TERMS FOR NATURAL DEDUCTION, SEQUENT CALCULUS AND CUT ELIMINATION IN CLASSICAL LOGIC

SILVIA GHILEZAN

Faculty of Engineering, University of Novi Sad, Novi Sad, Serbia
e-mail address: gsilvia@uns.ns.ac.yu

Abstract. This paper revisits the results of Barendregt and Ghilezan [3] and generalizes them for classical logic. Instead of λ-calculus, we use here $\lambda\mu$-calculus as the basic term calculus. We consider two extensionally equivalent type assignment systems for $\lambda\mu$-calculus, one corresponding to classical natural deduction, and the other to classical sequent calculus. Their relations and normalisation properties are investigated. As a consequence a short proof of Cut elimination theorem is obtained.

Introduction

The Curry-Howard correspondence provides a fundamental connection between logic and computation. Under the traditional Curry-Howard correspondence formulae provable in intuitionistic logic coincide with types inhabited in simply typed λ-calculus. This was observed already by Curry, first formulated by Howard [12], used intensively by de Brujin in the Authomath project and by Lambek in category theory. Parigot [14] extended this correspondence to classical logic based on natural deduction and $\lambda\mu$-calculus. Griffin [9] embodied a Curry-Howard correspondence for classical logic, by observing that classical tautologies provide typings for certain control operators. This initiated an active line of research both in natural deduction and sequent calculus formulations of classical logic. For an overview see Sorensen and Urzyczyn [17].

The $\lambda\mu_v$-calculus, a call-by-value variant of $\lambda\mu$, was proposed by Ong and Stewart [13]. The Symmetric lambda calculus of Barbanera and Berardi [2] is a calculus designed with the goal of extracting constructive content from classical proofs (Peano arithmetic). Curien and Herbelin [4] defined the system $\overline{\lambda}\mu\widetilde{\mu}$, which represents derivations in classical logic based on sequent calculus and reductions reflect cut-elimination. Cut-elimination in classical logic is known to be non-confluent, correspondingly $\overline{\lambda}\mu\widetilde{\mu}$-calculus is not confluent. However, restrictions to call-by-name or call-by-value discipline provide confluence. Urban and Bierman [18, 19] designed a calculus whose derivations correspond exactly to cut elimination. Wadler's Dual calculus, [20, 21], is a system closely related to $\overline{\lambda}\mu\widetilde{\mu}$.

2000 ACM Subject Classification: F.4.1 [Mathematical logic]: Computational logic, Lambda calculus and related systems, Proof theory.

Key words and phrases: classical logic, natural deduction, sequent calculus, normalisation, cut elimination.

This paper revisits the results of Barendregt and Ghilezan [3] and generalizes them to classical logic, as suggested in the conclusion. We use here $\lambda\mu$-calculus as the basic term calculus instead of λ calculus. We consider two extensionally equivalent type assignment systems for $\lambda\mu$-calculus, one corresponding to classical natural deduction ($\lambda\mu N$), and the other to classical sequent calculus ($\lambda\mu L$). Moreover, a cut-free variant of $\lambda\mu L$ is introduced ($\lambda\mu L^{\mathtt{cf}}$). The relations between these three systems and their normalisation properties are investigated. As a consequence a short proof of Cut elimination theorem for classical logic (Hauptsatz) is obtained.

The paper is organised as follows. Section 1 gives an overview of three classical logical systems. In Section 2 the corresponding three term calculi are considered. Their relations are investigated in Section 3. Final remarks and some future work are discussed in Section 4.

1. The systems of classical logic NK, LK, $LK^{\mathtt{cf}}$

We consider the natural deduction and sequent calculus formulation of the implicational fragment of classical logic. For further reading in this field we refer the reader to Prawitz[16], Ariola and Herbelin [1] and to the original work of Gentzen [8].

Definition 1.1. The set `form` of formulae (of minimal implicational propositional logic) is defined by the following abstract syntax.

$$\begin{array}{|lcl|} \hline \mathtt{form} & = & \mathtt{atom} \mid \mathtt{form} \to \mathtt{form} \\ \mathtt{atom} & = & \mathtt{p} \mid \mathtt{atom}' \\ \hline \end{array}$$

We write $p, q, r, \ldots$ for arbitrary atoms and $A, B, C, \ldots$ for arbitrary formulae. Sets of formulae are denoted by $\Gamma, \Delta, \ldots$. The set Γ, A stands for $\Gamma \cup \{A\}$ and $\Gamma \backslash A$ stands for $\Gamma \backslash \{A\}$.

Definition 1.2. A statement A is *derivable* from the set Γ in the system NK, notation $\Gamma \vdash_{NK} A$, if $\Gamma \vdash A$ can be generated by the axiom and rules given in Figure 1.

$$\frac{}{\Gamma, A \vdash A, \Delta}\ (\text{axiom})$$

$$\frac{\Gamma \vdash A \to B, \Delta \quad \Gamma \vdash A, \Delta}{\Gamma \vdash B, \Delta}\ (\to \text{elim}) \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ (\to \text{intro})$$

Figure 1: NK- classical natural deduction

Definition 1.3. A statement A is *derivable* from assumptions Γ in the system LK, notation $\Gamma \vdash_{LK} A$, if $\Gamma \vdash A$ can be generated by the axiom and rules given in Figure 2.

Definition 1.4. The system $LK^{\mathtt{cf}}$, given in Figure 3, is obtained from the system LK by omitting the rule (cut). Derivability in this system is denoted by $\Gamma \vdash_{LK^{\mathtt{cf}}} A$.

Lemma 1.5 (Weakening lemma). *Suppose* $\Gamma \subseteq \Gamma'$ *and* $\Delta \subseteq \Delta'$. *Then, in all logical systems*

$$\Gamma \vdash A, \Delta \implies \Gamma' \vdash A, \Delta'.$$

Proof. By an easy induction on derivations. $\square$

$$\frac{}{\Gamma, A \vdash A, \Delta}\ (\text{axiom})$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\ (\to \text{left}) \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ (\to \text{right})$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}\ (\text{cut})$$

Figure 2: LK- classical sequent calculus

$$\frac{}{\Gamma, A \vdash A, \Delta}\ (\text{axiom})$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\ (\to \text{left}) \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ (\to \text{right})$$

Figure 3: $LK^{\mathtt{cf}}$- classical sequent calculus without cut

Proposition 1.6. *For all* Γ *and* A *we have*

$$\Gamma \vdash_{NK} A, \Delta \iff \Gamma \vdash_{LK} A, \Delta.$$

Proof. ($\Longrightarrow$) By induction on derivations in NK. For the rule ($\to$ elim) we need the rule (cut). By the induction hypothesis $\Gamma \vdash_{LK} A \to B, \Delta$ and $\Gamma \vdash_{LK} A, \Delta$. Then by Lemma 1.5 $\Gamma \vdash_{LK} A, B, \Delta$ and $\Gamma \vdash_{LK} A \to B, B, \Delta$.

$$\frac{\Gamma \vdash_{LK} A \to B, B, \Delta \qquad \dfrac{\Gamma \vdash_{LK} A, B, \Delta \quad \Gamma, B \vdash_{LK} B, \Delta}{\Gamma, A \to B \vdash_{LK} B, \Delta}\ (\to \text{left})}{\Gamma \vdash_{LK} B, \Delta}\ (\text{cut})$$

($\Longleftarrow$) By induction on derivations in LK. The rule ($\to$ left) is treated as follows. By the induction hypothesis $\Gamma \vdash_{NK} A, \Delta$ and $\Gamma, B \vdash_{NK} C, \Delta$. On the one hand from the first premise by the Weakening lemma 1.5 we get $\Gamma, A \to B \vdash_{NK} A, \Delta$. By axiom $\Gamma, A \to B \vdash_{NK} A \to B, \Delta$, therefore by ($\to$ elim), $\Gamma, A \to B \vdash_{NK} B, \Delta$. On the other hand from the second premise using ($\to$ intro) we obtain $\Gamma \vdash_{NK} B \to C, \Delta$ and then by Weakening lemma 1.5, $\Gamma, A \to B \vdash_{NK} B \to C, \Delta$. Then

$$\frac{\Gamma, A \to B \vdash_{NK} B \to C, \Delta \quad \Gamma, A \to B \vdash_{NK} B, \Delta}{\Gamma, A \to B \vdash_{NK} C, \Delta}\ (\to \text{elim})$$

Admissibility of the (cut) rule in NK is treated as follows.

$$\frac{\Gamma \vdash_{NK} A, \Delta \qquad \dfrac{\Gamma, A \vdash_{NK} B, \Delta}{\Gamma \vdash_{NK} A \to B, \Delta}\ (\to \text{intr})}{\Gamma \vdash_{NK} B, \Delta}\ (\to \text{elim})$$

□

2. The type assignment systems $\lambda\mu N$, $\lambda\mu L$ and $\lambda\mu L^{\mathtt{cf}}$

We use $\lambda\mu$-calculus, introduced by Parigot [14, 15], as the basic term calculus. We consider two extensionally equivalent type assignment systems for $\lambda\mu$-calculus, one corresponding to classical natural deduction ($\lambda\mu N$), and the other to classical sequent calculus ($\lambda\mu L$). Moreover, a cut-free variant of $\lambda\mu L$ will be introduced ($\lambda\mu L^{\mathtt{cf}}$).

Definition 2.1. The set `term` of type–free $\lambda\mu$-terms is defined in Figure 4.

$$\begin{array}{rcrlllll} \mathtt{term} & = & \mathtt{var}\,| & \mathtt{term}\,\mathtt{term}\,| & \lambda\mathtt{var}.\mathtt{term}\,| & \mu\mathtt{mvar}.\mathtt{term}\,| & [\mathtt{mvar}]\mathtt{term} \\ \mathtt{var} & = & \mathtt{x}\,| & \mathtt{var}' \\ \mathtt{mvar} & = & \alpha\,| & \mathtt{mvar}' \end{array}$$

Figure 4: $\lambda\mu$ terms

We write $x, y, z, \ldots$ for arbitrary variables in terms, $\alpha, \beta, \gamma, \ldots$ for arbitrary co-variables in terms and $M, N, P, Q, R, \ldots$ for arbitrary terms. Equality of terms (up to renaming of bound variables) is denoted by $\equiv$.

The reduction relation of the $\lambda\mu$-calculus is induced by three different notions of reduction: usual notion of reduction β, structural reduction μ_{app}, and renaming reduction μ_{var}: The sets of free variables and co-variables of a term, denoted by $FV(-)$ and $FV_\mu(-)$, are defined as usual.

$$\begin{array}{llll} \beta: & (\lambda x.M)\,N & \to & M[x := N] \\ \mu_{app}: & (\mu\alpha.M)\,N & \to & \mu\beta.M[[\alpha]P := [\beta]PN] \quad \beta \text{ fresh} \\ \mu_{var}: & [\beta]\mu\alpha.c & \to & c[\alpha := \beta] \end{array}$$

Figure 5: Reductions

We write $\to$ for one-step reduction relation induced by the three notions of reduction given above. We write $\twoheadrightarrow$ for the reflexive, transitive closure of the one-step reduction relation. A $\beta\mu$ normal form ($\beta\mu$-nf) is a term that cannot be reduced. If $P \twoheadrightarrow Q$ and Q is a $\beta\mu$-nf, then Q is called the $\beta\mu$-nf of P (one can show it is unique). A collection $\mathbf{A}$ of terms is said to be *strongly normalising* if for no $P \in \mathbf{A}$ there is an infinite reduction path

$$P \twoheadrightarrow P_1 \twoheadrightarrow P_2 \ldots .$$

Definition 2.2.

(i) A *type assignment* is an expression of the form $P : A$, where P is a term and A is a formula.

(ii) A *variable declaration* is a type assignment of the form $x : A$. A *co-variable declaration* is a type assignment of the form $\alpha : A$.

(iii) A *variable context* $\Gamma_{\vec{x}} = \{x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n\}$ is a set of variable declarations such that for every variable x_i there is at most one declaration $x_i : A_i$ in $\Gamma_{\vec{x}}$. A *co-variable context* $\Delta_{\vec{\alpha}} = \{\alpha_1 : B_1, \alpha_2 : B_2, \ldots, \alpha_k : B_k\}$ is a set of co-variable declarations such that for every variable α_l there is at most one declaration $\alpha_l : B_l$ in $\Delta_{\vec{\alpha}}$.

Notation.

Let $\Gamma_{\vec{x}} = \{x_1 : A_1, \ldots, x_n : A_n\}$ be a variable context. We then say that

$$\Gamma = \{A_1, \ldots, A_n\}, \ \ \vec{x} = \{x_1, \ldots, x_n\} \ \text{ and } \ \Lambda^{\circ}(\vec{x}) = \{P \in \texttt{term} \mid FV(P) \subseteq \vec{x}\},$$

where $FV(P)$ is the set of free variables of P.

Similarly, let $\Delta_{\vec{\alpha}} = \{\alpha_1 : B_1, \ldots, \alpha_n : B_n\}$ be a co-variable context. We then say that

$$\Delta = \{B_1, \ldots, B_n\}, \ \ \vec{\alpha} = \{\alpha_1, \ldots, \alpha_n\} \ \text{ and } \ \Lambda^{\circ}_{\mu}(\vec{\alpha}) = \{P \in \texttt{term} \mid FV_{\mu}(P) \subseteq \vec{\alpha}\},$$

where $FV_{\mu}(P)$ is the set of free co-variables of P.

Definition 2.3. A type assignment $P : A$ is *derivable* from the contexts $\Gamma_{\vec{x}}$ and $\Delta_{\vec{\alpha}}$ in the system $\lambda\mu N$ (also known as simply typed $\lambda\mu$-calculus), notation

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}}$$

if $\Gamma_{\vec{x}} \vdash P : A, \Delta_{\vec{\alpha}}$ can be generated by the following axiom and rules given in Figure 6.

$$\frac{}{\Gamma_{\vec{x}}, y : A \vdash y : A, \Delta_{\vec{\alpha}}} \text{ (axiom)}$$

$$\frac{\Gamma_{\vec{x}} \vdash M : A \to B, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}} \vdash N : A, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash MN : B, \Delta_{\vec{\alpha}}} \text{ } (\to \text{elim}) \qquad \frac{\Gamma_{\vec{x}}, y : A \vdash M : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash \lambda y.M : A \to B, \Delta_{\vec{\alpha}}} \text{ } (\to \text{intro})$$

$$\frac{\Gamma_{\vec{x}} \vdash M : A, \Delta_{\vec{\alpha}}, \beta : A, \alpha : B}{\Gamma_{\vec{x}} \vdash \mu\alpha.[\beta]M : B, \Delta_{\vec{\alpha}}, \beta : A} \text{ } (\mu)$$

Figure 6: $\lambda\mu N$-calculus

Definition 2.4. A type assignment $P : A$ is *derivable* from the contexts $\Gamma_{\vec{x}}$ and $\Delta_{\vec{\alpha}}$ in the system $\lambda\mu L$, notation

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu L} P : A, \Delta_{\vec{\alpha}}$$

if $\Gamma_{\vec{x}} \vdash P : A, \Delta_{\vec{\alpha}}$ can be generated by the following axiom and rules given in Figure 7.

$$\frac{}{\Gamma_{\vec{x}}, y : A \vdash y : A, \Delta_{\vec{\alpha}}} \text{ (axiom)}$$

$$\frac{\Gamma_{\vec{x}} \vdash N : A, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}}, x : B \vdash M : C, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}}, y : A \to B \vdash M[x := yN] : C, \Delta_{\vec{\alpha}}} \text{ } (\to \text{left}) \qquad \frac{\Gamma_{\vec{x}}, y : A \vdash M : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash \lambda y.M : A \to B, \Delta_{\vec{\alpha}}} \text{ } (\to \text{right})$$

$$\frac{\Gamma_{\vec{x}} \vdash M : A, \Delta_{\vec{\alpha}}, \beta : A, \alpha : B}{\Gamma_{\vec{x}} \vdash \mu\alpha.[\beta]M : B, \Delta_{\vec{\alpha}}, \beta : A} \text{ } (\mu)$$

$$\frac{\Gamma_{\vec{x}} \vdash N : B, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}}, x : B \vdash M : A, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash M[x := N] : A, \Delta_{\vec{\alpha}}} \text{ (cut)}$$

Figure 7: $\lambda\mu L$-calculus

$$\frac{}{\Gamma_{\vec{x}}, y : A \vdash y : A, \Delta_{\vec{\alpha}}} \text{(axiom)}$$

$$\frac{\Gamma_{\vec{x}} \vdash N : A, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}}, x : B \vdash M : C, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}}, y : A \to B \vdash M[x := yN] : C, \Delta_{\vec{\alpha}}} (\to \text{left}) \qquad \frac{\Gamma_{\vec{x}}, y : A \vdash M : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash \lambda y.M : A \to B, \Delta_{\vec{\alpha}}} (\to \text{right})$$

$$\frac{\Gamma_{\vec{x}} \vdash M : A, \Delta_{\vec{\alpha}}, \beta : A, \alpha : B}{\Gamma_{\vec{x}} \vdash \mu\alpha.[\beta]M : B, \Delta_{\vec{\alpha}}, \beta : A} (\mu)$$

Figure 8: $\lambda\mu L^{\mathtt{cf}}$-calculus

Definition 2.5. The system $\lambda\mu L^{\mathtt{cf}}$, given in Figure 8, is obtained from the system $\lambda\mu L$ by omitting the rule (cut).

The following result is the propositions-as-types interpretation of classical logic given by Parigot [14]. This is an extension of the well-known propositions-as-types interpretation of intuitionistic logic that was observed by Curry, Howard, de Bruijn and Lambek.

Proposition 2.6 (Propositions–as–types interpretation)**.** *Let SK be one of the logical systems NK LK or $LK^{\mathtt{cf}}$ and let $\lambda\mu S$ be the corresponding type assignment system. Then*

$$\Gamma \vdash_{SK} A, \Delta \Longleftrightarrow \exists \vec{x}.\ \exists \vec{\alpha}.\ \exists P \in \Lambda^{\circ}(\vec{x}) \cup \Lambda^{\circ}_{\mu}(\vec{\alpha}).\ \Gamma_{\vec{x}} \vdash_{\lambda\mu S} P : A, \Delta_{\vec{\alpha}}.$$

Proof. ($\Rightarrow$) By an easy induction on derivations, just observing how the right lambda term can be constructed. ($\Leftarrow$) By omitting the terms and the (μ) rule. □

Since $\lambda\mu N$ is the first order restriction of Parigot's $\lambda\mu$-calculus, we know the following results. From Proposition 3.1 it follows that these results also hold for $\lambda\mu L$.

Proposition 2.7. (i) *(Normalisation theorem for $\lambda\mu N$)*

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}} \Longrightarrow P \textit{ has a } \beta\mu\textit{-nf } P^{nf}.$$

(ii) *(Subject reduction theorem for $\lambda\mu N$)*

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}} \textit{ and } P \twoheadrightarrow P' \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} P' : A, \Delta_{\vec{\alpha}}.$$

(iii) *(Generation lemma for $\lambda\mu N$) Type assignment for terms of a certain syntactic form is caused in the obvious way.*

(1) $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} x : A, \Delta_{\vec{\alpha}} \Longrightarrow (x : A) \in \Gamma_{\vec{x}}$.

(2) $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} PQ : B, \Delta_{\vec{\alpha}} \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : (A \to B), \Delta_{\vec{\alpha}}$ *and* $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} Q : A$, *for some type A.*

(3) $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} \lambda x.P : C, \Delta_{\vec{\alpha}} \Longrightarrow \Gamma_{\vec{x}}, x : A \vdash_{\lambda\mu N} P : B, \Delta_{\vec{\alpha}}$ *and* $C \equiv A \to B$, *for some types A, B.*

(4) $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} \mu\alpha[\beta].P : A, \Delta_{\vec{\alpha}}, \beta : B \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : B, \Delta_{\vec{\alpha}}, \beta : B, \alpha : A$.

Proof. (i) Normalisation, even strong normalisation, for terms typeable in $\lambda\mu N$ was proved by Parigot [15]. (ii) See Parigot [14]. (iii) Generation lemma is straightforward since $\lambda\mu N$ is syntax directed. □

3. Relating $\lambda\mu N$, $\lambda\mu L$ and $\lambda\mu L^{\mathrm{CF}}$

Now the proof of the equivalence between systems NK and LK will be 'lifted' to that of $\lambda\mu N$ and $\lambda\mu L$.

Proposition 3.1. *For all $\Gamma_{\vec{x}}$, $\Delta_{\vec{\alpha}}$ and A we have*

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}} \implies \Gamma_{\vec{x}} \vdash_{\lambda\mu L} P : A, \Delta_{\vec{\alpha}}.$$

Proof. By induction on derivations in $\lambda\mu N$. The rule ($\rightarrow$ elim), Modus ponens, is treated as follows.

$$\dfrac{\Gamma_{\vec{x}} \vdash_{\lambda\mu L} P : A \rightarrow B, \Delta_{\vec{\alpha}} \qquad \dfrac{\Gamma_{\vec{x}} \vdash_{\lambda\mu L} Q : A, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}}, x : B \vdash_{\lambda\mu L} x : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu L} yQ : B, \Delta_{\vec{\alpha}}}\ (\rightarrow \text{left})}{\Gamma_{\vec{x}} \vdash_{\lambda\mu L} PQ : B, \Delta_{\vec{\alpha}}}\ (\text{cut})$$

□

Proposition 3.2.

(i) $\Gamma_{\vec{x}} \vdash_{\lambda\mu L} M : C, \Delta_{\vec{\alpha}} \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} M' : C, \Delta_{\vec{\alpha}}$, *for some* $M' \twoheadrightarrow M$.

(ii) $\Gamma_{\vec{x}} \vdash_{\lambda\mu L} M : C, \Delta_{\vec{\alpha}} \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} M : C, \Delta_{\vec{\alpha}}$.

Proof. (i) By induction on derivations in $\lambda\mu L$. The rule ($\rightarrow$ left) is treated as follows. By the induction hypothesis $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} Q : A, \Delta_{\vec{\alpha}}$ and $\Gamma_{\vec{x}}, x : B \vdash_{\lambda\mu N} P : C, \Delta_{\vec{\alpha}}$. On the one hand from the first premise by the context Weakening lemma 1.5 we get $\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} Q : A, \Delta_{\vec{\alpha}}$. By axiom $\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} y : A \rightarrow B, \Delta_{\vec{\alpha}}$, therefore by ($\rightarrow$ elim), $\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} yQ : B, \Delta_{\vec{\alpha}}$. On the other hand from the second premise ($\rightarrow$ intro), $\Gamma_{\vec{x}} \vdash_{\lambda\mu N} \lambda x.P : B \rightarrow C, \Delta$ and then by context Weakening lemma 1.5, $\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} \lambda x.P : B \rightarrow C, \Delta_{\vec{\alpha}}$. Then

$$\dfrac{\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} \lambda x.P : B \rightarrow C, \Delta_{\vec{\alpha}} \quad \Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} yQ : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}}, y : A \rightarrow B \vdash_{\lambda\mu N} (\lambda x.P)(yQ) : C, \Delta_{\vec{\alpha}}}\ (\rightarrow \text{elim})$$

and $(\lambda x.P)(yQ) \rightarrow P[x := yQ]$, as required.

Admissibility of the (cut) rule in $\lambda\mu N$ is treated as follows.

$$\dfrac{\dfrac{\Gamma_{\vec{x}}, x : A \vdash_{\lambda\mu N} P : B, \Delta_{\vec{\alpha}}}{\Gamma_{\vec{x}} \vdash_{\lambda\mu N} \lambda x.P : A \rightarrow B, \Delta_{\vec{\alpha}}}\ (\rightarrow \text{intro}) \qquad \Gamma_{\vec{x}} \vdash_{\lambda\mu N} Q : A, \Delta_{\vec{\alpha}}}{\Gamma \vdash_{\lambda\mu N} (\lambda x.P)Q : B, \Delta_{\vec{\alpha}}}\ (\rightarrow \text{elim})$$

(ii) By (i) and the Subject reduction theorem for $\lambda\mu N$ (Proposition 2.7(ii)). □

Corollary 3.3. $\Gamma_{\vec{x}} \vdash_{\lambda\mu L} M : C, \Delta_{\vec{\alpha}} \Longleftrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu N} M : C, \Delta_{\vec{\alpha}}$.

Proof. By Propositions 3.1 and 3.2(ii). □

In the following we will investigate the role of $\lambda\mu L^{\mathrm{cf}}$.

Proposition 3.4.

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu L^{\mathrm{cf}}} P : A, \Delta_{\vec{\alpha}} \Longrightarrow P \textit{ is in } \beta\mu\textit{-nf}.$$

Proof. By an easy induction on derivations. □

Lemma 3.5. *Suppose* $\Gamma_{\vec{x}} \vdash_{\lambda\mu L^{\mathrm{cf}}} P_1 : A_1, \Delta_{\vec{\alpha}}$, ... , $\Gamma_{\vec{x}} \vdash_{\lambda\mu L^{\mathrm{cf}}} P_n : A_n, \Delta_{\vec{\alpha}}$. *Then*

$$\Gamma_{\vec{x}}, x : A_1 \to \ldots \to A_n \to B \vdash_{\lambda\mu L^{\mathrm{cf}}} xP_1 \ldots P_n : B, \Delta_{\vec{\alpha}}$$

for those variables x *such that* $\Gamma, x : A_1 \to \ldots \to A_n \to B$ *is a term context.*

Proof. Without loss of generality we may assume $n = 2$. The following derivation proves the statement

$$\dfrac{\Gamma_{\vec{x}} \vdash P_1 : A_1, \Delta_{\vec{\alpha}} \qquad \dfrac{\Gamma_{\vec{x}} \vdash P_2 : A_2, \Delta_{\vec{\alpha}} \qquad \overline{\Gamma_{\vec{x}}, z\, B \vdash z : B, \Delta_{\vec{\alpha}}}}{\Gamma_{\vec{x}}, y : A_2 \to B \vdash yP_2 : B, \Delta_{\vec{\alpha}}}\,(\to L)}{\Gamma_{\vec{x}}, x : A_1 \to A_2 \to B \vdash xP_1P_2 : B, \Delta_{\vec{\alpha}}}\,(\to L)$$

where $yP_2 \equiv z[z := yP_2]$ and $xP_1P_2 \equiv yP_2[y := xP_1]$. □

Proposition 3.6. *Suppose that* P *is a* $\beta\mu$*-nf. Then*

$$\Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}} \Longrightarrow \Gamma_{\vec{x}} \vdash_{\lambda\mu L^{\mathrm{cf}}} P : A, \Delta_{\vec{\alpha}}.$$

Proof. By induction on the following generation of normal forms.

$$\begin{array}{rcl} \mathrm{nf}_t & = & \mathtt{var} \quad \mid \mathtt{var}\,\mathrm{nf}^+ \quad \mid \lambda\mathtt{var}.\mathrm{nf} \\ \mathrm{nf} & = & \mathrm{nf}_t \quad \mid \mu\beta.[\alpha]\,\mathrm{nf}_t \end{array}$$

The easy cases are $P \equiv x$, $P \equiv \lambda x.P_1$ and $P \equiv \mu\beta.[\alpha]P_2$, where P_2 is not a μ abstraction. The case $P \equiv xP_1 \ldots P_n$ follows from the previous lemma, using the Generation lemma for λN (2.7(iii)(3)). □

As bonus, we now get the cut elimination property, Hauptsatz, of Gentzen [8] for classical implicational sequent calculus.

Theorem 3.7 (Cut elimination)**.**

$$\Gamma \vdash_{LK} A \Longrightarrow \Gamma \vdash_{LK^{\mathrm{cf}}} A.$$

Proof.

$$\begin{array}{rcll} \Gamma \vdash_{LK} A, \Delta & \Longleftrightarrow & \Gamma_{\vec{x}} \vdash_{\lambda\mu L} P : A, \Delta_{\vec{\alpha}} & \text{by Proposition 2.6} \\ & \Longrightarrow & \Gamma_{\vec{x}} \vdash_{\lambda\mu N} P : A, \Delta_{\vec{\alpha}}, & \text{by Proposition 3.2}(ii) \\ & \Longrightarrow & \Gamma_{\vec{x}} \vdash_{\lambda\mu N} P^{\mathtt{nf}} : A, \Delta_{\vec{\alpha}}, & \text{by Proposition 2.7}(i)\text{and}(ii) \\ & \Longrightarrow & \Gamma_{\vec{x}} \vdash_{\lambda\mu L^{\mathrm{cf}}} P^{\mathtt{nf}} : A, \Delta_{\vec{\alpha}}, & \text{by Proposition 3.6} \\ & \Longrightarrow & \Gamma \vdash_{LK^{\mathrm{cf}}} A, \Delta, & \text{by Proposition 2.6.} \end{array}$$

4. Discussion

There are several calculi for encoding proofs in classical sequent calculus: Symmetric lambda calculus of Barbanera and Berardi [2], Curien and Herbelin's $\overline{\lambda}\mu\widetilde{\mu}$-calculus [4, 11], the calculus of Urban and Bierman [18, 19], Dual calculus of Wadler [20, 21], the symmetric extension of $\lambda\mu$ by David and Nour [5]. We did not consider whether a direct encoding of derivations in some of the symmetric lambda calculi and their (strong) normalisation properties (Dougherty et al. [7, 6], David and Nour [5]) can lead to similar results.

The main technical tool in this paper is the type assignment system $\lambda\mu L$ based on $\lambda\mu$-terms which corresponds to classical sequent logic. This system is not any of the known systems for encoding proofs in classical sequent logic. The emphasis here is on $\lambda\mu$-terms rather than on derivations. The aim was to revisit the cut-elimination theorem for classical logic via normalisation of $\lambda\mu$-terms, in the style of [3]. In $\lambda\mu L$ formulae provable in LK and $\lambda\mu$-terms are first class citizens, whereas in the systems mentioned above derivations in LK are in the focus. There is an analogy with different approaches in [3] and Herbelin [10]. The former paper considers formulae provable in intuitionistic sequent logic and λ-terms as first class citizens, whereas in the latter one the encoding of derivations of intuitionistic sequent logic is in the focus.

Acknowledgement

I wish to thank Henk Barendregt with whom we developed the ideas on which this paper is based. I am grateful to Pierre Lescanne, Daniel Dougherty and Hugo Herbelin for research and discussions we have had during several years on the computational content of logic.

References

[1] Z. M. Ariola and H. Herbelin. Minimal classical logic and control operators. In *ICALP: Annual International Colloquium on Automata, Languages and Programming*, volume 2719 of *LNCS*, pages 871–885. Springer-Verlag, 2003.

[2] F. Barbanera and S. Berardi. A symmetric lambda calculus for classical program extraction. *Information and Computation*, 125(2):103–117, 1996.

[3] H. P. Barendregt and S. Ghilezan. Lambda terms for natural deduction, sequent calculus and cut-elimination. *J. of Functional Programming*, 10(1):121–134, 2000.

[4] P.-L. Curien and H. Herbelin. The duality of computation. In *Proc. of the 5th ACM SIGPLAN Int. Conference on Functional Programming, ICFP'00*, Montreal, Canada, 2000. ACM Press.

[5] R. David and K. Nour. Arithmetical proofs of strong normalization results for symmetric λ-calculi. *Fundamenta Informaticae*, 77(4):489–510, 2007.

[6] D. Dougherty, S. Ghilezan, and P. Lescanne. Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: extending the Coppo-Dezani heritage. *Theoretical Computer Science, to appear in Festschrift Coppo, Dezani, Ronchi*, 2007.

[7] D. Dougherty, S. Ghilezan, P. Lescanne, and S. Likavec. Strong normalization of the dual classical sequent calculus. In G. Sutcliffe and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 12th International Conference, LPAR 2005*, volume 3835 of *LNCS*, pages 169–183. Springer, 2005.

[8] G. Gentzen. Unterschungen über das logische Schliessen, Math Z. 39 (1935), 176–210. In M.E. Szabo, editor, *Collected papers of Gerhard Gentzen*, pages 68–131. North-Holland, 1969.

[9] T. Griffin. A formulae-as-types notion of control. In *Proc. of the 19th Annual ACM Symp. on Principles Of Programming Languages, (POPL'90)*, pages 47–58, San Fransisco (Ca., USA), 1990. ACM Press.

[10] H. Herbelin. A lambda calculus structure isomorphic to Gentzen-style sequent calculus structure. In *Computer Science Logic, CSL 1994*, volume 933 of *LNCS*, pages 61–75. Springer-Verlag, 1995.

[11] H. Herbelin. *C'est maintenant qu'on calcule, au cœur de la dualité*. Mémoire d'habiliation, Université de Paris-Orsay, December 2005.

[12] W. A. Howard. The formulas-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, New York, 1980. Academic Press.

[13] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *POPL 24*, pages 215–227, 1997.

[14] M. Parigot. An algorithmic interpretation of classical natural deduction. In *Proc. of Int. Conf. on Logic Programming and Automated Reasoning, LPAR'92*, volume 624 of *LNCS*, pages 190–201. Springer-Verlag, 1992.

[15] M. Parigot. Proofs of strong normalisation for second order classical natural deduction. *The J. of Symbolic Logic*, 62(4):1461–1479, December 1997.

[16] D Prawitz. *Natural Deduction*. Almqvist and Wiksell, 1965.

[17] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard isomorphism*. Studies in Logic and the Foundations of Mathematics, 149.

[18] C. Urban and G. M. Bierman. Strong normalisation of cut-elimination in classical logic. In *Typed Lambda Calculus and Applications, TLCA'99*, volume 1581 of *LNCS*, pages 365–380. Springer-Verlag, 1999.

[19] C. Urban and G. M. Bierman. Strong normalisation of cut-elimination in classical logic. *Fundamenta Informaticae*, 45(1-2):123–155, 2001.

[20] Ph. Wadler. Call-by-value is dual to call-by-name. In *Proc. of the 8th ACM SIGPLAN Int. Conference on Functional Programming, ICFP'03*, pages 189–201, 2003.

[21] Ph. Wadler. Call-by-value is dual to call-by-name, reloaded. In *Rewriting Technics and Application, RTA'05*, volume 3467 of *LNCS*, pages 185–203, 2005.