A Sequent Calculus for Subtyping Polymorphic Types

Jerzy Tiuryn*
Institute of Informatics
Warsaw University
Banacha 2
02-097 Warsaw, POLAND
(tiuryn@mimuw.edu.pl)

April 5, 1996

Abstract

We present two complete systems for subtyping polymorphic types. One system is in the style of natural deduction, while another is a Gentzen style sequent calculus system. We prove several metamathematical properties for these systems including cut elimination, subject reduction, coherence, and decidability of type reconstruction. Following the approach by J.Mitchell, the sequents are given a simple semantics using logical relations over applicative structures. The systems are complete with respect to this semantics. The logic which emerges from this paper can be seen as a successor to the original Hilbert style system proposed by J. Mitchell in 1988, and to the "half way" sequent calculus of G. Longo, K. Milsted and S. Soloviev proposed in 1995.

1 Introduction

The notion of subtyping is one of the most important concepts introduced recently into the theory of functional languages (see [6, 7, 8]). One of the basic motivations comes from the area of object-oriented programming via the concept of *inheritance*. Another, not necessarily disjoint, way of viewing subtyping is the possibility of treating objects of a given type σ as legal objects of any type which contains σ as a subtype. This is known as the *subsumption rule*.

The relation of subtyping for polymorphic types has been axiomatized by J. Mitchell in 1988 [15]. This axiomatization was proved complete with respect to the semantics based on set-theoretic containment of logical relations over applicative structures.[1] This relation of subtyping has been proved recently undecidable by P. Urzyczyn and the author of this paper (see [18]).[2]

*This work is partly supported by NSF Grants CCR-9417382, CCR-9304144, and by Polish KBN Grant 2 P301 031 06.

[1]These relations are called in [15] *simple inference models*. This semantics is based on the ideas of recursive realizability due to S.C. Kleene, see [19].

[2]There is available another proof of this result due to J. Wells [21].

An important progress towards understanding subtyping was done in 1995 by G. Longo, K. Milsted and S. Soloviev (see [14]) by looking at the subtyping expression $\sigma \sqsubseteq \tau$ as a sequent $\sigma \vdash \tau$ and designing a certain sequent calculus equivalent to the original Mitchell's system. The main contribution of [14] was to bring to the surface the logical contents of the relation of subtyping. However, the sequent calculus of [14] is neither a Gentzen style sequent calculus, nor a natural deduction system. We discuss this point in Section 4 in some detail.

The aim of the present paper is to propose two systems which are strongly related with subtyping. One system, $\vdash_{ND}$, is in the natural deduction style, while another, $\vdash_S$ is a Gentzen style sequent calculus. We show that the system of [14] is a proper sybsystem of $\vdash_{ND}$ in the sense that all rules of the former system are admissible in the latter. We also show several metamathematical properties of $\vdash_{ND}$ and $\vdash_S$, including their equivalence.

The paper is organized as follows. Section 2 contains basic definitions including the polymorphic lambda calculus, the Mitchell's system and the Longo-Milsted-Soloviev system, $\vdash_{LMS}$. In Section 3 we recall the semantics using logical relations over applicative structures. In Section 4 we introduce $\vdash_{ND}$, discuss cut rules and show a containment of $\vdash_{LMS}$ in $\vdash_{ND}$. We also give semantics of sequents with finitely many premises by using the logical relations and show completeness of $\vdash_{ND}$ with respect to this semantics. In Section 5 we prove subject reduction and observe strong normalization for terms typable in $\vdash_{ND}$. In Section 6 we prove the coherence property using an adaptation of the equational theory of [14] and of [6]. In Section 7 we introduce the Gentzen style sequent calculus $\vdash_S$ and prove cut elimination for $\vdash_S$. Finally, in Section 8 we discuss the issue of type inhabitation (provability) and type reconstruction. As we mentioned earlier provability is undecidable. However, we show in this section that type reconstruction problem for $\vdash_{ND}$ is decidable. This is a corollary of the chracterization of pure lambda terms which are erasures of the terms of $\vdash_{ND}$. In Conclusion (Section 9) we briefly discuss the relationship of $\vdash_S$ with *unified calculus*, of J.-Y. Girard [12].

2 Formal Systems

The notation for *polymorphic types* is given in the following grammar.

$$\sigma \ ::= \ X \mid (\sigma \to \sigma) \mid (\forall X.\sigma)$$

where X ranges over *type variables*. We will use X, Y, Z (possibly with indices) as metavariables which denote type variables and σ, ρ, τ are metavariables which denote types. Paretheses are usually omitted if this doesn't lead to a confusion. We will denote by $\mathcal{T}$ the set of all types.

Typed lambda terms are defined by the following grammar.

$$M \ ::= x \mid (MM) \mid (\lambda x : \sigma.M) \mid (M\sigma) \mid (\Lambda X.M)$$

where x ranges over *term variables* and σ ranges over polymorphic types. We use x, y, z, u, v and M, N, P, Q as metavariables ranging over term variables and over terms, respectively.

Pure lambda terms are obtained from the typed terms by erasing all the type information. Hence pure lambda terms are generated by the following grammar.

$$M \ ::= x \mid (MM) \mid (\lambda x.M)$$

For each typed lambda term M we have a pure lambda term, $erase(M)$, it is obtained from M by erasing all the type information.

For terms M and N and a term variable x, $N[M/x]$ denotes the result of substituting M for all free occurences of x in N (bound variables in N may have to be α-renamed in order to avoid capture of free variables in M). Similarly, $\tau[\sigma/X]$ denotes the result of substituting a type σ for all free occurences of X in τ.

We have two kinds of β reductions and two kinds of η reductions.

$$(\beta_1)\ (\lambda x:\sigma.\ M)N \ \rightarrow_\beta\ M[N/x] \qquad (\beta_2)\ (\Lambda X.\ M)\rho \ \rightarrow_\beta\ M[\rho/X]$$

and

$$(\eta_1)\ (\lambda x:\sigma.\ Mx) \ \rightarrow_\eta\ M \qquad (\eta_2)\ (\Lambda X.\ MX) \ \rightarrow_\eta\ M$$

In (η_1) we assume that x doesn't occur free in M. In (η_2) we assume that X doesn't occur free in M. We write $M \rightarrow_\beta^* N$ to indicate that N can be obtained from M by performing a certain number of β-reductions. We use a similar notation for $M \rightarrow_\eta^* N$. We can also mix reductions, $M \rightarrow_{\beta,\eta}^* N$ means that N can be obtained from M by a finite sequence of $\beta\eta$ reductions. Similar notions apply to pure lambda terms.[3]

2.1 The system F

The system **F** of *polymorphic λ calculus* is due to J.-Y. Girard (see [10, 11]) and independently to J. Reynolds (see [16]). It is a system for deriving *typing judgements* which are triples of the form $A \vdash M : \sigma$, where A is a finite partial function from object variables to types, (A is called an *environment*), M is a typed term and σ is a type. The environment A can be thought as containing typing assumptions about variables which may occur free in M. We represent A as a finite set of ordered pairs of the form $(x : \tau)$, where x is an object variable and τ is a type. $Dom(A)$ is the domain of A, and $FV(A)$ is the set of all type variables which occur free in types of A. By $A(x:\sigma)$ we mean a new environment which assigns to x the type σ, and is defined the same as A for all other object variables.

Axiom: $A \vdash (x:\sigma)$, provided $(x:\sigma) \in A$.

Rules:

$$(\rightarrow\text{-intro}) \quad \frac{A(x:\sigma) \vdash M:\tau}{A \vdash (\lambda x:\sigma.M):\sigma\rightarrow\tau}$$

$$(\rightarrow\text{-elim}) \quad \frac{A \vdash M:\sigma\rightarrow\tau \qquad A\vdash N:\sigma}{A\vdash (MN):\tau}$$

[3] Then we have just one notion of β-reduction and one notion of η-reduction.

$$(\forall\text{-intro}) \quad \frac{A \vdash M : \sigma}{A \vdash (\Lambda X.M) : \forall X.\sigma} \qquad (X \notin FV(A))$$

$$(\forall\text{-elim}) \quad \frac{A \vdash M : \forall X.\sigma}{A \vdash M\tau : \sigma[\tau/X]}$$

We write $A \vdash_{\mathbf{F}} M : \sigma$ to indicate that the typing judgement $A \vdash M : \sigma$ is derivable in the above system.

2.2 Mitchell's containment system

The system which we present here derives epressions of the form $\sigma \sqsubseteq \tau$, which are pronounced "type σ is contained in (or is a subtype of) a type τ". This system id due to J. Mitchell ([15]).

Axioms:

(refl) $\quad \sigma \sqsubseteq \sigma$

(inst) $\quad \forall X.\sigma \sqsubseteq \sigma[\rho/X]$

(dummy) $\quad \sigma \sqsubseteq \forall X.\sigma, \quad (X \notin FV(\sigma))$

(distr) $\quad (\forall X.\sigma \to \tau) \sqsubseteq (\forall X.\sigma) \to (\forall X.\tau)$

Rules:

$$(\to) \quad \frac{\sigma' \sqsubseteq \sigma \qquad \tau \sqsubseteq \tau'}{\sigma \to \tau \sqsubseteq \sigma' \to \tau'} \qquad\qquad (\forall) \quad \frac{\sigma \sqsubseteq \tau}{\forall X.\sigma \sqsubseteq \forall X.\tau}$$

$$(\text{trans}) \quad \frac{\sigma \sqsubseteq \rho \qquad \rho \sqsubseteq \tau}{\sigma \sqsubseteq \tau}$$

We write $\vdash \sigma \sqsubseteq \tau$ to indicate that the formula $\sigma \sqsubseteq \tau$ is derivable in the Mitchell's system.

An important result due to J. Mitchell (see [15]) relating the system **F** and the Mitchell's system is the following equivalence.

Theorem 1 *For all polymorphic types σ and τ the following conditions are equivalent:*
(i) $\vdash \sigma \sqsubseteq \tau$.
(ii) *There is a typed term M such that $(x : \sigma) \vdash_{\mathbf{F}} M : \tau$ and $erase(M) \to_{\eta}^{*} x$.*

2.3 The Longo-Milsted-Soloviev system

This system (see [14]) derives sequents of the form $(x : \sigma) \vdash M : \tau$, where σ and τ are polymorphic types, x is an object variable and M is a typed term.

Axiom: $(x : \sigma) \vdash x : \sigma.$

Rules:

$$(\rightarrow) \qquad \frac{(x : \tau_1) \vdash M : \sigma_1 \qquad (y : \sigma_2) \vdash N : \tau_2}{(y : \sigma_1 \rightarrow \sigma_2) \vdash (\lambda x : \tau_1.\ N[(yM)/y]) : \tau_1 \rightarrow \tau_2}$$

$$(\forall_L) \qquad \frac{(x : \sigma[\rho/X]) \vdash M : \tau}{(x : \forall X.\sigma) \vdash M[(x\rho)/x] : \tau}$$

For $0 \leq k \leq n$

$$(\forall_{k,n}) \qquad \frac{(x : \sigma) \vdash \lambda x_1 : \tau_1 \ldots \lambda x_k : \tau_k.\ M : \tau_1 \rightarrow \ldots \tau_n \rightarrow \tau}{(x : \sigma) \vdash \lambda x_1 : \tau_1 \ldots \lambda x_n : \tau_n.\Lambda X.\ M x_{k+1} \ldots x_n : \tau_1 \rightarrow \ldots \tau_n \rightarrow \forall X.\tau}$$

In the above rule we assume that X doesn't occur freely in $\sigma, \tau_1, \ldots \tau_n$, and either $k = n$, or $k < n$ and M doesn't start with λ.[4]

We write $(x : \sigma) \vdash_{LMS} M : \tau$ to denote that the sequent $(x : \sigma) \vdash M : \tau$ is derivable in the above system.

The main result connecting Mitchell's system and $\vdash_{LMS}$ was obtained by Longo, Milsted and Soloviev in [14].

Theorem 2 *The following conditions are equivalent:*
(i) $\vdash \sigma \sqsubseteq \tau$.
(ii) *There is a typed term M such that $(x : \sigma) \vdash_{LMS} M : \tau$.*

The above result implies that (trans) rule of the Mitchell's system is admisssible in $\vdash_{LMS}$. This can be seen as a sort of cut elimination theorem for $\vdash_{LMS}$.

The technical restriction[5] in rule $(\forall_{k,n})$ implies that terms typable in $\vdash_{LMS}$ are in β normal form. The consequence of dropping this restriction is that more terms would be typable but the stronger system would lack the subject reduction property, i.e, typable terms need not be closed under β reduction.

[4] In the original formulation of the system in [14] there are two rules $(\forall_{k,n})$, the first, as defined here, and the second defined for the case $0 \leq n < k$. The reader may check that the second rule is a special case of the first rule.

[5] The restriction which says that the term starts with strictly fewer λ's than the arity of the type on the right of $\vdash$.

3 Semantics via Logical Relations

For the purposes of this section it will be enough to work with very general structures. Let $\mathcal{D} = \langle D, \cdot \rangle$ be a groupoid, i.e. a nonempty set D with a binary operation $\cdot$. Such structures are called *applicative structures*. The intuition behind $\mathcal{D}$ is that every element of D can be viewed both as a code of a certain function or as an argument to a function. Then $d \cdot e$ can be viewed as a result of applying the function whose code is d to the value e.

The operation $\cdot$ can be extended to vectors componentwise. For $(d_1, \ldots, d_n) \in D^n$ and $(e_1, \ldots, e_n) \in D^n$, we define $(d_1, \ldots, d_n) \cdot (e_1, \ldots, e_n) = (d_1 \cdot e_1, \ldots, d_n \cdot e_n)$. Given an applicative structure $\mathcal{D}$ and $n \geq 1$, we can define a binary operation $\Rightarrow$ on n-ary relations over D,

$$R_1 \Rightarrow R_2 = \{\vec{d} \in D^n \mid \text{ for all } \vec{e} \in R_1, \vec{d} \cdot \vec{e} \in R_2\}$$

We can further extend $\cdot$ to n-ary relations elementwise: $R_1 \cdot R_2 = \{\vec{d} \cdot \vec{e} \mid \vec{d} \in R_1,\ \vec{e} \in R_2\}$. Let us observe that the following property holds for n-ary relations R, R_1, R_2 over $\mathcal{D}$. The proof is left for the reader.

Lemma 3 *If R is a relation over an applicative structure $\mathcal{D} = \langle D, \cdot \rangle$, then $R \subseteq (R_1 \Rightarrow R_2)$ iff $(R \cdot R_1) \subseteq R_2$.*

Hence $R_1 \Rightarrow R_2$ is the largest relation R which satisfies $(R \cdot R_1) \subseteq R_2$.

A n-ary *logical relation* over $\mathcal{D}$ is a family $\mathcal{R} = \{R^\sigma\}_{\sigma \in \mathcal{T}}$ of n-ary relations indexed by types, subject to the following two conditions.

$$(\rightarrow) \qquad R^{\sigma \rightarrow \tau} = R^\sigma \Rightarrow R^\tau$$

$$(\forall) \qquad R^{\forall X.\sigma} = \bigcap_{\tau \in \mathcal{T}} R^{\sigma[\tau/X]}$$

We say that type σ is a subtype of a type τ under a logical relation $\mathcal{R}$, written $\mathcal{R} \models \sigma \sqsubseteq \tau$, if $R^\sigma \subseteq R^\tau$. We say that σ is a *subtype* of τ, written $\models \sigma \sqsubseteq \tau$, if σ is a subtype of τ under every logical relation.

The following result, due to J. Mitchell ([15]), establishes soundness and completeness of Mitchell's system with respect to logical relations.

Theorem 4 *For arbitrary types σ and τ, the following two conditions are equivalent:*
(i) $\vdash \sigma \sqsubseteq \tau$.
(ii) $\models \sigma \sqsubseteq \tau$.
Moreover there is a unary logical relation $\mathcal{R}_*$ over a certain applicative structure[6] $\mathcal{D}$, such that for all types σ, τ, we have $\vdash \sigma \sqsubseteq \tau$ iff $\mathcal{R}_* \models \sigma \sqsubseteq \tau$.

[6]This structure is even a λ model.

There is a well known approach to the semantics of system F which uses *partial equivalence relations* (PER), see [1, 3, 5, 4]. A partial equivalence relation over D is a relation which is symmetric and transistive. The importatnt property of PER's is that they are closed under $\Rightarrow$. There is a one-to-one correspondence between subsets of D (i.e. unary relations) and PER's with at most one equivalence class — given $R \subseteq D$ we define a PER $\hat{R}$ such that $(d, e) \in \hat{R}$ iff $d, e \in R$. This correspondence preserves $\Rightarrow$ and arbitrary intersections. Hence, it follows from the second part of Theorem 4 that the Mitchell's system is also complete with respect to PER semantics.

4 Natural Deduction System

The Michell's containment system presented in Section 2.2 is in Hilbert style, i.e., its main emphasis is on the specific axioms. On the other hand, $\vdash_{LMS}$ presented in Section 2.3 can be seen as "half way" sequent calculus in the style of Gentzen — it has just one basic axiom and several rules which introduce the type constructors: $\rightarrow$ and $\forall$. Rule $(\forall_L)$ is a typical Gentzen style rule of introducing $\forall$ on the left of $\vdash$. However, the rule $(\rightarrow)$ introduces $\rightarrow$ on both sides of $\vdash$ simultaneously, while the rule $(\forall_{k,n})$ introduces $\forall$ on the right of $\vdash$ in the special context of a type, i.e., the $\forall$ is introduced to the right of n arrows. The reason for having $(\forall_{k,n})$ in $\vdash_{LMS}$ for each n is that the sequents have only one premise.[7] These additional restrictions are not Gentzen style. A typical Gentzen style system has rules, one for each type constructor, introducing it on the left of $\vdash$, and one for introducing it on the right of $\vdash$. The reader should compare $\vdash_{LMS}$ with the Gentzen style sequent calculus introduced in Section 7.

There is another style of presenting a logical system. It is called *natural deduction* style. The typical feature of this style is that for each type constructor there is one rule which introduces it,[8] and one which eliminates it. An example of a natural deduction system is the system **F** presented in Section 2.1.

In this section we will introduce a natural deduction system $\vdash_{ND}$ which can be seen as an extension of $\vdash_{LMS}$ in the sense that all proofs in $\vdash_{LMS}$ can be carried over in $\vdash_{ND}$. This extension is proper in several respects. First of all it is dealing with a broader class of sequents than $\vdash_{LMS}$ does. But even restricted to the $\vdash_{LMS}$ sequents it has more proofs than $\vdash_{LMS}$ does. The system is remarkably similar to the system **F**, though a few differences, notably non-commutativity and linearity, make the metamathematical theory quite simple. Nevertheless, as we will see, $\vdash_{ND}$ (and also $\vdash_{LMS}$) is undecidable.

In this approach we assume that environments are sequences rather than sets. Hence an *environment* E is a finite non-empty sequence of expressions of the form $(x : \sigma)$ such that no term variable occurs twice in E. By $Dom(E)$ we denote the set of term variables occuring in E, and by $FV(E)$ we denote the set of all type variables which ocur free in types of E. $E_1 * E_2$ denotes concatenation of environments.

We stress two important points in which this definition of an environment differs from the usual definition, e.g. in system **F**:

[7] Usually dealing with sequents having more than one premise makes $(\forall_{k,n})$ admissible, as a consequence of the usual rules which introduce $\rightarrow$ and $\forall$ on the right.

[8] On the right of $\vdash$.

- E is non-commutative, e.g. $(x : \sigma) * (y : \tau) \neq (y : \tau) * (x : \sigma)$.
- E is non-empty.

The first assumption is crucial. The second can be eliminated at the expense of complication of the system.

The following rules define our natural deduction system for coercions.

Axiom: $(x : \sigma) \vdash (x : \sigma)$

Rules:

$$(\to\text{-intro}) \quad \frac{E * (x : \sigma) \vdash M : \tau}{E \vdash (\lambda x : \sigma.M) : \sigma \to \tau}$$

$$(\to\text{-elim}) \quad \frac{E \vdash M : \sigma \to \tau \qquad (x : \rho) \vdash N : \sigma}{E * (x : \rho) \vdash (MN) : \tau} \qquad (x \notin Dom(E))$$

$$(\forall\text{-intro}) \quad \frac{E \vdash M : \sigma}{E \vdash (\Lambda X.M) : \forall X.\sigma} \qquad (X \notin FV(E))$$

$$(\forall\text{-elim}) \quad \frac{E \vdash M : \forall X.\sigma}{E \vdash M\tau : \sigma[\tau/X]}$$

In the above rules we assume that $E \neq \emptyset$. We will write $E \vdash_{ND} M : \sigma$ to denote that the sequent $E \vdash M : \sigma$ is derivable in the above system.

We conclude with some technical results which will be used later.

Lemma 5 *If $E \vdash_{ND} M : \sigma$, then for any type ρ and for any type variable X, $E[\rho/X] \vdash_{ND} M[\rho/X] : \sigma[\rho/X]$.*

Proof: Obvious induction on M. We leave the details for the reader. ∎

Lemma 6 *If $(x : X) * E \vdash_{ND} M : \sigma$, then $E = \emptyset$, and for some $m \geq 0$, $\sigma = \forall Y_1 \ldots \forall Y_m.X$, with $Y_j \neq X$ for $j = 1, \ldots, m$, and $M \to_\beta^* \Lambda Y_1 \ldots \Lambda Y_m.x$.*

Proof: The proof is by routine induction on M. We leave it for the reader. ∎

Lemma 7 *Let $(x:\sigma)*E \vdash_{ND} M:\tau$, and let M be in $\beta\eta$ normal form, such that M is neither of the form $\lambda y:\rho.\ M'$, nor of the form $\Lambda X.\ M'$. Then*
(i) If $M \equiv x$, then $E = \emptyset$ and $\tau \equiv \sigma$.
(ii) If $\sigma \equiv \sigma' \to \sigma''$, and $M \not\equiv x$, then $E = (y_1:\sigma_1)*E'$ is non-empty and M is of the form

$$M \equiv xM'G_1\ldots G_n$$

where $n \geq 0$, M' is a term such that $(y_1:\sigma_1) \vdash_{ND} M':\sigma'$ holds, and each G_j is either a term or a type.
(iii) If $\sigma \equiv \forall Y.\ \sigma'$, and $M \not\equiv x$, then M is of the form

$$M \equiv x\rho G_1\ldots G_n$$

where $n \geq 0$, ρ is a type, and each G_j is either a term or a type.

Proof: The proof is by a routine induction on the number of ($\to$-elim) and ($\forall$-elim) rules used in the derivation of $(x:\sigma)*E \vdash_{ND} M:\tau$. We omit the details. ∎

4.1 Cut rules

Depending on whether the cut is performed on the leftmost formula of the environment, or somwhere inside, we have two cut rules. When cut is performed on the leftmost formula we have the following rule

$$(\mathrm{cut}^l) \qquad \frac{(y:\rho)*R \vdash M:\sigma \qquad (x:\sigma)*E \vdash N:\tau}{(y:\rho)*R*E \vdash N[M/x]:\tau}$$

When the cut formula is not necessarily the leftmost formula of the environment the rule becomes more restrictive.

$$(\mathrm{cut}^r) \qquad \frac{(y:\rho) \vdash M:\sigma \qquad L*(x:\sigma)*E \vdash N:\tau}{L*(y:\rho)*E \vdash N[M/x]:\tau}$$

Let us observe that when $R = \emptyset = L$, then both rules coincide. These two rules can be conveniently represented by the following single rule.

$$(\mathrm{cut}) \qquad \frac{(y:\rho)*R \vdash M:\sigma \qquad L*(x:\sigma)*E \vdash N:\tau}{L*(y:\rho)*R*E \vdash N[M/x]:\tau} \qquad (L = \emptyset \text{ or } R = \emptyset)$$

By taking $L = \emptyset$ in (cut) we obtain (cut^r), while $R = \emptyset$ gives us (cut^l).

Proposition 8 *Rule* (cut) *is admissible in* $\vdash_{ND}$.

Proof: The proof is by induction on N. It is completely routine, with the exception of the case when N is an application $N_1 N_2$. Here we use the assumption that the environments are non-empty. This situation comes up when we consider the following subcase for deriving $L * (x : \sigma) * E \vdash_{ND} (N_1 N_2) : \tau$, with $E = \emptyset$.

$$\frac{L \vdash_{ND} N_1 : \xi \to \tau, \qquad (x : \sigma) \vdash_{ND} N_2 : \xi}{L * (x : \sigma) \vdash_{ND} (N_1 N_2) : \tau}$$

It follows that $L \neq \emptyset$. Hence $R = \emptyset$. By induction hypothesis we obtain.

$$(y : \rho) \vdash_{ND} N_2[M/x] : \xi$$

Thus,

$$L * (y : \rho) \vdash_{ND} N_1(N_2[M/x]) : \tau$$

Since $(N_1 N_2)[M/x] = N_1(N_2[M/x])$, the conclusion follows. The other cases are completely routine. ∎

The reader may have started wondering why do we assume that the environment E in the rules of $\vdash_{ND}$ is non-empty. If we allow $E = \emptyset$, then all the rules make perfect sense and the system can be shown to be equivalent to $\vdash_{ND}$ in the sense that it doesn't prove more sequents of the form $E \vdash M : \sigma$, with $E \neq \emptyset$. The problem with this extension is that (cut^l) is not admissible in it. To see this let us observe that assuming rule (cut^l) and empty environments, the following derivation is legal. Let $E = (z_1 : X \to X) * (z_2 : X)$.

$$\frac{E \vdash z_1 z_2 : X \qquad \dfrac{\vdash (\lambda x : X.\ x) : X \to X \qquad (y : X) \vdash y : X}{(y : X) \vdash (\lambda x : X.\ x) y : X}\,(\to\text{-elim})}{E \vdash (\lambda x : X.\ x)(z_1 z_2) : X}\,(\mathrm{cut}^l)$$

On the other hand it is easy to show that the sequent $(z_1 : X \to X) * (z_2 : X) \vdash (\lambda x : X.\ x)(z_1 z_2) : X$ is not derivable in $\vdash_{ND}$ which allows empty environments.

In order to have the admissibility of (cut) restored one needs an addtional rule which extends $(\to$-elim$)$.

$$(\to \text{-elim-empty}) \qquad \frac{\vdash M : \sigma \to \tau \qquad E \vdash N : \sigma}{E \vdash (MN) : \tau}$$

Environment E in the above rule is arbitrary. In fact, one can prove that (cut^l) and $(\to$ -elim-empty$)$ are equivalent in the extended system.

4.2 From $\vdash_{LMS}$ to $\vdash_{ND}$

In this part we shall prove that if $(x : \sigma) \vdash_{LMS} M : \tau$, then also $(x : \sigma) \vdash_{ND} M : \tau$, i.e. every derivation in $\vdash_{LMS}$ can be viewed as being in $\vdash_{ND}$.

Theorem 9 *If* $(x:\sigma) \vdash_{LMS} M:\tau$, *then* $(x:\sigma)\vdash_{ND} M:\tau$.

Proof: We prove that each of the rules of $\vdash_{LMS}$ is admissible in $\vdash_{ND}$. For $(\to)$ we have the following derivation in $\vdash_{ND}$ extended by (cut) .

$$\cfrac{\cfrac{(y:\sigma_1\to\sigma_2)\vdash y:\sigma_1\to\sigma_2 \quad (x:\tau_1)\vdash M:\sigma_1}{(y:\sigma_1\to\sigma_2)*(x:\tau_1)\vdash (yM):\sigma_2}\,(\to\text{-elim}) \quad (y:\sigma_2)\vdash N:\tau_2}{\cfrac{(y:\sigma_1\to\sigma_2)*(x:\tau_1)\vdash N[(yM)/y]:\tau_2}{(y:\sigma_1\to\sigma_2)\vdash \lambda x:\tau_1.\ N[(yM)/y]:\tau_1\to\tau_2}\,(\to\text{-intro})}\,(\text{cut}^l)$$

Hence, by Proposition 8, $(\to)$ is admissible in $\vdash_{ND}$.

For admissiblity of $(\forall_L)$ we prove that if $(x:\sigma[\rho/X])*E\vdash_{ND} M:\tau$, then $(x:\forall X.\sigma)*E\vdash_{ND} M[(x\rho)/x]:\tau$. The proof is by induction on M. It is routine. Let's just consider the case $M=M_1\xi$.

The last rule in the derivation must have been ($\forall$-elim). Hence the last step in the derivation must have looked as follows.

$$\frac{(x:\sigma[\rho/X])*E\vdash M_1:\forall Y.\tau'}{(x:\sigma[\rho/X])*E\vdash M_1\xi:\tau'[\xi/Y]}$$

And $\tau=\tau'[\xi/Y]$. By induction assumption we obtain

$$(x:\forall X.\sigma)*E\vdash_{ND} M_1[(x\rho)/x]:\forall Y.\tau'$$

Applying ($\forall$-elim) rule we obtain

$$(x:\forall X.\sigma)*E\vdash_{ND} (M_1[(x\rho)/x])\xi:\tau'[\xi/Y]$$

This case is concluded by observing that

$$(M_1[(x\rho)/x])\xi=(M_1\xi)[(x\rho)/x]$$

Let $0\leq k\leq n$. For admissibility of $(\forall_{k,n})$ we prove the following property. If $E\vdash_{ND}\lambda x_1:\tau_1\ldots\lambda x_k:\tau_k.\ M:\tau_1\to\ldots\tau_n\to\tau$ and X doesn't occur free in $\tau_1,\ldots\tau_n$, nor in E, then $E\vdash_{ND}\lambda x_1:\tau_1\ldots\lambda x_n:\tau_n.\Lambda X.\ Mx_{k+1}\ldots x_n:\tau_1\to\ldots\tau_n\to\forall X.\tau$. The proof is by induction on k. Let's just show the case $k=0$, leaving the induction step for the reader.

So assume $E\vdash_{ND} M:\tau_1\to\ldots\tau_n\to\tau$. Using n times ($\to$-elim) we obtain

$$E*(x_1:\tau_1)*\ldots*(x_n:\tau_n)\vdash_{ND} Mx_1\ldots x_n:\tau$$

where $x_1,\ldots,x_n$ are new variables, not occuring in M. By our assumption about X we can apply ($\forall$-intro) and get

$$(x:\sigma)*E*(x_1:\tau_1)*\ldots*(x_n:\tau_n)\vdash_{ND}\Lambda X.Mx_1\ldots x_n:\tau$$

Now, applying n times ($\rightarrow$-intro)we get the conclusion. ∎

Before passing farther let us observe that the converse to Theorem 9 doesn't hold. It follows from the observation that the way $\vdash_{LMS}$ is setup, it types only terms which contain no β_1-redexes (i.e. object-redexes). The reader may easily check that the following term is typable in $\vdash_{ND}$. (Let $\bot$ denote type $\forall X.X$.)

$$(x : \bot \rightarrow \bot) \vdash_{ND} \lambda y : \bot\ \Lambda X.\ ((\lambda z : \bot.\ xzX)y) : \bot \rightarrow \bot$$

4.3 Soundness and completeness

We first present semantics for sequents. Let $\mathcal{R}$ be a logical relation over an applicative structure. $\mathcal{R}$ *satisfies* a sequent $\sigma * \tau_1 * \ldots * \tau_n \vdash \tau$, written $\sigma * \tau_1 * \ldots * \tau_n \models_{\mathcal{R}} \tau$, if $R^{\sigma} \cdot R^{\tau_1} \cdot \ldots R^{\tau_n} \subseteq R^{\tau}$.[9] We say that the sequent $\sigma * \tau_1 * \ldots * \tau_n \vdash \tau$ is *valid*, written $\sigma * \tau_1 * \ldots * \tau_n \models \tau$, if for every applicative structure $\mathcal{D}$ and for every logical relation $\mathcal{R}$ over $\mathcal{D}$, $\sigma * \tau_1 * \ldots * \tau_n \models_{\mathcal{R}} \tau$ holds.

The next result is a straight generalization of Mitchell's Theorems 1 and 4.

Theorem 10 *For arbitrary types $\sigma, \tau_1, \ldots, \tau_n, \tau$ the following are equivalent:*
(i) *There is a typed term M such that $(x : \sigma) * (y_1 : \tau_1) * \ldots * (y_n : \tau_n) \vdash_{ND} M : \tau$.*
(ii) *There is a typed term M such that $\{(x : \sigma), (y_1 : \tau_1), \ldots, (y_n : \tau_n)\} \vdash_{\mathbf{F}} M : \tau$ and $erase(M) \rightarrow^*_{\eta} xy_1 \ldots y_n$.*
(iii) $\vdash \sigma \sqsubseteq \tau_1 \rightarrow \ldots \rightarrow \tau_n \rightarrow \tau$.
(iv) $\sigma * \tau_1 * \ldots * \tau_n \models \tau$.

Proof: (i)⇒(ii) is proved by an easy induction on M. We use the observation that every derivation in $\vdash_{ND}$ can be easily transformed into a derivation in $\vdash_{\mathbf{F}}$. The property of the system F used in this transformation is that the following weakening rule is admissible in $\vdash_{\mathbf{F}}$: if $A \vdash_{\mathbf{F}} M : \tau$, and B is an environment extending A, then $B \vdash_{\mathbf{F}} M : \tau$. We leave the details for the reader.

(ii)⇒(iii). Assuming (ii) we have $\{(x : \sigma)\} \vdash_{\mathbf{F}} (\lambda y_1 : \tau_1. \ldots \lambda y_n : \tau_n.M) : \tau_1 \rightarrow \ldots \rightarrow \tau_n \tau$ and $erase(\lambda y_1 : \tau_1. \ldots \lambda y_n : \tau_n.M) \rightarrow^*_{\eta} x$. Hence, by Theorem 1 we get (iii).

(iii)⇒(iv) follows from Theorem 4 and Lemma 3.

(iv)⇒(i). Assuming (iv), by Lemma 3 we have $\models \sigma \sqsubseteq \tau_1 \rightarrow \ldots \rightarrow \tau_n \rightarrow \tau$. Hence by Theorem 4, Theorem 2 and Theorem 9 we have that there is a typed term M such that

$$(x : \sigma) \vdash_{ND} M : \tau_1 \rightarrow \ldots \rightarrow \tau_n \rightarrow \tau$$

Applying n times ($\rightarrow$-elim) rule we obtain

$$(x : \sigma) * (y_1 : \tau_1) * \ldots * (y_n : \tau_n) \vdash_{ND} (My_1 \ldots y_n) : \tau$$

Hence $My_1 \ldots y_n$ is the sought term. ∎

[9] Partentheses associate to the left, i.e., $R^{\sigma} \cdot R^{\tau_1} \cdot \ldots R^{\tau_n}$ stands for $(\ldots(R^{\sigma} \cdot R^{\tau_1}) \cdot \ldots R^{\tau_n})$.

5 Subject Reduction and Strong Normalization

Subject reduction is a property of the system which says that each type is closed under β reduction of terms typable in that type. We have the following result.

Theorem 11 (Subject Reduction)
Let $E \vdash_{ND} M : \tau$ *and let* $M \ \to_\beta \ N$. *Then* $E \vdash_{ND} N : \tau$.

Proof: The proof is by induction on M. For β reduction let's consider just two cases:

I. $M \equiv (\lambda x : \sigma.\ P)Q$ and $N = P[Q/x]$.
II. $M \equiv (\Lambda X.\ P)\rho$ and $N \equiv P[\rho/X]$.

In case (I) the last part of the derivation of $E \vdash_{ND} M : \sigma$ must look as folows.

$$\dfrac{\dfrac{E' * (x : \sigma) \vdash P : \tau}{E' \vdash (\lambda x : \sigma.\ P) : \sigma \to \tau}(\to\text{-intro}) \qquad (y : \xi) \vdash Q : \sigma}{E' * (y : \xi) \vdash (\lambda x : \sigma.\ P)Q : \tau}(\to\text{-elim})$$

where $E = E' * (y : \xi)$. Then we use (cut^r) rule to get

$$\dfrac{(y : \xi) \vdash Q : \sigma \qquad E' * (x : \sigma) \vdash P : \tau}{E' * (y : \xi) \vdash P[Q/x]}(\text{cut}^r)$$

By Proposition 8 we conclude that $E \vdash_{ND} N : \tau$.

In case (II) we proceed in a similar way. The last part of the derivation of $E \vdash_{ND} M : \tau$ must look as follows.

$$\dfrac{\dfrac{E \vdash P : \tau'}{E \vdash (\Lambda X.P) : \forall X.\ \tau'}(\forall\text{-intro})}{E \vdash (\Lambda X.\ P)\rho : \tau'[\rho/X]}(\forall\text{-elim})$$

where $\tau \equiv \tau'[\rho/X]$. Since $X \notin FV(E)$, Proposition 5 applied to $E \vdash_{ND} P : \tau'$ yields $E \vdash_{ND} N : \tau$. This completes the proof. ∎

It turns out that terms typable in $\vdash_{ND}$ are also closed under η-reductions. An obvious proof of the next result is left for the reader.

Proposition 12 *Let* $E \vdash_{ND} M : \tau$ *and let* $M \ \to_\eta \ N$. *Then* $E \vdash_{ND} N : \tau$.

We conclude this section with a result which states *strong normalization* of terms typable in $\vdash_{ND}$. Since typability in $\vdash_{ND}$ can be easily seen as a part of typability in **F**, strong normalization of terms typable in $\vdash_{ND}$ follows from strong normalization for system **F** (see [11]). However, the result for $\vdash_{ND}$ has an obvious direct proof due to the special form of terms typable in $\vdash_{ND}$.

Theorem 13 (Strong Normalization)
If $E \vdash_{ND} M : \tau$, then every sequence of β-reductions starting from M is finite, i.e., leads to a normal form.

Proof: It is enough to observe that all terms typable in $\vdash_{ND}$ are linear, i.e. every lambda abstraction in such a term binds just one occurence of a variable. Hence every β-reduction decreases the size of the term. ∎

6 Coherence

Coherence of $\vdash_{ND}$ is the property which says that any two terms which represent coercions between the same types are provably equal. For this we need an equational theory for term typable in $\vdash_{ND}$.

6.1 Equational theory

We will present a formal system for deriving expressions of the form $E \vdash M = N : \tau$, which we read as 'terms M and N are provably equal in type τ'.

Axioms for equality:
We assume that for every axiom $E \vdash M = N : \tau$ stated below we assume that both $E \vdash_{ND} M : \tau$ and $E \vdash_{ND} N : \tau$ hold.

(eq-refl) $\quad E \vdash M = M : \tau$

(eq-β) $\quad E \vdash (\lambda x : \sigma.M)N = M[N/x] : \tau$

(eq-β2) $\quad E \vdash (\Lambda X.M)\sigma = M[\sigma/X] : \tau$

(eq-η) $\quad E \vdash \lambda x : \sigma.Mx = M : \sigma \to \tau, \quad x \notin FV(M)$

(eq-η2) $\quad E \vdash \Lambda X.MX = M : \forall X.\sigma, \quad X \notin FV(M)$

Rules for equality:

$$\text{(eq-appl)} \qquad \frac{E \vdash M = N : \sigma \to \tau \qquad (x : \xi) \vdash P = Q : \sigma}{E * (x : \xi) \vdash MP = NQ : \tau} \qquad x \notin dom(E)$$

$$\text{(eq appl2)} \qquad \frac{E \vdash M = N : \forall X.\sigma}{E * E' \vdash P[M\tau_1/y] = Q[N\tau_2/y] : \xi}$$

where in (eq appl2) we assume that $(y : \sigma[\tau_1/X]) * E' \vdash_{ND} P : \xi$ and $(y : \sigma[\tau_2/X]) * E' \vdash_{ND} Q : \xi$

$$\text{(eq abs)} \qquad \frac{E * (x : \sigma) \vdash M = N : \tau}{E \vdash (\lambda x : \sigma.M) = (\lambda x : \sigma.N) : \sigma \to \tau}$$

$$\text{(eq abs2)} \qquad \frac{E \vdash M = N : \tau}{E \vdash (\Lambda X.M) = (\Lambda X.N) : \forall X.\tau} \quad X \notin FV(E)$$

$$\text{(eq symm)} \qquad \frac{E \vdash M = N : \tau}{E \vdash N = M : \tau}$$

$$\text{(eq trans)} \qquad \frac{E \vdash M = P : \tau \qquad E \vdash P = N : \tau}{E \vdash N = M : \tau}$$

We will use notation $E \vdash_{ND} M = N : \sigma$ to denote that the terms M, N are provably equal (in the environment E and in the type σ) in the above system, i.e. the expression $E \vdash M = N : \sigma$ is derivable in the above system.

The above axioms and rules are pretty obvious and need no explanation, except perhaps the rule (eq appl2). This rule is a direct translation into our framework of explicit coercions of the rule (Eq appl2) from [6] which is a congruence rule for polymorphic type application. This is quite a powerful rule having many intriguing consequences. The reader may also consult [14] for a discussion of this rule and its "coercion version" for $\vdash_{LMS}$.

6.2 Proving coherence property

We first prove a sequence of lemmas which lead to the coherence.

Lemma 14 *If $E \vdash_{ND} P = Q : \sigma$ and $(y : \sigma) * E' \vdash_{ND} M : \tau$, then $E * E' \vdash_{ND} M[P/y] = M[Q/y] : \tau$.*

Proof: The proof is by a routine induction on the size of M. We leave the details for the reader. ∎

Lemma 15 *If $E \vdash_{ND} P = Q : \sigma$ and $(y : \sigma) * E' \vdash M = N : \tau$ is an instance of an axiom for equality, then $E * E' \vdash_{ND} M[P/y] = N[Q/y] : \tau$.*

Proof: When $(y : \sigma) * E' \vdash M = N : \tau$ is an instance of (eq refl) then we apply Lemma 14. We just show the case of (eq β). Hence M is $(\lambda x : \xi.M_1)M_2$ and N is $M_1[M_2/x]$. Moreover we have

$$(y : \sigma) * E'' * (x : \xi) \vdash_{ND} M_1 : \tau \qquad\qquad (z : \delta) \vdash_{ND} M_2 : \xi,$$

where $E' = E'' * (z : \delta)$.

By Lemma 14 we have

$$E * E'' * (x : \xi) \vdash_{ND} M_1[P/y] = M_1[Q/y] : \tau$$

Thus, applying (eq abs) and (eq appl) we obtain

$$E * E' \vdash_{ND} (\lambda x : \xi . M_1[P/y])M_2 = (\lambda x : \xi . M_1[Q/y])M_2 : \tau$$

and by (eq β) and (trans) we conclude that

$$E * E' \vdash_{ND} (\lambda x : \xi . M_1[P/y])M_2 = M_1[Q/y][M_2/x] : \tau$$

Since $y \notin FV(M_2)$ and $x \notin FV(Q)$, it follows that the left hand side of the above equality is $M[P/y]$ and the right hand side is $N[Q/y]$. The other cases we leave for the reader. ∎

Lemma 16 (Substitution)
If $E \vdash_{ND} P = Q : \sigma$ and $(y : \sigma) * E' \vdash_{ND} M = N : \tau$, then $E * E' \vdash_{ND} M[P/y] = N[Q/y] : \tau$.

Proof: This is a routine induction on the derivation of $(y : \sigma) * E' \vdash M = N : \tau$ in $\vdash_{ND}$. The base case is handled by Lemma 15. We leave the details for the reader. ∎

Theorem 17 (Coherence)
If $E \vdash_{ND} M_1 : \tau$ and $E \vdash_{ND} M_2 : \tau$, then $E \vdash_{ND} M_1 = M_2 : \tau$.

Proof: Let $E = (x_1 : \sigma_1) * \ldots * (x_n : \sigma_n)$. The proof is by induction on the sum of lengths of types in E and τ.

For the base step we have $E = (x_1 : X)$ and $\tau = X$. By the assumption we have $(x_1 : X) \vdash_{ND} M_i : X$, for $i = 1, 2$. By Lemma 6 we obtain that $M_i \rightarrow_\beta^* x_1$ for $i = 1, 2$, and thus $(x_1 : X) \vdash_{ND} M_1 = M_2 : X$.

For induction step we consider several cases.

I. $\tau \equiv \tau_1 \rightarrow \tau_2$.
In this case let's consider for $i = 1, 2$, $E * (y : \tau_1) \vdash_{ND} M_i y : \tau_2$, where y is a fresh variable. By induction hypothesis we have

$$E * (y : \tau_1) \vdash_{ND} M_1 y = M_2 y : \tau_2$$

Thus

$$E \vdash_{ND} (\lambda y : \tau_1 .\ M_1 y) = (\lambda y : \tau_1 .\ M_2 y) : \tau$$

Since left side is η equal to M_1 and right side is η equal to M_2, the conclusion follows.

II. $\tau \equiv \forall X.\ \tau_1$.
Let X be a fresh type variable. We have $E \vdash_{ND} M_i X : \tau_1$, for $i = 1, 2$. By induction assumption we obtain

$$E \vdash_{ND} M_1 X = M_2 X : \tau_1$$

Hence

$$E \vdash_{ND} (\Lambda X.\ M_1 X) = (\Lambda X.\ M_2 X) : \tau$$

And again by η equality we obtain $E \vdash_{ND} M_1 = M_2 : \tau$.

III. $\tau \equiv X$.
In this case let's consider the various forms for σ_1, the leftmost type in E.

III-a. $\sigma_1 \equiv Y$.
It follows (see Lemma 6) that E consists of just one item and $Y = X$ is the base case we have considered at the beginning of this proof.

III-b. $\sigma_1 \equiv \sigma_1' \to \sigma_2''$.
For $i = 1, 2$, let N_i be the $\beta\eta$ normal form of M_i. Since $\sigma_1 \not\equiv \tau$, it follows from Lemma 7(i) that $N_i \not\equiv x$. Thus, by Lemma 7(ii), it follows that $n > 1$ and there is $m_i \geq 0$ such that N_i is of the form

$$N_i \equiv x_1 N_i' G_1^{(i)} \ldots G_{m_i}^{(i)}$$

for some term N_i' such that $(x_2 : \sigma_2) \vdash_{ND} N_i' : \sigma_1'$ holds and each $G_j^{(i)}$ is either a term or a type. By induction hypothesis we obtain

$$(x_2 : \sigma_2) \vdash N_1' = N_2' : \sigma_1'$$

Since

$$(x_1 : \sigma_1)(x_2 : \sigma_2) \vdash_{ND} x_1 N_i' : \sigma_1'' \tag{1}$$

it follows that if z is a fresh variable then

$$(z : \sigma_1'') * (x_3 : \sigma_3) * \ldots * (x_n : \sigma_n) \vdash_{ND} z G_1^{(i)} \ldots G_{m_i}^{(i)} : X$$

By induction hypothesis we obtain

$$(z : \sigma_1'') * (x_3 : \sigma_3) * \ldots * (x_n : \sigma_n) \vdash_{ND} z G_1^{(1)} \ldots G_{m_1}^{(1)} = z G_1^{(2)} \ldots G_{m_2}^{(2)} : X$$

Thus, by (1), by Lemma 16 and the above equality we obtain

$$E \vdash_{ND} N_1 = N_2 : X$$

This completes the proof of III-b.

III-c. $\sigma_1 \equiv \forall Y.\ \sigma_1'$.
Again, for $i = 1, 2$, let N_i be the $\beta\eta$ normal form of M_i. As in the case III-b we conclude (by Lemma 7(i)) that $N_i \not\equiv x$. By Lemma 7(iii) it follows that there is $m_i \geq 0$ such that N_i is of the form

$$N_i \equiv x_1 \rho_i G_1^{(i)} \ldots G_{m_i}^{(i)}$$

for some type ρ_i, and each $G_j^{(i)}$ is either a term or a type. Since $(x_1 : \forall Y.\ \sigma_1') \vdash_{ND} x_1 \rho_i : \sigma_1'[\rho_i/Y]$, it follows that if y is a fresh variable, then

$$(y : \sigma_1'[\rho_i/Y]) * (x_2 : \sigma_2) * \ldots (x_n : \sigma_n) \vdash_{ND} y G_1^{(i)} \ldots G_{m_i}^{(i)} : X$$

Thus, by (eq appl2 co), we obtain

$$E \vdash_{ND} N_1 = N_2 : X$$

as desired. ∎

7 Gentzen Style Sequent Calculus and Cut Elimination

In this section we will present a Gentzen-style sequent calculus for the logic of coercions.

Axiom: $(x : \sigma) \vdash (x : \sigma)$

Rules:

$$(\rightarrow\text{-L}) \quad \frac{(y : \rho) \vdash M : \sigma_1 \qquad (x : \sigma_2) * E \vdash N : \tau}{(x : \sigma_1 \rightarrow \sigma_2) * (y : \rho) * E \vdash N[xM/x] : \tau} \qquad y \neq x \text{ and } y \notin Dom(E)$$

$$(\rightarrow\text{-R}) \quad \frac{E * (x : \sigma) \vdash M : \tau}{E \vdash (\lambda x : \sigma.M) : \sigma \rightarrow \tau} \quad (E \neq \emptyset)$$

$$(\forall\text{-L}) \quad \frac{(x : \sigma[\rho/X]) * E \vdash M : \tau}{(x : \forall X.\sigma) * E \vdash M[(x\rho)/x] : \tau}$$

$$(\forall\text{-R}) \quad \frac{E \vdash M : \sigma}{E \vdash (\Lambda X.M) : \forall X.\sigma} \qquad (X \notin FV(E))$$

We write $E \vdash_S M : \tau$ to denote that the sequent $E \vdash M : \tau$ is derivable in the above system.

The immediate goal is to prove that the natural deduction system and the above introduced Gentzen-style sequent calculus are equivalent. We will achieve this in two steps. First we show that $\vdash_{ND}$ is equivalent to $\vdash_S$ extended with (cut) and then we show that (cut) is admissible in $\vdash_S$. Let $\vdash_{S+}$ denote the system $\vdash_S$ augmented with (cut).

We first need a technical lemma.

Lemma 18 *If $E \vdash_{S+} M : \sigma$, then for every type ξ and for every type variable α, $E[\xi/\alpha] \vdash_{S+} M[\xi/\alpha] : \sigma[\xi/\alpha]$.*

Proof: A routine induction on the length of derivation of $E \vdash_{S+} M : \sigma$. We leave the details for the reader. ∎

Proposition 19 *The systems $\vdash_{ND}$ and $\vdash_{S+}$ are of equal power, i.e., for every sequent $E \vdash M : \tau$ we have $E \vdash_{ND} M : \tau$ iff $E \vdash_{S+} M : \tau$.*

Proof: For the implication $\Leftarrow$ it is enough to observe that ($\rightarrow$-L) is admissible in $\vdash_{ND}$ – a proof very similar to the proof of admissibility of ($\rightarrow$) in $\vdash_{ND}$ (see the proof of Theorem 9) is left for the reader. Also admissibility of (cut) and ($\forall$-L) in $\vdash_{ND}$ is proved in the proof of Theorem 9, when proving admissibility of ($\forall_L$). Admissibility of (cut) follows from Proposition 8.

For the proof in the opposite direction we have to show that both elimination rules of $\vdash_{ND}$ are admissible in $\vdash_{S+}$. For the proof of admissibility of ($\rightarrow$-elim) in $\vdash_{S+}$ let us assume

$$E \vdash_{S+} M : \sigma \rightarrow \tau \quad \text{and} \quad (x : \rho) \vdash_{S+} N : \sigma.$$

We prove $E * (x : \rho) \vdash_{S+} (MN) : \tau$ by induction on the length of derivation of $E \vdash_{S+} M : \sigma \rightarrow \tau$. The induction is routine. Let's just consider the case when the last rule in the derivation of $E \vdash_{S+} M : \sigma \rightarrow \tau$ was ($\rightarrow$-R) — this is the place of the proof where we need (cut). Thus we have $M = \lambda x : \sigma.M'$ and

$$E * (y : \sigma) \vdash_{S+} M' : \tau$$

Applying (cut) to $(x : \rho) \vdash_{S+} N : \sigma$ and the above sequence we obtain the conclusion.

For the proof of admissibility of ($\forall$-elim) let's assume that

$$E \vdash_{S+} M : \forall X\, \tau \tag{2}$$

we prove $E \vdash_{S+} M\xi : \tau(\xi/X)$ by induction on the length of derivation of (2). The proof is again routine. For the case when the last rule in derivation of (2) was ($\forall$-R) we use Lemma 18. We leave the details for the reader. ∎

Theorem 20 (Cut Elimination)
If $E \vdash_{S+} M : \tau$, then $E \vdash_S M' : \tau$, where M' is a β-normal form of M.

Proof: Let $E \vdash_{S+} M : \tau$. By Proposition 19, subject reduction (Theorem 11) and strong normalization (Theorem 13) we can assume without loss of generality that M is in β-normal form. It follows that in the derivation of $E \vdash M : \tau$ in $\vdash_{S+}$ all terms are in β-normal form — an easy proof by induction on the structure of this derivation is left for the reader.[10]

Next we introduce a cut measure on derivations Δ of $E \vdash M : \tau$. For each occurence of cut rule in Δ, cut measure of this occurence is the number of nodes in Δ which are above that occurence. *Cut measure* of a derivation Δ is the sum of cut measures of all occurences of cut rules in Δ. We prove that if $E \vdash_{S+} M : \tau$ and M is in β-normal form, then $E \vdash_S M : \tau$. The proof is by induction on the cut measure of a derivation of $E \vdash M : \tau$ in $\vdash_{S+}$.

Take a derivation Δ of $E \vdash M : \tau$ in $\vdash_{S+}$ with positive cut measure. Consider an occurence of cut rule in Δ such that no cut rule occurs above it. Let $\hat{\Delta}$ be the derivation which consists of all nodes of Δ which are above this occurence of the cut rule, including that occurence. Thus $\hat{\Delta}$ looks as follows.

[10] An important property of the type system used in this proof is that if $E \vdash M : \tau$ is derivable in $\vdash_{S+}$, then every object variable which occurs in E also occurs freely in M.

$$\dfrac{\begin{array}{c}\vdots\,\Delta_1\\ E_1 \vdash N_1 : \tau_1\end{array} \qquad \begin{array}{c}\vdots\,\Delta_2\\ E_2 \vdash N_2 : \tau_2\end{array}}{E \vdash N : \tau_2}\;(\text{cut})$$

If $E_1 \vdash N_1 : \tau_1$ is an axiom, then $E_2 \vdash N_2 : \tau_2$ and $E \vdash N : \tau_2$ are identical, up to variable renaming, and we replace $\hat{\Delta}$ by the following derivation

$$\begin{array}{c}\vdots\,\Delta_2'\\ E \vdash N : \tau_2\end{array}$$

where derivation Δ_2' is obtained from Δ_2 by variable renaming so as to yield $E \vdash N : \tau_2$. The resulting derivation has smaller cut measure and we can apply the induction hypothesis.

If the last rule in Δ_1 is ($\rightarrow$-L) then Δ_1 looks as follows

$$\dfrac{\begin{array}{c}\vdots\,\Delta_1'\\ E_1' \vdash N_1' : \tau_1'\end{array} \qquad \begin{array}{c}\vdots\,\Delta_1''\\ E_1'' \vdash N_1'' : \tau_1\end{array}}{E_1 \vdash N_1 : \tau_1}\;(\rightarrow\text{-L})$$

Thus we can replace $\hat{\Delta}$ by the following derivation.

$$\dfrac{\begin{array}{c}\vdots\,\Delta_1'\\ E_1' \vdash N_1' : \tau_1'\end{array} \qquad \dfrac{\begin{array}{c}\vdots\,\Delta_1''\\ E_1'' \vdash N_1'' : \tau_1\end{array} \qquad \begin{array}{c}\vdots\,\Delta_2\\ E_2 \vdash N_2 : \tau_2\end{array}}{E_1''' \vdash N_1''' : \tau_2}\;(\text{cut})}{E \vdash N : \tau_2}\;(\rightarrow\text{-L})$$

where $E_1''' \vdash N_1''' : \tau_2$ is a suitably chosen sequent. For this transformation to work we have to notice that if y is a variable which doesn't occur freely in a term P, then for arbitrary terms Q_1, Q_2 and for an arbitrary variable x we have

$$P[Q_1/x][Q_2/y] = P[Q_1[Q_2/y]/x]$$

We leave the easy details of this transformation for the reader. The derivation which results from this transformation has smaller cut measure and we can apply the induction hypothesis.

If the last rule in Δ_1 is ($\forall$-L) then we proceed in a similar way, pushing the cut rule one level up through the left branch.

If the last rule in Δ_1 is ($\rightarrow$-R) or ($\forall$-R) then, since N is in β-normal form, it follows that either $E_2 \vdash N_2 : \tau_2$ is an axiom, or the last rule in Δ_2 is either ($\rightarrow$-R) or ($\forall$-R) .[11] We proceed similarly as in the previous part of the proof, eliminating this cut, if $E_2 \vdash N_2 : \tau_2$ is an axiom, or pushing it up through the right branch. In any case this results in decreasing the cut measure of the derivation. We leave the easy details for the reader. ∎

[11] It can be neither ($\rightarrow$-L) , nor ($\forall$-L) , since this would result in a β-redex in M.

8 Type Inhabitation and Type Reconstruction

Despite simplicity of the metatheory of the sequent calculus presented in this paper, the following problem is undecidable: given a sequent $\sigma_1 * \ldots * \sigma_n \vdash \tau$, is there a term M such that $(y_1 : \sigma_1) * \ldots * (y_n : \sigma_n) \vdash_{ND} M : \tau$ holds? This problem is known as the *inhabitation problem*: assuming that we have objects of types $\sigma_1, \ldots, \sigma_n$, can we construct an object of type τ. This result has been proved by P. Urzyczyn and the author of this paper (see [18]). It is proved there that for types σ and τ the relation $\vdash \sigma \sqsubseteq \tau$ is undecidable. The reader may also consult [21] for another proof of this result. Thus none of the systems: Mitchell's containment relation, $\vdash_{LMS}$, $\vdash_{ND}$, $\vdash_S$, and $\vdash_{S+}$ is decidable. The reader may compare this result with undecidability of type inhabitation for the system **F** which was proved D.M. Gabbay ([9]) and independently by M.H. Löb (see [13]).

It is perhaps interesting to contrast the undecidability result of [18] with decidability of the following problem: given two types σ and τ, does $\vdash \sigma \sqsubseteq \tau$ and $\vdash \tau \sqsubseteq \sigma$ hold? This problem turns out to be decidable (see [17]). This is a consequence of a simple equational axiomatization of the above relation. This axiomatization is given in [17].

Type inhabitation problem is concerned with decidability of the set of theorems of a given logic. Hence it has a clear logical motivation. There is another problem which has a computational motivation and is in a ceratin sense dual to the above mentioned problem. It is known as *type reconstruction problem*. An instance of this problem is a pure lambda term M and we are interested whether it can be typed in $\vdash_{ND}$, i.e. whether there is a typed term Q, a noncommutative environment E and a type τ such that $erase(Q) = M$, and $E \vdash_{ND} M : \tau$. We will prove that this problem is decidable. The reader may find it interesting to compare it with undecidability of type reconstruction for system **F** (see [20]).

We are going to give a simple syntactic characterization of pure lambda terms which are typable in $\vdash_{ND}$. Call a term M *strongly linear* if for every subterm N of M, $N \to^*_{\eta\beta} x_1 \ldots x_n$, where $x_1, \ldots, x_n$ are pairwise different variables, and $n \geq 1$.

Proposition 21 *If $\lambda x.M$ is a strongly linear term, then M has exactly one free occurence of x. Hence all β-reductions performed on strongly linear terms shrink their size. In particular, it follows that the set of strongly linear terms is decidable and that strongly linear terms are strongly normalizing.*

Proof: This is a routine induction on M of the following more general satement: if M strongly linear, then every variable occuring free in M has exactly one occurence and for every subterm of M of the form $\lambda y.N$, N has exactly one free occurence of y. We leave the easy details for the reader.

Decidability of the property "being strongly linear" follows from the previous part — given a term M, if it is strongly linear it has to shrink its size during every step of $\beta\eta$-reductions, hence the normal form (if it exists), or the information that there is none, is computable in polynomial time. ∎

Before giving the main result of this section we state the following technical lemma which will be needed further. It says that the converse to one of the cut rules is admissible in $\vdash_{ND}$.

Lemma 22 *Let Q and P be typed terms and let x, y be variables such that $x \in FV(Q)$ and $erase(P) \to^*_{\eta\beta} y$. If for some E, F and types ρ, τ we have*

$$E * (y : \rho) * F \vdash_{ND} Q[P/x] : \tau,$$

then for some type σ with $FV(\sigma) \subseteq FV(\rho) \cup FV(P)$ we have

$$E * (x : \sigma) * F \vdash_{ND} Q : \tau \qquad \textit{and} \qquad (y : \rho) \vdash_{ND} P : \sigma.$$

Proof: We proceed by induction on Q. If $Q = x$, then we have $E * (y : \rho) * F \vdash_{ND} P : \tau$. Since $erase(P) \to^*_{\eta\beta} y$ it follows[12] that $E = F = \emptyset$. If $X_1, \ldots, X_n$ are free type variables of τ which occur neither in ρ nor in P, then, by Lemma 5, we have $(y : \rho) \vdash_{ND} P : \tau[\bot/X_1, \ldots, \bot/X_n]$. Thus $\sigma = \tau[\bot/X_1, \ldots, \bot/X_n]$ is the sought type.[13]

The cases $Q = \lambda z : \xi.Q_1$ and $Q = Q_1\xi$ are completely routine. For the case $Q = Q_1Q_2$ we use the observation that x occurs exacly once in Q (otherwise $y \in FV(P)$ would occur more than once in $Q[P/x]$, which is a contradiction with typability in $\vdash_{ND}$). For the case $Q = \Lambda X.Q_1$ we use the assumption that $FV(\sigma) \subseteq FV(\rho) \cup FV(P)$. We leave the details for the reader. ∎

The next result gives a full charcterization of terms typable in $\vdash_{ND}$. It also yields decidability of type reconstruction for $\vdash_{ND}$.

Theorem 23 (Type Reconstruction)
The following conditions are equivalent:
(i) *M is strongly linear;*
(ii) *There is a typed term Q, such that $erase(Q) = M$ and for some E and τ, $E \vdash_{ND} Q : \tau$.*
Hence the type reconstruction problem for $\vdash_{ND}$ is decidable.

Proof: The implication (ii)⇒(i) is proved by an obvious induction on Q. For the proof of (i)⇒(ii) we prove by induction on M the following property. If M is strongly linear and $M \to^*_{\eta\beta} y_1 \ldots y_n$, then there are types $\tau_1, \ldots, \tau_n, \tau$ and a typed term Q such that $erase(Q) = M$ and $(y_1; \tau_1) * \ldots * (y_n : \tau_n) \vdash_{ND} Q : \tau$.

The base case is obvious. If $M = \lambda x.M_1$, then strong linearity of M implies that if $M \to^*_{\eta\beta} y_1 \ldots y_n$, then $M_1 \to^*_{\eta\beta} y_1 \ldots y_n x$ (the outermost lambda abstraction can only be eliminated by η-reduction). We apply the induction hypothesis and we are done.

If $M = xM_1 \ldots M_n$, $(n \geq 1)$, then strong linearity implies that each $M_i \to^*_{\eta\beta} y_i$ and $y_1, \ldots, y_n$ are pairwise different. We apply the induction hypothesis to all $M_1, \ldots, M_n$ and we are done.

The last case is $M = (\lambda x.M_0)M_1 \ldots M_n$, with $n \geq 1$. By Proposition 21 $M_0[M_1/x]M_2 \ldots M_n$ has smaller size than M. Moreover we have that for some $m \geq 1$,

$$M_0 \to^*_{\eta\beta} y_1 \ldots y_m x \tag{3}$$

[12] An easy induction on P we leave for the reader.
[13] Let's recall that $\bot$ denotes the type $\forall X.X$.

Since $(\lambda x.M_0)M_1$ is strongly linear, it follows from (3) and the Church-Rosser property (see [2]) that for pairwise different variables $z_1, \ldots, z_n$ we have $M_i \rightarrow^*_{\eta\beta} z_i$ for $i = 1, \ldots, n$, and $M \rightarrow^*_{\eta\beta} y_1 \ldots y_m z_1 \ldots z_n$. By induction hypothesis there are types $\sigma_1, \ldots, \sigma_m, \tau_1, \ldots, \tau_n, \tau$ and a typed term Q such that $erase(Q) = M_0[M_1/x]M_2 \ldots M_n$ and $E * (z_1 : \tau_1) * F \vdash_{ND} Q : \tau$, where $E = (y_1 : \sigma_1) * \ldots * (y_m : \sigma_m)$ and $F = (z_2 : \tau_2) * \ldots * (z_n : \tau_n)$.

It follows that Q must be of the form $Q_0[Q_1/x]Q_2 \ldots Q_n$, for some typed terms[14] $Q_0, \ldots Q_n$, with $erase(Q_i) = M_i$, for $i = 0, \ldots, n$. By Lemma 22, there is a type σ such that

$$E * (x : \sigma) * F \vdash_{ND} Q_0 Q_2 \ldots Q_n : \tau \tag{4}$$

and

$$(z_1 : \tau_1) \vdash_{ND} Q_1 : \sigma. \tag{5}$$

The last $n-1$ rules in the derivation of (4) are ($\rightarrow$-elim) rules. Thus there are types $\xi_2, \ldots, \xi_n$ such that $E * (x : \sigma) \vdash_{ND} Q_0 : \xi_2 \rightarrow \ldots \rightarrow \xi_n \rightarrow \tau$ and $(z_i : \tau_i) \vdash_{ND} Q_i : \xi$ for $i = 2, \ldots n$. By (5) and the above we get $E * (z_1 : \tau_1) * F \vdash_{ND} (\lambda x : \sigma.Q_0)Q_1 \ldots Q_n : \tau$. Thus $(\lambda x : \sigma.Q_0)Q_1 \ldots Q_n$ is the sought term.

The second part of the theorem follows from the first part and from Proposition 21. This completes the proof. ∎

9 Conclusion

We have introduced a sequent calculus for deriving sequents of the form $\sigma_1 * \ldots * \sigma_n \vdash \tau$, with $n \geq 1$. The calculus comes in two flavors: as a natural deduction system and as a Gentzen style sequent calculus. The rules of the systems are annotated with terms in a smooth and natural way. Hence we have an instance of the Curry-Howard isomorphism which links logical and computational aspects of the logic. The logic which emerges from these systems has many properties including: coherence, subject reduction, strong normalization and decidable problem of type reconstruction. This logic is undecidable.

It is interesting to compare our logic with the *unified calculus*, **LU** of J.-Y. Girard [12]. This system deals with sequents of the form $\Gamma; \Gamma' \vdash \Delta'; \Delta$, i.e., sequents have four zones: two *linear zones* Γ and Δ and two *classical zones* Γ', Δ'. Our logic can be embedded in a certain sense[15] into **LU** by representing the sequent $\sigma_1 * \ldots * \sigma_n \vdash \tau$ by $\sigma_1, \ldots, \sigma_n; \vdash ; \tau$ (i.e. classical zones are empty). Thus our logic can be qualified as non-commutative linear intuitionistic logic.

Acknowledgment:
Part of the results presented in this paper was obtained by the author during his visits in 1994 and 1995 at Ecole Normale Superieure in Paris. He would like to thank French Ministry of Research, the French Embassy and Polish State Research Agency (KBN) for generous grants to support these

[14] These terms need not be unique.

[15] There is problem here since **LU** is a commutative logic, i.e. it has exchange rules, while our logic is non-commutative. The reader has to be careful with this embedding since some rules of *LU*, notably introduction of linear implication on the left, have to be rewritten in order to accomodate to the non-commutative situation.

visits. He would also like to thank Pierre-Louis Curien, Jean-Yves Girard, Giuseppe Longo and Sergei Soloviev for stimulating discussions on the topics presented here. Thanks are also for Paweł Urzyczyn for his comments on the previous version of this paper.

References

[1] E.S. Bainbridge, P.J. Freyd, A. Scedrov, and P.J. Scott. Functorial polymorphism. *Theoretical Computer Science*, 70:35–64, 1989. Corrigendum *ibid.*, 71:431, 1990. Preliminary report in *Logical Foundations of Functional Programming,* ed. G. Huet, Addison-Wesley (1990) 315–327.

[2] H.P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* North-Holland, Amsterdam, 1984. Second edition.

[3] K. Bruce and G. Longo. A modest model of records, inheritance and bounded quantification. *Information and Computation*, 87(1/2):196–240, 1990.

[4] K. Bruce and J.C. Mitchell. PER models of subtyping, recursive types and higher-order polymorphism. In *Proc. 19th ACM Symp. on Principles of Programming Languages*, pages 316–327, January 1992.

[5] L. Cardelli and G. Longo. A semantic basis for Quest. Technical Report 55, DEC Systems Research Center, 1990. To appear in *J. Functional Programming.*

[6] L. Cardelli, S. Martini, J. Mitchell, and A. Scedrov. An extension of System F with subtyping. *Information and Computation*, 109:4–56, 1994. Preliminary version appeared *Proc. Theor. Aspects of Computer Software,* Springer LNCS 526, September 1991, pages 750–770.

[7] L. Cardelli and P. Wegner. On understanding types, data abstraction, and polymorphism. *Computing Surveys*, 17(4):471–522, 1985.

[8] P.-L. Curien and G. Ghelli. Coherence of subsumption, minimum typing and type-checking in $F_{\leq}$. *Math. Struct. in Comp. Sci.*, 2:55–91, 1992.

[9] D.M. Gabbay. On 2nd order intuitionistic propositional calculus with full comprehension. *Arch. Math. Logik*, 16:177–186, 1974.

[10] J.-Y. Girard. Une extension de l'interpretation de Gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types. In J.E. Fenstad, editor, *2nd Scandinavian Logic Symposium*, pages 63–92. North-Holland, Amsterdam, 1971.

[11] J.-Y. Girard. Interpretation fonctionelle et elimination des coupures de l'arithmetique d'ordre superieur. These D'Etat, Universite Paris VII, 1972.

[12] J.-Y. Girard. On the unity of logic. *Annals of Pure and Applied Logic*, 59:201–217, 1993.

[13] M.H. Löb. Embedding first order predicate logic in fragments of intuitionistic logic. *Journal of Symbolic Logic*, 41:705–718, 1976.

[14] G. Longo, K. Milsted, and S. Soloviev. A logic of subtyping. In *Proc. IEEE Symp. on Logic in Computer Science*, pages 292–299, 1995.

[15] J.C. Mitchell. Polymorphic type inference and containment. *Information and Computation*, 76(2/3):211–249, 1988. Reprinted in *Logical Foundations of Functional Programming,* ed. G. Huet, Addison-Wesley (1990) 153–194.

[16] J.C. Reynolds. Towards a theory of type structure. In *Paris Colloq. on Programming*, pages 408–425, Berlin, 1974. Springer-Verlag LNCS 19.

[17] J. Tiuryn. Equational axiomatization of bicoercibility for polymorphic types. In Ed. P.S. Thiagarajan, editor, *Proc. 15th Conference Foundations of Software Technology and Theoretical Computer Science*, volume 1026 of *Lecture Notes in Computer Science*, pages 166–179. Springer Verlag, 1995.

[18] J. Tiuryn and P. Urzyczyn. The subtyping problem for second-order types is undecidable. In *Proc. LICS 96, to appear*, 1996.

[19] A.S. Troelstra and D. van Dalen. *Constructivism in Mathematics. An Introduction. Volume 1.* North-Holland, 1988. ISBN-0-444-705066.

[20] J. Wells. Typability and type checking in the second-order λ-calculus are equivalent and undecidable. In *Proc. 9th Ann. IEEE Symp. Logic in Comput. Sci.*, pages 176–185, 1994.

[21] J. Wells. The undecidability of Mitchell's subtyping relation. Technical report, Computer Sci. Dept., Boston University, December 1995.