma \vdash_{sfnsd} a : \Rightarrow^{wh} \Pi x:B_0.A \quad \Gamma \vdash_{sfnsd} b : B_1}{\Gamma \vdash_{sfnsd} a b : A[x := b]} \quad B_0 \simeq B_1
\end{array}$$

Lemma 29 *Equivalence of \vdash_{sf} and \vdash_{sfnsd} . For semi-full PTS*

- i* If $\Gamma \vdash_{sfnsd} a : A$ then $\Gamma \vdash_{sf} a : A$
- ii* If $\Gamma \vdash_{sf} a : A$ then $\exists A_0 [A \simeq A_0 \text{ and } \Gamma \vdash_{sfnsd} a : A_0]$

Proof. We prove both parts.

i Induction on $\Gamma \vdash_{sfnsd} a : A$, using the fact that closure holds for \vdash_{sf} by equivalence (lemma 27) and closure for \vdash (lemma 10). We treat the case of **App-sfnsd**: $\Gamma \vdash_{sfnsd} a b : A[x := b]$ from $\Gamma \vdash_{sfnsd} a : A_0$, $A_0 \Rightarrow^{wh} \Pi x:B_0.A$, $\Gamma \vdash_{sfnsd} b : B_1$ and $B_0 \simeq B_1$. By induction $\Gamma \vdash_{sf} a : A_0$ and $\Gamma \vdash_{sf} b : B_0$. Take a common reduct B of B_0 and B_1 . It follows by closure for \vdash_{sf} that $\Gamma \vdash_{sf} a : \Pi x:B.A$ and $\Gamma \vdash_{sf} b : B$, and therefore $\Gamma \vdash_{sfnsd} a b : A[x := b]$ by **App-sf**.

ii Induction on $\Gamma \vdash_{sf} a : A$, using correctness of types for \vdash_{sf} , which is again a consequence of equivalence (lemma 27) and correctness of types for \vdash (lemma 9). Again we do only one case, **Lda1-sf**: $\Gamma \vdash_{sf} \lambda x:A.b : \Pi x:A.B$ from $\Gamma \vdash_{sf} A : s_1$, $\Gamma, x:A \vdash_{sf} b : B$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$, while $B \notin \mathcal{S}$. By induction hypothesis $\Gamma \vdash_{sfnsd} A : A_0 \simeq s_1$ and $\Gamma, x:A \vdash_{sfnsd} b : B_0 \simeq B$. It remains to show that $B_0 \in \mathcal{S}$ implies $\langle B_0, s \rangle \in \mathcal{A}$ for some s . So suppose $B_0 \in \mathcal{S}$, hence $B \Rightarrow B_0$. By correctness of types $\Gamma \vdash_{sf} B : s_B$, hence $\Gamma \vdash_{sf} B_0 : s_B$ by closure, so by generation $\langle B_0, s_B \rangle \in \mathcal{A}$ and we are done. \square

3.2 A Syntax Directed system for Semi-full PTS's

The only failure of syntax-directedness in the rules of system \vdash_{sfnsd} is due to possible non-functionality of the PTS. For example, if $\langle s_1, s_2 \rangle \in \mathcal{A}$ and $\langle s_1, s'_2 \rangle \in \mathcal{A}$, the rule **Srt-sfnsd** can be used to derive the judgements $\odot \vdash_{sfnsd} s_1 : s_2$ and $\odot \vdash_{sfnsd} s_1 : s'_2$. Similarly **Pi-sfnsd**, **Lda1-sfnsd** and **Lda2-sfnsd** may be non-syntax-directed due to non-functionality of \mathcal{R} . For functional PTS \vdash_{sfnsd} is already syntax directed, and may be used directly for typechecking.

For non-functional semi-full PTS we will remove this non-determinism by a method suggested in [Hue87, HP91, Pol92], and for this purpose we first introduce some technical machinery.

Schematic Terms Introduce a new set Σ , disjoint from \mathcal{V} and \mathcal{S} . Elements of Σ will be called *sort variables* and will be denoted by $\sigma, \sigma_0, \sigma_1, \dots$. The symbols $\alpha, \beta, \gamma, \dots$ will range over $\Sigma \cup \mathcal{S}$.

Definition 30 Schematic Terms. The set, T^Σ , of *schematic terms* is the smallest set satisfying:

$$\begin{aligned} \Sigma &\subseteq \mathsf{T}^\Sigma, \\ \text{If } X \in \mathsf{T}^\Sigma \text{ and } b \in \mathsf{T} &\text{ then } Xb \in \mathsf{T}^\Sigma, \\ \text{If } A \in \mathsf{T}, X \in \mathsf{T}^\Sigma \text{ and } x \in \mathcal{V} &\text{ then } \Pi x:A.X \in \mathsf{T}^\Sigma. \end{aligned}$$

We use X, Y, Z to range over $\mathsf{T} \cup \mathsf{T}^\Sigma$.

In this section we will only need schematic terms of the form $\Pi x_1:A_1 \dots \Pi x_n:A_n.\sigma$, but in section 7 the more general notion will be used.

If $X \in \mathsf{T} \cup \mathsf{T}^\Sigma$ we denote by $\mathsf{SV}(X)$ the sort variables of X . Substitution and β - and π -reduction are easily extended to T^Σ .

A *sort assignment* is a partial function from Σ to \mathcal{S} . If $X \in \mathsf{T}^\Sigma$ and $\mathsf{SV}(X) \in \mathbf{dom}(\phi)$ then $\phi X \in \mathsf{T}$ is defined in the obvious way. Substitution of pseudoterms and reduction are conserved by assignments, i.e. $X \rightarrow Y$ iff $\phi X \rightarrow \phi Y$ for any assignment ϕ .

A *constraint* is a finite set of formulas of the form $\langle \alpha, \beta \rangle \in \mathcal{A}$, $\langle \alpha, \beta, \gamma \rangle \in \mathcal{R}$ and $\alpha = \beta$. The set of all constraints is denoted by \mathbf{Cnstr} . If \mathcal{C} is a constraint then $\mathsf{SV}(\mathcal{C})$ denotes the set of sort variables occurring in \mathcal{C} . A sort assignment ϕ is said to *satisfy* a constraint \mathcal{C} , written $\phi \models \mathcal{C}$, iff $\mathsf{SV}(\mathcal{C}) \subseteq \mathbf{dom}(\phi)$ and each of the propositions in $\phi\mathcal{C}$ is true. A constraint, \mathcal{C} , is said to be *consistent* or *satisfiable*, written $\mathcal{C}\mathbf{con}$, if there is a sort assignment satisfying it.

We say that a PTS has *decidable constraints* if for every constraint, \mathcal{C} , it is decidable whether or not \mathcal{C} is satisfiable. If \mathcal{S} is finite then this condition is clearly fulfilled, but PTS with \mathcal{S} infinite may also have this property, as the following example shows.

Example 2. Consider the PTS with infinitely many stratified universes

$$\begin{aligned} \mathcal{S} &= \{ \mathit{Type}_i \mid i \in \mathbf{N} \} \\ \mathcal{A} &= \{ \langle \mathit{Type}_i, \mathit{Type}_j \rangle \mid i < j \} \\ \mathcal{R} &= \{ \langle \mathit{Type}_i, \mathit{Type}_j, \mathit{Type}_k \rangle \mid i \leq k, j \leq k \} \end{aligned}$$

The constraints of this theory only contain formulas of the form $\alpha < \beta$ and $\alpha \leq \beta$ ($\alpha = \beta$ being expressible by the pair $\alpha \leq \beta$ and $\beta \leq \alpha$). All such sets are decidable (in time polynomial in both the number of constraints and the number of variables and constants) by checking acyclicity of a directed graph whose nodes are variables and constants, and whose edges are the relations $<$ and \leq . This result is due to Chan [Cha77], and its application for solving constraints of typechecking was suggested by Huet [Hue87] and detailed in [HP91]. \square

Our purpose will be to compute for any pair Γ, a a term $X \in \mathsf{T} \cup \mathsf{T}^\Sigma$ and a constraint \mathcal{C} , (we denote this by $\Gamma \vdash_{\text{sfnsd}} a : X \mid \mathcal{C}$), such that

- i If $\Gamma \vdash_{\text{sfnsd}} a : X \mid \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\Gamma \vdash a : \phi X$
- ii If $\Gamma \vdash a : A$ then $\Gamma \vdash_{\text{sfnsd}} a : X \mid \mathcal{C}$ and $\exists \phi [\phi \models \mathcal{C} \text{ and } \phi X \simeq A]$

The tools developed so far will enable us to make the rules **Srt-sfnsd** and **Pi-sfnsd** syntax directed. Let us turn our attention now to the rule **App-sfnsd**.

$$\text{App-sfnsd} \quad \frac{\Gamma \vdash_{\text{sfnsd}} a : \rightarrow^{wh} \Pi x : B_0 . A \quad \Gamma \vdash_{\text{sfnsd}} b : B_1}{\Gamma \vdash_{\text{sfnsd}} a b : A[x := b]} \quad B_0 \simeq B_1$$

We have already made the reduction of the type of a to $\Pi x : B_0 . A$ deterministic by requiring weak head-reduction; now consider the convertibility of B_0 and B_1 . B_1 could be a schematic term, so we must define the constraints under which two terms $X, Y \in \mathsf{T} \cup \mathsf{T}^\Sigma$ are convertible, in order to give the application rule of \vdash_{sfnsd} .

Definition 31 Schematic Equality and Schematic Conversion. *schematic equality* is the smallest relation $=_{\Sigma} _ \mid _ \subseteq (\mathsf{T} \cup \mathsf{T}^\Sigma) \times (\mathsf{T} \cup \mathsf{T}^\Sigma) \times \mathbf{Cnstr}$, satisfying

- $\alpha =_{\Sigma} \beta \mid \{\alpha = \beta\}$.
- If $X =_{\Sigma} Y \mid \mathcal{C}$ then $\Pi x : A . X =_{\Sigma} \Pi x : A . Y \mid \mathcal{C}$ and $X a =_{\Sigma} Y a \mid \mathcal{C}$.
- If $X = Y$ then $X =_{\Sigma} Y \mid \emptyset$.

schematic β -conversion is the relation $\simeq _ \mid _ \subseteq (\mathsf{T} \cup \mathsf{T}^\Sigma) \times (\mathsf{T} \cup \mathsf{T}^\Sigma) \times \mathbf{Cnstr}$ defined by

$$X \simeq Y \mid \mathcal{C} \triangleq \exists X_0, Y_0 [X \rightarrow X_0, Y \rightarrow Y_0 \text{ and } X_0 =_{\Sigma} Y_0 \mid \mathcal{C}]$$

Clearly for normalizing X and Y the relation $\exists \mathcal{C} [X \simeq Y \mid \mathcal{C}]$ is decidable, and for arbitrary X and Y this relation is semi-decidable.

Lemma 32 Properties of Schematic Conversion.

- If $X \simeq Y \mid \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\phi X \simeq \phi Y$
- If $\phi X \simeq \phi Y$ then $\exists \mathcal{C} [X \simeq Y \mid \mathcal{C} \text{ and } \phi \models \mathcal{C}]$

The syntax directed system

Notation 33. We extend notation 12 and write

$$\Gamma \vdash_{\text{sfnsd}} a : \rightarrow X \mid \mathcal{C} \quad \text{for} \quad \Gamma \vdash_{\text{sfnsd}} a : X_0 \mid \mathcal{C} \quad \text{and} \quad X_0 \rightarrow X.$$

Definition 34 \vdash_{sfsd} . The relation $\vdash_{sfsd} \subseteq \mathbf{C} \times \mathbf{T} \times (\mathbf{T} \cup \mathbf{T}^\Sigma) \times \mathbf{Cnstr}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l}
\text{Srt-sfsd} \quad \bigcirc \vdash_{sfsd} s : \sigma \mid \{\langle s, \sigma \rangle \in \mathcal{A}\} \\
\\
\text{Var-sfsd} \quad \frac{\Gamma \vdash_{sfsd} A : \rightarrow \alpha \mid \mathcal{C}}{\Gamma, x:A \vdash_{sfsd} x : A \mid \mathcal{C}} \\
\\
\text{Wk-sfsd} \quad \frac{\Gamma \vdash_{sfsd} b : X \mid \mathcal{C} \quad \Gamma \vdash_{sfsd} A : \rightarrow \alpha \mid \mathcal{D}}{\Gamma, x:A \vdash_{sfsd} b : X \mid \mathcal{C} \cup \mathcal{D}} \quad b \in \mathbf{S} \cup \mathbf{V} \\
\\
\text{Pi-sfsd} \quad \frac{\Gamma \vdash_{sfsd} A : \rightarrow \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{sfsd} B : \rightarrow \beta \mid \mathcal{D}}{\Gamma \vdash_{sfsd} \Pi x:A. B : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}} \\
\\
\text{Lda1-sfsd} \quad \frac{\Gamma \vdash_{sfsd} A : \rightarrow \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{sfsd} b : X \mid \mathcal{D}}{\Gamma \vdash_{sfsd} \lambda x:A. b : \Pi x:A. X \mid \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R}\}} \quad X \notin \Sigma \cup \mathbf{S} \\
\\
\text{Lda2-sfsd} \quad \frac{\Gamma \vdash_{sfsd} A : \rightarrow \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{sfsd} b : \beta \mid \mathcal{D}}{\Gamma \vdash_{sfsd} \lambda x:A. b : \Pi x:A. \beta \mid \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R}\} \cup \{\langle \beta, \sigma_4 \rangle \in \mathcal{A}\}} \\
\\
\text{App-sfsd} \quad \frac{\Gamma \vdash_{sfsd} a : \rightarrow^{wh} \Pi x:B. X \mid \mathcal{C} \quad \Gamma \vdash_{sfsd} b : Y \mid \mathcal{D}}{\Gamma \vdash_{sfsd} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}} \quad Y \simeq B \mid \mathcal{E}
\end{array}$$

In the definition we assume all the sort variables which are introduced to be fresh. For example in rule **Pi-sfsd** we assume σ to be a fresh variable, and also $\mathbf{SV}(\mathcal{C})$ and $\mathbf{SV}(\mathcal{D})$ should be disjoint. Similarly in rule **App-sfsd** $\mathbf{SV}(\mathcal{C})$ and $\mathbf{SV}(\mathcal{D})$ will be disjoint, but $\mathbf{SV}(\mathcal{D})$ and $\mathbf{SV}(\mathcal{E})$ might intersect.

Note that for any Γ and a there is (up to the names of sort variables) at most one derivable judgement $\Gamma \vdash_{sfsd} a : X \mid \mathcal{C}$, and at most one derivation of such a judgement. Further, derivations in \vdash_{sfsd} depend on the language of the PTS, but are independent of the axioms, \mathcal{A} , and Π -rules, \mathcal{R} , of the PTS; it is in the satisfaction of constraints that a particular PTS is observed.

Lemma 35 Equivalence of \vdash_{sfnsd} and \vdash_{sfsd} .

- i* If $\Gamma \vdash_{sfsd} a : X \mid \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\Gamma \vdash_{sfnsd} a : \phi X$
- ii* If $\Gamma \vdash_{sfnsd} a : A$ then $\exists X, \mathcal{C}, \phi [\phi \models \mathcal{C}, A = \phi X \text{ and } \Gamma \vdash_{sfsd} a : X \mid \mathcal{C}]$

Proof.

i Induction on $\Gamma \vdash_{sfsd} a : X \mid \mathcal{C}$. We treat **App-sfsd**:

$$\text{App-sfsd} \quad \frac{\Gamma \vdash_{sfsd} a : X' \mid \mathcal{C} \quad X' \rightarrow^{wh} \Pi x:B. X \quad \Gamma \vdash_{sfsd} b : Y \mid \mathcal{D}}{\Gamma \vdash_{sfsd} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}} \quad Y \simeq B \mid \mathcal{E}$$

Now suppose $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$. By induction

$$\Gamma \vdash_{sfnsd} a : \phi X' \rightarrow^{wh} \Pi x:B. \phi X \quad \text{and} \quad \Gamma \vdash_{sfnsd} b : \phi Y.$$

By lemma 32 $\phi Y \simeq B$, so $\Gamma \vdash_{sfnsd} a b : (\phi X)[x := b] = \phi(X[x := b])$.

ii Induction on $\Gamma \vdash_{sfn\text{sd}} a : A$. We consider **Lda1-sfn\text{sd}**:

$$\frac{\Gamma \vdash_{sfn\text{sd}} A : C \rightarrow s_1 \quad \Gamma, x:A \vdash_{sfn\text{sd}} b : B}{\Gamma \vdash_{sfn\text{sd}} \lambda x:A. b : \Pi x:A. B} \quad \begin{array}{l} \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\ B \notin \mathcal{S} \end{array}$$

By induction

$$\begin{array}{l} \exists X, \mathcal{C}, \phi_{\mathcal{C}}, \alpha [\phi_{\mathcal{C}} \models \mathcal{C}, C = \phi_{\mathcal{C}} X \text{ and } \Gamma \vdash_{sfn\text{sd}} A : X \mid \mathcal{C} \text{ where } X \rightarrow \alpha] \\ \exists Y, \mathcal{D}, \phi_{\mathcal{D}} [\phi_{\mathcal{D}} \models \mathcal{D}, B = \phi_{\mathcal{D}} Y \text{ and } \Gamma \vdash_{sfn\text{sd}} b : Y \mid \mathcal{D}] \end{array}$$

where we may assume $\text{SV}(\mathcal{C})$ disjoint from $\text{SV}(\mathcal{D})$ (change the names of sort variables if necessary). Since $\phi_{\mathcal{D}} Y = B \notin \mathcal{S}$, $Y \notin \Sigma \cup \mathcal{S}$, and by **Lda1-sfn\text{sd}**

$$\Gamma \vdash_{sfn\text{sd}} \lambda x:A. b : \Pi x:A. Y \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R} \}$$

We must have $\phi_{\mathcal{C}} \alpha = s_1$, so take

$$\phi = \phi_{\mathcal{C}} \cup \phi_{\mathcal{D}} \cup \{ \sigma_2 \mapsto s_2, \sigma_3 \mapsto s_3 \}$$

and $\phi \models \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R} \}$ as required. \square

An algorithm for typechecking normalizing semi-full PTS with decidable constraints We cannot use $\vdash_{sfn\text{sd}}$ to decide \vdash because $\vdash_{sfn\text{sd}}$ does not incrementally check consistency of constraints, so we may try to normalize a schematic term that has no well-typed instance. We now fix this deficiency.

Definition 36 \vdash_{sfa} . The relation $\vdash_{sfa} \subseteq \mathbf{C} \times \mathbf{T} \times (\mathbf{T} \cup \mathbf{T}^{\Sigma}) \times \mathbf{Cnstr}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l} \text{Srt-sfa} \quad \circ \vdash_{sfa} s : \sigma \mid \{ \langle s, \sigma \rangle \in \mathcal{A} \} \\ \\ \text{Var-sfa} \quad \frac{\Gamma \vdash_{sfa} A : X \mid \mathcal{C} \quad \mathcal{C} \text{con} \quad X \rightarrow \alpha}{\Gamma, x:A \vdash_{sfa} x : A \mid \mathcal{C}} \\ \\ \text{Wk-sfa} \quad \frac{\Gamma \vdash_{sfa} b : X \mid \mathcal{C} \quad \Gamma \vdash_{sfa} A : Y \mid \mathcal{D} \quad \mathcal{D} \text{con} \quad Y \rightarrow \alpha}{\Gamma, x:A \vdash_{sfa} b : X \mid \mathcal{C} \cup \mathcal{D}} \quad b \in \mathcal{S} \cup \mathcal{V} \\ \\ \text{Pi-sfa} \quad \frac{\Gamma \vdash_{sfa} A : X \mid \mathcal{C} \quad \mathcal{C} \text{con} \quad X \rightarrow \alpha \quad \Gamma, x:A \vdash_{sfa} B : Y \mid \mathcal{D} \quad \mathcal{D} \text{con} \quad Y \rightarrow \beta}{\Gamma \vdash_{sfa} \Pi x:A. B : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \beta, \sigma \rangle \in \mathcal{R} \}} \\ \\ \text{Lda1-sfa} \quad \frac{\Gamma \vdash_{sfa} A : Y \mid \mathcal{C} \quad \mathcal{C} \text{con} \quad Y \rightarrow \alpha \quad \Gamma, x:A \vdash_{sfa} b : X \mid \mathcal{D}}{\Gamma \vdash_{sfa} \lambda x:A. b : \Pi x:A. X \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R} \}} \quad X \notin \Sigma \cup \mathcal{S} \\ \\ \text{Lda2-sfa} \quad \frac{\Gamma \vdash_{sfa} A : Y \mid \mathcal{C} \quad \mathcal{C} \text{con} \quad Y \rightarrow \alpha \quad \Gamma, x:A \vdash_{sfa} b : \beta \mid \mathcal{D}}{\Gamma \vdash_{sfa} \lambda x:A. b : \Pi x:A. \beta \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \sigma_2, \sigma_3 \rangle \in \mathcal{R} \} \cup \{ \langle \beta, \sigma_4 \rangle \in \mathcal{A} \}} \\ \\ \text{App-sfa} \quad \frac{\Gamma \vdash_{sfa} a : Z \mid \mathcal{C} \quad \mathcal{C} \text{con} \quad Z \rightarrow^{wh} \Pi x:B. X \quad \Gamma \vdash_{sfa} b : Y \mid \mathcal{D} \quad \mathcal{D} \text{con} \quad Y \simeq B \mid \mathcal{E}}{\Gamma \vdash_{sfa} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}} \end{array}$$

4 The typing relations \vdash_o and \vdash_{tp}

We now begin a deeper study of the general PTS. In section 1.3 we noticed that the third premiss of the **Lda** rule is a serious obstacle to a syntax directed characterization of \vdash . In section 3 we showed that for a special class of PTS that premiss could be replaced by innocuous side conditions, and proceeded to characterize that class by a syntax directed presentation, as advertised. Now we show that in general the troublesome third premiss asks for too much checking; that in the presence of the first two premisses of the **Lda** rule we can replace the third premiss by a more liberal judgement. For this purpose we introduce two liberal typing judgements, \vdash_o and \vdash_{tp} , which are closely related to the typing operators in [vBJ93]. We remind the reader here of the extensions to the reduction and conversion relations arising from our π -contractions

$$(IIx:A.B) a \rightarrow_{\pi} B[x := a].$$

4.1 Well-formed Contexts

We start with a very weak notion of well-formedness for contexts.

Definition 37. \vdash_{wfctx} is the smallest predicate on \mathbb{C} satisfying the following rules.

$$\text{Empty-wfctx} \quad \emptyset \vdash_{wfctx}$$

$$\text{Extend-wfctx} \quad \frac{\Gamma \vdash_{wfctx} \quad \text{FV}(A) \subseteq \text{dom}(\Gamma)}{\Gamma, x:A \vdash_{wfctx}}$$

Recall convention 2. It is easily seen that $\Gamma \vdash a : A$ implies $\Gamma \vdash_{wfctx}$.

4.2 A Nearly Syntax Directed Relation, \vdash_o

We define a relation, \vdash_o , closely connected with \vdash , which is nearly syntax directed. We will prove (lemma 39) that \vdash_o is complete for \vdash . \vdash_o is too liberal to be sound for \vdash , but if $\Gamma \vdash a : A$ for some A , and if also $\Gamma \vdash_o a : B$, then B is, in a sense, convertible to a type for a . This is expressed in our Key Theorem 46.

Definition 38 \vdash_o . The relation $\vdash_o \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the following rules.

$$\text{Srt-o} \quad \frac{\Gamma \vdash_{wfctx}}{\Gamma \vdash_o s_1 : s_2} \quad \langle s_1, s_2 \rangle \in \mathcal{A}$$

$$\text{Var-o} \quad \frac{\Gamma \vdash_{wfctx}}{\Gamma \vdash_o x : A} \quad x:A \in \Gamma$$

$$\text{Pi-o} \quad \frac{\Gamma \vdash_o A : \rightarrow_{\pi} s_1 \quad \Gamma, x:A \vdash_o B : \rightarrow_{\pi} s_2}{\Gamma \vdash_o IIx:A.B : s_3} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

$$\text{Lda-o} \quad \frac{\Gamma, x:A \vdash_o b : B}{\Gamma \vdash_o \lambda x:A.b : IIx:A.B}$$

$$\text{App-o} \quad \frac{\Gamma \vdash_o a : A \rightarrow_{\pi} IIx:B.A_0 \quad \Gamma \vdash_o b : \rightarrow_{\pi} B}{\Gamma \vdash_o ab : Ab}$$

Notice that the rule **App-o** may cause Π -application, and therefore create π -redexes. Also, the rules for \vdash_o are nearly syntax directed, and consequently it is easy to see that \vdash_o has a strong generation lemma (compare with 6) which we will freely use without further comment.

Lemma 39 Completeness of \vdash_o .

If $\Gamma \vdash a : A$
then $\exists A_0 [A_0 \simeq_\pi A \text{ and } \Gamma \vdash_o a : A_0]$.

Proof. By induction on $\Gamma \vdash a : A$. We consider **App**:

$$\Gamma \vdash a b : A[x := b] \quad \text{because} \quad \Gamma \vdash a : \Pi x:B.A \text{ and } \Gamma \vdash b : B.$$

By induction hypothesis we have $\Gamma \vdash_o a : A_0 \simeq_\pi \Pi x:B.A$ and $\Gamma \vdash_o b : B_0 \simeq_\pi B$. Let A_1, B_1 be such that $A_0 \rightarrow_\pi \Pi x:B_1.A_1$, and $B_0 \rightarrow_\pi B_1$. Then

$$\Gamma \vdash_o a b : A_0 b \quad \text{where} \quad A_0 b \simeq_\pi (\Pi x:B_1.A_1) b \rightarrow_\pi A[x := b]$$

as required. \square

Now we derive properties of \vdash_o leading to a weak closure (subject reduction) theorem.

Lemma 40 Free variables for \vdash_o .

If $\Gamma \vdash_o a : A$
then $\Gamma \vdash_{wfctx}$ and $FV(a) \cup FV(A) \subseteq \text{dom}(\Gamma)$

Lemma 41 Weakening for \vdash_o .

If $\Gamma_1 \vdash_o a : A$, $\Gamma_1 \sqsubseteq \Gamma_2$ and $\Gamma_2 \vdash_{wfctx}$
then $\Gamma_2 \vdash_o a : A$

Lemma 42 Approximate substitution for \vdash_o .

If $\Gamma_1, x:A, \Gamma_2 \vdash_o b : B$, $\Gamma_1 \vdash_o a : A_0$ and $A_0 \simeq_\pi A$
then $\Gamma_1, \Gamma_2[x := a] \vdash_o b[x := a] : B_0[x := a]$ where $B_0 \simeq_\pi B$

Proof. Use induction on $\Gamma_1, x:A, \Gamma_2 \vdash_o b : B$, noticing that, since $FV(a) \subseteq \text{dom}(\Gamma_1)$, we have $\Gamma_1, \Gamma_2[x := a] \vdash_{wfctx}$. We consider some cases.

Var-o $\Gamma_1, x:A, \Gamma_2 \vdash_o y : B$ because $y:B \in \Gamma_1, x:A, \Gamma_2$. We discern between cases $y = x$ and $y \neq x$.

For $y = x$, $\Gamma_1, \Gamma_2[x := a] \vdash_o a : A_0$ by weakening. Also $a = y[x := a]$, and $A_0 = A_0[x := a]$ because $x \notin FV(A_0)$ by the free variables lemma.

For $y \neq x$ we have either $y:B \in \Gamma_1$ or $y:B \in \Gamma_2$. In the first case we use the well-formedness of contexts to show that $x \notin FV(B)$, in the second case we are done immediately.

App-o. $\Gamma_1, x:A, \Gamma_2 \vdash_o c d : C d$ because

$$\Gamma_1, x:A, \Gamma_2 \vdash_o c : C \rightarrow_\pi \Pi y:D_0.C_0 \quad \text{and} \quad \Gamma_1, x:A, \Gamma_2 \vdash_o d : D \rightarrow_\pi D_0.$$

By induction hypothesis

$$\Gamma_1, \Gamma_2[x := a] \vdash_o c[x := a] : C_1[x := a] \quad \text{and} \quad \Gamma_1, \Gamma_2[x := a] \vdash_o d[x := a] : D_1[x := a]$$

where $C_1 \simeq_\pi C$ and $D_1 \simeq_\pi D$. Let C_2, D_2 be such that $C_1 \rightarrow_\pi \Pi y:D_2.C_2$ and $D_1 \rightarrow_\pi D_2$. Then

$$C_1[x := a] \rightarrow_\pi \Pi y:D_2[x := a].C_2[x := a] \quad \text{and} \quad D_1[x := a] \rightarrow_\pi D_2[x := a].$$

It follows that $\Gamma_1, \Gamma_2[x := a] \vdash_o (c d)[x := a] : (C_1 d)[x := a]$, where $C_1 d \simeq_\pi C d$. \square

Lemma 43 Reduction of contexts.

If $\Gamma_1 \vdash_o a : A_1$ and $\Gamma_1 \twoheadrightarrow \Gamma_2$
then $\Gamma_2 \vdash_o a : A_2$ where $A_1 \twoheadrightarrow A_2$

Proof. Use induction on $\Gamma_1 \vdash_o a : A_1$, noticing that that since $\Gamma_1 \vdash_{wfcxt}$ we have $\Gamma_2 \vdash_{wfcxt}$. Consider the rule **App-o**: $\Gamma_1 \vdash_o a b : A b$ as a consequence of $\Gamma_1 \vdash_o a : A \twoheadrightarrow_\pi \Pi x : B_0 . A_0$ and $\Gamma_1 \vdash_o b : B \twoheadrightarrow_\pi B_0$. By the induction hypothesis we have $\Gamma_2 \vdash_o a : A_1$ and $\Gamma_2 \vdash_o b : B_1$, where $A \twoheadrightarrow A_1$ and $B \twoheadrightarrow B_1$. Let A_2, B_2 be such that $A_1 \twoheadrightarrow_\pi \Pi x : B_2 . A_2$ and $B_1 \twoheadrightarrow_\pi B_2$. Then $\Gamma_2 \vdash_o a b : A_0 b$ where $A b \twoheadrightarrow A_1 b$. \square

Lemma 44 One step Closure.

If $\Gamma \vdash_o a : A$ and $a \rightarrow b$
then $\Gamma \vdash_o b : B$ where $B \simeq_\pi A$

Proof. Induction on $\Gamma \vdash_o a : A$. The interesting case is the rule **App-o**, where we have $\Gamma \vdash_o a b : A b$ as a consequence of $\Gamma \vdash_o a : A \twoheadrightarrow_\pi \Pi x : B_0 . A_0$ and $\Gamma \vdash_o b : B \twoheadrightarrow_\pi B_0$. We discern cases.

$a = \lambda x : B_1 . a_0$ and $a b \rightarrow a_0[x := b]$. As $\Gamma \vdash_o a : A$ we have $\Gamma, x : B_1 \vdash_o a_0 : A_1$ and $A = \Pi x : B_1 . A_1$. (\vdash_o has a stronger generation lemma than \vdash .) Hence $B \simeq_\pi B_1$, and by the approximate substitution lemma (42), $\Gamma \vdash_o a_0[x := b] : A_2[x := b]$ where $A_2 \simeq_\pi A_1$. It follows that

$$A b = (\Pi x : B_1 . A_1) b \rightarrow_\pi A_1[x := b] \simeq_\pi A_2[x := b],$$

so we are done.

$a \rightarrow c$ so $a b \rightarrow c b$. We have by the induction hypothesis $\Gamma \vdash_o c : C$ where $C \simeq_\pi A$. It follows that $\Gamma \vdash_o c b : C b$ where $C b \simeq_\pi A b$.

$b \rightarrow c$ so $a b \rightarrow a c$. By the induction hypothesis $\Gamma \vdash_o c : C$ where $C \simeq_\pi B$. It follows that $\Gamma \vdash_o a c : A c$ where $A c \simeq_\pi A b$. \square

Now we can easily prove the main result of the current subsection, analogous to lemma 10.

Lemma 45 Weak Closure for \vdash_o .

If $\Gamma \vdash_o a : A, \Gamma \twoheadrightarrow \Gamma_0$ and $a \twoheadrightarrow a_0$
then $\Gamma_0 \vdash_o a_0 : A_0$ where $A_0 \simeq_\pi A$

How is it possible to have proved a closure lemma for \vdash_o without having a type correctness lemma such as lemma 9? (See [GN91, Bar92, MP93] for discussion of the proof of closure for \vdash .) The central point is that \vdash_o , having no conversion rule, has a stronger generation lemma than \vdash , as noted in the proof of 44 above. To take advantage of this, we use an approximate substitution lemma: compare 8 with 42. In the case of \vdash_o it is also possible to separate context reduction from term reduction, while for \vdash these must be proved by simultaneous induction; this latter point, however, is inessential.

4.3 The Key Theorem

In lemma 39 we proved completeness of \vdash_o for \vdash . In this section we will show that \vdash_o is sound, in the weak sense that, if a is some term already well-typed in \vdash , \vdash_o gives it types “correct up to $\beta\pi$ -conversion”. The precise statement is as follows:

Theorem 46 Key Theorem for \vdash_o .

If $\Gamma \vdash a : A_0$ and $\Gamma \vdash_o a : A_1$
then either $A_1 \simeq_\pi A_0$
or $A_1 \twoheadrightarrow_\pi \Pi x_1:C_1 \dots \Pi x_n:C_n.s_0$,
 $a \twoheadrightarrow \lambda x_1:C_1 \dots \lambda x_n:C_n.a_0$
and $\Gamma, x_1:C_1, \dots, x_n:C_n \vdash a_0 : s_0$

Notice that the second alternative is phrased to allow for the fact that the abstractions $\Pi x_1:C_1 \dots \Pi x_n:C_n.s_0$ and $\lambda x_1:C_1 \dots \lambda x_n:C_n.a_0$ may not be well-formed in the absence of sufficient rule instances in \mathcal{R} . The reader should compare the statement of this theorem with that of Theorem 5.5 of [vBJ93].

Proof. By induction on $\Gamma \vdash a : A_0$. We discuss some interesting cases.

Pi $\Gamma \vdash \Pi x:A.B : s_3$ from $\Gamma \vdash A : s_1$, $\Gamma, x:A \vdash B : s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. Suppose $\Gamma \vdash_o \Pi x:A.B : s$. Then $\Gamma \vdash_o A : A_1 \twoheadrightarrow_\pi s_A$ and $\Gamma, x:A \vdash_o B : B_1 \twoheadrightarrow_\pi s_B$, where $\langle s_A, s_B, s \rangle \in \mathcal{R}$. Define C such that $A \twoheadrightarrow C$ and $\Gamma \vdash C : s_A$ as follows:

– If $s_A = s_1$ we take $C = A$.

– Otherwise, by the induction hypothesis $A \twoheadrightarrow C_0$ such that $\Gamma \vdash C_0 : s_A$, and we take $C = C_0$. Similarly define a reduct D of B such that $\Gamma, x:A \vdash D : s_B$. It follows by closure that $\Gamma, x:C \vdash D : s_B$, and hence $\Gamma \vdash \Pi x:C.D : s$.

Lda $\Gamma \vdash \lambda x:A.b : \Pi x:A.B$ from $\Gamma \vdash A : s_1$, $\Gamma, x:A \vdash b : B$, $\Gamma, x:A \vdash B : s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. Suppose $\Gamma \vdash_o \lambda x:A.b : C$. Then we have $\Gamma, x:A \vdash_o b : B_0$ and $C = \Pi x:A.B_0$. Now by induction

either $B \simeq_\pi B_0$
or $B_0 \twoheadrightarrow_\pi \Pi x_1:C_1 \dots \Pi x_n:C_n.s_0$,
 $b \twoheadrightarrow \lambda x_1:C_1 \dots \lambda x_n:C_n.b_0$,
and $\Gamma, x:A, x_1:C_1, \dots, x_n:C_n \vdash b_0 : s_0$.

In both cases we are done.

App $\Gamma \vdash \lambda b : A[x := b]$ from $\Gamma \vdash a : \Pi x:B.A$ and $\Gamma \vdash b : B$. Suppose $\Gamma \vdash_o \lambda b : C$. Then

$$\Gamma \vdash_o a : A_1 \twoheadrightarrow_\pi \Pi x:B_0.A_0, \quad \Gamma \vdash_o b : B_1 \twoheadrightarrow_\pi B_0 \quad \text{and} \quad C = A_1 b.$$

By the left induction hypothesis we have

either $A_1 \simeq_\pi \Pi x:B.A$
or $A_1 \twoheadrightarrow_\pi \Pi x_0:C_0 \dots \Pi x_n:C_n.s_0$,
 $a \twoheadrightarrow \lambda x_0:C_0 \dots \lambda x_n:C_n.a_0$,
and $\Gamma, x_0:C_0, \dots, x_n:C_n \vdash a_0 : s_0$.

(Notice $n \geq 0$ because $A_1 \simeq_\pi \Pi x:B_0.A_0$ cannot reduce to a sort.) In the first case we have

$$C = A_1 b \simeq_\pi (\Pi x:B.A) b \simeq_\pi A[x := B]$$

so we are done. In the second case we have by closure for \vdash (lemma 10) that

$$\Gamma \vdash \lambda x_0:C_0 \dots \lambda x_n:C_n. a_0 : \Pi x:B. A.$$

By generation for \vdash (lemma 6) $C_0 \simeq_\pi B$ and $\Gamma \vdash C_0 : s$ for some $s \in \mathcal{S}$; hence $\Gamma \vdash b : C_0$ by **Cnv**. Therefore we have by substitution for \vdash (lemma 8) that

$$\Gamma, x_1:C_1[x := b], \dots, x_n:C_n[x := b] \vdash a_0[x := b] : s_0.$$

Also

$$C = A_1 b \rightarrow_\pi (\Pi x_0:C_0 \dots \Pi x_n:C_n. s_0) b \rightarrow_\pi \Pi x_1:C_1[x := b] \dots \Pi x_n:C_n[x := b]. s_0$$

and

$$a b \rightarrow (\lambda x_0:C_0 \dots \lambda x_n:C_n. a_0) b \rightarrow \lambda x_1:C_1[x := b] \dots \lambda x_n:C_n[x := b]. a_0[x := b]$$

as required. \square

Corollary 47.

If $\Gamma \vdash a : A$ and $\Gamma \vdash_o a : \rightarrow_\pi s$
then $a \rightarrow a_0$ where $\Gamma \vdash a_0 : s$

Proof. By the Key Theorem we have either $A \simeq_\pi s$ or $a \rightarrow a_0$ where $\Gamma \vdash a_0 : s$. In the first case notice that A , occurring in a \vdash -judgement, contains no π -redexes, so $A \rightarrow s$, and we are done by closure (lemma 10). \square

The following example shows that in general we cannot expect more with respect to soundness; given $\Gamma \vdash a : A$ and $\Gamma \vdash_o a : s$, reduction of a may be necessary to obtain a term of type s .

Example 3. Recall example 1; we have $\odot \vdash a : \Delta$, and $\odot \vdash_o a : \nabla$ but $\odot \not\vdash a : \nabla$. The best we can say is $a \rightarrow \star$ and $\odot \vdash \star : \nabla$. \square

Obviously this is due to our liberal **Lda-o** rule.

4.4 The relation \vdash_{tp}

We now introduce a relation \vdash_{tp} , similar to \vdash_o but with no correctness check for application. \vdash_{tp} is more efficient for type checking, but it lacks the beautiful properties of \vdash_o . In fact it is easy to see that weak closure does not hold for \vdash_{tp} . We will show that \vdash_{tp} -judgements “lift” to \vdash_o -judgements in a sense that is sufficient to prove the Key Theorem for \vdash_{tp} .

Definition 48 \vdash_{tp} . The relation $\vdash_{tp} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the rules for \vdash_o , but having instead of the rule **App-o**:

$$\text{App-tp} \quad \frac{\Gamma \vdash_{tp} a : A}{\Gamma \vdash_{tp} a b : A b}$$

Clearly \vdash_o is contained in \vdash_{tp} , hence, from lemma 39, we have:

Lemma 49 Completeness of \vdash_{tp} .

If $\Gamma \vdash a : A$
then $\exists A_0 [A_0 \simeq_\pi A \text{ and } \Gamma \vdash_{tp} a : A_0]$

In order to prove the Key Theorem for \vdash_{tp} we borrow a technical notion from [vD80]:

Definition 50 Similarity. The relation $\approx \subseteq \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying:

- $a \approx a$
- $s_1 \approx s_2$,
- If $A_1 \approx A_2$ then $\prod x:B.A_1 \approx \prod x:B.A_2$,
- If $A_1 \approx A_2$ then $A_1 b \approx A_2 b$.

Lemma 51 Properties of \approx . If $A \approx B$ then

- i* $A[x := c] \approx B[x := c]$.
- ii* If $A \rightarrow_\pi A_0$ then $B \rightarrow_\pi B_0$ where $A_0 \approx B_0$.
- iii* If $A \twoheadrightarrow_\pi A_0$ then $B \twoheadrightarrow_\pi B_0$ where $A_0 \approx B_0$.
- iv* If $A = \prod x:C.A_0$ then $B = \prod x:C.B_0$ where $A_0 \approx B_0$.

Proof. *i* is proved by induction on $A \approx B$. *ii* is proved similarly, using *i*. *iii* follows immediately from *ii*. *iv* is a simple generation property of \approx . \square

Lemma 52 \vdash_{tp} uniqueness of types.

- If $\Gamma \vdash_{tp} a : A_0$ and $\Gamma \vdash_{tp} a : A_1$
- then $A_0 \approx A_1$

In particular, if $\Gamma \vdash_o a : A_0$ and $\Gamma \vdash_o a : A_1$ then $A_0 \approx A_1$.

Proof. By induction on the structure of a , using the generation lemma for \vdash_{tp} . \square

Lemma 53 Lifting \vdash_{tp} to \vdash_o .

- If $\Gamma \vdash_o a : A_0$ and $\Gamma \vdash_{tp} a : A_1$
- then $\Gamma \vdash_o a : A_1$

Proof. Induction on $\Gamma \vdash_o a : A_0$. Obviously the only interesting case is **App-o**; $\Gamma \vdash_o a b : A b$ as a consequence of $\Gamma \vdash_o a : A \twoheadrightarrow_\pi \prod x:B_2.A_2$ and $\Gamma \vdash_o b : B \twoheadrightarrow_\pi B_2$. As $\Gamma \vdash_{tp} a b : A_1$, it follows that $\Gamma \vdash_{tp} a : A_3$ where $A_1 = A_3 b$. By induction we have $\Gamma \vdash_o a : A_3$. Hence $A_3 \approx A$ by lemma 52, and, by 51(iii), $A_3 \twoheadrightarrow_\pi C$ where $\prod x:B_2.A_2 \approx C$. Now $C = \prod x:B_2.C_2$ by 51(iv), and $\Gamma \vdash_o a b : A_3 b = A_1$ as required. \square

As a corollary we have

Theorem 54 Key Theorem for \vdash_{tp} .

- If $\Gamma \vdash a : A_0$ and $\Gamma \vdash_{tp} a : A_1$
- then either $A_1 \simeq_\pi A_0$
- or $A_1 \twoheadrightarrow_\pi \prod x_1:C_1 \dots \prod x_n:C_n.s_0$,
- $a \twoheadrightarrow \lambda x_1:C_1 \dots \lambda x_n:C_n.a_0$
- and $\Gamma, x_1:C_1, \dots, x_n:C_n \vdash a_0 : s_0$

Proof. By completeness for \vdash_o (39) we have $\Gamma \vdash_o a : A_0$ and hence, by the previous lemma $\Gamma \vdash_o a : A_1$, so we can apply the Key Theorem for \vdash_o . \square

Corollary 55.

- If $\Gamma \vdash a : A$ and $\Gamma \vdash_{tp} a : \twoheadrightarrow_\pi s$
- then $a \twoheadrightarrow a_0$ where $\Gamma \vdash a_0 : s$

4.5 A First Application of \vdash_{tp}

\vdash_o is more liberal than \vdash , and \vdash_{tp} is still more liberal. In the Introduction we hinted that \vdash checks too much to be used in the third premiss of **Lda**, and in the next section we will be precise about using \vdash_{tp} for this purpose. Another application where, intuitively, \vdash does too much checking is computing the type of a subterm of a term already known to be well typed. In application, this problem arises when implementing a unification algorithm for a PTS, for the variables of unification carry types, and before instantiating a variable with some subterm we must check that subterm has the correct type. For this purpose, \vdash_{tp} is more efficient than a correct typechecking algorithm, and this is so even for well behaved PTS such as the Calculus of Constructions, which is full, functional and normalizing.

Assume we have some judgement $\Gamma \vdash a : A$, and b is a “locally closed” subterm of a , i.e. $FV(b) \subseteq \Gamma$. Then it is easy to see that $\Gamma \vdash b : B$ for some B , and our goal is to compute B . By completeness for \vdash_{tp} , $\Gamma \vdash_{tp} b : B_0 \simeq_\pi B$.

5 Application to Pure Type Systems

We present a nearly syntax directed system for arbitrary PTS which is, in a sense to be made precise, equivalent to the original system. We start by giving the definition, in which we replace the third premise of the **Lda** rule by an appeal to \vdash_{tp} , motivated by corollary 55 above.

Definition 56 \vdash_{nsdtp} . The relation $\vdash_{nsdtp} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying

$$\begin{array}{l}
\text{Srt-nsdtp} \quad \circlearrowleft \vdash_{nsdtp} s_1 : s_2 \qquad \langle s_1, s_2 \rangle \in \mathcal{A} \\
\\
\text{Var-nsdtp} \quad \frac{\Gamma \vdash_{nsdtp} A : \rightarrow s}{\Gamma, x:A \vdash_{nsdtp} x : A} \\
\\
\text{Wk-nsdtp} \quad \frac{\Gamma \vdash_{nsdtp} b : B \quad \Gamma \vdash_{nsdtp} A : \rightarrow s}{\Gamma, x:A \vdash_{nsdtp} b : B} \qquad b \in \mathcal{S} \cup \mathcal{V} \\
\\
\text{Pi-nsdtp} \quad \frac{\Gamma \vdash_{nsdtp} A : \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsdtp} B : \rightarrow s_2}{\Gamma \vdash_{nsdtp} \Pi x:A. B : s_3} \qquad \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \\
\\
\text{Lda-nsdtp} \quad \frac{\Gamma \vdash_{nsdtp} A : \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsdtp} b : \rightarrow B}{\Gamma \vdash_{nsdtp} \lambda x:A. b : \Pi x:A. B} \qquad \begin{array}{l} \Gamma, x:A \vdash_{tp} B : \rightarrow_\pi s_2 \\ \langle s_1, s_2, s_3 \rangle \in \mathcal{R} \end{array} \\
\\
\text{App-nsdtp} \quad \frac{\Gamma \vdash_{nsdtp} a : \rightarrow^{wh} \Pi x:B_1. A \quad \Gamma \vdash_{nsdtp} b : B_2}{\Gamma \vdash_{nsdtp} a b : A[x := b]} \qquad B_1 \simeq B_2
\end{array}$$

Informally notice that \vdash_{nsdtp} is better than \vdash_{nsd} from an algorithmic viewpoint, as the occurrence of \vdash_{tp} in **Lda-nsdtp** is cheaper to compute than the corresponding occurrence of \vdash_{nsd} in **Lda-nsd**. That is, our inability to prove Expansion Postponement may be costing us simplicity, but it is not costing efficiency.

Scanning the rules of \vdash_{nsdtp} we see that they are not fully syntax directed. First there are the rules **Srt-nsdtp** and **Pi-nsdtp**, which may give various types to the a term. We have seen how to solve such difficulties by introducing schematic terms. This requires the extension of \vdash_{tp} to schematic terms, which seems to pose no problems.

Second there is the rule **Lda-nsdtp**

$$\mathbf{Lda-nsdtp} \frac{\Gamma \vdash_{nsdtp} A : \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsdtp} b : \rightarrow B \quad \Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2}{\Gamma \vdash_{nsdtp} \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

which fails to be syntax directed because the reduction strategy for the type of b is not fixed. In section 2.1 we repaired the non syntax directed aspect of rule **App-nsd**, replacing reduction by weak head reduction to *some* Π -type, as in **App-nsdtp**. But a Π -type is a weak head normal form, and if weak head reduction fails to terminate in the first premiss of **App-nsdtp** then the term in question has no type. When should we stop reducing in the second premiss of **Lda-nsdtp**? We do not in general know if the type of b has a normal form. In section 6 below, we see that, for functional PTS, no reduction is required in this premiss, but for general PTS this is not complete (example 4).

To begin the theory of \vdash_{nsdtp} , observe:

Lemma 57 Completeness of \vdash_o for \vdash_{nsdtp} .

$$\begin{array}{l} \text{If} \quad \Gamma \vdash_{nsdtp} a : A \\ \text{then} \quad i \quad \exists A_0 \quad A_0 \simeq_{\pi} A \text{ and } \Gamma \vdash_o a : A_0 \\ \quad \quad ii \quad A \in \mathcal{S} \text{ or } \Gamma \vdash_o A : \rightarrow_{\pi} s \end{array}$$

Proof. By induction on $\Gamma \vdash_{nsdtp} a : A$. The interesting case is **Lda-nsdtp**:

$$\mathbf{Lda-nsdtp} \frac{\Gamma \vdash_{nsdtp} A : \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsdtp} b : B' \rightarrow B \quad \Gamma, x:A \vdash_{tp} B : D \rightarrow_{\pi} s_2}{\Gamma \vdash_{nsdtp} \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

By induction we have $\Gamma \vdash_o A : \rightarrow_{\pi} s_1$ and $\Gamma, x:A \vdash_o b : B'_0 \simeq_{\pi} B'$ where $B' \in \mathcal{S}$ or $\Gamma, x:A \vdash_o B' : \rightarrow_{\pi} s_{B'}$. By **Lda-o** we have $\Gamma \vdash_o \lambda x:A.b : \Pi x:A.B'_0 \simeq_{\pi} \Pi x:A.B$. To finish, we claim $\Gamma \vdash_o \Pi x:A.B : \rightarrow_{\pi} s_3$. To show this claim using **Pi-o** and the left induction hypothesis, it remains to show $\Gamma, x:A \vdash_o B : \rightarrow_{\pi} s_2$. Consider the two cases $B' \in \mathcal{S}$ or $\Gamma, x:A \vdash_o B' : \rightarrow_{\pi} s_{B'}$. In the first case $B \in \mathcal{S}$ so it must be that $\langle B, D \rangle \in \mathcal{A}$ (a generation property of \vdash_{tp}); but then $\Gamma, x:A \vdash_o B : D$ and $D = s_2$. In the second case, B' has some type in \vdash_o , and by lemma 45 (weak closure for \vdash_o) B also has some type in \vdash_o . Now we are done by lemma 53 (lifting \vdash_{tp} to \vdash_o) and the side condition of our original instance of **Lda-nsdtp**. \square

Now we prove that \vdash_{nsdtp} is complete and sound with respect to \vdash .

Lemma 58 Completeness of \vdash_{nsdtp} .

$$\begin{array}{l} \text{If} \quad \Gamma \vdash a : A \\ \text{then} \quad \exists A_0 [A_0 \simeq A \text{ and } \Gamma \vdash_{nsdtp} a : A_0]. \end{array}$$

Proof. By induction on $\Gamma \vdash a : A$. We give two cases.

Lda

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x:A \vdash b : B \quad \Gamma, x:A \vdash B : s_2}{\Gamma \vdash \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

By induction

$$\Gamma \vdash_{nsdtp} A : \rightarrow s_1 \text{ and } \Gamma, x:A \vdash_{nsdtp} b : B_0 \simeq B.$$

Also $\Gamma, x:A \vdash_o B : \rightarrow_{\pi} s_2$ by completeness for \vdash_o . Taking a common reduct B_1 of B and B_0 we have $\Gamma, x:A \vdash_o B_1 : \rightarrow_{\pi} s_2$ by closure for \vdash_o , so also $\Gamma, x:A \vdash_{tp} B_1 : \rightarrow_{\pi} s_2$. Moreover $\Gamma, x:A \vdash_{nsdtp} b : \rightarrow B_1$, and therefore by **Lda-nsdtp** $\Gamma \vdash_{nsdtp} \lambda x:A.b : \Pi x:A.B_1 \simeq \Pi x:A.B$.

App $\Gamma \vdash ab : A[x := b]$ as a consequence of $\Gamma \vdash a : \Pi x:B.A$ and $\Gamma \vdash b : B$. By the induction hypothesis $\Gamma \vdash_{nsdtp} a : A_0$ and $\Gamma \vdash_{nsdtp} b : B_0$ where $A_0 \simeq \Pi x:B.A$ and $B_0 \simeq B$. It follows that $A_0 \rightarrow^{wh} \Pi x:B_1.A_1$ where $A_1 \simeq A$ and $B_1 \simeq B$. Therefore $B_0 \simeq B_1$ so $\Gamma \vdash_{nsdtp} ab : A_1[x := b]$ where $A[x := b] \simeq A_1[x := b]$. \square

Lemma 59 Soundness of \vdash_{nsdtp} .

If $\Gamma \vdash_{nsdtp} a : A$
then $\exists A_0 [A \rightarrow A_0, \Gamma \vdash a : A_0, \text{ and } [A \in \mathcal{S} \text{ or } \exists s_0 \Gamma \vdash A_0 : s_0]]$

Proof. By induction on $\Gamma \vdash_{nsdtp} a : A$. Again we treat the lambda rule and the application rule.

Lda-nsdtp

$$\frac{\Gamma \vdash_{nsdtp} A : A_0 \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsdtp} b : B_0 \rightarrow B \quad \Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2}{\Gamma \vdash_{nsdtp} \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

It follows by the induction hypothesis that $\Gamma \vdash A : A_1$ and $\Gamma, x:A \vdash b : B_1$, where $A_0 \rightarrow A_1$ and $B_0 \rightarrow B_1$. Therefore $A_1 \rightarrow s_1$ and hence $\Gamma \vdash A : s_1$ by closure. Also either $B_0 \in \mathcal{S}$ or $\Gamma, x:A \vdash_{\circ} B_0 : B_2 \rightarrow_{\pi} s$ by lemma 57(ii).

In the first case $B = B_0 = B_1$, hence $\langle B, s_2 \rangle \in \mathcal{A}$ and therefore $\Gamma, x:A \vdash B : s_2$. Hence we have $\Gamma \vdash \lambda x:A.b : \Pi x:A.B$ and $\Gamma \vdash \Pi x:A.B : s_3$.

In the second case we have by closure for \vdash_{\circ} that $\Gamma, x:A \vdash_{\circ} B : B_3 \simeq_{\pi} B_2$. It follows by lemma 53 that $\Gamma, x:A \vdash_{\circ} B : \rightarrow_{\pi} s_2$. Taking a common β -reduct B_4 of B and B_1 we have by closure for \vdash that $\Gamma, x:A \vdash b : B_4$ and by closure for \vdash_{\circ} that $\Gamma, x:A \vdash_{\circ} B_4 : \rightarrow_{\pi} s_2$. By correctness of types for \vdash either $B_4 \in \mathcal{S}$ or $\Gamma, x:A \vdash B_4 : s_0$ for some s_0 . In the first case we have $\langle B_4, s_2 \rangle \in \mathcal{A}$ so $\Gamma, x:A \vdash B_4 : s_2$, hence $\Gamma \vdash \lambda x:A.b : \Pi x:A.B_4$ by **Lda** and $\Gamma \vdash \Pi x:A.B_4 : s_3$ by **Pi**, while $B \rightarrow B_4$. In the second case we have by the corollary to the Key Theorem for \vdash_{\circ} (47) that $\Gamma, x:A \vdash B_5 : s_2$ for some reduct B_5 of B_4 , and therefore as before $\Gamma \vdash \lambda x:A.b : \Pi x:A.B_5$ by **Lda** and $\Gamma \vdash \Pi x:A.B_5 : s_3$ by **Pi**, while $B \rightarrow B_5$.

App-nsdtp We have

$$\frac{\Gamma \vdash_{nsdtp} a : A_0 \rightarrow^{wh} \Pi x:B.A \quad \Gamma \vdash_{nsdtp} b : B_0 \quad B_0 \simeq B}{\Gamma \vdash_{nsdtp} ab : A[x := b]}$$

By induction hypothesis $\Gamma \vdash a : A_1$ and $\Gamma \vdash b : B_1$ where $A_0 \rightarrow A_1$ and $B_0 \rightarrow B_1$. Take a common reduct $\Pi x:B_2.A_2$ of A_1 and $\Pi x:B.A$ and note that $B_2 \simeq B_1$. Taking again a common reduct B_3 we have by closure $\Gamma \vdash a : \Pi x:B_3.A_2$ and $\Gamma \vdash b : B_3$, so $\Gamma \vdash ab : A_2[x := b]$ while $A \rightarrow A_2$. Moreover we have by correctness of types for \vdash that $\Gamma \vdash \Pi x:B_3.A_2 : s$ for some s , hence $\Gamma, x:B_3 \vdash A_2 : s_0$ by generation and $\Gamma \vdash A_2[x := b] : s_0$ by the substitution lemma. \square

As a consequence we can characterize \vdash completely in terms of \vdash_{nsdtp} .

Corollary 60 \vdash_{nsdtp} characterizes \vdash .

$$\Gamma \vdash a : A \Leftrightarrow \exists A_0 [A_0 \simeq A \text{ and } \Gamma \vdash_{nsdtp} a : A_0] \quad \text{and} \quad [A \in \mathcal{S} \text{ or } \exists s_0 \Gamma \vdash_{nsdtp} A : \rightarrow s_0]$$

Proof.

\Rightarrow Suppose $\Gamma \vdash a : A$ then the first statement is completeness of \vdash_{nsdtp} . Also we have by correctness of types that either $A \in \mathcal{S}$ or $\Gamma \vdash A : s_0$. In the first case we are done and in the second case we apply once more completeness of \vdash_{nsdtp} .

\Leftarrow Now suppose $\Gamma \vdash_{nsdtp} a : A_0 \simeq A$. Then we have by soundness of \vdash_{nsdtp} that $\Gamma \vdash a : A_1$, where $A_0 \rightarrow A_1$. Now if $A \in \mathcal{S}$ we have $A_1 \rightarrow A$ so $\Gamma \vdash a : A$ by closure. And if $\Gamma \vdash_{nsdtp} A : A_2 \rightarrow s_0$, then again by soundness of \vdash_{nsdtp} we see $\Gamma \vdash A : A_3$ where $A_3 \rightarrow s_0$, hence $\Gamma \vdash A : s_0$; and $\Gamma \vdash a : A$ follows by **Cnv**. \square

As corollaries we have also:

Lemma 61 Weak Closure for \vdash_{nsdtp} .

If $\Gamma \vdash_{nsdtp} a : A$, $\Gamma \rightarrow \Gamma_0$ and $a \rightarrow a_0$
then $\Gamma_0 \vdash_{nsdtp} a_0 : A_0$ where $A \simeq A_0$

Lemma 62 Weak Correctness of Types for \vdash_{nsdtp} .

If $\Gamma \vdash_{nsdtp} a : A$
then either $A \in \mathcal{S}$ or $\exists A_0, s_0 [A \rightarrow A_0$ and $\Gamma \vdash_{nsdtp} A_0 : \rightarrow s_0]$.

5.1 Strengthening

Strengthening is the property that a type assignment to a variable which is never used may be dropped from the context. Formally it is the proposition

Theorem 63 Strengthening for \vdash .

If $\Gamma_1, x:A, \Gamma_2 \vdash b : B$ and $x \notin FV(\Gamma_2) \cup FV(b)$
then $\Gamma_1, \Gamma_2 \vdash b : B_0$ where $B \rightarrow B_0$.

This formulation was first stated and proved for Constructions-like calculi by Luo [Luo90]. A proof for functional PTS appears in [GN91]; strengthening for arbitrary PTS was proved in [vBJ93]. Our results above have strengthening for arbitrary PTS as an easy consequence.

Lemma 64 Strengthening for \vdash_{tp} .

If $\Gamma_1 \vdash_{tp} b : B$, $\Gamma_2 \vdash_{wfcxt}$, $\Gamma_2 \sqsubseteq \Gamma_1$ and $FV(b) \subseteq \text{dom}(\Gamma_2)$
then $\Gamma_2 \vdash_{tp} b : B$.

As usual, we have

Lemma 65 Free Variables for \vdash_{nsdtp} .

If $\Gamma \vdash_{nsdtp} a : A$
then $FV(a) \cup FV(A) \subseteq \text{dom}(\Gamma)$.

More interestingly, because \vdash_{nsdtp} has no conversion rule, we have:

Lemma 66.

If $\Gamma_1, x:A, \Gamma_2 \vdash_{nsdtp} b : B$ and $x \notin FV(\Gamma_2) \cup FV(b)$
then $x \notin FV(B)$.

Lemma 67 Strengthening for \vdash_{nsdtp} .

If $\Gamma_1, x:A, \Gamma_2 \vdash_{nsdtp} b : B$ and $x \notin FV(\Gamma_2) \cup FV(b)$
then $\Gamma_1, \Gamma_2 \vdash_{nsdtp} b : B$

The proofs of these lemmas are by easy induction. They give us strengthening for \vdash .

Proof of Theorem 63. Assume $\Gamma_1, x:A, \Gamma_2 \vdash b : B$. By completeness (58), strengthening for \vdash_{nsdtp} (67), and soundness (59), $\Gamma_1, \Gamma_2 \vdash b : B_1$, where $B \simeq B_1$. Taking a common reduct B_0 of B and B_1 we have by closure $\Gamma_1, \Gamma_2 \vdash b : B_0$. \square

6 Functional Pure Type Systems

We observed that \vdash_{nsdtp} fails to be syntax directed only because of the axiom side condition of **Srt-nsdtp**, the *II*-rule side condition of **Lda-nsdtp** and the non-deterministic reduction of the type of b in the second premiss of **Lda-nsdtp**. In the case of functional PTS the side conditions mentioned are in fact single valued, so the only remaining problem is the reduction in the second premiss of **Lda-nsdtp**. We have also seen that, although terms really can get more types by reduction (example 3), this does not happen for functional PTS (lemma 21). Thus for functional PTS we might hope to remove reduction in the right premiss of **Lda-nsdtp**, and in fact this is the case. For the most part this section parallels the development of section 5, but the following result shows what the difference is: for functional PTS, the relations \vdash_{tp} and \vdash_o are functions.

Lemma 68 Uniqueness of types for \vdash_{tp} . *For functional PTS*

If $\Gamma \vdash_{tp} a : A_0$ and $\Gamma \vdash_{tp} a : A_1$
then $A_0 = A_1$

In particular, if $\Gamma \vdash_o a : A_0$ and $\Gamma \vdash_{tp} a : A_1$ then $A_0 = A_1$.

Now we define a syntax directed system \vdash_f for functional PTS's which is similar to \vdash_{nsdtp} but has a syntax directed **Lda** rule. For the rest of this section we assume that the PTS under discussion is functional.

Definition 69 \vdash_f . The relation $\vdash_f \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the rules of the system \vdash_{nsdtp} , defined in definition 56, but having instead of **Lda-nsdtp** the rule

$$\text{Lda-f} \quad \frac{\Gamma \vdash_f A : \rightarrow s_1 \quad \Gamma, x:A \vdash_f b : B}{\Gamma \vdash_f \lambda x:A.b : \Pi x:A.B} \quad \Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2 \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

Trivially $\vdash_f \subseteq \vdash_{nsdtp}$, so from lemma 57 we have:

Lemma 70 Completeness of \vdash_o for \vdash_f .

If $\Gamma \vdash_f a : A$
then *i* $\exists A_0 [A_0 \simeq_{\pi} A \text{ and } \Gamma \vdash_o a : A_0]$
ii $A \in \mathcal{S} \text{ or } \Gamma \vdash_o A : \rightarrow_{\pi} s$

We prove that \vdash_f is complete and sound with respect to \vdash ; in fact we have even $\vdash_f \subseteq \vdash$.

Lemma 71 Completeness of \vdash_f .

If $\Gamma \vdash a : A$
then $\exists A_0 [A_0 \simeq A \text{ and } \Gamma \vdash_f a : A_0]$.

Proof. By induction on $\Gamma \vdash a : A$. All cases are as in the proof of completeness for \vdash_{nsdtp} (lemma 58) except **Lda**: we have $\Gamma \vdash \lambda x:A.b : \Pi x:A.B$ because $\Gamma \vdash A : s_1$, $\Gamma, x:A \vdash b : B$ and $\Gamma, x:A \vdash B : s_2$, where $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. It follows by induction that $\Gamma \vdash_f A : \rightarrow s_1$ and $\Gamma, x:A \vdash_f b : B_0$ where $B_0 \simeq B$. Also $\Gamma, x:A \vdash_o B : \rightarrow_{\pi} s_2$ by completeness of \vdash_o with respect to \vdash (lemma 39). Either $B_0 \in \mathcal{S}$ or $\Gamma, x:A \vdash_o B_0 : \rightarrow_{\pi} s$ for some s by completeness of \vdash_o with respect to \vdash_f (lemma 70). If $B_0 \in \mathcal{S}$ then $B \rightarrow B_0$, hence $\Gamma, x:A \vdash_o B_0 : \rightarrow_{\pi} s_2$ by closure for \vdash_o . It follows that $\Gamma \vdash_f \lambda x:A.b : \Pi x:A.B_0$. Alternatively, assume $\Gamma, x:A \vdash_o B_0 : \rightarrow_{\pi} s$. Taking a common reduct B_2 of B and B_1 we have $\Gamma, x:A \vdash_o B_2 : \rightarrow_{\pi} s_2$ and $\Gamma, x:A \vdash_o B_2 : \rightarrow_{\pi} s$, both by closure for \vdash_o . It follows by functionality that $s = s_2$ and therefore again $\Gamma \vdash_f \lambda x:A.b : \Pi x:A.B_0$. \square

Lemma 72 Soundness for \vdash_f .

If $\Gamma \vdash_f a : A$
then $\Gamma \vdash a : A$

Proof. By induction on $\Gamma \vdash_f a : A$. We treat the **Lda-f** rule: $\Gamma \vdash_f \lambda x:A.b : \Pi x:A.B$ as a consequence of $\Gamma \vdash_f A : A_0 \rightarrow s_1$ and $\Gamma, x:A \vdash b : B$, where $\Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. It follows by induction that $\Gamma \vdash A : A_0$ and $\Gamma, x:A \vdash b : B$. Hence $\Gamma \vdash A : s_1$ by closure. Also by correctness of types (lemma 9) either $B \in \mathcal{S}$ or $\Gamma, x:A \vdash B : s$ for some s . If $B \in \mathcal{S}$ then $\langle B, s_2 \rangle \in \mathcal{A}$ (because of $\Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2$) and therefore $\Gamma, x:A \vdash B : s_2$. It follows that $\Gamma \vdash \lambda x:A.b : \Pi x:A.B$ by **Lda**. Alternatively assume $\Gamma, x:A \vdash B : s$; we have by completeness for \vdash_o (lemma 39) that $\Gamma, x:A \vdash_o B : \rightarrow_{\pi} s$. It follows by functionality (lemma 68) that $s = s_2$ and therefore again $\Gamma \vdash \lambda x:A.b : \Pi x:A.B$ by **Lda**. \square

As corollaries we have:

Lemma 73 Weak Closure for \vdash_f .

If $\Gamma \vdash_f a : A, \Gamma \rightarrow \Gamma_0$ and $a \rightarrow a_0$
then $\Gamma_0 \vdash_f a_0 : A_0$ where $A \simeq A_0$

Lemma 74 Correctness of Types for \vdash_f .

If $\Gamma \vdash_f a : A$
then either $A \in \mathcal{S}$ or $\exists s \Gamma \vdash_f A : \rightarrow s$.

Proof. Assume $\Gamma \vdash_f a : A$ then by soundness for \vdash_f we have $\Gamma \vdash a : A$, hence by correctness of types for \vdash either $A \in \mathcal{S}$ or $\Gamma \vdash A : s$ for some s . In the latter case $\Gamma \vdash_f A : \rightarrow s$ by completeness for \vdash_f . \square

An important consequence is the following theorem.

Theorem 75 Decidability for normalizing functional PTS. *If a PTS is functional and normalizing, and $\exists s \in \mathcal{S}[\langle s_1, s \rangle \in \mathcal{A}]$ and $\exists s \in \mathcal{S}[\langle s_1, s_2, s \rangle \in \mathcal{R}]$ are decidable relations then the relation \vdash is decidable.*

Proof. It follows from 72 that $\Gamma \vdash_f a : A$ implies that a and A are normalizing. Therefore all the side conditions in the rules for \vdash_f are decidable. \square

6.1 Incompleteness of \vdash_f

We promised to show that \vdash_f is in fact incomplete for the general PTS, i.e. that **Lda-nsdtp** is essentially non-syntax-directed.

Example 4. Consider the following PTS, extending example 3:

$$\begin{aligned} \mathcal{S} &= \{\star, \Delta, \nabla, \square\} \\ \mathcal{A} &= \{\langle \star, \Delta \rangle, \langle \star, \nabla \rangle, \langle \Delta, \square \rangle\} \\ \mathcal{R} &= \{\langle \square, \square, \square \rangle, \langle \Delta, \nabla, \square \rangle\} \end{aligned}$$

As before let $a = (\lambda x: \Delta .x) \star$. We know that $\circlearrowleft \vdash a : \Delta$, and now verify that in fact $\circlearrowleft \vdash_f a : \Delta$, hence also $\circlearrowleft \vdash_{nsdtp} a : \Delta$

$$\frac{\frac{\frac{\circlearrowleft \vdash_f \Delta : \square}{x: \Delta \vdash_f x : \Delta} \quad \frac{x: \Delta \vdash_{wfcxt}}{x: \Delta \vdash_{tp} \Delta : \square}}{\circlearrowleft \vdash_f \lambda x: \Delta .x : \Pi x: \Delta . \Delta} \quad \langle \square, \square, \square \rangle \in \mathcal{R}}{\circlearrowleft \vdash_f (\lambda x: \Delta .x) \star : \Delta} \quad \circlearrowleft \vdash_f \star : \Delta$$

Now we verify $y:a \vdash_{nsdtp} \lambda z: \star .y : \Pi z: \star . \star$

$$\frac{\frac{\vdots}{y:a \vdash_{nsdtp} \star : \Delta} \quad \frac{\vdots}{y:a, z: \star \vdash_{nsdtp} y : a \twoheadrightarrow \star} \quad \frac{y:a, z: \star \vdash_{wfcxt}}{y:a, z: \star \vdash_{tp} \star : \nabla}}{y:a \vdash_{nsdtp} \lambda z: \star .y : \Pi z: \star . \star} \quad \langle \Delta, \nabla, \square \rangle \in \mathcal{R}$$

Obviously (because \vdash_f is nearly syntax directed) the only possible derivation in \vdash_f of a type for $\lambda z: \star .y$ in the context $y:a$ has shape

$$\frac{\frac{\vdots}{y:a \vdash_f \star : X \twoheadrightarrow \alpha} \quad \frac{\vdots}{y:a, z: \star \vdash_f y : a} \quad \frac{\frac{y:a, z: \star, x: \Delta \vdash_{tp} x : \Delta}{y:a, z: \star \vdash_{tp} \lambda x: \Delta .x : \Pi x: \Delta . \Delta}}{y:a, z: \star \vdash_{tp} a : (\Pi x: \Delta . \Delta) \star \twoheadrightarrow \pi \Delta}}{y:a \vdash_f \lambda z: \star .y : \Pi z: \star .a} \quad \langle \alpha, \Delta, ? \rangle \in \mathcal{R}$$

As there is no rule $\langle \alpha, \Delta, ? \rangle \in \mathcal{R}$, \vdash_f does not derive any type for $\lambda z: \star .y$ in the context $y:a$. We see that \vdash_f is strictly weaker than \vdash_{nsdtp} , and is incomplete for non-functional PTS. \square

7 A syntax directed system for arbitrary PTS's

Before defining a syntax directed system we must decide on the lambda rule; given the pseudoterm $\lambda x:A.b$ how far will we reduce B , the type of b , in order to find its sorts? To make the reduction path unique, we reduce B using complete developments.

Definition 76 complete development. The relation $\overset{\lambda}{\Rightarrow} \subseteq \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the following rules.

- $a \overset{\lambda}{\Rightarrow} a$ if $a \in \mathcal{V} \cup \mathcal{S}$.
- If $A_1 \overset{\lambda}{\Rightarrow} A_2$ and $B_1 \overset{\lambda}{\Rightarrow} B_2$ then $\Pi x:A_1.B_1 \overset{\lambda}{\Rightarrow} \Pi x:A_2.B_2$.
- If $A \overset{\lambda}{\Rightarrow} B$ and $a \overset{\lambda}{\Rightarrow} b$ then $\lambda x:A.a \overset{\lambda}{\Rightarrow} \lambda x:B.b$.
- If $a \overset{\lambda}{\Rightarrow} a_0$ and $b \overset{\lambda}{\Rightarrow} b_0$ then $(\lambda x:A.a)b \overset{\lambda}{\Rightarrow} a_0[x := b_0]$.
- If $a \overset{\lambda}{\Rightarrow} a_0$, $b \overset{\lambda}{\Rightarrow} b_0$ and $a \neq \lambda x:A.a_1$ then $ab \overset{\lambda}{\Rightarrow} a_0b_0$.

Clearly $\overset{\lambda}{\Rightarrow}$ is a function, $\overset{\lambda}{\Rightarrow} \subset \twoheadrightarrow$, and if $a \twoheadrightarrow b$, $a \overset{\lambda}{\Rightarrow} a_1$ and $b \overset{\lambda}{\Rightarrow} b_1$ then $a_1 \twoheadrightarrow b_1$. Note that $a \overset{\lambda}{\Rightarrow} a$ iff a is normal. We will denote by $\overset{\lambda}{\Rightarrow}^n$ the result of applying $\overset{\lambda}{\Rightarrow}$ n times, and have by induction: if $a \twoheadrightarrow b$, $a \overset{\lambda}{\Rightarrow}^n a_n$ and $b \overset{\lambda}{\Rightarrow}^n b_n$ then $a_n \twoheadrightarrow b_n$.

The complete development strategy has the advantage of simplicity, and is moreover a *cofinal* reduction strategy, in the sense that if $a \twoheadrightarrow b$, then for some n and c , $a \overset{\lambda}{\Rightarrow}^n c$, with $b \twoheadrightarrow c$. So by closure (lemma 10), we know that if *some* reduct of B has sort s , it suffices to consider complete developments of B in order to compute s . This behaviour should be contrasted with non-cofinal strategies, such as leftmost-outermost reduction; in the case of non-normalising systems such strategies may not capture all possible sorts for B .

Now for every $n \in \mathbb{N}$ we define a relation, \vdash_{nsd-n} , by a nearly syntax directed set of rules whose only non syntax directedness (in rules **Srt-nsd-n** and **Pi-nsd-n**) is a consequence of the non-functionality of the PTS. In general \vdash_{nsd-n} is not equivalent to \vdash for any particular n , but by using unbounded n we will derive a semi-algorithm for typechecking an arbitrary PTS.

Definition 77 \vdash_{nsd-n} . For each $n \in \mathbb{N}$ the relation $\vdash_{nsd-n} \subseteq \mathbb{C} \times \mathbb{T} \times \mathbb{T}$ is the smallest relation satisfying the rules of the relation \vdash_{nsdtp} but for the **Lda**-rule which is replaced by:

$$\text{Lda-nsd-n} \quad \frac{\Gamma \vdash_{nsd-n} A \rightarrow s_1 \quad \Gamma, x:A \vdash_{nsd-n} b \xrightarrow{n} B \quad \Gamma, x:A \vdash_{tp} B \rightarrow_{\pi} s_2}{\Gamma \vdash_{nsd-n} \lambda x:A.b : \Pi x:A.B} \quad \langle s_1, s_2, s_3 \rangle \in \mathcal{R}$$

\vdash_{nsd-n} is sound and complete with respect to \vdash_{nsdtp} .

Lemma 78 Soundness of \vdash_{nsd-n} with respect to \vdash_{nsdtp} .

If $\Gamma \vdash_{nsd-n} a : A$
then $\Gamma \vdash_{nsdtp} a : A$.

For proving completeness we need some properties of \vdash_{nsd-n} .

Lemma 79 Completeness of \vdash_o for \vdash_{nsd-n} .

If $\Gamma \vdash_{nsd-n} a : A$
then $i \quad \exists A_0 [A_0 \simeq_{\pi} A \text{ and } \Gamma \vdash_o a : A_0].$
 $ii \quad A \in \mathcal{S} \text{ or } \exists s [\Gamma \vdash_o A \rightarrow_{\pi} s].$

Lemma 80 Monotonicity of \vdash_{nsd-n} .

If $\Gamma \vdash_{nsd-m} a : A$ and $n \geq m$
then $\exists B [A \rightarrow B \text{ and } \Gamma \vdash_{nsd-n} a : B].$

Proof. By induction on $\Gamma \vdash_{nsd-m} a : A$. We treat some cases.

Lda-nsd-m We have $\Gamma \vdash_{nsd-m} \lambda x:A.c : \Pi x:A.C$ as a consequence of

- $\Gamma \vdash_{nsd-m} A : A_0$, where $A_0 \rightarrow s_1$,
- $\Gamma, x:A \vdash_{nsd-m} c : C_0$, where $C_0 \xrightarrow{m} C$, and
- $\Gamma, x:A \vdash_{tp} C \rightarrow_{\pi} s_2$ where $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$.

The induction hypothesis gives us: first $\Gamma \vdash_{nsd-n} A : A_1$ with $A_0 \rightarrow A_1$ (and hence $A_1 \rightarrow s_1$); second $\Gamma, x:A \vdash_{nsd-n} c : C_1$, with $C_0 \rightarrow C_1$. Now define D by $C_1 \xrightarrow{n} D$ then we have by the properties of \xrightarrow{n} that $C \rightarrow D$. As $\vdash_o \subseteq \vdash_{tp}$ it follows by the lemmas 79 and 45 that $\Gamma, x:A \vdash_{tp} D \rightarrow_{\pi} s_2$. Therefore $\Gamma \vdash_{nsd-n} \lambda x:A.c : \Pi x:A.D$.

App-nsd-m We have $\Gamma \vdash_{nsd-m} a b : A[x := b]$ because of $\Gamma \vdash_{nsd-m} a : A_0$ where $A_0 \rightarrow^{wh} \Pi x:B_1.A$ and $\Gamma \vdash_{nsd-m} b : B_2$, while $B_1 \simeq B_2$. By induction, first $\Gamma \vdash_{nsd-n} a : A_1$ with $A_0 \rightarrow A_1$ (and hence $A_1 \rightarrow^{wh} \Pi x:B_3.B$ with $A \rightarrow B$ and $B_1 \rightarrow B_3$); secondly $\Gamma \vdash_{nsd-n} b : B_4$ with $B_2 \rightarrow B_4$. It follows that $B_3 \simeq B_4$ and hence $\Gamma \vdash_{nsd-n} a b : B[x := b]$. \square

Now we can prove completeness.

Lemma 81 Completeness of \vdash_{nsd-n} with respect to \vdash_{nsdtp} .

If $\Gamma \vdash_{nsdtp} a : A$
then $\exists n, A_0 [A \rightarrow A_0 \text{ and } \Gamma \vdash_{nsd-n} a : A_0].$

Proof. Induction on $\Gamma \vdash_{nsdtp} a : A$. We select interesting cases.

Wk-nsdtp We have $\Gamma, x:A \vdash_{nsd} b : B$ as a consequence of $\Gamma \vdash_{nsd} b : B$, $\Gamma \vdash_{nsd} A : A_0$ where $A_0 \twoheadrightarrow s$ and $b \in \mathcal{S} \cup \mathcal{V}$. By induction hypothesis we have $\Gamma \vdash_{nsd-k} b : B_0$ where $B \twoheadrightarrow B_0$ and $\Gamma \vdash_{nsd-m} A : A_1$ where $A_0 \twoheadrightarrow A_1$. Take $n = \max(k, m)$ and, by 80, $\Gamma \vdash_{nsd-n} b : B_1$ where $B_0 \twoheadrightarrow B_1$ and $\Gamma \vdash_{nsd-n} A : A_2$ where $A_1 \twoheadrightarrow A_2$. It follows that $A_2 \twoheadrightarrow s$ and hence $\Gamma, x:A \vdash_{nsd-n} b : B_1$.

Lda-nsdtp We have $\Gamma \vdash_{nsdtp} \lambda x:A. b : \Pi x:A. B$ as a consequence of $\Gamma \vdash_{nsdtp} A : A_0$ where $A_0 \twoheadrightarrow s_1$ and $\Gamma, x:A \vdash_{nsdtp} b : B_0$ where $B_0 \twoheadrightarrow B$, in k steps, say. And also we know $\Gamma, x:A \vdash_{tp} B : D$ where $D \twoheadrightarrow_{\pi} s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. By the induction hypothesis we have $\Gamma \vdash_{nsd-l} A : A_1$ where $A_0 \twoheadrightarrow A_1$ and $\Gamma, x:A \vdash_{nsd-m} b : B_1$ where $B_0 \twoheadrightarrow B_1$. Take $n = \max(k, l, m)$, getting, by lemma 80, $\Gamma \vdash_{nsd-n} A : A_2$ where $A_1 \twoheadrightarrow A_2$ and $\Gamma, x:A \vdash_{nsd-n} b : B_2$ where $B_1 \twoheadrightarrow B_2$. First observe that $A_1 \twoheadrightarrow s_1$. Next define C_0 and C_2 by $B_0 \xrightarrow{\cong} C_0$ and $B_2 \xrightarrow{\cong} C_2$. Then we have $B \twoheadrightarrow C_0 \twoheadrightarrow C_2$ by the properties of $\xrightarrow{\cong}$. Also, by 57(ii), either $B_0 \in \mathcal{S}$ or $\Gamma \vdash_{\circ} B_0 : D_0$ where $D_0 \twoheadrightarrow_{\pi} s \in \mathcal{S}$. If $B_0 \in \mathcal{S}$ then $B = B_0 = C_2$ and hence $\Gamma, x:A \vdash_{tp} C_2 \twoheadrightarrow_{\pi} s_2$. And if $\Gamma \vdash_{\circ} B_0 : D_0$ then by 45 we have $\Gamma, x:A \vdash_{\circ} B : D_1$ and, it follows by 53 that $\Gamma, x:A \vdash_{\circ} B : D$ and hence again by 45 $\Gamma, x:A \vdash_{\circ} C_2 : D_2$ where $D_2 \twoheadrightarrow_{\pi} s_2$. Therefore $\Gamma \vdash_{nsd-n} \lambda x:A. b : \Pi x:A. C_2$.

App-nsdtp We have

$$\frac{\Gamma \vdash_{nsdtp} a : A_0 \twoheadrightarrow^{wh} \Pi x:B_1. A \quad \Gamma \vdash_{nsdtp} b : B_2}{\Gamma \vdash_{nsdtp} a b : A[x := b]} \quad B_1 \simeq B_2$$

By induction hypothesis

$$\Gamma \vdash_{nsd-k} a : A_1, \quad A_0 \twoheadrightarrow A_1, \quad \Gamma \vdash_{nsd-m} b : B_3, \quad \text{and} \quad B_2 \twoheadrightarrow B_3.$$

Take $n = \max(k, m)$ and, by lemma 80, $\Gamma \vdash_{nsd-n} a : A_2$ where $A_1 \twoheadrightarrow A_2$, and $\Gamma \vdash_{nsd-n} b : B_4$ where $B_3 \twoheadrightarrow B_4$. It follows that $A_2 \twoheadrightarrow^{wh} \Pi x:B_0. A_3$ and $A \twoheadrightarrow A_3$. Hence $\Gamma \vdash_{nsd-n} a b : A_3[x := b]$ where $A[x := b] \twoheadrightarrow A_3[x := b]$. \square

Now we prove that \vdash and \vdash_{nsd-n} are – in a sense – equivalent.

Lemma 82 Equivalence of \vdash and \vdash_{nsd-n} .

$$\Gamma \vdash a : A \quad \Leftrightarrow \quad \begin{array}{l} i \quad \exists n, A_0 [A \simeq A_0 \text{ and } \Gamma \vdash_{nsd-n} a : A_0] \\ ii \quad \text{either } A \in \mathcal{S} \text{ or } \exists n, s [\Gamma \vdash_{nsd-n} A \twoheadrightarrow s]. \end{array}$$

Proof.

\Rightarrow Suppose $\Gamma \vdash a : A$, then by 60 we have that $\Gamma \vdash_{nsdtp} a : A_0$] where $A_0 \simeq A$, and also that either $A \in \mathcal{S}$ or $\Gamma \vdash_{nsdtp} A \twoheadrightarrow s$. It follows from 81 that for some $n \in \mathbb{N}$ there exists A_1 with $\Gamma \vdash_{nsd-n} a : A_1$ and $A \simeq A_1$. And also either $A \in \mathcal{S}$ or by the same argument $\Gamma \vdash_{nsd-n} A \twoheadrightarrow s$.

\Leftarrow Now suppose that $A \simeq A_0$ and $\Gamma \vdash_{nsd-n} a : A_0$, and that either $A \in \mathcal{S}$ or $\Gamma \vdash_{nsd-m} A \twoheadrightarrow s$. Then we have by 78 that $\Gamma \vdash_{nsdtp} a : A_0$ and either $A \in \mathcal{S}$ or $\Gamma \vdash_{nsdtp} A \twoheadrightarrow s$ and again by 60 $\Gamma \vdash a : A$. \square

Monotonicity ensures that any strictly increasing sequence n_j in \mathbb{N} gives rise to a nearly syntax directed search for possible types via the systems \vdash_{nsd-n_j} .

Applying the methods of section 3.2 we will define a syntax directed set of rules for arbitrary PTS. But as the types in this system will be schematic terms, we have to extend our relations

\vdash_o and \vdash_{tp} to schematic terms. In order to do this we extend the notion of β -convertibility of schematic terms modulo a constraint as defined in 31 to $\pi\beta$ -convertibility.

$$X \simeq_\pi Y \mid \mathcal{C} \triangleq \exists X_0, Y_0 [X \twoheadrightarrow_\pi X_0, Y \twoheadrightarrow_\pi Y_0 \text{ and } X_0 =_\Sigma Y_0 \mid \mathcal{C}]$$

Just as in the case of β -reduction we have

Lemma 83 Equivalence of schematic conversion and conversion.

- i* If $X \simeq_\pi Y \mid \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\phi X \simeq_\pi \phi Y$
- ii* If $\phi X \simeq_\pi \phi Y$ then $\exists \mathcal{C} [X \simeq Y \mid \mathcal{C} \text{ and } \phi \models \mathcal{C}]$
- iii* If $\phi_1 X \simeq_\pi \phi_2 Y$ then $\exists \mathcal{C} [X \simeq_\pi Y \mid \mathcal{C}]$

Now we can define the schematized version of \vdash_o .

Definition 84 \vdash_{o+} . The relation $\vdash_{o+} \subseteq \mathbf{C} \times (\mathbf{T} \cup \mathbf{T}^\Sigma) \times (\mathbf{T} \cup \mathbf{T}^\Sigma) \times \mathbf{Cnstr}$ is the smallest relation satisfying the following rules.

$$\begin{array}{l} \text{Srt-}o+ \quad \frac{\Gamma \vdash_{wfcxt}}{\Gamma \vdash_{o+} \alpha : \sigma \mid \{ \langle \alpha, \sigma \rangle \in \mathcal{A} \}} \quad \alpha \in \mathcal{S} \cup \Sigma \\ \\ \text{Var-}o+ \quad \frac{\Gamma \vdash_{wfcxt}}{\Gamma \vdash_{o+} x : A \mid \emptyset} \quad x : A \in \Gamma \\ \\ \text{Pi-}o+ \quad \frac{\Gamma \vdash_{o+} A : \twoheadrightarrow_\pi \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{o+} X : \twoheadrightarrow_\pi \beta \mid \mathcal{D}}{\Gamma \vdash_{o+} \Pi x:A. X : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \beta, \sigma \rangle \in \mathcal{R} \}} \\ \\ \text{Lda-}o+ \quad \frac{\Gamma, x:A \vdash_{o+} b : X \mid \mathcal{C}}{\Gamma \vdash_{o+} \lambda x:A. b : \Pi x:A. X \mid \mathcal{C}} \\ \\ \text{App-}o+ \quad \frac{\Gamma \vdash_{o+} X : Y \mid \mathcal{C} \quad \Gamma \vdash_{o+} b : Z \mid \mathcal{D}}{\Gamma \vdash_{o+} X b : Y b \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}} \quad \begin{array}{l} Y \twoheadrightarrow_\pi \Pi x:B. Y_0 \\ Z \simeq_\pi B \mid \mathcal{E} \end{array} \end{array}$$

Lemma 85 Equivalence of \vdash_o and \vdash_{o+} .

- i* If $\Gamma \vdash_{o+} X : Y \mid \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\Gamma \vdash_o \phi X : \phi Y$
- ii* If $\Gamma \vdash_o \phi X : A$ then $\exists Y, \mathcal{C}, \phi_1 [\phi_1 \supseteq \phi, \phi_1 \models \mathcal{C}, A = \phi_1 Y \text{ and } \Gamma \vdash_{o+} X : Y \mid \mathcal{C}]$

Proof.

i Induction on $\Gamma \vdash_{o+} X : Y \mid \mathcal{C}$. We treat **App- $o+$** : $\Gamma \vdash_{o+} X b : Y b \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$ because $\Gamma \vdash_{o+} X : Y \mid \mathcal{C}$ and $\Gamma \vdash_{o+} b : Z \mid \mathcal{D}$, where $Y \twoheadrightarrow_\pi \Pi x:B. Y_0$ and $Z \simeq_\pi B \mid \mathcal{E}$. Now suppose $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$. By induction $\Gamma \vdash_o \phi(X) : \phi(Y)$ and $\Gamma \vdash_o \phi(b) : \phi(Z)$. Moreover by 83(i), $\phi Z \simeq_\pi \phi B$ and hence $\phi Y \twoheadrightarrow_\pi \phi(\Pi x:B. Y_0) = \Pi x:\phi B. \phi Y_0 \simeq_\pi \Pi x:\phi Z. \phi Y_0$. Therefore $\Gamma \vdash_o \phi(X) \phi(b) : \phi(Y) \phi(b)$ while $\phi(X) \phi(b) = \phi(X b)$ and $\phi(Y) \phi(b) = \phi(Y b)$.

ii Induction on $\Gamma \vdash_o \phi X : A$. We consider some cases.

Srt- o $X \in \mathcal{S} \cup \Sigma$ and $A = s \in \mathcal{S}$, with $\langle \phi X, s \rangle \in \mathcal{A}$. It follows that $\Gamma \vdash_{o+} X : \sigma \mid \{ \langle X, \sigma \rangle \in \mathcal{A} \}$ and defining $\phi_1 = \phi \cup \{ \langle \sigma, s \rangle \}$ we are done.

Lda- o As there are no schematic λ -terms we have $\Gamma \vdash_o \lambda x:A. b : \Pi x:A. B$ as a consequence of $\Gamma, x:A \vdash_o b : B$. By induction hypothesis $\Gamma, x:A \vdash_{o+} b : X \mid \mathcal{C}$ and we have a sort assignment ϕ satisfying \mathcal{C} such that $B = \phi X$. We conclude that $\Gamma \vdash_{o+} \lambda x:A. b : \Pi x:A. X \mid \mathcal{C}$ and also $\Pi x:A. B = \Pi x:A. \phi(X) = \phi(\Pi x:A. X)$.

App-o $\Gamma \vdash_o \phi(X) b : Ab$ from $\Gamma \vdash_o \phi(X) : A$, $\Gamma \vdash_o b : B$ and $A \simeq_\pi \Pi x:B.A_0$. By induction hypothesis $\Gamma \vdash_{o+} X : Y \mid \mathcal{C}$ and $\Gamma \vdash_{o+} b : Z \mid \mathcal{D}$ where we may assume $\text{SV}(\mathcal{C})$ and $\text{SV}(\mathcal{D})$ to be disjoint. Also we have a sort assignments ϕ_1 and ϕ_2 respectively satisfying \mathcal{C} and \mathcal{D} such that $A = \phi_1 Y$ and $B = \phi_2 Z$. Taking $\phi_3 = \phi_1 \cup \phi_2$, we have $\phi_3 Y \simeq_\pi \Pi x:B.A_0$, hence $Y \rightarrow_\pi \Pi x:B_0.Y_0$ where $B_0 \simeq_\pi B = \phi_3 Z$. Now, as $\phi_3 B_0 = B_0$ it follows by 83(ii) that $B_0 \simeq_\pi Z \mid \mathcal{E}$ for some constraint \mathcal{E} which is satisfied by ϕ_3 . Hence $\Gamma \vdash_{o+} X b : Y b \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$, $\phi_3 \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$ and $Ab = \phi_3(Y) b = \phi_3(Y b)$. \square

We want to show that the substitution lemma and the closure lemma of \vdash_o carry over to \vdash_{o+} . In order to avoid reasoning about schematic derivations to prove these, we introduce a technical device. Define a new PTS, $\{\mathcal{S}_0, \mathcal{V}, \mathcal{A}_0, \mathcal{R}_0\}$, the *completely full* PTS derived as follows from the PTS $\{\mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{R}\}$ under consideration:

$$\begin{aligned} \mathcal{S}_0 &= \mathcal{S} \\ \mathcal{A}_0 &= \{ \langle s_1, s_2 \rangle \mid s_1, s_2 \in \mathcal{S} \} \\ \mathcal{R}_0 &= \{ \langle s_1, s_2, s_3 \rangle \mid s_1, s_2, s_3 \in \mathcal{S} \} \end{aligned}$$

The possible constraints in the two systems are identical, but in our new PTS every formula of the form $\langle \alpha, \beta \rangle \in \mathcal{A}$ or $\langle \alpha, \beta, \gamma \rangle \in \mathcal{R}$ is satisfied by every sort assignment; only the equations in a constraint are important for satisfiability. Further, consider the assignment, ϕ_0 , that maps every $\sigma \in \Sigma$ to some one, arbitrary, $s_0 \in \mathcal{S}$, and hence satisfies every constraint.

Lemma 86 Approximate substitution for \vdash_{o+} .

If $\Gamma_1, x:A, \Gamma_2 \vdash_{o+} b : Y \mid \mathcal{D}$, $\Gamma_1 \vdash_{o+} a : X \mid \mathcal{C}$ and $X \simeq_\pi A \mid \mathcal{E}$
then $\exists Y_0, \mathcal{D}_0, \mathcal{E}_0 [\Gamma_1, \Gamma_2[x := a] \vdash_{o+} b[x := a] : Y_0 \mid \mathcal{D}_0$ and $Y_0 \simeq_\pi Y[x := a] \mid \mathcal{E}_0]$.

Proof. Suppose $\Gamma_1, x:A, \Gamma_2 \vdash_{o+} b : Y \mid \mathcal{D}$, $\Gamma_1 \vdash_{o+} a : X \mid \mathcal{C}$ and $X \simeq_\pi A \mid \mathcal{E}$. As in our new system $\phi_0 \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$, it follows from 85(i) that in this system $\Gamma_1, x:A, \Gamma_2 \vdash_o b : \phi_0 Y$, $\Gamma_1 \vdash_o a : \phi_0 X$ and $\phi_0 X \simeq_\pi A$. Hence we have by 42 that $\Gamma_1, \Gamma_2[x := a] \vdash_o b[x := a] : B_0[x := a]$ where $B_0 \simeq_\pi \phi_0 Y$, and by 85(ii) there is Y_0, \mathcal{D}_0 and ϕ such that $\Gamma_1, \Gamma_2[x := a] \vdash_{o+} b[x := a] : Y_0 \mid \mathcal{D}_0$ and $\phi Y_0 = B_0[x := a]$. But $\phi Y_0 = B_0[x := a] \simeq_\pi (\phi_0 Y)[x := a] = \phi_0(Y[x := a])$ and it follows from 83 that $Y_0 \simeq_\pi Y[x := a] \mid \mathcal{E}_0$ for some constraint \mathcal{E}_0 . \square

Lemma 87 Weak Closure for \vdash_{o+} .

If $\Gamma \vdash_{o+} a : X \mid \mathcal{C}$, $\Gamma \rightarrow \Gamma_0$ and $a \rightarrow a_0$
then $\exists X_0, \mathcal{C}_0, \mathcal{D}_0 [\Gamma_0 \vdash_{o+} a_0 : X_0 \mid \mathcal{C}_0$ and $X_0 \simeq_\pi X \mid \mathcal{D}_0]$.

Proof. Suppose $\Gamma \vdash_{o+} a : X \mid \mathcal{C}$, $\Gamma \rightarrow \Gamma_0$ and $a \rightarrow a_0$. As $\phi_0 \models \mathcal{C}$ in the new system we have by 85 that $\Gamma \vdash_o a : \phi_0 X$, and hence by closure for \vdash_o (45) that $\Gamma_0 \vdash_o a_0 : A_0$ where $A_0 \simeq_\pi \phi_0 X$. Again by 85 there is X_0, \mathcal{C}_0 and ϕ such that $\Gamma_0 \vdash_{o+} a_0 : X_0 \mid \mathcal{C}_0$ where $\phi X_0 = A_0$. It follows from 83 that $\phi X_0 = A_0 \simeq_\pi \phi_0 X$ and hence $X_0 \simeq_\pi X \mid \mathcal{D}_0$ for some \mathcal{D}_0 . \square

Now we define the relation \vdash_{tp+} .

Definition 88 \vdash_{tp+} . The relation $\vdash_{tp+} \subseteq \mathbf{C} \times (\mathbf{T} \cup \mathbf{T}^\Sigma) \times (\mathbf{T} \cup \mathbf{T}^\Sigma) \times \mathbf{Cnstr}$ is the smallest relation satisfying the rules for \vdash_{o+} , but having instead of the rule **App-o+** the following rule for application.

$$\text{App-tp+} \quad \frac{\Gamma \vdash_{tp+} a : X \mid \mathcal{C}}{\Gamma \vdash_{tp+} a b : X b \mid \mathcal{C}}$$

Clearly $\vdash_{o+} \subseteq \vdash_{tp+}$, or more precisely, if $\Gamma \vdash_{o+} a : X | \mathcal{C}$ then $\Gamma \vdash_{tp+} a : X | \mathcal{D}$ for some \mathcal{D} . As the systems are syntax directed we have also: if $\Gamma \vdash_{tp+} a : X | \mathcal{D}$ and $\Gamma \vdash_{o+} a : Y | \mathcal{C}$ then $X = Y$.

Lemma 89 Equivalence of \vdash_{tp} and \vdash_{tp+} .

- i* If $\Gamma \vdash_{tp+} X : Y | \mathcal{C}$ and $\phi \models \mathcal{C}$ then $\Gamma \vdash_{tp} \phi X : \phi Y$
- ii* If $\Gamma \vdash_{tp} \phi X : A$ then $\exists Y, \mathcal{C}, \phi_1 [\phi_1 \supseteq \phi, \phi_1 \models \mathcal{C}, A = \phi Y \text{ and } \Gamma \vdash_{tp+} X : Y | \mathcal{C}]$

The proof is similar to the proof of 85.

We are ready to define a syntax directed set of rules for arbitrary PTS.

Definition 90 \vdash_{sd-n} . For each $n \in \mathbb{N}$ the relation $\vdash_{sd-n} \subseteq \mathbf{C} \times \mathbf{T} \times (\mathbf{T} \cup \mathbf{T}^{\mathcal{J}}) \times \mathbf{Cnstr}$ is the smallest relation satisfying the following rules

$$\begin{array}{l}
\text{Srt-sd-n} \quad \circlearrowleft \vdash_{sd-n} s : \sigma \mid \{ \langle s, \sigma \rangle \in \mathcal{A} \} \\
\\
\text{Var-sd-n} \quad \frac{\Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{C}}{\Gamma, x:A \vdash_{sd-n} x : A \mid \mathcal{C}} \\
\\
\text{Wk-sd-n} \quad \frac{\Gamma \vdash_{sd-n} b : X \mid \mathcal{C} \quad \Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{D}}{\Gamma, x:A \vdash_{sd-n} b : X \mid \mathcal{C} \cup \mathcal{D}} \quad b \in \mathcal{S} \cup \mathcal{V} \\
\\
\text{Pi-sd-n} \quad \frac{\Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{sd-n} B : \Rightarrow \beta \mid \mathcal{D}}{\Gamma \vdash_{sd-n} \Pi x:A. B : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{ \langle \alpha, \beta, \sigma \rangle \in \mathcal{R} \}} \\
\\
\text{Lda-sd-n} \quad \frac{\Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{C} \quad \Gamma, x:A \vdash_{sd-n} b : \overset{\lambda}{\Rightarrow} X \mid \mathcal{D}}{\Gamma \vdash_{sd-n} \lambda x:A. b : \Pi x:A. X \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E} \cup \{ \langle \alpha, \beta, \sigma \rangle \in \mathcal{R} \}} \quad \Gamma, x:A \vdash_{tp+} X : \Rightarrow_{\pi} \beta \mid \mathcal{E} \\
\\
\text{App-sd-n} \quad \frac{\Gamma \vdash_{sd-n} a : \Rightarrow^{wh} \Pi x:B. X \mid \mathcal{C} \quad \Gamma \vdash_{sd-n} b : Y \mid \mathcal{D}}{\Gamma \vdash_{sd-n} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}} \quad Y \simeq B \mid \mathcal{E}
\end{array}$$

We prove soundness and completeness of \vdash_{sd-n} with respect to \vdash_{nsd-n} .

Lemma 91 Soundness of \vdash_{sd-n} with respect to \vdash_{nsd-n} .

If $\Gamma \vdash_{sd-n} a : X | \mathcal{C}$ and $\phi \models \mathcal{C}$
then $\Gamma \vdash_{nsd-n} a : \phi X$.

Proof. Induction on $\Gamma \vdash_{sd-n} a : A$.

Srt-sd-n $\circlearrowleft \vdash_{sd-n} s : \sigma \mid \{ \langle s, \sigma \rangle \in \mathcal{A} \}$ If $\phi \models \{ \langle s, \sigma \rangle \in \mathcal{A} \}$ then $\phi \sigma = s_0$ where $\langle s, s_0 \rangle \in \mathcal{A}$. Hence $\circlearrowleft \vdash_{nsd-n} s : s_0$.

Var-sd-n $\Gamma, x:A \vdash_{sd-n} x : A \mid \mathcal{C}$ because $\Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{C}$. Suppose $\phi \models \mathcal{C}$, then by induction hypothesis $\Gamma \vdash_{nsd-n} A : \Rightarrow \phi \alpha$. As $\phi \alpha \in \mathcal{S}$ it follows that $\Gamma, x:A \vdash_{nsd-n} x : A$.

Wk-sd-n $\Gamma, x:A \vdash_{sd-n} b : X \mid \mathcal{C} \cup \mathcal{D}$ because $\Gamma \vdash_{sd-n} b : X \mid \mathcal{C}$, $\Gamma \vdash_{sd-n} A : \Rightarrow \alpha \mid \mathcal{D}$ and $b \in \mathcal{S} \cup \mathcal{V}$. Suppose $\phi \models \mathcal{C} \cup \mathcal{D}$, then by induction hypothesis $\Gamma \vdash_{nsd-n} b : \phi X$ and $\Gamma \vdash_{nsd-n} A : \Rightarrow \phi \alpha$. Again $\phi \alpha \in \mathcal{S}$, so $\Gamma, x:A \vdash_{nsd-n} b : \phi X$.

Pi-sd-n $\Gamma \vdash_{sd-n} \Pi x:A.B : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$ because

$$\Gamma \vdash_{sd-n} A := \alpha \mid \mathcal{C} \quad \text{and} \quad \Gamma, x:A \vdash_{sd-n} B := \beta \mid \mathcal{D}.$$

Suppose $\phi \models \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$, then by induction hypothesis $\Gamma \vdash_{nsd-n} A := \phi \alpha$ and $\Gamma, x:A \vdash_{nsd-n} B := \phi \beta$. As earlier we conclude $\Gamma \vdash_{nsd-n} \Pi x:A.B : \phi \sigma$ because $\langle \phi \alpha, \phi \beta, \phi \sigma \rangle \in \mathcal{R}$.

Lda-sd-n $\Gamma \vdash_{sd-n} \lambda x:A.b : \Pi x:A.X \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$ because

$$\Gamma \vdash_{sd-n} A := \alpha \mid \mathcal{C}, \quad \Gamma, x:A \vdash_{sd-n} b \stackrel{\approx}{=} X \mid \mathcal{D} \quad \text{and} \quad \Gamma, x:A \vdash_{tp+} X := \beta \mid \mathcal{E}.$$

Suppose $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$. By induction hypothesis $\Gamma \vdash_{nsd-n} A := \phi \alpha$ and $\Gamma, x:A \vdash_{nsd-n} b \stackrel{\approx}{=} \phi X$. Also by 89 we have $\Gamma, x:A \vdash_{tp} \phi X := \pi \phi \beta$ and it follows that

$$\Gamma \vdash_{nsd-n} \lambda x:A.b : \Pi x:A.\phi X$$

because $\langle \phi \alpha, \phi \beta, \phi \sigma \rangle \in \mathcal{R}$.

App-sd-n $\Gamma \vdash_{sd-n} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$ because

$$\Gamma \vdash_{sd-n} a := \text{wh} \Pi x:B.X \mid \mathcal{C}, \quad \Gamma \vdash_{sd-n} b : Y \mid \mathcal{D} \quad \text{and} \quad \text{quad} Y \simeq B \mid \mathcal{E}.$$

Suppose $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$, then by induction hypothesis

$$\Gamma \vdash_{nsd-n} a := \text{wh} \Pi x:B.\phi X \quad \text{and} \quad \Gamma \vdash_{nsd-n} b : \phi Y.$$

Also we have $\phi Y \simeq_{\pi} B$ by 32. Hence

$$\Gamma \vdash_{nsd-n} a b : \phi (X[x := b])$$

because $(\phi X)[x := b] = \phi (X[x := b])$. □

Lemma 92 Completeness of \vdash_{sd-n} with respect to \vdash_{nsd-n} .

If $\Gamma \vdash_{nsd-n} a : A$
then $\exists X \in \mathcal{T} \cup \mathcal{T}^{\exists} \exists C \in \mathcal{Cnstr} \exists \phi \models \mathcal{C} [A = \phi X \text{ and } \Gamma \vdash_{sd-n} a : X \mid \mathcal{C}]$.

Proof. Induction on $\Gamma \vdash_{nsd-n} a : A$.

Srt-nsd-n We have $\circlearrowleft \vdash_{nsd-n} s_1 : s_2$ because $\langle s_1, s_2 \rangle \in \mathcal{A}$. Hence $\circlearrowleft \vdash_{sd-n} s_1 : \sigma \mid \{\langle s_1, \sigma \rangle \in \mathcal{A}\}$ where σ is a fresh sort variable. Defining $\phi \sigma = s_2$ we have $\phi \models \{\langle s_1, \sigma \rangle \in \mathcal{A}\}$ and we are done.

Var-nsd-n We have $\Gamma, x:A \vdash_{nsd-n} x : A$ as a consequence of $\Gamma \vdash_{nsd-n} A : A_0$ where $A_0 \twoheadrightarrow s$. By the induction hypothesis $\Gamma \vdash_{sd-n} A : X_0 \mid \mathcal{C}$ and we have $\phi \models \mathcal{C}$ and $\phi X_0 = A_0$. Now as $A_0 \twoheadrightarrow s$, also $X_0 \twoheadrightarrow \alpha$ where $\phi \alpha = s$. It follows that $\Gamma, x:A \vdash_{sd-n} x : A \mid \mathcal{C}$.

Wk-nsd-n $\Gamma, x:A \vdash_{nsd-n} b : B$ because $\Gamma \vdash_{nsd-n} b : B$ and $\Gamma \vdash_{nsd-n} A : A_0$ where $A_0 \twoheadrightarrow s$, and $b \in \mathcal{S} \cup \mathcal{V}$. Induction gives $\Gamma \vdash_{sd-n} b : Y \mid \mathcal{C}$ and $\Gamma \vdash_{sd-n} A : X_0 \mid \mathcal{D}$, where we assume $\mathcal{SV}(\mathcal{C})$ and $\mathcal{SV}(\mathcal{D})$ to be disjoint. Also we have ϕ_1 satisfying \mathcal{C} and ϕ_2 satisfying \mathcal{D} with $\phi_1 Y = B$ and $\phi_2 X_0 = A_0$. As $A_0 \twoheadrightarrow s$ it follows that $X_0 \twoheadrightarrow \alpha$. So, taking $\phi = \phi_1 \cup \phi_2$ we have $\Gamma, x:A \vdash_{sd-n} b : Y \mid \mathcal{C} \cup \mathcal{D}$ where $\phi \models \mathcal{C} \cup \mathcal{D}$ and $\phi Y = B$.

Pi-nsd-n $\Gamma \vdash_{nsd-n} \Pi x:A.B : s_3$ as a consequence of $\Gamma \vdash_{nsd-n} A : A_0$ and $\Gamma, x:A \vdash_{nsd-n} B : B_0$ where $A_0 \twoheadrightarrow s_1$, $B_0 \twoheadrightarrow s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. As before we have $\Gamma \vdash_{sd-n} A : X_0 \mid \mathcal{C}$ and $\Gamma, x:A \vdash_{sd-n} B : Y_0 \mid \mathcal{D}$ where we assume $\mathcal{SV}(\mathcal{C})$ and $\mathcal{SV}(\mathcal{D})$ disjoint, ϕ_1 and ϕ_2 satisfying \mathcal{C} and \mathcal{D} respectively, and $\phi_1 X_0 = A_0$, $\phi_2 Y_0 = B_0$. Hence again $X_0 \twoheadrightarrow \alpha$ and $Y_0 \twoheadrightarrow \beta$, where $\phi_1 \alpha = s_1$ and $\phi_2 \beta = s_2$. It follows that $\Gamma \vdash_{sd-n} \Pi x:A.B : \sigma \mid \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$. So taking $\phi = \phi_1 \cup \phi_2 \cup \{\langle \sigma, s_3 \rangle\}$ we have $\phi \models \mathcal{C} \cup \mathcal{D} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$ and $\phi \sigma = s_3$.

Lda-nsd-n $\Gamma \vdash_{nsd-n} \lambda x:A.b : \Pi x:A.B$ because
 $\Gamma \vdash_{nsd-n} A : A_0$ where $A_0 \rightarrow s_1$, $\Gamma, x:A \vdash_{nsd-n} b : B_0$ where $B_0 \xrightarrow{n} B$, $\Gamma, x:A \vdash_{tp} B : \rightarrow_{\pi} s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. By induction hypothesis we have $\Gamma \vdash_{sd-n} A : X_0 \mid \mathcal{C}$ and $\Gamma, x:A \vdash_{sd-n} b : Y_0 \mid \mathcal{D}$. We observe that $\mathbf{SV}(\mathcal{C})$ and $\mathbf{SV}(\mathcal{D})$ might be chosen to be disjoint. Also we have ϕ_1 and ϕ_2 satisfying \mathcal{C} and \mathcal{D} respectively, where $\phi_1 X_0 = A_0$ and $\phi_2 Y_0 = B_0$. We define $\phi_3 = \phi_1 \cup \phi_2$. Then we have $\phi_3 \alpha = s_1$ and $\phi_3 Y_0 = B_0$. And because $B_0 \xrightarrow{n} B$ we have also $Y_0 \xrightarrow{n} Y$ with $\phi_3 Y = B$. Hence we have $\Gamma, x:A \vdash_{tp} \phi_3 Y : \rightarrow_{\pi} s_2$ and therefore by 89 $\Gamma \vdash_{tp+} Y : \rightarrow_{\pi} \beta \mid \mathcal{E}$ and there is an extension ϕ_4 of ϕ_3 with $\phi_4 \models \mathcal{E}$ and $\phi_4 \beta = s_2$. It follows that

$$\Gamma \vdash_{sd-n} \lambda x:A.b : \Pi x:A.Y \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}.$$

Also if we take $\phi = \phi_4 \cup \{\langle \sigma, s_3 \rangle\}$ we have $\phi Y = B$, so

$$\phi(\Pi x:A.Y) = \Pi x:A.B \quad \text{and} \quad \phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E} \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}.$$

App-nsd-n We have $\Gamma \vdash_{nsd-n} a b : A[x := b]$ because

$$\Gamma \vdash_{nsd-n} a : A_0, \quad \text{and} \quad \Gamma \vdash_{nsd-n} b : B_2 \quad \text{where} \quad A_0 \rightarrow^{wh} \Pi x:B_1.A \quad \text{and} \quad B_1 \simeq B_2.$$

By induction hypothesis $\Gamma \vdash_{sd-n} a : X_0 \mid \mathcal{C}$ and $\Gamma \vdash_{sd-n} b : Y_2 \mid \mathcal{D}$ and as before we consider $\mathbf{SV}(\mathcal{C})$ and $\mathbf{SV}(\mathcal{D})$ to be disjoint, and we have also ϕ_1 and ϕ_2 satisfying \mathcal{C} and \mathcal{D} respectively, where $\phi_1 X_0 = A_0$ and $\phi_2 Y_2 = B_2$. It follows that $X_0 \rightarrow^{wh} \Pi x:B_1.X$ and as $\phi_2 Y_2 \simeq_{\pi} \phi_2 B_1$ we have also $Y_2 \simeq_{\pi} B_1 \mid \mathcal{E}$ where $\phi_2 \models \mathcal{E}$. Hence we have $\Gamma \vdash_{sd-n} a b : X[x := b] \mid \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$ and taking $\phi = \phi_1 \cup \phi_2$ we have also $\phi X = A$ and $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$. \square

For proving our second completeness result we need the following property of \vdash_{sd-n} .

Lemma 93 Weak Completeness of \vdash_{o+} for \vdash_{sd-n} .

$$\begin{array}{l} \text{If} \quad \Gamma \vdash_{sd-n} a : X \mid \mathcal{C} \\ \text{then} \quad i \quad \exists Y, \mathcal{D} \quad Y \simeq_{\pi} X \quad \text{and} \quad \Gamma \vdash_{o+} a : Y \mid \mathcal{D} \\ \quad \quad ii \quad \exists \alpha, \mathcal{D} \quad \Gamma \vdash_{o+} X : \rightarrow_{\pi} \alpha \mid \mathcal{D} \end{array}$$

Now we prove a second completeness lemma for \vdash_{sd-n} .

Lemma 94 Weak Completeness of \vdash_{sd-n} with respect to \vdash_{nsdtp} .

$$\begin{array}{l} \text{If} \quad \Gamma \vdash_{nsdtp} a : A \\ \text{then} \quad \forall n \exists X, \mathcal{C}, \mathcal{D} \quad [A \simeq X \mid \mathcal{C} \quad \text{and} \quad \Gamma \vdash_{sd-n} a : X \mid \mathcal{D}]. \end{array}$$

Proof. Induction on $\Gamma \vdash_{nsdtp} a : A$. We select interesting cases.

Wk-nsdtp We have $\Gamma, x:A \vdash_{nsdtp} b : B$ as a consequence of $\Gamma \vdash_{nsdtp} b : B$, $\Gamma \vdash_{nsdtp} A : A_0$ where $A_0 \rightarrow s$ and $b \in \mathcal{S} \cup \mathcal{V}$. By induction hypothesis we have $\Gamma \vdash_{sd-n} b : X \mid \mathcal{D}_1$ where $B \simeq X \mid \mathcal{C}_1$ and $\Gamma \vdash_{sd-n} A : Y \mid \mathcal{D}_2$ where $A_0 \simeq Y \mid \mathcal{C}_2$. Now as $A_0 \rightarrow s$ it follows that $Y \rightarrow \alpha$ and hence $\Gamma, x:A \vdash_{sdtp-n} b : X \mid \mathcal{D}_1 \cup \mathcal{D}_2$.

Lda-nsdtp We have $\Gamma \vdash_{nsdtp} \lambda x:A.b : \Pi x:A.B$ as a consequence of $\Gamma \vdash_{nsdtp} A : A_0$ where $A_0 \rightarrow s_1$ and $\Gamma, x:A \vdash_{nsdtp} b : B_0$ where $B_0 \rightarrow B$. And also we know $\Gamma, x:A \vdash_{tp} B : C$ where $C \rightarrow_{\pi} s_2$ and $\langle s_1, s_2, s_3 \rangle \in \mathcal{R}$. By the induction hypothesis we have: $\Gamma \vdash_{sd-n} A : X \mid \mathcal{D}_1$ where $A_0 \simeq X \mid \mathcal{C}_1$, and $\Gamma, x:A \vdash_{sd-n} b : Y \mid \mathcal{D}_2$ where $B_0 \simeq Y \mid \mathcal{C}_2$. We observe that $X \rightarrow \alpha$. Also we have by 93 that $\Gamma, x:A \vdash_{o+} Y : \rightarrow_{\pi} \beta \mid \mathcal{E}$. Now we define Y_n by $Y \xrightarrow{n} Y_n$. Then also $\Gamma, x:A \vdash_{o+} Y_n : \rightarrow_{\pi} \beta \mid \mathcal{E}_1$ by 87, and therefore also $\Gamma, x:A \vdash_{tp+} Y_n : \rightarrow_{\pi} \beta \mid \mathcal{E}_2$. It follows that

$$\Gamma \vdash_{sdtp-n} \lambda x:A.b : \Pi x:A.Y_n \mid \mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{E}_2 \cup \{\langle \alpha, \beta, \sigma \rangle \in \mathcal{R}\}$$

by **Lda-sd-n**, and also $\Pi x:A.Y_n \simeq \Pi x:A.B \mid \mathcal{C}_2$, because $B_0 \simeq Y \mid \mathcal{C}_2$, $B \simeq B_0$ and $Y_n \simeq Y$.

App-nsdtp We have $\Gamma \vdash_{nsdtp} ab : A[x := b]$ as a consequence of $\Gamma \vdash_{nsdtp} a : A_0$ where $A_0 \rightarrow^{wh} \Pi x : B. A$ and $\Gamma \vdash_{nsdtp} b : B_0$ where $B_0 \simeq B$. By the induction hypothesis $\Gamma \vdash_{sd-n} a : X | \mathcal{D}_1$ where $A_0 \simeq X | \mathcal{C}_1$ and $\Gamma \vdash_{sd-n} b : Y | \mathcal{D}_2$ where $B_0 \simeq Y | \mathcal{C}_2$. It follows that $X \rightarrow^{wh} \Pi x : B_1. X_1$ where $B \simeq B_1$ and therefore also $Y \simeq B_0 | \mathcal{C}_2$. Hence $\Gamma \vdash_{sd-n} ab : X[x := b] | \mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{C}_2$ by **App-sd-n**. \square

It is the place to collect results. We will characterize \vdash in terms of the relations \vdash_{sd-n} . As the latter have syntax directed presentations there is an obvious way to construct checking algorithms for them, and hence we have an (efficient) algorithm for typechecking \vdash .

Lemma 95 Soundness of \vdash_{sd-n} for \vdash .

If $\Gamma \vdash_{sd-n} a : X | \mathcal{C}$, $\Gamma \vdash_{sd-n} A : \rightarrow \alpha | \mathcal{D}$, $X \simeq A | \mathcal{E}$ and $\phi \models \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}$
then $\Gamma \vdash a : A$

Proof. We have by 91 that $\Gamma \vdash_{nsd-n} a : \phi X$ and $\Gamma \vdash_{nsd-n} A : \rightarrow \phi \alpha$. Hence we have by 78 that $\Gamma \vdash_{nsdtp} a : \phi X$ and $\Gamma \vdash_{nsdtp} A : \rightarrow \phi \alpha$. And therefore by 14 $\Gamma \vdash a : \phi X$ and $\Gamma \vdash A : A_0$ where $A_0 \rightarrow \phi \alpha$. Now $\phi \alpha = s \in \mathcal{S}$ and it follows by closure that $\Gamma \vdash A : s$. And as $\phi \models \mathcal{E}$ we have $\phi X \simeq A$, so $\Gamma \vdash a : A$ by **Cnv**. \square

The next lemma states that for all n , \vdash_{sd-n} succeeds on every well typed subject; the rub is that, if n is not large enough, the constraints may not be satisfiable.

Lemma 96 Weak Completeness of \vdash_{sd-n} for \vdash .

If $\Gamma \vdash a : A$
then $\forall n \exists X, \alpha, \mathcal{C}, \mathcal{D}, \mathcal{E} [\Gamma \vdash_{sd-n} a : X | \mathcal{C}, A \simeq X | \mathcal{E}$ and
 $[A \in \mathcal{S}$ or $\Gamma \vdash_{sd-n} A : \rightarrow \alpha | \mathcal{D}]]$.

Proof. If $\Gamma \vdash a : A$ then we have by 58 that $\Gamma \vdash_{nsdtp} a : A_0$ where $A_0 \simeq A$. It follows by lemma 94 that $\Gamma \vdash_{sd-n} a : X | \mathcal{C}$ where $A_0 \simeq X | \mathcal{E}$ and hence also $A \simeq X | \mathcal{E}$. And we have also either $A \in \mathcal{S}$ or $\Gamma \vdash A : s$. In the first case we are done. In the second case we repeat our argument, getting $\Gamma \vdash_{sd-n} A : Y | \mathcal{D}$ with $s \simeq Y | \mathcal{E}_1$ and therefore $Y \rightarrow \alpha \in \mathcal{S} \cup \Sigma$. \square

Lemma 97 Completeness of \vdash_{sd-n} for \vdash .

If $\Gamma \vdash a : A$
then $\exists n, X, \alpha, \mathcal{C}, \mathcal{D}, \phi [\phi \models \mathcal{C} \cup \mathcal{D}, \Gamma \vdash_{sd-n} a : X | \mathcal{C}, A \simeq \phi X$ and
 $[A \in \mathcal{S}$ or $\Gamma \vdash_{sd-n} A : \rightarrow \alpha | \mathcal{D}]]$.

The proofs are similar to the proofs of the corresponding lemma's for \vdash_{sd-n} , and use the fact that all terms and all schematic terms considered will normalize, as they are β -convertible to terms of the underlying PTS.

8 Conclusion

We have presented efficient syntax directed presentations of two subclasses of PTS:

- the semi-full systems, via the \vdash_{sd-f} relation
- the functional systems, via the \vdash_f relation

The only remaining defect in these presentations lies in the possible failure of tests for conversion in the application rule. Thus for normalizing functional and semi-full systems, everything has been said.

For non-functional systems the situation is less clear. We know of no *a priori* bound on the amount of reduction necessary to correctly type λ -abstractions, so we must be content with the collective completeness of the family of syntax directed systems \vdash_{sd-n} .

We have made little impact on the Expansion Postponement problem, which we leave as future work. We can however bask in the relative peace of mind gained from the machine-checked presentation of most (i.e. those not concerning schematic judgments) of the above results.

References

- [Bar91] Henk Barendregt. Introduction to Generalised Type Systems. *J. Functional Programming*, 1(2):125–154, April 1991.
- [Bar92] Henk Barendregt. Lambda calculi with types. In Abramsky, Gabbai, and Maibaum, editors, *Handbook of Logic in Computer Science*, volume II. Oxford University Press, 1992.
- [Ber90] Stefano Berardi. *Type Dependence and Constructive Mathematics*. PhD thesis, Dipartimento di Informatica, Torino, Italy, 1990.
- [CH88] Thierry Coquand and Gérard Huet. The calculus of constructions. *Information and Computation*, 76(2/3):95–120, February/March 1988.
- [Cha77] Tat-Hung Chan. An algorithm for checking PL/CV arithmetical inferences. Technical Report 77–236, Computer Science Department, Cornell University, Ithaca, New York, 1977.
- [Geu] Herman Geuvers. The calculus of constructions and higher order logic. In preparation.
- [Geu93] Herman Geuvers. *Logics and Type Systems*. PhD thesis, Department of Mathematics and Computer Science, University of Nijmegen, 1993.
- [GN91] Herman Geuvers and Mark-Jan Nederhof. A modular proof of strong normalization for the calculus of constructions. *Journal of Functional Programming*, 1(2):155–189, April 1991.
- [Hel91] Leen Helmkink. Goal directed proof construction in type theory. In *Logical Frameworks*. Cambridge University Press, 1991.
- [HHP87] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. In *Proceedings of the Symposium on Logic in Computer Science*, pages 194–204, Ithaca, New York, June 1987.
- [HHP92] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, 1992. Preliminary version in LICS’87.
- [HP91] Robert Harper and Robert Pollack. Type checking with universes. *Theoretical Computer Science*, 89:107–136, 1991.
- [Hue87] Gérard Huet. Extending the calculus of constructions with Type:Type. Unpublished manuscript, April 1987.
- [Hue89] Gérard Huet. The constructive engine. In R. Narasimhan, editor, *A Perspective in Theoretical Computer Science*. World Scientific Publishing, 1989. Commemorative Volume for Gift Siromoney.
- [LP92] Zhaohui Luo and Robert Pollack. LEGO proof development system: User’s manual. Technical Report ECS-LFCS-92-211, LFCS, Computer Science Dept., University of Edinburgh, The King’s Buildings, Edinburgh EH9 3JZ, May 1992. Updated version. See <http://www.dcs.ed.ac.uk/packages/lego/>
- [Luo90] Zhaohui Luo. *An Extended Calculus of Constructions*. PhD thesis, Department of Computer Science, University of Edinburgh, June 1990.
- [Mar72] Per Martin-Löf. An intuitionistic theory of types. Technical report, University of Stockholm, 1972.
- [MP93] James McKinna and Robert Pollack. Pure Type Systems formalized. In M. Bezem and J.F. Groote, editors, *Proceedings of the International Conference on Typed Lambda Calculi and Applications, TLCA’93, Utrecht*, pages 289–305. Springer-Verlag, LNCS 664, March 1993.
- [MP94] James McKinna and Robert Pollack. Pure Type Systems formalized. Available by anonymous ftp from <ftp.dcs.ed.ac.uk>, directory `export/lego`, file `PTSproofs.tar.Z`, 1994.
- [Pfe89] Frank Pfenning. Elf: A language for logic definition and verified metaprogramming. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science, Asilomar, California*, June 1989.
- [Pol92] R. Pollack. Typechecking in Pure Type Systems. In *Informal Proceedings of the 1992 Workshop on Types for Proofs and Programs, Båstad, Sweden*, pages 271–288, June 1992. Available by ftp.

- [Pol94] Robert Pollack. *The Theory of LEGO: A Proof Checker for the Extended Calculus of Constructions*. PhD thesis, University of Edinburgh, 1994. Available by anonymous ftp from `ftp.cs.chalmers.se` in directory `pub/users/pollack`.
- [vBJ93] L.S. van Benthem Jutting. Typing in Pure Type Systems. *Information and Computation*, 105(1):30–41, July 1993.
- [vD80] D. T. van Daalen. *The Language Theory of Automath*. PhD thesis, Technische Hogeschool Eindhoven, 1980.