On the Strong Normalisation of Natural Deduction with Permutation-Conversions

Philippe de Groote

LORIA UMR n^o 7503 – INRIA
Campus Scientifique, B.P. 239
54506 Vandœuvre lès Nancy Cedex – France
e-mail: Philippe.de.Groote@loria.fr

Abstract. We present a modular proof of the strong normalisation of intuitionistic logic with permutation-conversions. This proof is based on the notions of negative translation and CPS-simulation.

1 Introduction

Natural deduction systems provide a notion of proof that is more compact (or, quoting Girard [6], more *primitive*) than that of sequent calculi. In particular, natural deduction is better adapted to the study of proof-normalisation procedures. This is true, at least, for the intuitionistic systems, where proof-normalisation expresses the computational content of the logic. Nevertheless, even in the intuitionistic case, the treatments of disjunction and existential quantification are problematic. This is due to the fact that the elimination rules of these connectives introduce arbitrary formulas as their conclusions. Consequently, in order to satisfy the subformula property, the so-called permutation-conversions are needed.

Strong normalisation proofs for intuitionistic logic [12] are more intricate in the presence of permutation-conversions. For instance, the proofs given in textbooks such as [6] and [14] do not take permutation-conversions into account.[1] In this paper, we revisit this problem and present a simple proof of the strong normalisation of intuitionistic logic with permutation-conversions. This proof, which is inspired by a similar proof in [4], has several advantages:

- It is modular and, therefore, easily adaptable to other systems. Indeed, the problem related to the interaction between permutation- and detour-conversions is avoided (see Lemma 6, in Section 5).
- It is based on a continuation-passing-style interpretation of intuitionistic logic, which sheds light on the computational content of the several conversion rules. In particular, it shows that the computational content of permutation-conversions is nil.

[1] In [6], extending the proof to permutation-conversions is left to the reader while in [14], the technique that is used is not adapted to the case of permutation-conversions.

– It is based on an arithmetisable translation of intuitionistic logic into the simply typed λ-calculus. Consequently, when combined with an arithmetisable proof of strong normalisation of the simply typed λ-calculus (see [3], for instance), it yields a completely arithmetisable proof of the strong normalisation of intuitionistic logic. This must be contrasted with the proof in [14], which is based on an interpretation into higher-order Heyting arithmetic.

The paper is organised as follows.

Section 2 is an introduction to the proof-theory of intuitionistic positive propositional logic (**IPPL**). In particular, we define the notions of detour- and permutation-conversions by means of an associated λ-calculus ($\lambda^{\to\wedge\vee}$). Our presentation is essentially inspired by [14].

In Section 3, we establish the strong normalisation of **IPPL** with respect to permutation-conversions. The proof consists simply in assigning a norm to the untyped terms of $\lambda^{\to\wedge\vee}$, and showing that this norm strictly decreases by permutation conversion.

Section 4 provides a negative translation of **IPPL** into the implicative fragment of intuitionistic logic. At the level of the proofs, this negative translation corresponds to a CPS-simulation of $\lambda^{\to\wedge\vee}$ into the simply typed λ-calculus. We show that this CPS-simulation (slightly modified) commutes with the relations of detour-conversion and β-reduction, from which we conclude that $\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to detour-conversions.

In Section 5, we collect the results of the two previous sections in order to show that $\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to detour- and permutation-conversions mixed together. To this end, we show that the modified CPS-translation of Section 4 interprets the relation of permutation-conversion as equality. This means that we have found a negative translation commuting with the permutation-conversions, answering a problem raised by Mints [9].

Our proof may be easily adapted to full intuitionistic propositional calculus (by adding negation), and to first-order intuitionistic logic (by adding quantifiers). We do not present this extension here, for the lack of space.

2 Intuitionistic positive propositional logic

2.1 Natural deduction

The formulas of intuitionistic positive propositional logic (**IPPL**) are built up from an alphabet of atomic propositions $\mathcal{A}$ and the connectives $\to$, $\wedge$, and $\vee$ according to the following grammar:

$$\mathcal{F} \ ::= \ \mathcal{A} \mid \mathcal{F} \to \mathcal{F} \mid \mathcal{F} \wedge \mathcal{F} \mid \mathcal{F} \vee \mathcal{F}$$

Following Gentzen [5], the meaning of the connectives is specified by the introduction and elimination rules of the following natural deduction system (where a bracketed formula corresponds to a hypothesis that may be discarded when applying the rule).

$$\dfrac{\begin{array}{c}[\,\alpha\,]\\ \beta\end{array}}{\alpha \rightarrow \beta} \qquad \dfrac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

$$\dfrac{\alpha \quad \beta}{\alpha \wedge \beta} \qquad \dfrac{\alpha \wedge \beta}{\alpha} \qquad \dfrac{\alpha \wedge \beta}{\beta}$$

$$\dfrac{\alpha}{\alpha \vee \beta} \qquad \dfrac{\beta}{\alpha \vee \beta} \qquad \dfrac{\alpha \vee \beta \quad \begin{array}{c}[\,\alpha\,]\\ \gamma\end{array} \quad \begin{array}{c}[\,\beta\,]\\ \gamma\end{array}}{\gamma}$$

As observed by Prawitz [11], an introduction immediately followed by an elimination corresponds to a *detour* that can be (locally) eliminated. Consider, for instance, the case of implication:

$$\dfrac{\dfrac{\begin{array}{c}[\,\alpha\,]\\ \vdots\ \Pi_1\\ \beta\end{array}}{\alpha \rightarrow \beta}\text{(Intro.)} \qquad \begin{array}{c}\vdots\ \Pi_2\\ \alpha\end{array}}{\beta}\text{(Elim.)}$$

Π_1 is a proof of β under the hypothesis α. On the other hand, Π_2 is a proof of α. Consequently, one may obtain a direct proof of β by grafting Π_2 at every place where α occurs in Π_1 as a hypothesis discarded by Rule (Elim.):

$$\begin{array}{c}\vdots\ \Pi_2\\ \alpha\\ \vdots\ \Pi_1\\ \beta\end{array}$$

When such local reduction steps allow any proof to be transformed into a proof without detour, one says that the given natural deduction system satisfies the *normalisation property.* Moreover, when this property holds independently of the strategy that is used in applying the reduction steps, one says that the system satisfies the *strong normalisation property.*

2.2 Natural deduction as a term calculus

As well-known, there exists a correspondence between natural deduction systems and typed λ-calculi, namely, the Curry-Howard isomorphism [2, 7]. This correspondence, which is described in the table below, allows natural deduction proofs to be denoted by terms.

Natural deduction	λ-calculus
propositions	types
connectives	type constructors
proofs	terms
introduction rules	term constructors
elimination rules	term destructors
active hypothesis	free variables
discarded hypothesis	bound variables

In the case of **IPPL**, the corresponding term calculus is the simply typed λ-calculus with product and coproduct, which we will call $\lambda^{\rightarrow\wedge\vee}$. In particular, introduction and elimination rules for conjunction correspond to pairing and projection functions, while introduction and elimination rules for disjunction correspond to injection and case analysis functions. The syntax is described by the following grammar, where $\mathcal{X}$ is a set of variables:

$$\mathcal{T} ::= \mathcal{X} \mid \lambda\mathcal{X}.\,\mathcal{T} \mid (\mathcal{T}\,\mathcal{T}) \mid \mathbf{p}(\mathcal{T},\mathcal{T}) \mid \mathbf{p}_1\,\mathcal{T} \mid \mathbf{p}_2\,\mathcal{T} \mid \mathbf{k}_1\,\mathcal{T} \mid \mathbf{k}_2\,\mathcal{T} \mid \mathbf{D}_{\mathcal{X},\mathcal{X}}(\mathcal{T},\mathcal{T},\mathcal{T})$$

and the typing rules are as follows:

$$\frac{\begin{array}{c}[\,x:\alpha\,]\\ M:\beta\end{array}}{\lambda x.\,M:\alpha\to\beta} \qquad \frac{M:\alpha\to\beta \quad N:\alpha}{M\,N:\beta}$$

$$\frac{M:\alpha \quad N:\beta}{\mathbf{p}(M,N):\alpha\wedge\beta} \qquad \frac{M:\alpha\wedge\beta}{\mathbf{p}_1\,M:\alpha} \qquad \frac{M:\alpha\wedge\beta}{\mathbf{p}_2\,M:\beta}$$

$$\frac{M:\alpha}{\mathbf{k}_1\,M:\alpha\vee\beta} \qquad \frac{M:\beta}{\mathbf{k}_2\,M:\alpha\vee\beta} \qquad \frac{M:\alpha\vee\beta \quad \begin{array}{c}[\,x:\alpha\,]\\ N:\gamma\end{array} \quad \begin{array}{c}[\,y:\beta\,]\\ O:\gamma\end{array}}{\mathbf{D}_{x,y}(M,N,O):\gamma}$$

2.3 Detour-conversion rules

The Curry-Howard isomorphism is not only a simple matter of notation. Its deep meaning lies in the relation existing between proof normalisation and λ-term evaluation. Indeed an introduction immediately followed by an elimination corresponds to a term destructor applied to term constructor. Consequently, Prawitz's detour elimination steps amount to evaluation steps. For instance, the detour elimination step for implication corresponds exactly to the familiar notion of β-reduction:

$$\frac{\dfrac{\begin{array}{c}[\,x:\alpha\,]\\ \vdots\ \Pi_1\\ M:\beta\end{array}}{\lambda x.\,M:\alpha\to\beta} \quad \begin{array}{c}\vdots\ \Pi_2\\ N:\alpha\end{array}}{(\lambda x.\,M)\,N:\beta} \quad \longrightarrow \quad \begin{array}{c}\vdots\ \Pi_2\\ N:\alpha\\ \vdots\ \Pi_1\\ M[x{:=}N]:\beta\end{array}$$

It is therefore possible to specify the detour elimination steps as simple rewriting rules between λ-terms. Following [14], these rules are called *detour-conversion* rules. We use "$\rightarrow_D$" to denote the corresponding one-step reduction relation between λ-terms and, following [1], we use "$\xrightarrow{+}_D$" and "$\twoheadrightarrow_D$" to denote, repectively, the transitive closure and the transitive, reflexive closure of "$\rightarrow_D$".

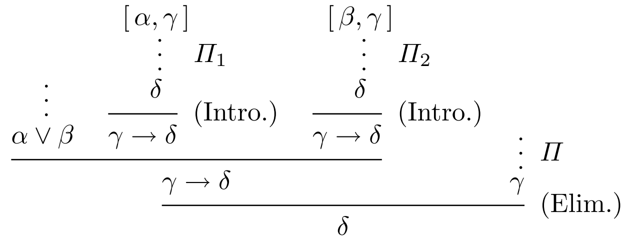
Definition 1. *(Detour-conversions)*

1. $(\lambda x.\, M)\, N \rightarrow_D M[x{:=}N]$
2. $\mathbf{p}_1\, \mathbf{p}(M, N) \rightarrow_D M$
3. $\mathbf{p}_2\, \mathbf{p}(M, N) \rightarrow_D N$
4. $\mathbf{D}_{x,y}(\mathbf{k}_1\, M, N, O) \rightarrow_D N[x{:=}M]$
5. $\mathbf{D}_{x,y}(\mathbf{k}_2\, M, N, O) \rightarrow_D O[y{:=}M]$ ■

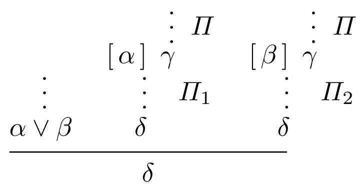
It must be clear that the Curry-Howard isomorphism allows one to reduce proof normalisation problems to λ-term normalisation problems. In particular, the strong normalisation of intuitionistic implicative logic corresponds to the strong normalisation of the simply typed λ-calculus [1] (which is, by the way, the only normalisation result we assume in this paper). There is, however, a slight difference that must be stressed. The grammar given in Section 2.2 defines untyped λ-terms, some of which do not correspond to natural deduction proofs. Consequently, the question whether the untyped λ-terms satisfy some normalisation property has no direct equivalent in the logical setting.

2.4 Permutation-conversion rules

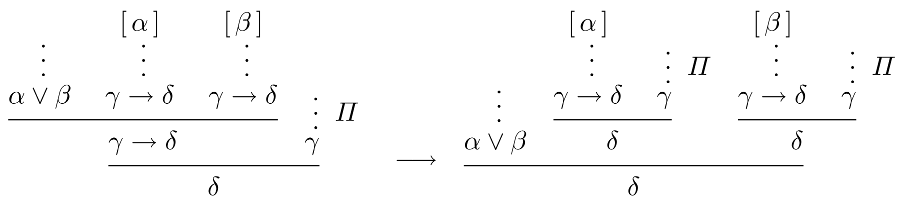
In the disjunction free fragment of **IPPL**, the normal proofs (i.e., the proofs without detour) satisfy the subformula property. This means that if Π is a normal proof of a formula α under a set of hypotheses Γ, then each formula occurring in Π is a subformula of a formula in $\Gamma \cup \{\alpha\}$. In the presence of disjunction, the detour-conversions of Definition 1 are no longer sufficient to guarantee the subformula property. Consider the following example:



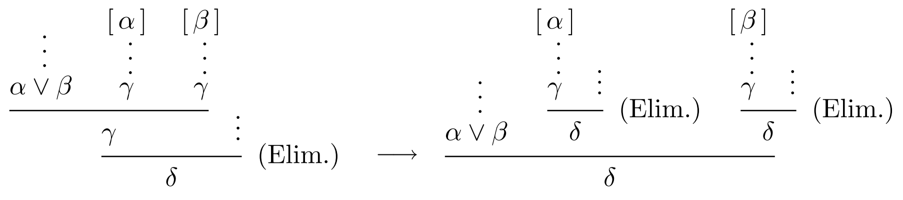
A priori there is no reason why γ and $\gamma \rightarrow \delta$ would be subformulas of δ or of any hypothesis from which δ is derived. This is due to the fact that there are introduction rules followed by an elimination rule. Indeed, one would like to reduce the above example as follows:



However, such a reduction is not possible by applying only the detour-conversion rules of Definition 1 because, in the above example, the elimination rule does not *immediately* follow the introduction rules. For this reason, some other conversion rules are needed, the so-called *permutation-conversion*. For instance, to reduce the above example, we need the following rule:



The above conversion, which concerns the implication elimination rule, obeys a general scheme:



All the permutation-conversions may be obtained, from the above scheme, by replacing Rule (Elim.) by the different elimination rules of **IPPL**. Of course, it is also possible to express these permutation-conversions as rewriting rules between λ-terms. This is achieved in the following definition.

Definition 2. *(Permutation-conversions)*

1. $\mathbf{D}_{x,y}(M, N, O)\, P \to_P \mathbf{D}_{x,y}(M, N\, P, O\, P)$
2. $\mathbf{p}_1\, \mathbf{D}_{x,y}(M, N, O) \to_P \mathbf{D}_{x,y}(M, \mathbf{p}_1\, N, \mathbf{p}_1\, O)$
3. $\mathbf{p}_2\, \mathbf{D}_{x,y}(M, N, O) \to_P \mathbf{D}_{x,y}(M, \mathbf{p}_2\, N, \mathbf{p}_2\, O)$
4. $\mathbf{D}_{u,v}(\mathbf{D}_{x,y}(M, N, O), P, Q) \to_P \mathbf{D}_{x,y}(M, \mathbf{D}_{u,v}(N, P, Q), \mathbf{D}_{u,v}(O, P, Q))$ ■

We are now in a position to state precisely the question addressed by the present paper: how can we give a modular proof of the strong normalisation of **IPPL** with respect to both the detour- and permutation-conversions? or, equivalently, how can we prove that the typed λ-terms of Section 2.2 satisfy the strong normalisation property with respect to the reduction relation induced by the union of the rewriting systems of definitions 1 and 2?

3 Strong Normalisation of permutation-conversions

In this section, we establish the strong normalisation of $\lambda^{\rightarrow\wedge\vee}$ with respect to permutation-conversions. The proof consists simply in assigning a norm to the λ-terms and then in proving that this norm (which we call the permutation degree) is decreasing under the reduction relation $\rightarrow_P$.

Definition 3. *(Permutation degree)*

1. $|x| = 1$
2. $|\lambda x.\, M| = |M|$
3. $|M\, N| = |M| + \#M \times |N|$
4. $|\mathbf{p}(M, N)| = |M| + |N|$
5. $|\mathbf{p}_1\, M| = |M| + \#M$
6. $|\mathbf{p}_2\, M| = |M| + \#M$
7. $|\mathbf{k}_1\, M| = |M|$
8. $|\mathbf{k}_2\, M| = |M|$
9. $|\mathbf{D}_{x,y}(M, N, O)| = |M| + \#M \times (|N| + |O|)$

where:

10. $\#x = 1$
11. $\#\lambda x.\, M = 1$
12. $\#M\, N = \#M$
13. $\#\mathbf{p}(M, N) = 1$
14. $\#\mathbf{p}_1\, M = \#M$
15. $\#\mathbf{p}_2\, M = \#M$
16. $\#\mathbf{k}_1\, M = 1$
17. $\#\mathbf{k}_2\, M = 1$
18. $\#\mathbf{D}_{x,y}(M, N, O) = 2 \times \#M \times (\#N + \#O)$ ■

Lemma 1. *Let M and N be two λ-terms of $\lambda^{\rightarrow\wedge\vee}$ such that $M \rightarrow_P N$. Then $\#M = \#N$.*

Proof. Let $C[\,]$ be any context, i.e., a λ-term with a hole. It is straightforward that $\#C[M] = \#C[N]$ whenever $\#M = \#N$. Hence it remains to show that # is invariant under each rewriting rule of Definition 2.

$$\begin{aligned}
&\#\mathbf{D}_{x,y}(M, N, O)\, P \\
&\qquad = \#\mathbf{D}_{x,y}(M, N, O) \\
&\qquad = 2 \times \#M \times (\#N + \#O) \\
&\qquad = 2 \times \#M \times (\#N\, P + \#O\, P) \\
&\qquad = \#\mathbf{D}_{x,y}(M, N\, P, O\, P) \\
&\#\mathbf{p}_i\, \mathbf{D}_{x,y}(M, N, O) \\
&\qquad = \#\mathbf{D}_{x,y}(M, N, O) \\
&\qquad = 2 \times \#M \times (\#N + \#O) \\
&\qquad = 2 \times \#M \times (\#\mathbf{p}_i\, N + \#\mathbf{p}_i\, O) \\
&\qquad = \#\mathbf{D}_{x,y}(M, \mathbf{p}_i\, N, \mathbf{p}_i\, O)
\end{aligned}$$

$$\begin{aligned}
&\#\mathbf{D}_{u,v}(\mathbf{D}_{x,y}(M,N,O),P,Q)\\
&\qquad = 2\times\#\mathbf{D}_{x,y}(M,N,O)\times(\#P+\#Q)\\
&\qquad = 4\times\#M\times(\#N+\#O)\times(\#P+\#Q)\\
&\qquad = 2\times\#M\times(2\times\#N\times(\#P+\#Q)+2\times\#O\times(\#P+\#Q))\\
&\qquad = 2\times\#M\times(\#\mathbf{D}_{u,v}(N,P,Q)+\#\mathbf{D}_{u,v}(O,P,Q))\\
&\qquad = \#\mathbf{D}_{x,y}(M,\mathbf{D}_{u,v}(N,P,Q),\mathbf{D}_{u,v}(O,P,Q))
\end{aligned}$$

□

Lemma 2. *Let M and N be two λ-terms of $\lambda^{\to\wedge\vee}$ such that $M \to_P N$. Then $|M| > |N|$.*

Proof. The proof is similar to that of the previous lemma. We show that the permutation degree is strictly decreasing under the rewriting rules of Definition 2.

$$\begin{aligned}
&|\mathbf{D}_{x,y}(M,N,O)\,P|\\
&\qquad = |\mathbf{D}_{x,y}(M,N,O)|+\#\mathbf{D}_{x,y}(M,N,O)\times|P|\\
&\qquad = |M|+\#M\times(|N|+|O|)+2\times\#M\times(\#N+\#O)\times|P|\\
&\qquad > |M|+\#M\times(|N|+|O|)+\#M\times(\#N+\#O)\times|P|\\
&\qquad = |M|+\#M\times(|N|+\#N\times|P|+|O|+\#O\times|P|)\\
&\qquad = |M|+\#M\times(|N\,P|+|O\,P|)\\
&\qquad = |\mathbf{D}_{x,y}(M,N\,P,O\,P)|\\
&|\mathbf{p}_i\,\mathbf{D}_{x,y}(M,N,O)|\\
&\qquad = |\mathbf{D}_{x,y}(M,N,O)|+\#\mathbf{D}_{x,y}(M,N,O)\\
&\qquad = |M|+\#M\times(|N|+|O|)+2\times\#M\times(\#N+\#O)\\
&\qquad > |M|+\#M\times(|N|+|O|)+\#M\times(\#N+\#O)\\
&\qquad = |M|+\#M\times(|N|+\#N+|O|+\#O)\\
&\qquad = |M|+\#M\times(|\mathbf{p}_i\,N|+|\mathbf{p}_i\,O|)\\
&\qquad = |\mathbf{D}_{x,y}(M,\mathbf{p}_i\,N,\mathbf{p}_i\,O)|\\
&|\mathbf{D}_{u,v}(\mathbf{D}_{x,y}(M,N,O),P,Q)|\\
&\qquad = |\mathbf{D}_{x,y}(M,N,O)|+\#\mathbf{D}_{x,y}(M,N,O)\times(|P|+|Q|)\\
&\qquad = |M|+\#M\times(|N|+|O|)+2\times\#M\times(\#N+\#O)\times(|P|+|Q|)\\
&\qquad > |M|+\#M\times(|N|+|O|)+\#M\times(\#N+\#O)\times(|P|+|Q|)\\
&\qquad = |M|+\#M\times(|N|+\#N\times(|P|+|Q|)+|O|+\#O\times(|P|+|Q|))\\
&\qquad = |M|+\#M\times(|\mathbf{D}_{u,v}(N,P,Q)|+|\mathbf{D}_{u,v}(O,P,Q)|)\\
&\qquad = |\mathbf{D}_{x,y}(M,\mathbf{D}_{u,v}(N,P,Q),\mathbf{D}_{u,v}(O,P,Q))|
\end{aligned}$$

□

We immediately obtain the expected strong normalisation result from the above lemma.

Proposition 1. *$\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to permutation-conversions.* □

Remark that this proposition also holds for the untyped terms. This fact confirms that the permutation-conversions do not have a real computational meaning. The fact that they are needed to obtain the subformula property may be seen as a defect of the syntax.

4 Negative translation and CPS-simulation

We now establish the strong normalisation of $\lambda^{\rightarrow\wedge\vee}$ with respect to detour-conversions. To this end we interpret **IPPL** into intuitionistic implicative logic by means of a negative translation. This corresponds to a translation of $\lambda^{\rightarrow\wedge\vee}$ into the simply typed λ-calculus. This translation must satisfy two requirements. On the one hand, it must provide a simulation of the detour-conversions. On the other hand it must be compatible with the permutation-conversions in a sense that will be explained. In order to satisfy the first requirement, the negative translation we use is a generalisation of the one used by Meyer and Wand in the implicative case [8], i.e., a generalisation of the translation induced by Plotkin's call-by-name CPS-translation [10].

Definition 4. *(Negative translation) The negative translation $\overline{\alpha}$ of any formula α is defined as:*

$$\overline{\alpha} = \sim\sim\alpha^{\circ}$$

where

$$\sim\alpha = \alpha \rightarrow o$$

for some distinguished atomic proposition o (that is not used elsewhere), and where:

1. $a^{\circ} = a$
2. $(\alpha \rightarrow \beta)^{\circ} = \overline{\alpha} \rightarrow \overline{\beta}$
3. $(\alpha \wedge \beta)^{\circ} = \sim(\overline{\alpha} \rightarrow \sim\overline{\beta})$
4. $(\alpha \vee \beta)^{\circ} = \sim\overline{\alpha} \rightarrow \sim\sim\overline{\beta}$ ■

Then we accomodate Plotkin's call-by-name simulation to the case of $\lambda^{\rightarrow\wedge\vee}$.

Definition 5. *(CPS-translation)*

1. $\overline{x} = \lambda k.\, x\, k$
2. $\overline{\lambda x.\, M} = \lambda k.\, k\, (\lambda x.\, \overline{M})$
3. $\overline{(M\, N)} = \lambda k.\, \overline{M}\, (\lambda m.\, m\, \overline{N}\, k)$
4. $\overline{\mathbf{p}(M, N)} = \lambda k.\, k\, (\lambda p.\, p\, \overline{M}\, \overline{N})$
5. $\overline{\mathbf{p}_1\, M} = \lambda k.\, \overline{M}\, (\lambda p.\, p\, (\lambda i.\, \lambda j.\, i\, k))$
6. $\overline{\mathbf{p}_2\, M} = \lambda k.\, \overline{M}\, (\lambda p.\, p\, (\lambda i.\, \lambda j.\, j\, k))$
7. $\overline{\mathbf{k}_1\, M} = \lambda k.\, k\, (\lambda i.\, \lambda j.\, i\, \overline{M})$
8. $\overline{\mathbf{k}_2\, M} = \lambda k.\, k\, (\lambda i.\, \lambda j.\, j\, \overline{M})$
9. $\overline{\mathbf{D}_{x,y}(M, N, O)} = \lambda k.\, \overline{M}\, (\lambda m.\, m\, (\lambda x.\, \overline{N}\, k)\, (\lambda y.\, \overline{O}\, k))$

where k, m, p, i and j are fresh variables. ■

We now prove that the translations of Definition 4 and 5 commute with the typing relation.

Proposition 2. *Let M be a λ-term of $\lambda^{\rightarrow\wedge\vee}$ typable with type α under a set of declarations Γ. Then $\overline{M}$ is a λ-term of the simply typed λ-calculus, typable with type $\overline{\alpha}$ under the set of declarations $\overline{\Gamma}$.*

Proof. *See Appendix A.* □

The translation of Definition 5 does not map normal forms to normal forms. This is due to the so-called administrative redexes that are introduced by the translation. The modified translation below circumvents this problem.

Definition 6. *(Modified CPS-translation) The modified CPS-translation $\overline{\overline{M}}$ of any λ-term M of $\lambda^{\to\wedge\vee}$ is defined as:*

$$\overline{\overline{M}} = \lambda k.\,(M : k)$$

where k is a fresh variable, and where the infix operator ":" obeys the following definition:

1. $x : K = x\,K$
2. $\lambda x.\,M : K = K\,(\lambda x.\,\overline{\overline{M}})$
3. $(M\,N) : K = M : \lambda m.\,m\,\overline{\overline{N}}\,K$
4. $\mathbf{p}(M, N) : K = K\,(\lambda p.\,p\,\overline{\overline{M}}\,\overline{\overline{N}})$
5. $\mathbf{p}_1\,M : K = M : \lambda p.\,p\,(\lambda i.\,\lambda j.\,i\,K)$
6. $\mathbf{p}_2\,M : K = M : \lambda p.\,p\,(\lambda i.\,\lambda j.\,j\,K)$
7. $\mathbf{k}_1\,M : K = K\,(\lambda i.\,\lambda j.\,i\,\overline{\overline{M}})$
8. $\mathbf{k}_2\,M : K = K\,(\lambda i.\,\lambda j.\,j\,\overline{\overline{M}})$
9. $\mathbf{D}_{x,y}(M, N, O) : K = M : \lambda m.\,m\,(\lambda x.\,(N : K))\,(\lambda y.\,(O : K))$

where m, p, i and j are fresh variables, and where, in Clause 9, x and y do not occur free in K. Remark that this last condition is not restrictive since it may always be satisfied by renaming. ■

As expected, the modified translation is a β-reduced form of the CPS-translation.

Lemma 3. *Let M and K be terms of $\lambda^{\to\wedge\vee}$. Then:*

1. $\overline{M} \twoheadrightarrow_\beta \overline{\overline{M}}$,
2. $\overline{M}\,K \twoheadrightarrow_\beta M : K$.

Proof. *We proceed by induction on the structure of M. Property 1 is the property of interest, while Property 2 is needed to make the induction work.* □

From this lemma, we get the analogue of Proposition 2 for the modified translation.

Proposition 3. *Let M be a λ-term of $\lambda^{\to\wedge\vee}$ typable with type α under a set of declarations Γ. Then $\overline{\overline{M}}$ is a λ-term of the simply typed λ-calculus, typable with type $\overline{\alpha}$ under the set of declarations $\overline{\Gamma}$.*

Proof. *The proposition follows from Proposition 2, Lemma 3, and the subject reduction property of the simply typed λ-calculus.*

The modified CPS-translation allows the detour-conversions of $\lambda^{\rightarrow\wedge\vee}$ to be simulated by β-reduction. This is established by the next lemmas.

Lemma 4. *Let M and N be λ-terms of $\lambda^{\rightarrow\wedge\vee}$ and K be a simple λ-term. Then:*

1. $(M : K)[x := \overline{\overline{N}}] \twoheadrightarrow_\beta (M[x := N]) : (K[x := \overline{\overline{N}}])$,
2. $\overline{\overline{M}}[x := \overline{\overline{N}}] \twoheadrightarrow_\beta \overline{\overline{M[x := N]}}$.

Proof. *Property 2 is a direct consequence of Property 1, which is established by a straightforward induction on the structure of M.* □

Lemma 5. *Let M and N be two λ-terms of $\lambda^{\rightarrow\wedge\vee}$ such that $M \rightarrow_D N$. Then:*

1. $M : K \overset{+}{\rightarrow}_\beta N : K$, *for any simple λ-term K,*
2. $\overline{\overline{M}} \overset{+}{\rightarrow}_\beta \overline{\overline{N}}$.

Proof. *Property 2 may be established as a direct consequence of Property 1. Then proving that $C[M] : K \overset{+}{\rightarrow}_\beta C[N] : K$ whenever $M : K \overset{+}{\rightarrow}_\beta N : K$ consists in a straightforward induction on the structure of the context $C[\,]$. Hence it remains to establish Property 1 for the five rewriting rules of Definition 1.*

$$\begin{aligned}
((\lambda x.\, M)\, N) : K &= (\lambda x.\, M) : \lambda m.\, m\, \overline{\overline{N}}\, K \\
&= (\lambda m.\, m\, \overline{\overline{N}}\, K)\, (\lambda x.\, \overline{\overline{M}}) \\
&\rightarrow_\beta (\lambda x.\, \overline{\overline{M}})\, \overline{\overline{N}}\, K \\
&\rightarrow_\beta \overline{\overline{M}}[x := \overline{\overline{N}}]\, K \\
&\twoheadrightarrow_\beta \overline{\overline{M[x := N]}}\, K \\
&\rightarrow_\beta (M[x := N]) : K \\
(\mathbf{p}_i\, \mathbf{p}(M, N)) : K &= \mathbf{p}(M_1, M_2) : \lambda p.\, p\, (\lambda j_1.\, \lambda j_2.\, j_i\, K) \\
&= (\lambda p.\, p\, (\lambda j_1.\, \lambda j_2.\, j_i\, K))\, (\lambda p.\, p\, \overline{\overline{M}}_1\, \overline{\overline{M}}_2) \\
&\rightarrow_\beta (\lambda p.\, p\, \overline{\overline{M}}_1\, \overline{\overline{M}}_2)\, (\lambda j_1.\, \lambda j_2.\, j_i\, K) \\
&\rightarrow_\beta (\lambda j_1.\, \lambda j_2.\, j_i\, K)\, \overline{\overline{M}}_1\, \overline{\overline{M}}_2 \\
&\overset{+}{\rightarrow}_\beta \overline{\overline{M}}_i\, K \\
&\rightarrow_\beta M_i : K
\end{aligned}$$

$$\begin{aligned}
&\mathbf{D}_{x_1,x_2}(\mathbf{k}_i\, M, N_1, N_2) : K \\
&\quad= (\mathbf{k}_i\, M) : \lambda m.\, m\, (\lambda x_1.\, (N_1 : K))\, (\lambda x_2.\, (N_2 : K)) \\
&\quad= (\lambda m.\, m\, (\lambda x_1.\, (N_1 : K))\, (\lambda x_2.\, (N_2 : K)))\, (\lambda j_1.\, \lambda j_2.\, j_i\, \overline{\overline{M}}) \\
&\quad\rightarrow_\beta (\lambda j_1.\, \lambda j_2.\, j_i\, \overline{\overline{M}})\, (\lambda x_1.\, (N_1 : K))\, (\lambda x_2.\, (N_2 : K)) \\
&\quad\overset{+}{\rightarrow}_\beta (\lambda x_i.\, (N_i : K))\, \overline{\overline{M}} \\
&\quad\rightarrow_\beta (N_i : K)[x_i := \overline{\overline{M}}] \\
&\quad\twoheadrightarrow_\beta (N_i[x_i := M]) : K
\end{aligned}$$

□

The above lemma allows any sequence of detour-conversion steps to be simulated by a longer sequence of β-reduction steps in the simply typed λ-calculus. Therefore, since the simply typed λ-calculus is strongly β-normalisable, we immediately obtain the following proposition.

Proposition 4. *$\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to detour-conversions.* □

5 Strong normalisation

In this section we prove that $\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to both detour- and permutation-conversion. This is not a direct consequence of Propositions 1 and 4 because detour-conversions can create permutation-redexes and, conversely, permutation conversions can create detour-redexes. Therefore we first show that the modified CPS-translation maps the relation of permutation-conversion to syntactic equality.

Lemma 6. *Let M and N be two λ-terms of $\lambda^{\to\wedge\vee}$ such that $M \to_P N$. Then:*

1. *$M : K = N : K$, for any simple λ-term K,*
2. *$\overline{\overline{M}} = \overline{\overline{N}}$.*

Proof. *Property 2 is a direct consequence of Property 1. To show that $C[M] : K = C[N] : K$ whenever $M : K = N : K$ consists in a routine induction on the structure of the context $C[\,]$. It remains to establish Property 1 for the four rewriting rules of Definition 2.*

$$
\begin{array}{l}
(\mathbf{D}_{x,y}(M,N,O)\,P) : K \\
\qquad = \mathbf{D}_{x,y}(M,N,O) : \lambda m.\, m\,\overline{\overline{P}}\,K \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(N : \lambda m.\, m\,\overline{\overline{P}}\,K))\,(\lambda y.\,(O : \lambda m.\, m\,\overline{\overline{P}}\,K)) \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(N\,P : K))\,(\lambda y.\,(O\,P : K)) \\
\qquad = \mathbf{D}_{x,y}(M, N\,P, O\,P) : K \\
(\mathbf{p}_i\,\mathbf{D}_{x,y}(M,N,O)) : K \\
\qquad = \mathbf{D}_{x,y}(M,N,O) : \lambda p.\, p\,(\lambda j_1.\,\lambda j_2.\, j_i\,K) \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(N : \lambda p.\, p\,(\lambda j_1.\,\lambda j_2.\, j_i\,K))) \\
\qquad\qquad\qquad (\lambda y.\,(O : \lambda p.\, p\,(\lambda j_1.\,\lambda j_2.\, j_i\,K))) \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(\mathbf{p}_i\,N : K))\,(\lambda y.\,(\mathbf{p}_i\,O : K)) \\
\qquad = \mathbf{D}_{x,y}(M, \mathbf{p}_i\,N, \mathbf{p}_i\,O) : K \\
\mathbf{D}_{u,v}(\mathbf{D}_{x,y}(M,N,O),P,Q) : K \\
\qquad = \mathbf{D}_{x,y}(M,N,O) : \lambda m.\, m\,(\lambda u.\,(P : K))\,(\lambda v.\,(Q : K)) \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(N : \lambda m.\, m\,(\lambda u.\,(P : K))\,(\lambda v.\,(Q : K)))) \\
\qquad\qquad\qquad (\lambda y.\,(O : \lambda m.\, m\,(\lambda u.\,(P : K))\,(\lambda v.\,(Q : K)))) \\
\qquad = M : \lambda m.\, m\,(\lambda x.\,(\mathbf{D}_{u,v}(N,P,Q) : K))\,(\lambda y.\,(\mathbf{D}_{u,v}(O,P,Q) : K)) \\
\qquad = \mathbf{D}_{x,y}(M, \mathbf{D}_{u,v}(N,P,Q), \mathbf{D}_{u,v}(O,P,Q)) : K
\end{array}
$$

□

We may now prove the main result.

Theorem 1. *$\lambda^{\to\wedge\vee}$ is strongly normalisable with respect to the reduction relation induced by the union of the detour- and permutation-conversions.*

Proof. Suppose it is not the case. Then there would exist an infinite sequence of detour- and permutation-conversion steps starting from a typable term (say, M) of $\lambda^{\rightarrow\wedge\vee}$. *If this infinite sequence contains infinitely many detour-conversion steps, there must exist, by Lemmas 5 and 6 an infinite sequence of* β*-reduction steps starting from* $\overline{\overline{\overline{M}}}$. *But this, by Proposition 3, would contradict the strong normalisation of the simply typed* λ*-calculus. Hence the infinite sequence may contain only a finite number of detour conversion steps. But then, it would contain an infinite sequence of consecutive permutation-conversion steps, which contradicts Proposition 1.* □

References

1. H.P. Barendregt. *The lambda calculus, its syntax and semantics.* North-Holland, revised edition, 1984.
2. H.B. Curry and R. Feys. *Combinatory Logic, Vol. I.* North-Holland, 1958.
3. Ph. de Groote. The conservation theorem revisited. In M. Bezem and J.F. Groote, editors, *Proceedings of the International Conference on Typed Lambda Calculi and Applications*, pages 163–178. Lecture Notes in Computer Science, 664, Springer Verlag, 1993.
4. Ph. de Groote. A simple calculus of exception handling. In M. Dezani and G. Plotkin, editors, *Second International Conference on Typed Lambda Calculi and Applications, TLCA'95*, volume 902 of *Lecture Notes in Computer Science*, pages 201–215. Springer Verlag, 1995.
5. G. Gentzen. *Recherches sur la déduction logique (Untersuchungen über das logische schliessen).* Presses Universitaires de France, 1955. Traduction et commentaire par R. Feys et J. Ladrière.
6. J.-Y. Girard. *Proof Theory and Logical Complexity.* Bibliopolis, 1987.
7. W.A. Howard. The formulae-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *to H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, 1980.
8. A. Meyer and M. Wand. Continuation semantics in typed lambda-calculi (summary). In R. Parikh, editor, *Logics of Programs*, pages 219–224. Lecture Notes in Computer Science, 193, Springer Verlag, 1985.
9. G. Mints. Private communication, 1997.
10. G. D. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1:125–159, 1975.
11. D. Prawitz. *Natural Deduction, A Proof-Theoretical Study.* Almqvist & Wiksell, Stockholm, 1965.
12. D. Prawitz. Ideas and results in proof-theory. In J.E. Fenstad, editor, *Proceedings of the Second Scandinavian Logic Symposium*, pages 237–309. North-Holland, 1971.
13. A. Troelstra and D. van Dalen. *Constructivism in Mathematics*, volume I. North-Holland, 1988.
14. A. Troelstra and D. van Dalen. *Constructivism in Mathematics*, volume II. North-Holland, 1988.

A Proof of Proposition 2

Variable

$$\dfrac{\dfrac{x : {\sim}{\sim}a \quad k : {\sim}a}{x\,k : o}}{\lambda k.\, x\,k : {\sim}{\sim}a}$$

Abstraction

$$\dfrac{\dfrac{k : {\sim}(\overline{\alpha} \to \overline{\beta}) \qquad \dfrac{\begin{matrix}[\,x : \overline{\alpha}\,] \\ \vdots \\ \overline{M} : \overline{\beta}\end{matrix}}{\lambda x.\, \overline{M} : \overline{\alpha} \to \overline{\beta}}}{k\,(\lambda x.\,\overline{M}) : o}}{\lambda k.\, k\,(\lambda x.\,\overline{M}) : {\sim}{\sim}(\overline{\alpha} \to \overline{\beta})}$$

Application

$$\dfrac{\dfrac{\overline{M} : {\sim}{\sim}(\overline{\alpha} \to \overline{\beta}) \qquad \dfrac{\dfrac{\dfrac{m : \overline{\alpha} \to \overline{\beta} \quad \overline{N} : \overline{\alpha}}{m\,\overline{N} : \overline{\beta}} \qquad k : {\sim}\beta^{\circ}}{m\,\overline{N}\,k : o}}{\lambda m.\, m\,\overline{N}\,k : {\sim}(\overline{\alpha} \to \overline{\beta})}}{\overline{M}\,(\lambda m.\, m\,\overline{N}\,k) : o}}{\lambda k.\, \overline{M}\,(\lambda m.\, m\,\overline{N}\,k) : \overline{\beta}}$$

Pairing

$$\dfrac{\dfrac{k : {\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta}) \qquad \dfrac{\dfrac{\dfrac{p : \overline{\alpha} \to {\sim}\overline{\beta} \quad \overline{M} : \overline{\alpha}}{p\,\overline{M} : {\sim}\overline{\beta}} \qquad \overline{N} : \overline{\beta}}{p\,\overline{M}\,\overline{N} : o}}{\lambda p.\, p\,\overline{M}\,\overline{N} : {\sim}(\overline{\alpha} \to {\sim}\overline{\beta})}}{k\,(\lambda p.\, p\,\overline{M}\,\overline{N}) : o}}{\lambda k.\, k\,(\lambda p.\, p\,\overline{M}\,\overline{N}) : {\sim}{\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta})}$$

Left projection

$$\dfrac{\dfrac{\overline{M} : {\sim}{\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta}) \qquad \dfrac{\dfrac{p : {\sim}(\overline{\alpha} \to {\sim}\overline{\beta}) \quad \dfrac{\dfrac{\dfrac{i : \overline{\alpha} \quad k : {\sim}\alpha^{\circ}}{i\,k : o}}{\lambda j.\, i\,k : {\sim}\overline{\beta}}}{\lambda i.\, \lambda j.\, i\,k : \overline{\alpha} \to {\sim}\overline{\beta}}}{p\,(\lambda i.\, \lambda j.\, i\,k) : o}}{\lambda p.\, p\,(\lambda i.\, \lambda j.\, i\,k) : {\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta})}}{\overline{M}\,(\lambda p.\, p\,(\lambda i.\, \lambda j.\, i\,k)) : o}}{\lambda k.\, \overline{M}\,(\lambda p.\, p\,(\lambda i.\, \lambda j.\, i\,k)) : \overline{\alpha}}$$

Right projection

$$\dfrac{\overline{M} : {\sim}{\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta}) \qquad \dfrac{\dfrac{p : {\sim}(\overline{\alpha} \to {\sim}\overline{\beta}) \quad \dfrac{\dfrac{\dfrac{j : \overline{\beta} \quad k : {\sim}\beta^{\circ}}{j\,k : o}}{\lambda j.\, j\,k : {\sim}\overline{\beta}}}{\lambda i.\, \lambda j.\, j\,k : \overline{\alpha} \to {\sim}\overline{\beta}}}{p\,(\lambda i.\, \lambda j.\, j\,k) : o}}{\lambda p.\, p\,(\lambda i.\, \lambda j.\, j\,k) : {\sim}{\sim}(\overline{\alpha} \to {\sim}\overline{\beta})}}{\dfrac{\overline{M}\,(\lambda p.\, p\,(\lambda i.\, \lambda j.\, j\,k)) : o}{\lambda k.\, \overline{M}\,(\lambda p.\, p\,(\lambda i.\, \lambda j.\, j\,k)) : \overline{\beta}}}$$

Left injection

$$\dfrac{\dfrac{k : {\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta}) \qquad \dfrac{\dfrac{\dfrac{i : {\sim}\overline{\alpha} \quad \overline{M} : \overline{\alpha}}{i\,\overline{M} : o}}{\lambda j.\, i\,\overline{M} : {\sim}{\sim}\overline{\beta}}}{\lambda i.\, \lambda j.\, i\,\overline{M} : {\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta}}}{k\,(\lambda i.\, \lambda j.\, i\,\overline{M}) : o}}{\lambda k.\, k\,(\lambda i.\, \lambda j.\, i\,\overline{M}) : {\sim}{\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta})}$$

Right injection

$$\dfrac{\dfrac{k : {\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta}) \qquad \dfrac{\dfrac{\dfrac{j : {\sim}\overline{\beta} \quad \overline{M} : \overline{\beta}}{j\,\overline{M} : o}}{\lambda j.\, j\,\overline{M} : {\sim}{\sim}\overline{\beta}}}{\lambda i.\, \lambda j.\, j\,\overline{M} : {\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta}}}{k\,(\lambda i.\, \lambda j.\, j\,\overline{M}) : o}}{\lambda k.\, k\,(\lambda i.\, \lambda j.\, j\,\overline{M}) : {\sim}{\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta})}$$

Case analysis

$$\dfrac{\overline{M} : {\sim}{\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta}) \qquad \dfrac{\dfrac{\dfrac{m : {\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta} \quad \dfrac{\dfrac{\begin{array}{c}[\,x : \overline{\alpha}\,] \\ \vdots \\ \overline{N} : \overline{\gamma}\end{array} \quad k : {\sim}\gamma^{\circ}}{\overline{N}\,k : o}}{\lambda x.\, \overline{N}\,k : {\sim}\overline{\alpha}}}{m\,(\lambda x.\, \overline{N}\,k) : {\sim}{\sim}\overline{\beta}} \qquad \dfrac{\dfrac{\begin{array}{c}[\,y : \overline{\beta}\,] \\ \vdots \\ \overline{O} : \overline{\gamma}\end{array} \quad k : {\sim}\gamma^{\circ}}{\overline{O}\,k : o}}{\lambda y.\, \overline{O}\,k : {\sim}\overline{\beta}}}{m\,(\lambda x.\, \overline{N}\,k)\,(\lambda y.\, \overline{O}\,k) : o}}{\lambda m.\, m\,(\lambda x.\, \overline{N}\,k)\,(\lambda y.\, \overline{O}\,k) : {\sim}({\sim}\overline{\alpha} \to {\sim}{\sim}\overline{\beta})}}{\dfrac{\overline{M}\,(\lambda m.\, m\,(\lambda x.\, \overline{N}\,k)\,(\lambda y.\, \overline{O}\,k)) : o}{\lambda k.\, \overline{M}\,(\lambda m.\, m\,(\lambda x.\, \overline{N}\,k)\,(\lambda y.\, \overline{O}\,k)) : \overline{\gamma}}}$$