A completeness result for $\lambda\mu$

Phillipe Audebaud[*] Steffen van Bakel[†]

Inria Sophia Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia Antipolis, France

E-mail: Philippe.Audebaud@sophia.inria.fr,svb@doc.ic.ac.uk

Abstract

We study the expressivity of Parigot's $\lambda\mu$-calculus, and show that each statement $\Gamma \vdash_{\text{LK}} \Delta$ that is provable in Gentzen's LK has a proof in $\lambda\mu$. This result is obtained through defining an interpretation from nets *from the $\mathcal{X}$-calculus into both the λ-calculus and $\lambda\mu$; $\mathcal{X}$ enjoys the full Curry-Howard isomorphism for (the implicative fragment of) LK, and cut-elimination in LK is represented by reduction in $\mathcal{X}$.*

This interpretation will be shown to preserve reduction in $\mathcal{X}$ via equality in the target calculi, and to preserve typeability using the standard double negation translation technique of types. Using the fact that, in $\lambda\mu$, we can inhabit $\neg\neg A{\to}A$ for all types A, a completeness result as well as a consistency result are shown for $\lambda\mu$.

1 Introduction

The sequent calculus LK, introduced by Gentzen [7], is a logical system in which the rules only introduce connectives (but on both sides of a sequent), on the contrary to natural deduction which uses introduction and elimination rules. The only way to eliminate a connective is to eliminate the whole formula in which it appears, with an application of the (cut)-rule. Gentzen's calculus for classical logic LK allows sequents of the form $A_1,\ldots,A_n \vdash B_1,\ldots,B_m$, where $A_1,\ldots,A_n$ is to be understood as $A_1{\wedge}\ldots{\wedge}A_n$ and $B_1,\ldots,B_m$ is to be understood as $B_1{\vee}\ldots{\vee}B_m$. Thus, LK appears as a very symmetrical system.

For this calculus, a *cut-elimination procedure* has been defined that eliminates all applications of the (cut)-rule from the proof of a sequent, generating a proof in *normal form* of the same sequent, that is, with no cut. It is defined via local rewriting steps, reductions of the proof-tree, which has the flavour of the evaluation of explicit substitutions [4].

The calculus $\mathcal{X}$, as presented in [1, 13] represents a correspondence *à la* Curry-Howard-de Bruijn for LK$(\to)$, the implicational fragment of LK, bringing together the various features of two different approaches: that of Urban [17] and that of Curien and Herbelin [5]. The aim of this paper is to relate $\mathcal{X}$ to the standard presentation of $\lambda\mu$, as we target the analysis of two subreduction systems, both designed to avoid unrecoverable critical pairs. We will show that there exist faithful mappings from $\mathcal{X}$ to both the λ-calculus and the $\lambda\mu$-calculus, which establishes a strong link between provable sequents in LK$(\to)$ and the $\lambda\mu$-calculus.

The relevance of this result can be understood from the observation that the $\mathcal{X}$-calculus expresses the full symmetry of LK, while the $\lambda\mu$-calculus does not. However, we will show that the latter is expressive enough to reflect the propagation rules of $\mathcal{X}$, thus exhibiting that the actual loss is due to natural deduction presentation of the calculus, which forces the choice of a distinguished, active conclusion.

The symmetric nature of the $\mathcal{X}$-calculus is evident from the fact that reduction is not confluent; as the target calculus is confluent, the price to pay is, as for the pure λ-calculus case, to restrict ourselves to particular subsystems of the full reduction, that do not cause unrecoverable critical pairs to occur.

From the logical point of view, the natural deduction presentation introduces a lack of symmetry which is going to require more work for the translation of $\mathcal{X}$-terms (called circuits) into $\lambda\mu$-terms. Our translation therefore will consist of first applying a CPS-like transformation, followed by a recovery of the type information for the global derivation tree:

$$\mathcal{X}{\to}\lambda\mu_{\text{CPS}}{\to}\lambda\mu$$

This will allow us to prove:

Theorem COMPLETENESS. *If $\Gamma \vdash_{\text{LK}} \Delta$ in* LK$(\to)$*, then for any (fresh) type Ω, there exists a $\lambda\mu$-term M, and contexts Γ',Δ' such that $\Gamma' \vdash_{\lambda\mu} M{:}\Omega \mid \Delta'$ such that Γ,Δ can be obtained from Γ',Δ' by erasure of names.*

[*] École Normale Supérieure de Lyon, 46 Allée d'Italie 69364 Lyon 07, FRANCE

[†] On sabbatical leave from Department of Computing, Imperial College London, 180 Queen's Gate London SW7 2BZ, U.K.

We will also prove:

Theorem CONSISTENCY. *If P, P' are proofs in* LK$(\to)$, *such that P can be changed into P' performing cut-elimination steps, then their interpretations in $\lambda\mu$ are equivalent, i.e. share a common reduct.*

This result is obtained via the interpretation of $\mathcal{X}$ circuits into pure λ-terms, which requires a double-negation translation on types and loses the syntactical distinction between inputs and outputs. As we deal with a fragment of classical logic, a minimum requirement is to extend the pure λ-calculus in this direction. Possible extensions are the λ_C-Calculus (where C stands for Griffin's C operator) [8, 12] or Parigot's $\lambda\mu$-Calculus [15]. Since we want sockets and plugs being kept distinct from each other, we favour the second solution.

Limiting ourselves to the implicative fragment of LK might seem to be too much of a restriction, but this not so. In fact, extending the calculus with (rules and constructs for) the logical connectives $\wedge, \vee, \forall, \exists, \neg$ is straightforward, and brings no added complexity for achievable results. Also, arrow types are the natural types for the λ-calculus and $\lambda\mu$.

2 The $\mathcal{X}$-calculus

2.1 From LK to a calculus

There exist a number of systems that link Classical Logic with a notion of computation. In Parigot's $\lambda\mu$-calculus is a natural deduction system in which there is one main conclusion that is being manipulated and possibly several alternative ones. On the other hand, there exists the sequent calculus and the necessity for the left-introduction rules to manipulate hypotheses, and the concept of *stoup* that Herbelin has thoroughly studied in [10, 5, 11].

As mentioned in the introduction, $\mathcal{X}$ is inspired by the sequent calculus, so it is worthwhile to recall some of the principles.

Definition 2.1 LK$(\to)$. The sequent calculus we consider has only implication, no structural rules and a changed axiom. It offers an extremely natural presentation of the classical propositional calculus with implication, and is a variant of system LK.

It has four rules: *axiom*, *right introduction* of the arrow, *left introduction* and *cut*.

$$(ax) : \dfrac{}{\Gamma, A \vdash A, \Delta} \qquad (cut) : \dfrac{\Gamma \vdash A, \Delta \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}$$

$$(\Rightarrow R) : \dfrac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \Rightarrow B, \Delta} \qquad (\Rightarrow L) : \dfrac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \Rightarrow B \vdash \Delta}$$

It is well known that the rule (cut) plays a major role in proofs, since cut-free proofs enjoy nice properties; proof reductions via cut-elimination have been proposed by Gentzen. Those reductions become the fundamental principle of computation in $\mathcal{X}$.

One of the key points of Herbelin's $\overline{\lambda}\mu\tilde{\mu}$-calculus is to notice that the stoup and the main conclusion of $\lambda\mu$ are the dual notions of each other, and to express this duality in a very symmetrical syntax. But the duality goes beyond that: for instance, the symmetry of the reduction rules display syntactically the duality between the CBV and CBN evaluations (see also [18]). However, this duality notwithstanding, $\overline{\lambda}\mu\tilde{\mu}$ does not fully represent LK. The LK proof

$$\dfrac{\dfrac{\Gamma, A \vdash_{\text{LK}} B, \Delta}{\Gamma \vdash_{\text{LK}} A{\to}B, \Delta}(\to R) \quad \dfrac{\Gamma \vdash_{\text{LK}} A, \Delta \quad \Gamma, B \vdash_{\text{LK}} \Delta}{\Gamma, A{\to}B \vdash_{\text{LK}} \Delta}(\to L)}{\Gamma \vdash_{\text{LK}} \Delta}(cut)$$

reduces to both

$$\dfrac{\Gamma \vdash_{\text{LK}} A, \Delta \quad \dfrac{\Gamma, A \vdash_{\text{LK}} B, \Delta \quad \dfrac{\Gamma, B \vdash_{\text{LK}} \Delta}{\Gamma, A, B \vdash_{\text{LK}} \Delta}}{\Gamma, A \vdash_{\text{LK}} \Delta}}{\Gamma \vdash_{\text{LK}} \Delta}$$

and

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash_{\text{LK}} A, \Delta}{\Gamma \vdash_{\text{LK}} A, B, \Delta} \quad \Gamma, A \vdash_{\text{LK}} B, \Delta}{\Gamma \vdash_{\text{LK}} B, \Delta} \quad \Gamma, B \vdash_{\text{LK}} \Delta}{\Gamma \vdash_{\text{LK}} \Delta}$$

The first result is represented in the normal reduction system of $\overline{\lambda}\mu\tilde{\mu}$, but the second is not, whereas both are represented in $\mathcal{X}$.

2.2 Syntax

The Curry-Howard correspondence for $\mathcal{X}$ with classical propositional calculus is achieved by giving propositions names; those that appear in the left part of a sequent receive names like $x, y, z, \ldots$ and those that appear in the right part of a sequent receive name like $\alpha, \beta, \gamma, \ldots$, and to associate formulae with types.

The circuits that are the objects of $\mathcal{X}$ are built with three kinds of building stones, or constructors, called *capsule*, *export* and *import*. In addition there is an operator we call *cut*, which is handy for describing circuit construction, and which will be eliminated eventually by *rules*. These four will be the natural representatives for the four logical rules given above.

Circuits are connected through *wires* that are named. In our description wires are oriented. This means we know in which direction the 'ether running through our circuits' moves, and can say when a wire provides an entrance to a circuit or when a wire provides an exit. Thus we make the distinction between exit wires which we call *plugs* and enter wires which we call *sockets*. Plugs are named with Greek letters $\alpha, \beta, \gamma, \delta, \ldots$ and sockets are named with Latin letters $x, y, z, \ldots$.

When connecting two circuits P and Q, we may suppose that P has a plug α and Q has a socket x which we want to connect together to create a flow from P to Q. After the link has been established, the wires have been plugged, and the name of the plug and the name of the socket are forgotten. To be more precise, in $P\widehat{\alpha} \dagger \widehat{x}Q$, the name α is bound over P and the name x is bound over Q, bound in the interaction. We use the *"hat"*-notation, keeping in line with the old tradition of *Principia Mathematica* [19], writing $\hat{x}$ to say that x is bound.

Definition 2.2 [1] SYNTAX. The circuits of the $\mathcal{X}$-calculus are defined by the following grammar, where $x, y, \ldots$ range over the infinite set of *sockets*, and α, β over the infinite set of *plugs*.

$$\begin{array}{rccccccc} P, Q ::= & \langle x.\alpha\rangle & \mid & \widehat{y}P\widehat{\beta}\cdot\alpha & \mid & P\widehat{\beta}\,[y]\,\widehat{x}Q & \mid & P\widehat{\alpha}\dagger\widehat{x}Q \\ & \textit{capsule} & & \textit{export} & & \textit{import} & & \textit{cut} \end{array}$$

Notice that, using the intuition sketched above, for example, the connector β is supposed not to occur outside of $\widehat{y}P\widehat{\beta}\cdot\alpha$.

The calculus, defined by the reduction rules (Section *2.3*) explains in detail how cuts are distributed through circuits to be eventually erased at the level of capsules.

We spoke above about bound names; we will introduce now formally those notions with that of free sockets and plugs into $\mathcal{X}$.

Definition 2.3 The *free sockets* and *free plugs* in a circuit are:

$$\begin{array}{lcl} \mathit{fs}(\langle x.\alpha\rangle) &=& \{x\} \\ \mathit{fs}(\widehat{x}P\widehat{\beta}\cdot\alpha) &=& \mathit{fs}(P) \setminus \{x\} \\ \mathit{fs}(P\widehat{\alpha}\,[y]\,\widehat{x}Q) &=& \mathit{fs}(P) \cup \{y\} \cup (\mathit{fs}(Q) \setminus \{x\}) \\ \mathit{fs}(P\widehat{\alpha}\dagger\widehat{x}Q) &=& \mathit{fs}(P) \cup (\mathit{fs}(Q) \setminus \{x\}) \end{array}$$

$$\begin{array}{lcl} \mathit{fp}(\langle x.\alpha\rangle) &=& \{\alpha\} \\ \mathit{fp}(\widehat{x}P\widehat{\beta}\cdot\alpha) &=& (\mathit{fp}(P) \setminus \{\beta\}) \cup \{\alpha\} \\ \mathit{fp}(P\widehat{\alpha}\,[y]\,\widehat{x}Q) &=& (\mathit{fp}(P) \setminus \{\alpha\}) \cup \mathit{fp}(Q) \\ \mathit{fp}(P\widehat{\alpha}\dagger\widehat{x}Q) &=& (\mathit{fp}(P) \setminus \{\alpha\}) \cup \mathit{fp}(Q) \end{array}$$

A socket x or plug α which is not free is called *bound*. We will write $x \notin \mathit{fs}(P, Q)$ for $x \notin \mathit{fs}(P)$ & $x \notin \mathit{fs}(Q)$.

We will normally adopt Barendregt's convention (called *convention on variables* by Barendregt, but here it will be a convention on names).

2.3 The rules

For reduction, it is important to know when a socket or a plug is introduced, i.e. is connectable, i.e. is exposed and unique. Informally, a circuit P introduces a socket x if P is constructed from subcircuits which do not contain x as free socket, so x only occurs at the "top level." This means that P is either an import with a middle connector $[x]$ or a capsule with left part x. Similarly, a circuit introduces a plug α if it is an export that "creates" α or a capsule with right part α (Urban [17] uses the terminology "freshly introduce").

Definition 2.4 [1] INTRODUCTION. $(P$ *introduces* $x)$: $P = \langle x.\beta\rangle$ or $P = R\widehat{\alpha}\,[x]\,\widehat{y}Q$, with $x \notin \mathit{fs}(R, Q)$.
$(P$ *introduces* $\alpha)$: $P = \langle y.\alpha\rangle$ or $P = \widehat{x}Q\widehat{\beta}\cdot\alpha$ with $\alpha \notin \mathit{fp}(Q)$.

We first present a simple family of reduction rules. They say how to reduce a circuit that cuts subcircuits that both introduce connectors.

Definition 2.5 LOGICAL REDUCTION [1]. The logical rules are (assume that the circuits of the left-hand sides of the rules *introduce* the socket x and the plug α)

$$\begin{array}{rrcl} (\textit{cap}): & \langle y.\alpha\rangle\widehat{\alpha}\dagger\widehat{x}\langle x.\beta\rangle &\to& \langle y.\beta\rangle \\ (\textit{exp}): & (\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha}\dagger\widehat{x}\langle x.\gamma\rangle &\to& \widehat{y}P\widehat{\beta}\cdot\gamma \\ (\textit{imp}): & \langle y.\alpha\rangle\widehat{\alpha}\dagger\widehat{x}(P\widehat{\beta}\,[x]\,\widehat{z}Q) &\to& P\widehat{\beta}\,[y]\,\widehat{z}Q \\ (\textit{exp-imp}): & (\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha}\dagger\widehat{x}(Q\widehat{\gamma}\,[x]\,\widehat{z}R) &\to& \\ & & & \left\{\begin{array}{l} (Q\widehat{\gamma}\dagger\widehat{y}P)\widehat{\beta}\dagger\widehat{z}R \\ Q\widehat{\gamma}\dagger\widehat{y}(P\widehat{\beta}\dagger\widehat{z}R) \end{array}\right. \end{array}$$

We now need to define how to reduce a cut circuit in case when one of its sub-circuits does not introduce a socket or a plug. This requires to extend the syntax with two new operators that we call *activated* cuts:

$$P ::= \ldots \mid P\widehat{\alpha} \not\nearrow \widehat{x}Q \mid P\widehat{\alpha} \not\nwarrow \widehat{x}Q$$

Reduction on circuits with activated cuts will make sure these are propagated through the circuits.

Definition 2.6 [1] ACTIVATING THE CUTS.

$$\begin{array}{rl} (\textit{act-}\mathrm{L}): & P\widehat{\alpha}\dagger\widehat{x}Q \to P\widehat{\alpha} \not\nearrow \widehat{x}Q, \textit{ if } P \textit{ does not introduce } \alpha \\ (\textit{act-}\mathrm{R}): & P\widehat{\alpha}\dagger\widehat{x}Q \to P\widehat{\alpha} \not\nwarrow \widehat{x}Q, \textit{ if } Q \textit{ does not introduce } x \end{array}$$

Notice that both side-conditions can be valid simultaneously, thereby validating both rewrite rules at the same moment. This gives, in fact, a *critical pair* or *superposition* for our notion of reduction, and is the cause for the loss of confluence.

We will now define how to propagate an activated cut through sub-circuits. The direction of the activating shows in which direction the cut should be propagated, hence the two sets of reduction rules.

Definition 2.7 [1] PROPAGATION REDUCTION. The rules of propagation are:

Left propagation

$$\begin{array}{rl}
(d\text{L}): & \langle y.\alpha\rangle\widehat{\alpha} \not\dagger \widehat{x}P \;\to\; \langle y.\alpha\rangle\widehat{\alpha} \dagger \widehat{x}P \\
(cap\not\dagger): & \langle y.\beta\rangle\widehat{\alpha} \not\dagger \widehat{x}P \;\to\; \langle y.\beta\rangle, \quad \beta \neq \alpha \\
(exp\text{-}outs\not\dagger): & (\widehat{y}Q\widehat{\beta}\cdot\alpha)\widehat{\alpha} \not\dagger \widehat{x}P \;\to \\
& (\widehat{y}(Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta}\cdot\gamma)\widehat{\gamma} \dagger \widehat{x}P, \gamma \textit{ fresh} \\
(exp\text{-}ins\not\dagger): & (\widehat{y}Q\widehat{\beta}\cdot\gamma)\widehat{\alpha} \not\dagger \widehat{x}P \;\to \\
& \widehat{y}(Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta}\cdot\gamma, \gamma \neq \alpha \\
(imp\not\dagger): & (Q\widehat{\beta}\,[z]\,\widehat{y}R)\widehat{\alpha} \not\dagger \widehat{x}P \;\to \\
& (Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta}\,[z]\,\widehat{y}(R\widehat{\alpha} \not\dagger \widehat{x}P) \\
(cut\not\dagger): & (Q\widehat{\beta} \dagger \widehat{y}R)\widehat{\alpha} \not\dagger \widehat{x}P \;\to \\
& (Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta} \dagger \widehat{y}(R\widehat{\alpha} \not\dagger \widehat{x}P)
\end{array}$$

Right propagation

$$\begin{array}{rl}
(d\text{R}): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}\langle x.\beta\rangle \;\to\; P\widehat{\alpha} \dagger \widehat{x}\langle x.\beta\rangle \\
(\backslash\!\!\!\dagger cap): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}\langle y.\beta\rangle \;\to\; \langle y.\beta\rangle, \quad y \neq x \\
(\backslash\!\!\!\dagger exp): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}(\widehat{y}Q\widehat{\beta}\cdot\gamma) \;\to\; \widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q)\widehat{\beta}\cdot\gamma \\
(\backslash\!\!\!\dagger imp\text{-}outs): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}(Q\widehat{\beta}\,[x]\,\widehat{y}R) \;\to \\
& P\widehat{\alpha} \dagger \widehat{z}((P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q)\widehat{\beta}\,[z]\,\widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}R)), \; z \textit{ fresh} \\
(\backslash\!\!\!\dagger imp\text{-}ins): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}(Q\widehat{\beta}\,[z]\,\widehat{y}R) \;\to \\
& (P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q)\widehat{\beta}\,[z]\,\widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}R), z \neq x \\
(\backslash\!\!\!\dagger cut): & P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}(Q\widehat{\beta} \dagger \widehat{y}R) \;\to \\
& (P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q)\widehat{\beta} \dagger \widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}R)
\end{array}$$

The rules $(exp\text{-}outs\not\dagger)$ and $(\backslash\!\!\!\dagger imp\text{-}outs)$ deserve some attention. For instance, in the left-hand side of $(exp\text{-}outs\not\dagger)$, α is not introduced, hence α occurs more than once in $\widehat{y}Q\widehat{\beta}\cdot\alpha$, that is once after the dot and again in Q. The occurrence after the dot is dealt with separately by creating a new name γ. Note that the cut associated with that γ is then unactivated; this is because, after the cut has been pushed through $\widehat{y}(Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta}\cdot\gamma$ (so leaves a circuit with no activated cut), the resulting circuit $(\widehat{y}R\widehat{\beta}\cdot\gamma)\widehat{\gamma} \dagger \widehat{x}P$ needs to be considered in its entirety: although we now that now γ is introduced, we know not if x is. A similar reasoning holds for x in $(\backslash\!\!\!\dagger imp\text{-}outs)$ and a new name z is created and the external cut is not active.

2.4 Call-by-name and call-by-value

In this section we will define two sub-systems of reduction, that correspond to call-by-name (CBN) and call-by-value (CBV) reduction. Notice that this is essentially different from the approach of [18], where, as in $\overline{\lambda}\mu\tilde{\mu}$, only one notion of reduction is defined.

As mentioned above, when P does not introduce α and Q does not introduce x, $P\widehat{\alpha} \dagger \widehat{x}Q$ is a *superposition*, meaning that two rules, namely $(act\text{-L})$ and $(act\text{-R})$, can both be fired. The *critical pair* $\langle P\widehat{\alpha} \not\dagger \widehat{x}Q, P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q\rangle$. may lead to different irreducible circuits. This is to say that the reduction relation $\to$ is *not confluent*. Non-determinism is a key feature of both classical logic and term rewriting.

We introduce two strategies which explicitly favour one kind of activating whenever the above critical pair occurs. Consider a circuit $P\widehat{\alpha} \dagger \widehat{x}Q$ where P does not introduce α and Q does not introduce x, intuitively CBV tends to push Q through P and CBN tends to do the other way around.

Definition 2.8 • The CBV strategy only activates a cut via $(act\text{-L})$ when it could be activated in two ways; we write $P \to_{\text{V}} Q$ in that case, and replace rule $(act\text{-R})$ by:

$$\begin{array}{rl}
(act\text{-R}): & P\widehat{\alpha} \dagger \widehat{x}Q \;\to\; P\widehat{\alpha} \backslash\!\!\!\dagger \widehat{x}Q, \textit{ if } P \textit{ introduces } \alpha \\
& \textit{and } Q \textit{ does not introduce } x.
\end{array}$$

• The CBN strategy only activates such a cut via $(act\text{-R})$; like above, we write $P \to_{\text{N}} Q$, and replace rule $(act\text{-L})$ by:

$$\begin{array}{rl}
(act\text{-L}): & P\widehat{\alpha} \dagger \widehat{x}Q \;\to\; P\widehat{\alpha} \not\dagger \widehat{x}Q, \textit{ if } Q \textit{ introduces } x \\
& \textit{and } P \textit{ does not introduce } \alpha.
\end{array}$$

2.5 Typing for $\mathcal{X}$

We will now formally define a notion of type assignment on $\mathcal{X}$, which will establish the Curry-Howard-de Bruijn isomorphism between $\mathcal{X}$ and LK$(\to)$.

Definition 2.9 TYPES AND CONTEXTS. *i*) The set of types is defined by the grammar:

$$A, B ::= \varphi \mid A{\to}B.$$

The types considered in this paper are normally known as *simple* (or *Curry*) types.

ii) A *context of sockets* Γ is a mapping from sockets to types, denoted as a finite set of *statements* $x{:}A$, such that the *subject* of the statements (x) are distinct. We write $\Gamma, x{:}A$ for the context defined by:

$$\begin{array}{rll}
\Gamma, x{:}A & = \Gamma \cup \{x{:}A\}, & \text{if } \Gamma \text{ is not defined on } x \\
& = \Gamma, & \text{otherwise}
\end{array}$$

So, when writing a context as $\Gamma, x{:}A$, this implies that $x{:}A \in \Gamma$, or Γ is not defined on x. When we write Γ_1, Γ_2 we mean the union of Γ_1 and Γ_2 when Γ_1 and Γ_2 are coherent (if Γ_1 contains $x{:}A_1$ and Γ_2 contains $x{:}A_2$ then $A_1 = A_2$).

iii) Contexts of *plugs* Δ are defined in a similar way.

Definition 2.10 [1] TYPING FOR $\mathcal{X}$. *i*) *Type judgements* are expressed via a ternary relation $P :\cdot \Gamma \vdash \Delta$, where Γ is a context of *sockets* and Δ is a context of *plugs*, and P is a circuit. We say that P is the *witness* of this judgement.

ii) *Type assignment for $\mathcal{X}$* is defined by the following sequent calculus:

$$(cap): \frac{}{\langle y.\alpha\rangle :\cdot \Gamma, y{:}A \vdash \alpha{:}A, \Delta}$$

$$(imp): \frac{P :\cdot \Gamma \vdash \alpha{:}A, \Delta \quad Q :\cdot \Gamma, x{:}B \vdash \Delta}{P\widehat{\alpha}\,[y]\,\widehat{x}Q :\cdot \Gamma, y{:}A{\to}B \vdash \Delta}$$

$$(exp): \frac{P :\cdot \Gamma, x{:}A \vdash \alpha{:}B, \Delta}{\widehat{x}P\widehat{\alpha}\cdot\beta :\cdot \Gamma \vdash \beta{:}A{\to}B, \Delta}$$

$$(cut): \frac{P :\cdot \Gamma \vdash \alpha{:}A, \Delta \quad Q :\cdot \Gamma, x{:}A \vdash \Delta}{P\widehat{\alpha}\dagger\widehat{x}Q :\cdot \Gamma \vdash \Delta}$$

We write $P :\cdot \Gamma \vdash \Delta$ if there exists a derivation that has this judgement in the bottom line.

Γ and Δ carry the types of the free connectors in P, as unordered sets. There is no notion of type for P itself, instead the derivable statement shows how P is connectable. The Curry-Howard property for the implicative fragment of LK is easily achieved by erasing all term-information.

The soundness result of simple type assignment with respect to reduction is stated as usual:

Theorem 2.11 [1] WITNESS REDUCTION. *If $P :\cdot \Gamma \vdash \Delta$, and $P \to Q$, then $Q :\cdot \Gamma \vdash \Delta$.*

3 The λ-calculus

The expressive power of $\mathcal{X}$ is illustrated in [1] by showing that the λ-calculus [3], $\lambda\mathbf{x}$, $\lambda\mu$, and $\overline{\lambda}\mu\tilde{\mu}$ can be faithfully interpreted via the mapping $[\![M]\!]_\alpha$. Using the notion of Curry type assignment, assignable types are preserved by the interpretation: if $\Gamma \vdash_\lambda M : A$, then $[\![M]\!]_\alpha :\cdot \Gamma \vdash \alpha{:}A$. Even more, the interpretation encompasses CBV and CBN reduction. However, in $\mathcal{X}$ we have no need of two separate interpretation functions, but will define only *one*. Combining this with the two sub-reduction systems $\to_{\mathrm{V}}$ and $\to_{\mathrm{N}}$ we can encode the the CBV- and CBN-λ-calculus.

We assume the reader to be familiar with the λ-calculus [3]; we just recall the definition of lambda terms and β-contraction.

Definition 3.1 LAMBDA TERMS AND REDUCTION [3].

i) The set Λ of *lambda terms* is defined by the syntax:

$$M ::= x \mid \lambda x.M \mid M_1 M_2$$

ii) The reduction relation $\to_\beta \;\subseteq \Lambda \times \Lambda$ is defined as the contextual, reflexive, symmetric, and transitive (i.e. compatible [3]) closure of the rule:

$$(\lambda x.M)N \;\to_\beta\; M[N/x]$$

iii) The notion of reduction $\to_\beta$ can be restricted to *Call by Value* reduction by: the set of *values* $\subseteq \Lambda$ is defined by the syntax:

$$V ::= x \mid \lambda x.M$$

Then the *Call by Value* reduction relation $\to_{\mathrm{V}}$ is defined as the compatible closure of the rule:

$$(\lambda x.M)V \;\to_\beta\; M[V/x]$$

The full reduction system is then called *Call by Name*, and we will write $\to_{\mathrm{N}}$ when necessary.

This calculus has a notion of type assignment that corresponds nicely to implicative propositional logic, in the framework of natural deduction.

Definition 3.2 TYPE ASSIGNMENT FOR THE λ-CALCULUS.

$$(ax): \frac{}{\Gamma, x{:}A \vdash_\lambda x : A} \qquad (\to I): \frac{\Gamma, x{:}A \vdash_\lambda M : B}{\Gamma \vdash_\lambda \lambda x.M : A{\to}B}$$

$$(\to E): \frac{\Gamma \vdash_\lambda M : A{\to}B \qquad \Gamma \vdash_\lambda N : A}{\Gamma \vdash_\lambda MN : B}$$

The logic that the above system establishes a Curry-Howard isomorphism for is Implicative Intuitionistic Logic.

4 The $\lambda\mu$-calculus

Parigot [15] presented the $\lambda\mu$-calculus as a calculus which extends the proofs-as-programs paradigm of the λ-calculus to classical logic. The $\lambda\mu$-calculus gives a natural deduction system which allows to deal with multi conclusions by choosing at most one *active* formula at once. This is achieved by introducing two kinds of variables, as found in more recent calculi like $\mathcal{X}$ itself, $\overline{\lambda}\mu\tilde{\mu}$[5], Wadler's dual calculus [18], and others.

Definition 4.1 $\lambda\mu$ TERMS. Terms of the $\lambda\mu$–calculus are generated by the grammar

$$\begin{array}{rl}(terms): & M, N ::= x \mid \lambda x.M \mid M\,N \mid \mu\alpha.C \\ (commands): & C ::= [\alpha]M\end{array}$$

where x ranges over (ordinary) term variables, and α over formula names (also called μ-variables).

The set of *values* is defined by:

$$(values): V ::= x \mid \lambda x.M \mid \mu\alpha.[\beta]V$$

We emphasise the clear distinction made above between regular terms and the so called commands; the original presentation of the calculus would have the case $\mu\alpha.[\beta]M$ in the syntax for terms. We use this separation for convenience only as our interpretations rely on it.

Definition 4.2 $\lambda\mu$ REDUCTION. Reduction for the $\lambda\mu$-calculus is defined as the compatible closure of the following reduction rules:

$$\begin{array}{rrcl}
\textit{logical}\ (\beta): & \lambda x.M\,N & \to & (M)\,\langle N/x\rangle \\
\textit{structural}\ (\mu): & (\mu\alpha.C)\,M & \to & \mu\alpha.C[[\alpha]\Box\,M/[\alpha]\Box] \\
\textit{renaming}\ (\nu): & [\beta]\mu\alpha.C & \to & C\langle\alpha/\beta\rangle \\
\textit{erasing}: & \mu\alpha.[\alpha]M & \to & M \textit{ if } \alpha \textit{ does not occur in } M.
\end{array}$$

As usual, the substitution mechanism $C[[\alpha]\Box\,M/[\alpha]\Box]$ used in the μ-reduction rule consists of replacing *recursively* every occurrence in the command C of a command $[\alpha]N$ *labelled* with α, by the command $[\alpha](N\,M)$.

Call-by-Value reduction is defined by restricting the rules as follows:

$$\begin{array}{rcl}
\lambda x.M\,V & \to & (M)\,\langle V/x\rangle \\
(\mu\alpha.C)\,V & \to & \mu\alpha.C[[\alpha](\Box\,V)/[\alpha]\Box]
\end{array}$$

Equipped with these reductions, the $\lambda\mu$-calculus is well known to be confluent.

In this paper, strictly speaking, we do not need to deal with μ-reduction, but, instead, with the compounded reduction defined by:

$$(\nu\circ\mu):\ [\alpha]((\mu\beta.C)\,(\lambda x.M)) \to C\langle\lambda x.M\cdot\alpha/\beta\rangle$$

where the substitution mechanism consists of replacing recursively in C every occurrence of a command of the shape $[\beta]N$ by the command $[\alpha](N\,\lambda x.M)$.

In this paper, we shall assign types to $\lambda\mu$-terms much along the same lines as for the λ-calculus. Actually, we will use more general judgements such as $\Gamma\vdash_{\lambda\mu} M{:}T \mid \Delta$ where Δ holds types for μ-variables, and is void as far as pure λ-calculus-terms are concerned. Formally:

Definition 4.3 TYPE ASSIGNMENT FOR $\lambda\mu$. Type assignment for $\lambda\mu$ is defined by the following natural deduction system:

$$(ax):\quad \frac{}{\Gamma, x{:}A\vdash_{\lambda\mu} x{:}A \mid \Delta}$$

$$(\to I):\quad \frac{\Gamma, x{:}A\vdash_{\lambda\mu} M{:}B \mid \Delta}{\Gamma\vdash_{\lambda\mu} \lambda x.M : A\to B \mid \Delta}$$

$$(\to E):\quad \frac{\Gamma\vdash_{\lambda\mu} M{:}A\to B \mid \Delta \qquad \Gamma\vdash_{\lambda\mu} N{:}A \mid \Delta}{\Gamma\vdash_{\lambda\mu} MN{:}B \mid \Delta}$$

$$(\mu):\quad \frac{\Gamma\vdash_{\lambda\mu} C \mid \alpha{:}A,\Gamma}{\Gamma\vdash_{\lambda\mu} \mu\alpha.C{:}A \mid \Gamma}$$

$$(\textsc{Cmd}):\quad \frac{\Gamma\vdash_{\lambda\mu} M{:}A \mid \alpha{:}A,\Gamma}{\Gamma\vdash_{\lambda\mu} [\alpha]M \mid \alpha{:}A,\Gamma}$$

5 Interpreting $\mathcal{X}$ into the λ-calculus

As can be expected, the interpretation of $\mathcal{X}$'s circuits into pure λ-terms requires a double-negation translation on types and loses syntactical distinction between inputs and outputs. As we deal with a fragment of classical logic, a minimum requirement is to extend the pure λ-calculus is this direction.

However, in this section we will show that we can still faithfully interpret $\mathcal{X}$ into the λ-calculus, and obtain a type-preservation result using the 'double negation' technique. A similar result was obtained in [13]; the main difference between that result and the one obtained here is that we interpret left- and right-cuts in different ways, whereas they are interpreted in the same way in [13].

5.1 Call by Name

We will now show that we can interpret the CBN-subreduction system of $\mathcal{X}$ in the CBN-λ-calculus. It should be noted that, in the CBN reduction system, a left-cut $P\widehat{\alpha}\not\nearrow\widehat{x}Q$ is only generated if Q introduces x, so if $Q=\langle x.\beta\rangle$, or $Q=R\widehat{\beta}\,[x]\,\widehat{y}S$, and y not free in R,S; this observation will prove important when dealing with activation and deactivation rules.

The CBN-interpretation of circuits in $\mathcal{X}$ as terms in Λ is defined as follows:

Definition 5.1 CBN INTERPRETATION.

$$\begin{array}{rcl}
\ulcorner\langle x.\alpha\rangle\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & x\lambda u.\alpha u \\
\ulcorner\widehat{x}P\widehat{\alpha}\cdot\beta\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & \beta\,\lambda x\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}} \\
\ulcorner P\widehat{\alpha}\,[y]\,\widehat{x}Q\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & \\
 & & y\,\lambda u.(\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(u\,\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) \\
\ulcorner P\widehat{\alpha}\dagger\widehat{x}Q\urcorner^{\lambda}_{\mathrm{N}} = \ulcorner P\widehat{\alpha}\not\nwarrow\widehat{x}Q\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & (\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) \\
\ulcorner P\widehat{\alpha}\not\nearrow\widehat{x}\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\beta x) \\
\ulcorner P\widehat{\alpha}\not\nearrow\widehat{x}(Q\widehat{\beta}\,[x]\,\widehat{y}R)\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & \\
 & & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.(\lambda y.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(x\,\lambda\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}))
\end{array}$$

Notice that, defining

$$\begin{array}{rcl}
\ulcorner\langle x.\beta\rangle\urcorner_{\mathrm{A}} & = & \beta x \\
\ulcorner Q\widehat{\beta}\,[x]\,\widehat{y}R\urcorner_{\mathrm{A}} & = & (\lambda y.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(x\,\lambda\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})
\end{array}$$

for the Q that appears in $P\widehat{\alpha}\not\nearrow\widehat{x}Q$, we have

$$\ulcorner Q \urcorner^{\lambda}_{\mathrm{N}} = x\lambda u.\ulcorner Q \urcorner_{\mathrm{A}}[u/x]$$

and could have defined

$$\ulcorner P\widehat{\alpha} \not\nearrow \widehat{x}Q \urcorner^{\lambda}_{\mathrm{N}} \triangleq (\lambda\alpha.\ulcorner P \urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner Q \urcorner_{\mathrm{A}})$$

This will be used in the cases dealing with propagation of left-cuts.

In order to show that typeability is preserved by the interpretation, we need first to define a CBN-translation of types:

Definition 5.2 Give a type constant Ω, we define a CBN-interpretation of types, that is split in two independent parts, inductively defined by

$$\begin{aligned} \langle \phi \rangle^{l}_{\mathrm{N}} &= (\phi \to \Omega) \to \Omega \\ \langle A \to B \rangle^{l}_{\mathrm{N}} &= ((\langle A \rangle^{l}_{\mathrm{N}} \to \langle B \rangle^{l}_{\mathrm{N}}) \to \Omega) \to \Omega \\[4pt] \langle \phi \rangle^{r}_{\mathrm{N}} &= \phi \to \Omega \\ \langle A \to B \rangle^{r}_{\mathrm{N}} &= ((\langle A \rangle^{r}_{\mathrm{N}} \to \Omega) \to \langle B \rangle^{r}_{\mathrm{N}} \to \Omega) \to \Omega \end{aligned}$$

We define $\langle \Gamma, x{:}A \rangle^{l}_{\mathrm{N}} = \langle \Gamma \rangle^{l}_{\mathrm{N}}, x{:}\langle A \rangle^{l}_{\mathrm{N}}$, and $\langle \alpha{:}A, D \rangle^{r}_{\mathrm{N}} = \alpha{:}\langle A \rangle^{r}_{\mathrm{N}}, \langle \Delta \rangle^{r}_{\mathrm{N}}$.

The following result links the two interpretations.

Lemma 5.3 $\langle A \rangle^{l}_{\mathrm{N}} = \langle A \rangle^{r}_{\mathrm{N}} \to \Omega$

Using these interpretations, we can show:

Theorem 5.4 *If $P \mathrel{:\cdot} \Gamma \vdash \Delta$, then $\langle \Gamma \rangle^{l}_{\mathrm{N}}, \langle \Delta \rangle^{r}_{\mathrm{N}} \vdash_{\lambda} \ulcorner P \urcorner^{\lambda}_{\mathrm{N}} : \Omega$.*

We can show that reduction in $\mathcal{X}$ is modeled by equality after interpretation:

Theorem 5.5 *If $P \to_{\mathrm{N}} Q$, then $\ulcorner P \urcorner^{\lambda}_{\mathrm{N}} =_{\mathrm{N}} \ulcorner Q \urcorner^{\lambda}_{\mathrm{N}}$.*

5.2 Call by Value

The results obtained above can be repeated for the CBV-subreduction. Again, note that, in the CBV reduction system, a right-cut $P\widehat{\alpha} \not\nwarrow \widehat{x}Q$ is only generated if P introduces α, so if $P = \langle y.\alpha \rangle$, or $P = \widehat{y}R\widehat{\beta}\cdot\alpha$, and α not free in R.

The CBV-interpretation of nets in $\mathcal{X}$ as terms in λ is defined as follows:

Definition 5.6 CBV INTERPRETATION.

$$\begin{aligned} \ulcorner \langle x.\alpha \rangle \urcorner^{\lambda}_{\mathrm{V}} &\triangleq \alpha\, x \\ \ulcorner \widehat{x}P\widehat{\alpha}\cdot\beta \urcorner^{\lambda}_{\mathrm{V}} &\triangleq \beta\, \lambda\alpha x.\ulcorner P \urcorner^{\lambda}_{\mathrm{V}} \\ \ulcorner P\widehat{\alpha}\,[y]\,\widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} &\triangleq (\lambda\alpha.\ulcorner P \urcorner^{\lambda}_{\mathrm{V}})(y\, \lambda x.\ulcorner Q \urcorner^{\lambda}_{\mathrm{V}}) \\ \ulcorner P\widehat{\alpha} \dagger \widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} = \ulcorner P\widehat{\alpha} \not\nearrow \widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} &\triangleq (\lambda\alpha.\ulcorner P \urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner Q \urcorner^{\lambda}_{\mathrm{V}}) \\ \ulcorner \langle y.\alpha \rangle\widehat{\alpha} \not\nwarrow \widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} &\triangleq (\lambda x.\ulcorner Q \urcorner^{\lambda}_{\mathrm{V}})\, y \\ \ulcorner (\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha} \not\nwarrow \widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} &\triangleq (\lambda x.\ulcorner Q \urcorner^{\lambda}_{\mathrm{V}})(\lambda\beta y.\ulcorner P \urcorner^{\lambda}_{\mathrm{V}}) \end{aligned}$$

Again, defining

$$\begin{aligned} \ulcorner \langle x.\beta \rangle \urcorner_{\mathrm{A}} &= x \\ \ulcorner \widehat{y}R\widehat{\beta}\cdot\alpha \urcorner_{\mathrm{A}} &= \lambda\beta y.\ulcorner R \urcorner^{\lambda}_{\mathrm{V}} \end{aligned}$$

for the P that appears in $P\widehat{\alpha} \not\nwarrow \widehat{x}Q$, we have

$$\ulcorner P \urcorner^{\lambda}_{\mathrm{V}} = \alpha \ulcorner P \urcorner_{\mathrm{A}} \textit{ where } P \textit{ introduces } \alpha$$

Notice that $\ulcorner P \urcorner_{\mathrm{A}}$ is a value in those cases, which is important below to make sure that the reduction is call-by-value. We could have defined

$$\ulcorner P\widehat{\alpha} \not\nwarrow \widehat{x}Q \urcorner^{\lambda}_{\mathrm{V}} \triangleq (\lambda x.\ulcorner Q \urcorner^{\lambda}_{\mathrm{V}})\ulcorner P \urcorner_{\mathrm{A}}$$

This will be used in the proofs below for the cases dealing with propagation of right-cuts.

In order to show that typeability is preserved by the interpretation, we need first to define a CBV-translation of types (notice that this is not Plotkin's CBV-interpretation):

Definition 5.7 Give again a type constant Ω, we now define a CBN-interpretation of types, that is also split in two independent parts, inductively by

$$\begin{aligned} \langle \phi \rangle^{l}_{\mathrm{V}} &= \phi \\ \langle A \to B \rangle^{l}_{\mathrm{V}} &= (\langle B \rangle^{l}_{\mathrm{V}} \to \Omega) \to \langle A \rangle^{l}_{\mathrm{V}} \to \Omega \\[4pt] \langle \phi \rangle^{r}_{\mathrm{V}} &= \phi \to \Omega \\ \langle A \to B \rangle^{r}_{\mathrm{V}} &= (\langle B \rangle^{r}_{\mathrm{V}} \to \langle A \rangle^{r}_{\mathrm{V}}) \to \Omega \end{aligned}$$

Again, we define $\langle \Gamma, x{:}A \rangle^{l}_{\mathrm{V}} = \langle \Gamma \rangle^{l}_{\mathrm{V}}, x{:}\langle A \rangle^{l}_{\mathrm{V}}$, and $\langle \alpha{:}A, D \rangle^{r}_{\mathrm{V}} = \alpha{:}\langle A \rangle^{l}_{\mathrm{V}}, \langle \Delta \rangle^{l}_{\mathrm{V}}$.

The following result links the two interpretations.

Lemma 5.8 $\langle A \rangle^{r}_{\mathrm{V}} = \langle A \rangle^{l}_{\mathrm{V}} \to \Omega$.

Using these interpretations, we can show:

Theorem 5.9 *If $P \mathrel{:\cdot} \Gamma \vdash \Delta$, then $\langle \Gamma \rangle^{l}_{\mathrm{V}}, \langle \Delta \rangle^{r}_{\mathrm{V}} \vdash_{\lambda} \ulcorner P \urcorner^{\lambda}_{\mathrm{V}} : \Omega$.*

We can show that, also for the CBV-interpretation, reduction in $\mathcal{X}$ is modeled by equality after interpretation:

Theorem 5.10 *If $P \to_{\mathrm{V}} Q$, then $\ulcorner P \urcorner^{\lambda}_{\mathrm{V}} =_{\mathrm{V}} \ulcorner Q \urcorner^{\lambda}_{\mathrm{V}}$.*

6 From $\mathcal{X}$ to $\lambda\mu$

We have seen above, for the pure λ-calculus, that besides the fact that we translate a symmetric calculus to a strongly right-oriented one, reflecting the orientation of propagation rules requires some particular attention in the translation. In this section, we will show that the results achieved above are obtainable for $\lambda\mu$ as well, first for Call by Name, and then for

Call by Value. However, the details of the interpretations will differ significantly.

The $\lambda\mu$-calculus is expressive enough to allow for the preservation of types. Its asymmetric nature will prove to be no obstacle; it is overcome by splitting the interpretation into two steps. We perform first a compositional translation into $\lambda\mu$ which mimics the double negation translation in the sense of Plotkin's CBN, and thus alter the types as well. The second step consists in recovering the types, which can only be done globally. This first result implies that we can faithfully interpret $\mathcal{X}$ into $\lambda\mu$, the second that typeability is fully preserved.

A natural idea would be to start from our former λ-calculus interpretations (see the previous section) to build this interpretation. However, this is not going to help much, as there the two kinds of variables have been merged. Instead, we need to handle circuits directly, which means we are actually working out a genuine new interpretation.

Let us focus on the CBN-interpretation; the CBV one comes along very similar guidelines. Towards our main theorem, sockets are translated into ordinary λ-variables, and plugs are embedded into μ-variables; for both, we keep the same name, to ease the reading. We interpret both ordinary and right cuts the same way, while the left cut still requires switching the position of the sub-terms, as the reduction rules in the (standard) $\lambda\mu$-calculus are intrinsically call-by-name. Therefore, the flipping mechanism, used above to left-activate a cut or to de-activate a left-cut, is still required for the interpretation to hold along the bi-directional propagation rules from $\mathcal{X}$.

The formal definitions given below should be self-explanatory, except from the observation that a double-negation step is still used. This point deserves an explanation. The circuit $P\widehat{\alpha} \dagger \widehat{x}Q$ for which we know

$$P\widehat{\alpha} \dagger \widehat{x}Q \;:\!\cdot\; \Gamma \vdash \Delta$$

is translated into a command $[\omega]((\lambda x.M)\,(\mu\alpha.C))$ for some $\lambda\mu$-term M and command C and some μ-variable ω. The translation cannot be an ordinary term for both a syntactic reason and a semantic one. Since we target the standard $\lambda\mu$-calculus, the interpretation of the sub-circuit P has to be a command, otherwise the μ-binding cannot be defined; as a matter of consequence, the translation of Q is going to be of the shape $[\omega']M'$, which means that $M = \mu\eta.[\omega']M'$ for some μ-variables η, ω'.

From the $\lambda\mu$-calculus perspective, this observation means we are switching contexts intensively. This provides actually a semantic view on the way the $\mathcal{X}$-calculus manages derivations trees. The cut shows no privileged assumption nor conclusion, just connecting P and Q circuits. The lack of a selected assumption is reflected by the interpretation not to be an abstraction; also, the lack of any particular conclusion has the consequence we cannot use a μ binder at this point. Therefore, cuts in $\mathcal{X}$ correspond to proofs handled at the level of commands in $\lambda\mu$.

At this stage, we could have choosen to treat commands as proofs for the $\perp$ negation type [14, 9]. This would have been a fair choice, since $\lambda\mu$ allows us to build a proof in any type from such a proof. However, the translation itself can carry out a more elegant and precise information. The definition that we propose shows that, given an arbitrary, initial choice of a named conclusion $\omega{:}\Omega$, building circuits into $\mathcal{X}$ can be read as syntax rules for dealing with judgments, by switching over the targeted conclusion Ω as a *pivot*. (Then we can have $\omega = \omega'$ in the previous informal analysis, and the formal definition.)

As a matter of consequence, we get that the interpretation of any given proof net can be built by targeting one conclusion $\alpha{:}A$ among its non-empty set of conclusions, leading to a $\lambda\mu$-term M which expresses the view that the proof for A is done by switching back and forth with the conclusion.

Paradigmatic examples of such proofs are provided by Peirce's law or the *ex falso quodlibet* formula $\neg\neg T \to T$. Our interpretation shows that this mechanism is general, at least as far as the restricted fragment of sequent calculus $\mathcal{X}$ deals with can tell.

6.1 Call by Name

As for the case of the λ-calculus, our results will depend not only on an interpretation of terms, but also on one for types. Let Ω be, as before, any (fixed) type and denote $\neg T \equiv T{\to}\Omega$ for convenience. The following lemma will provide the necessary trick.

The following lemma will provide the necessary flippng trick which comes along switching propagation directions.

There exists two $\lambda\mu$-terms *force* and *delay* such that

$$\textit{force} \circ \textit{delay} \;=_{\mathrm{N}}\; I =_{\mathrm{V}} I$$

and for all type T, *force* : $\neg\neg T{\to}T$ and *delay* : $T{\to}\neg\neg T$. Take

$$\begin{aligned} \textit{force}\, F &\triangleq \mu\tau.[\omega](F\,\lambda t.\mu !.[\tau]t) \triangleq F^{-} \\ \textit{delay}\, t &\triangleq \lambda f.f\,t \triangleq (t)^{\star} \end{aligned}$$

Definition 6.1 PLOTKIN'S CBN. The CBN interpretation of type T, denoted $\ulcorner T \urcorner_{\mathrm{N}} \triangleq \neg\neg \ulcorner T \urcorner_{\mathrm{N}}'$ is defined inductively by

$$\begin{aligned} \ulcorner X \urcorner_{\mathrm{N}}' &\triangleq X, \qquad\qquad X \textit{ type variable} \\ \ulcorner A{\to}B \urcorner_{\mathrm{N}}' &\triangleq \ulcorner A \urcorner_{\mathrm{N}} {\to} \ulcorner B \urcorner_{\mathrm{N}} \end{aligned}$$

Also $\ulcorner \Gamma, x{:}T \urcorner_{\mathrm{N}} \triangleq \ulcorner \Gamma \urcorner_{\mathrm{N}}, x{:}\ulcorner T \urcorner_{\mathrm{N}}$.

Type recovery is possible, owing to the following result.

Lemma 6.2 For any type T, there exist $\varphi_T : \ulcorner T \urcorner_{\mathrm{N}}{\to}T$ and $\psi_T : T{\to}\ulcorner T \urcorner_{\mathrm{N}}$.

The first step of the interpretation is type-free, athough our definition aims at complying with types later.

The notation $\mu!.C$ is a shortcut for $\mu\eta.C$ where η is a fresh μ-variable wrt C.

Definition 6.3 CALL BY NAME. Let

$$\begin{aligned} \ulcorner\langle x.\alpha\rangle\urcorner_{\mathrm{N}}' &\triangleq \lambda v.\mu!.[\alpha](v)^\star \\ \ulcorner P\widehat{\beta}\,[x]\,\widehat{y}Q\urcorner_{\mathrm{N}}' &\triangleq \lambda v.\mu!.[\omega](\lambda y.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,(v\,\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})) \end{aligned}$$

For P any $\mathcal{X}$-term, we define $\ulcorner P\urcorner_{\mathrm{N}}$ by structural induction

$$\begin{aligned} \ulcorner\langle x.\alpha\rangle\urcorner_{\mathrm{N}} &\triangleq [\omega](x\,\ulcorner\langle x.\alpha\rangle\urcorner_{\mathrm{N}}') \\ \ulcorner\widehat{y}P\widehat{\beta}\cdot\alpha\urcorner_{\mathrm{N}} &\triangleq [\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})^\star \\ \ulcorner P\widehat{\beta}\,[x]\,\widehat{y}Q\urcorner_{\mathrm{N}} &\triangleq [\omega](x\,\ulcorner P\widehat{\beta}\,[x]\,\widehat{y}Q\urcorner_{\mathrm{N}}') \\ \ulcorner P\widehat{\alpha}\dagger\widehat{x}Q\urcorner_{\mathrm{N}} \triangleq \ulcorner P\widehat{\alpha}\not\nwarrow\widehat{x}Q\urcorner_{\mathrm{N}} &\triangleq [\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,\mu\alpha.\ulcorner P\urcorner_{\mathrm{N}}) \\ \ulcorner P\widehat{\alpha}\not\nearrow\widehat{x}Q\urcorner_{\mathrm{N}} &\triangleq [\omega](\mu\alpha.\ulcorner P\urcorner_{\mathrm{N}}\,\ulcorner)\urcorner_{\mathrm{N}}'Q \end{aligned}$$

Proposition 6.4 CONSERVATION OF TYPES IN CBN. $P \;\vdots\cdot\; \Gamma \vdash \Delta$*, then* $\ulcorner\Gamma\urcorner_{\mathrm{N}} \vdash_{\lambda\mu} \ulcorner P\urcorner_{\mathrm{N}} \mid \omega{:}\Omega, \ulcorner\Delta\urcorner_{\mathrm{N}}$.

Theorem 6.5 *For any* $P \;\vdots\cdot\; \Gamma \vdash \Delta$ *in* $\mathcal{X}$*, and type* Ω *there exists a* $\lambda\mu$*-term* $\ulcorner P\urcorner_{\mathrm{N}}^{\Omega}$ *such that* $\Gamma \vdash_{\lambda\mu} \ulcorner P\urcorner_{\mathrm{N}}^{\Omega}{:}\Omega \mid \Delta$.

Theorem 6.6 *For all* $\mathcal{X}$*-terms* P, P'*, if* $P \to_{\mathrm{N}}{}^{*} P'$*, then* $\ulcorner P\urcorner_{\mathrm{N}}^{\Omega} =_{\mathrm{N}} \ulcorner P'\urcorner_{\mathrm{N}}^{\Omega}$.

6.2 Call by Value

Definition 6.7 PLOTKIN'S CBV. The CBV interpretation of type T, denoted $\ulcorner T\urcorner_{\mathrm{V}} \triangleq \neg\neg\ulcorner T\urcorner_{\mathrm{V}}'$ is defined inductively by

$$\begin{aligned} \ulcorner X\urcorner_{\mathrm{V}}' &\triangleq X, && X \textit{ type variable} \\ \ulcorner A\to B\urcorner_{\mathrm{V}}' &\triangleq \ulcorner A\urcorner_{\mathrm{V}}'\to\ulcorner B\urcorner_{\mathrm{V}} \end{aligned}$$

If Δ is a typing judgement, and define accordingly $\ulcorner\Delta, x{:}T\urcorner_{\mathrm{V}} \triangleq \ulcorner\Gamma\urcorner_{\mathrm{V}}, x{:}\ulcorner T\urcorner_{\mathrm{V}}$ and $\ulcorner\Delta, x{:}T\urcorner_{\mathrm{V}}' \triangleq \ulcorner\Gamma\urcorner_{\mathrm{V}}, x{:}\ulcorner T\urcorner_{\mathrm{V}}'$.

Definition 6.8 CALL BY VALUE. For P any $\mathcal{X}$-term, we define $\ulcorner P\urcorner_{\mathrm{V}}$ by structural induction:

$$\begin{aligned} \ulcorner\langle x.\alpha\rangle\urcorner_{\mathrm{V}} &\triangleq [\alpha](x)^\star \\ \ulcorner\widehat{y}P\widehat{\beta}\cdot\alpha\urcorner_{\mathrm{V}} &\triangleq [\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}})^\star \\ \ulcorner P\widehat{\beta}\,[x]\,\widehat{y}Q\urcorner_{\mathrm{V}} &\triangleq \\ &\quad [\omega](\mu\beta.\ulcorner P\urcorner_{\mathrm{V}}\,\lambda v.\mu!.[\omega](x\,v\,\lambda y.\mu!.\ulcorner Q\urcorner_{\mathrm{V}})) \\ \ulcorner P\widehat{\alpha}\dagger\widehat{x}Q\urcorner_{\mathrm{V}} \triangleq \ulcorner P\widehat{\alpha}\not\nearrow\widehat{x}Q\urcorner_{\mathrm{V}} &\triangleq [\omega](\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}}\,\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}) \\ \ulcorner P\widehat{\alpha}\not\nwarrow\widehat{x}Q\urcorner_{\mathrm{V}} &\triangleq [\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}\,\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}}{}^{-}) \end{aligned}$$

Proposition 6.9 CONSERVATION OF TYPES IN CBV. *If* $P \;\vdots\cdot\; \Delta \vdash \Gamma$*, then* $\ulcorner\Gamma\urcorner_{\mathrm{V}}' \vdash_{\lambda\mu} \ulcorner P\urcorner_{\mathrm{V}} \mid \omega{:}\Omega, \ulcorner\Delta\urcorner_{\mathrm{V}}$.

Lemma 6.10 If α *is introduced in* P *and* V *any value, then*

$$\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}}\,V \;=_{\mathrm{V}}\; V\,(\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}$$

The last result of this section is:

Theorem 6.11 *For all* $\mathcal{X}$*-terms* P, P'*, if* $P \to_{\mathrm{V}}{}^{*} P'$*, then* $\ulcorner P\urcorner_{\mathrm{V}}^{\Omega} =_{\mathrm{V}} \ulcorner P'\urcorner_{\mathrm{V}}^{\Omega}$.

7 Main results

Using the results achieved above, we can now link provability in LK($\to$) with typability in $\lambda\mu$, and formulate the main result of this paper:

Theorem 7.1 COMPLETENESS. *If* $\Gamma \vdash_{\mathrm{LK}} \Delta$ *in* LK($\to$)*, then for any (fresh) type* Ω*, there exists a* $\lambda\mu$*-term* M*, and contexts* Γ', Δ' *such that* $\Gamma' \vdash_{\lambda\mu} M{:}\Omega \mid \Delta'$ *such that* Γ, Δ *can be obtained from* Γ', Δ' *by erasure of names.*

We also prove:

Theorem 7.2 CONSISTENCY. *If* P, P' *are proofs in* LK($\to$)*, such that* P *can be changed into* P' *performing cut-elimination steps that correspond to* $\mathcal{X}$ *reduction rules, then their interpretations in* $\lambda\mu$ *are equivalent, i.e. share a common reduct.*

Conclusion and Future work

The $\mathcal{X}$-calculus offers a correspondence *à la* Curry-Howard-de Bruijn for the implicational fragment of LK, bringing together various features from [17, 5] with a particular emphasis on local bi-directional propagation rules which describe transformations on derivations. Since $\mathcal{X}$ deals with classical logic, one cannot ensure confluence from taking into account the whole set of reduction rules. Two sub-reduction systems can be described to avoid critical pairs, reflecting Call by Name and Call by Value in $\mathcal{X}$.

On the other hand, $\lambda\mu$ has been designed from a fine analysis of which transformations could be applied on derivations, and has limited itself to a confluent extension of the λ-calculus, presenting classical logic in natural deduction form.

This paper has focused on the intepretation of $\mathcal{X}$ into $\lambda\mu$ (in its standard presentation) toward a stronger understanding of $\mathcal{X}$ and its comparison with the former, as a better understood calculus. Given one of the CBV or CBN strategies, our interpretation proceeds internally through a(n intermediate) CPS translation on types and terms, and transforms a proof net into a $\lambda\mu$-term which structure generalises those found in proofs for the Pierce law, or ex falso quodlibet.

We have shown this interpretation to be consistent with equality in the target calculus, and proved a completeness result for $\lambda\mu$ in the process. As expected, trying out such an interpretation in the pure λ-calculus cannot provides the same benefits; however, it is worth noticing the simulation of propagation rules does not require as complex constructions as in $\lambda\mu$. This can be understood from the fact we would be able to switch from one conclusion to another, as the latter allows by design. In fact, the interpretation keeps the focus on a conclusion, that we named Ω, which is the usual output type in the usual CPS transformations. On the other hand, the interpretation in $\lambda\mu$ requires that we encompass the evaluation model found in this calculus, strictly Call by Name in its design. It is a noticeable result that, nevertheless, we do not need further extension of $\lambda\mu$ for propagation rules defined in $\mathcal{X}$ to be faithfully reflected in the target calculus.

This work leaves space for some important issues we want to address as a consequence of this analysis. An extension of $\lambda\mu$-calculus towards a more symmetric one has been proposed for some time, and studied in [6] with respect to normalisation. Although confluence is lost, the reader might have already noticed a more simpler interpretation that comes to mind where terms switching position in the interpretation of cuts could be handled with more straightforward design. Symmetry is the main issue going behind all these calculi, and we expect to have provided some new light on the computational behaviour of classical proofs. Meanwhile, Barbanera and Berardi's symmetric calculus [2] and Wadler's analysis of duality [18] should be investigated further within the context of this paper towards a finer understanding in this area.

References

[1] S. Bakel, S. Lengrand, and P. Lescanne. The language $\mathcal{X}$: circuits, computations and classical logic. In Mario Coppo, Elena Lodi, and G. Michele Pinna, editors, *Proceedings of Ninth Italian Conference on Theoretical Computer Science (ICTCS'05), Siena, Italy*, volume 3701 of *Lecture Notes in Computer Science*, pages 81–96. Springer-Verlag, 2005.

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

[3] H. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. North-Holland, Amsterdam, revised edition, 1984.

[4] N. G. Bruijn. A namefree lambda calculus with facilities for internal definition of expressions and segments. TH-Report 78-WSK-03, Technological University Eindhoven, Netherlands, Department of Mathematics, 1978.

[5] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *Proceedings of the 5 th ACM SIGPLAN International Conference on Functional Programming (ICFP'00)*, pages 233–243. ACM, 2000.

[6] R. David and K. Nour. Arithmetical proofs of strong normalization resultsfor the symmetric lambda-mu-calculus. In *TLCA*, number 3461 in Lecture Notes in Computer Science, 2005.

[7] G. Gentzen. Untersuchungen über das Logische Schliessen. *Mathematische Zeitschrift*, 39:176–210 and 405–431, 1935. English translation in [16], Pages 68–131.

[8] T. Griffin. A formulae-as-types notion of control. In *Proceedings of the 17th Annual ACM Symposium on Principles Of Programming Languages, Orlando (Fla., USA)*, pages 47–58, 1990.

[9] Ph. Groote. On the relation between the $\lambda\mu$-calculus and the syntactic theory of sequential control. In Springer-Verlag, editor, *Proceedings of the (th International Conference on Logic Programming and Automated Reasoning, (LPAR'94)*, volume 822 of *Lecture Notes in Computer Science*, pages 31–43, 1994.

[10] H. Herbelin. *Séquents qu'on calcule : de l'interprétation du calcul des séquents comme calcul de λ-termes et comme calcul de stratégies gagnantes*. Thèse d'université, Université Paris 7, Janvier 1995.

[11] H. Herbelin. *C'est maintenant qu'on calcule: au cœur de la dualité*. Mémoire de habilitation, Université Paris 11, Décembre 2005.

[12] Jean-Louis Krivine. Classical logic, storage operators and second-order lambda-calculus. *Ann. Pure Appl. Logic*, 68(1):53–78, 1994.

[13] Stéphane Lengrand. Call-by-value, call-by-name, and strong normalization for the classical sequent calculus. In Bernhard Gramlich and Salvador Lucas, editors, *Electronic Notes in Theoretical Computer Science*, volume 86. Elsevier, 2003.

[14] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *Proceedings of the 24th Annual ACM Symposium on Principles Of Programming Languages, Paris (France)*, pages 215–227, 1997.

[15] 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 *Lecture Notes in Computer Science*, pages 190–201. Springer-Verlag, 1992.

[16] M. E. Szabo, editor. *The Collected Papers of Gerhard Gentzen*. Studies in Logic and the Foundations of Mathematics. North-Holland, 1969.

[17] Christian Urban. *Classical Logic and Computation*. PhD thesis, University of Cambridge, October 2000.

[18] Philip Wadler. Call-by-Value is Dual to Call-by-Name. In *Proceedings of the eighth ACM SIGPLAN international conference on Functional programming*, pages 189 – 201, 2003.

[19] A N. Whitehead and B. Russell. *Principia Mathematica*. Cambridge University Press, 2nd edition, 1925.

A Proofs for the main results

Theorem 5.5 *If $P \rightarrow_{\mathrm{N}} Q$, then $\ulcorner P\urcorner^{\lambda}_{\mathrm{N}} =_{\mathrm{N}} \ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}$.*

Proof: By induction on the length of the reduction path. We just check the rules, of which we show the interesting cases.

$$\begin{array}{rcl}
(\textit{exp-imp}) : & & \ulcorner(\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha} \dagger \widehat{x}(Q\widehat{\gamma}\,[x]\,\widehat{z}R)\urcorner^{\lambda}_{\mathrm{N}} \quad \triangleq \\
& & (\lambda x.x\,\lambda u.(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(u\,\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}))(\lambda\alpha.\alpha\,\lambda y\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) \\
& =_{\mathrm{N}} & (\lambda\alpha.\alpha\,\lambda y\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda u.(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(u\,\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})) \\
(\alpha \notin \ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) & =_{\mathrm{N}} & (\lambda u.(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(u\,\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}))(\lambda y\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) \\
& =_{\mathrm{N}} & (\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})((\lambda y\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})) \\
& =_{\mathrm{N}} & (\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}/y\rangle) \\
& =_{\mathrm{N}} & (\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(\lambda\beta.(\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})) \\
& \triangleq & \ulcorner(Q\widehat{\gamma} \dagger \widehat{y}P)\widehat{\beta} \dagger \widehat{z}R\urcorner^{\lambda}_{\mathrm{N}}
\end{array}$$

$$\begin{array}{rcl}
\text{Also:} & & (\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}/y\rangle) \\
& =_{\mathrm{N}} & \ulcorner R\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}/y\rangle/z\rangle \\
(y \notin R, \beta \notin Q) & = & \ulcorner R\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}/z\rangle\langle\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}/y\rangle \\
& =_{\mathrm{N}} & (\lambda y.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}/z\rangle)(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}) \\
& =_{\mathrm{N}} & (\lambda y.(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{N}})(\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}))(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}) \\
& \triangleq & \ulcorner Q\widehat{\gamma} \dagger \widehat{y}(P\widehat{\beta} \dagger \widehat{z}R)\urcorner^{\lambda}_{\mathrm{N}}
\end{array}$$

$(\textit{act-}\mathrm{L})$: Notice that either $Q = \langle x.\beta\rangle$, or $Q = Q_1\widehat{\beta}\,[x]\,\widehat{y}Q_2$, with $x \notin \mathit{fs}(Q_1, Q_2)$, so we can use $\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}} \triangleq x\,\lambda u.\ulcorner Q\urcorner_{\mathrm{A}}[u/x]$, in all cases below.

$$\begin{array}{rcl}
\ulcorner P\widehat{\alpha} \dagger \widehat{x}Q\urcorner^{\lambda}_{\mathrm{N}} & \triangleq & (\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}) \\
& =_{\mathrm{N}} & \ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}/x\rangle \\
& \triangleq & (x\,\lambda u.\ulcorner Q\urcorner_{\mathrm{A}}[u/x])\langle\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}}/x\rangle \\
& = & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda u.\ulcorner Q\urcorner_{\mathrm{A}}[u/x]) \\
(\alpha) & =_{\mathrm{N}} & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner Q\urcorner_{\mathrm{A}}) \\
& \triangleq & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner Q\urcorner_{\mathrm{A}}) \\
& \triangleq & \ulcorner P\widehat{\alpha} \not\dagger \widehat{x}Q\urcorner^{\lambda}_{\mathrm{N}}
\end{array}$$

$$\begin{array}{rcl}
(d\mathrm{L}) : & & \ulcorner\langle y.\alpha\rangle\widehat{\alpha} \not\dagger \widehat{x}P\urcorner^{\lambda}_{\mathrm{N}} \\
& \triangleq & (\lambda\alpha.y\lambda u.\alpha u)(\lambda x.\ulcorner P\urcorner_{\mathrm{A}}) \\
& =_{\mathrm{N}} & y\lambda u.(\lambda x.\ulcorner P\urcorner_{\mathrm{A}})u \\
& =_{\mathrm{N}} & (\lambda\alpha.y\lambda u.\alpha u)(\lambda x.\ulcorner P\urcorner_{\mathrm{A}}) \\
& =_{\mathrm{N}} & (\lambda v.v\lambda x.\ulcorner P\urcorner_{\mathrm{A}})(\lambda\alpha.y\lambda u.\alpha u) \\
(\alpha) & =_{\mathrm{N}} & (\lambda x.x\lambda v.\ulcorner P\urcorner_{\mathrm{A}}[v/x])(\lambda\alpha.y\lambda u.\alpha u) \\
& \triangleq & (\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda\alpha.y\lambda u.\alpha u) \\
& \triangleq & \ulcorner\langle y.\alpha\rangle\widehat{\alpha} \dagger \widehat{x}P\urcorner^{\lambda}_{\mathrm{N}}
\end{array}$$

$$\begin{array}{rl}
(\textit{exp-outs}\not\dagger) : & \ulcorner(\widehat{y}Q\widehat{\beta}\cdot\alpha)\widehat{\alpha} \not\dagger \widehat{x}P\urcorner^{\lambda}_{\mathrm{N}} \\
\triangleq & (\lambda\alpha.\alpha\,\lambda y\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}}) \\
=_{\mathrm{N}} & (\alpha\,\lambda y\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})\langle\lambda x.\ulcorner P\urcorner_{\mathrm{A}}/\alpha\rangle \\
= & (\lambda x.\ulcorner P\urcorner_{\mathrm{A}})(\lambda y\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}}\langle\lambda x.\ulcorner P\urcorner_{\mathrm{A}}/\alpha\rangle) \\
=_{\mathrm{N}} & (\lambda x.\ulcorner P\urcorner_{\mathrm{A}})(\lambda y\beta.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}})) \\
=_{\mathrm{N}} & (\lambda\delta.\delta\,\lambda y\beta.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}}))(\lambda x.\ulcorner P\urcorner_{\mathrm{A}}) \\
=_{\mathrm{N}} & (\lambda u.u\,\lambda x.\ulcorner P\urcorner_{\mathrm{A}})(\lambda\delta.\delta\,\lambda y\beta.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}})) \\
(\alpha) =_{\mathrm{N}} & \\
& (\lambda x.x\lambda u.\ulcorner P\urcorner_{\mathrm{A}}[u/x])(\lambda\delta.\delta\,\lambda y\beta.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}})) \\
=_{\mathrm{N}} & (\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{N}})(\lambda\delta.\delta\,\lambda y\beta.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{N}})(\lambda x.\ulcorner P\urcorner_{\mathrm{A}})) \\
\triangleq & \ulcorner(\widehat{y}(Q\widehat{\alpha} \not\dagger \widehat{x}P)\widehat{\beta}\cdot\delta)\widehat{\delta} \dagger \widehat{x}P\urcorner^{\lambda}_{\mathrm{N}}
\end{array}$$

Theorem 5.10 *If $P \rightarrow_{\mathrm{V}} Q$, then $\ulcorner P\urcorner^{\lambda}_{\mathrm{V}} =_{\mathrm{V}} \ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}$.*

Proof: By induction on the length of the reduction path. We just check the rules.

$$\begin{array}{rcl}
(\textit{exp-imp}) : & & \ulcorner(\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha} \dagger \widehat{x}(Q\widehat{\gamma}\,[x]\,\widehat{z}R)\urcorner^{\lambda}_{\mathrm{V}} \\
& \triangleq & (\lambda\alpha.\alpha\,\lambda\beta y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(x\,\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}})) \\
(\alpha \notin \ulcorner P\urcorner^{\lambda}_{\mathrm{V}}) & =_{\mathrm{V}} & (\lambda x.(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(x\,\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}))(\lambda\beta.\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}) \\
& =_{\mathrm{V}} & (\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})((\lambda\beta y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}})) \\
& =_{\mathrm{V}} & (\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda y.(\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}/\beta\rangle)) \\
& =_{\mathrm{V}} & (\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda y.(\lambda\beta.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}})) \\
& \triangleq & \ulcorner Q\widehat{\gamma} \dagger \widehat{y}(P\widehat{\beta} \dagger \widehat{z}R)\urcorner^{\lambda}_{\mathrm{V}}
\end{array}$$

$$\begin{array}{rcl}
\text{Also:} & & (\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}/\beta\rangle) \\
& =_{\mathrm{V}} & \ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}/\beta\rangle/\gamma\rangle \\
(y \notin R, \beta \notin Q) & = & \ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}/\gamma\rangle\langle\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}/\beta\rangle \\
& =_{\mathrm{V}} & (\lambda\beta.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}/\gamma\rangle)(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}) \\
& =_{\mathrm{V}} & (\lambda\beta.(\lambda\gamma.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda y.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}))(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}) \\
& \triangleq & (\lambda\beta.\ulcorner Q\widehat{\gamma} \dagger \widehat{y}P\urcorner^{\lambda}_{\mathrm{V}})(\lambda z.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}) \\
& \triangleq & \ulcorner(Q\widehat{\gamma} \dagger \widehat{y}P)\widehat{\beta} \dagger \widehat{z}R\urcorner^{\lambda}_{\mathrm{V}}
\end{array}$$

$(\textit{act-}\mathrm{R})$: $P\widehat{\alpha} \dagger \widehat{x}Q \rightarrow P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q$, x not introduced.
Notice that either $P = \langle y.\alpha\rangle$, or $P = \widehat{y}R\widehat{\beta}\cdot\alpha$, with $\alpha \notin \mathit{fp}(R)$, so we can use $\ulcorner P\urcorner^{\lambda}_{\mathrm{V}} \triangleq \alpha\,\ulcorner P\urcorner_{\mathrm{A}}$, in all cases below.

$$\begin{array}{rcl}
\ulcorner\langle y.\alpha\rangle\widehat{\alpha} \dagger \widehat{x}Q\urcorner^{\lambda}_{\mathrm{V}} & \triangleq & (\lambda\alpha.\ulcorner\langle y.\alpha\rangle\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}) \\
& \triangleq & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}) \\
& \triangleq & (\lambda\alpha.\alpha\ulcorner P\urcorner_{\mathrm{A}})(\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}) \\
& =_{\mathrm{V}} & (\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})\,\ulcorner P\urcorner_{\mathrm{A}} \\
& \triangleq & \ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q\urcorner^{\lambda}_{\mathrm{V}}
\end{array}$$

$$\begin{array}{rcl}
(d\mathrm{R}) : \ \ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{V}} & \triangleq & (\lambda x.\ulcorner\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{V}})\,\ulcorner P\urcorner_{\mathrm{A}} \\
& =_{\mathrm{V}} & (\lambda\alpha.\alpha\,\ulcorner P\urcorner_{\mathrm{A}})(\lambda x.\ulcorner\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{V}}) \\
& = & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{V}}) \\
& \triangleq & \ulcorner P\widehat{\alpha} \dagger \widehat{x}\langle x.\beta\rangle\urcorner^{\lambda}_{\mathrm{V}}
\end{array}$$

$$\begin{array}{rl}
(\backslash\!\!\!\dagger\,\textit{imp-outs}) : & \ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}(Q\widehat{\nu}\,[x]\,\widehat{y}R)\urcorner^{\lambda}_{\mathrm{V}} \\
\triangleq & (\lambda x.\ulcorner Q\widehat{\nu}\,[x]\,\widehat{y}R\urcorner^{\lambda}_{\mathrm{V}})\ulcorner P\urcorner_{\mathrm{A}} \\
\triangleq & (\lambda x.(\lambda\nu.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(x\,\lambda y.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}))\ulcorner P\urcorner_{\mathrm{A}} \\
=_{\mathrm{V}} & (\lambda\nu.(\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}\langle\ulcorner P\urcorner_{\mathrm{A}}/x\rangle))(\ulcorner P\urcorner_{\mathrm{A}}\,\lambda y.(\ulcorner R\urcorner^{\lambda}_{\mathrm{V}}\langle\ulcorner P\urcorner_{\mathrm{A}}/x\rangle)) \\
=_{\mathrm{V}} & (\lambda\nu.((\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})\ulcorner P\urcorner_{\mathrm{A}}))(\ulcorner P\urcorner_{\mathrm{A}}\,\lambda y.((\lambda x.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}})\ulcorner P\urcorner_{\mathrm{A}})) \\
=_{\mathrm{V}} & (\lambda z.(\lambda\nu.((\lambda x.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})\ulcorner P\urcorner_{\mathrm{A}}))(z\,\lambda y.((\lambda x.\ulcorner R\urcorner^{\lambda}_{\mathrm{V}})\ulcorner P\urcorner_{\mathrm{A}})))\ulcorner P\urcorner_{\mathrm{A}} \\
=_{\mathrm{V}} & (\lambda\alpha.\alpha\ulcorner P\urcorner_{\mathrm{A}})(\lambda z.(\lambda\nu.\ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q\urcorner^{\lambda}_{\mathrm{V}})(z\,\lambda y.\ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}R\urcorner^{\lambda}_{\mathrm{V}})) \\
= & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda z.(\lambda\nu.\ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q\urcorner^{\lambda}_{\mathrm{V}})(z\,\lambda y.\ulcorner P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}R\urcorner^{\lambda}_{\mathrm{V}})) \\
\triangleq & (\lambda\alpha.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})(\lambda z.\ulcorner(P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q)\widehat{\nu}\,[z]\,\widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}R)\urcorner^{\lambda}_{\mathrm{V}}) \\
\triangleq & \ulcorner P\widehat{\alpha} \dagger \widehat{z}((P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}Q)\widehat{\nu}\,[z]\,\widehat{y}(P\widehat{\alpha} \backslash\!\!\!\dagger\, \widehat{x}R))\urcorner^{\lambda}_{\mathrm{V}}
\end{array}$$

(*exp-outs*$\not\dagger$) : $\ulcorner(\widehat{y}Q\widehat{\beta}\cdot\alpha)\widehat{\alpha}\not\dagger\widehat{x}P\urcorner^{\lambda}_{\mathrm{V}}$

$$\begin{aligned}
&\triangleq (\lambda\alpha.\alpha\,\lambda\beta y.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})\\
&=_{\mathrm{V}} (\alpha\,\lambda\beta y.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})\langle\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}/\alpha\rangle\\
&=_{\mathrm{V}} (\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})\,\lambda\beta y.(\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}}\langle\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}/\alpha\rangle)\\
&=_{\mathrm{V}} (\lambda\delta.\delta\,\lambda\beta y.(\lambda\alpha.\ulcorner Q\urcorner^{\lambda}_{\mathrm{V}})(\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}}))(\lambda x.\ulcorner P\urcorner^{\lambda}_{\mathrm{V}})\\
&\triangleq \ulcorner(\widehat{y}(Q\widehat{\alpha}\not\dagger\widehat{x}P)\widehat{\beta}\cdot\delta)\widehat{\delta}\dagger\widehat{x}P\urcorner^{\lambda}_{\mathrm{V}}
\end{aligned}$$

Theorem 6.6 *For all $\mathcal{X}$-circuits P, P', if $P \rightarrow_{\mathrm{N}}{}^{*} P'$, then* $\ulcorner P\urcorner^{\Omega}_{\mathrm{N}} =_{\mathrm{N}} \ulcorner P'\urcorner^{\Omega}_{\mathrm{N}}$.

Proof: By induction on the length of reduction sequences; we only consider the base case, for which we show the more interesting cases.

(*exp-imp*) : $\ulcorner(\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha}\dagger\widehat{x}(Q\widehat{\gamma}\,[x]\,\widehat{z}R)\urcorner_{\mathrm{N}}$

$$\begin{aligned}
&\triangleq [\omega](\lambda x.\mu!.[\omega](x\,\ulcorner Q\widehat{\gamma}\,[x]\,\widehat{z}R\urcorner_{\mathrm{N}}{}')\,\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})^{\star})\\
&=_{\mathrm{N}} [\omega]\mu!.[\omega](\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})^{\star}\,\ulcorner Q\widehat{\gamma}\,[x]\,\widehat{z}R\urcorner_{\mathrm{N}}{}')\\
&=_{\mathrm{N}} [\omega](\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})^{\star}\,\ulcorner Q\widehat{\gamma}\,[x]\,\widehat{z}R\urcorner_{\mathrm{N}}{}')\\
&=_{\mathrm{N}} [\omega]((\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})^{\star}\,\ulcorner Q\widehat{\gamma}\,[x]\,\widehat{z}R\urcorner_{\mathrm{N}}{}')\\
&=_{\mathrm{N}} [\omega](\ulcorner Q\widehat{\gamma}\,[x]\,\widehat{z}R\urcorner_{\mathrm{N}}{}'\,\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}})\\
&=_{\mathrm{N}} [\omega]\mu!.[\omega](\lambda z.\mu!.\ulcorner R\urcorner_{\mathrm{N}}\,\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}}\,\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}})\\
&=_{\mathrm{N}} [\omega](\lambda z.\mu!.\ulcorner R\urcorner_{\mathrm{N}}\,\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}}\,\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}})\\
&=_{\mathrm{N}} [\omega]\mu!.(\ulcorner R\urcorner_{\mathrm{N}})\,\langle\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}}\,\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}}/z\rangle\\
&=_{\mathrm{N}} (\ulcorner R\urcorner_{\mathrm{N}})\,\langle\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{N}}\,\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}}/z\rangle\\
&=_{\mathrm{N}} (\ulcorner R\urcorner_{\mathrm{N}})\,\langle\mu\beta.(\ulcorner P\urcorner_{\mathrm{N}})\,\langle\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}}/y\rangle/z\rangle\\
&=_{\mathrm{N}} (\ulcorner R\urcorner_{\mathrm{N}})\,\langle\mu\beta.[\omega]\mu!.(\ulcorner P\urcorner_{\mathrm{N}})\,\langle\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}}/y\rangle/z\rangle\\
&=_{\mathrm{N}} (\ulcorner R\urcorner_{\mathrm{N}})\,\langle\mu\beta.[\omega](\lambda y.\mu!.\ulcorner P\urcorner_{\mathrm{N}}\,\mu\gamma.\ulcorner Q\urcorner_{\mathrm{N}})/z\rangle\\
&=_{\mathrm{N}} [\omega]\mu!.(\ulcorner R\urcorner_{\mathrm{N}})\,\langle\mu\beta.\ulcorner Q\widehat{\gamma}\dagger\widehat{y}P\urcorner_{\mathrm{N}}/z\rangle\\
&=_{\mathrm{N}} [\omega](\lambda z.\mu!.\ulcorner R\urcorner_{\mathrm{N}}\,\mu\beta.\ulcorner Q\widehat{\gamma}\dagger\widehat{y}P\urcorner_{\mathrm{N}})\\
&\triangleq \ulcorner(Q\widehat{\gamma}\dagger\widehat{y}P)\widehat{\beta}\dagger\widehat{z}R\urcorner_{\mathrm{N}}
\end{aligned}$$

(*act*-L) : x introduced and α not introduced,

$$\begin{aligned}
&\ulcorner P\widehat{\alpha}\dagger\widehat{x}Q\urcorner_{\mathrm{N}}\\
&\quad\triangleq [\omega]((\lambda x.\mu!.[\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,(x)^{\star}))^{\star}\,\mu\alpha.\ulcorner P\urcorner_{\mathrm{N}})\\
&\quad=_{\mathrm{N}} [\omega](\mu\alpha.\ulcorner P\urcorner_{\mathrm{N}}\,\lambda x.\mu!.[\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,(x)^{\star}))\\
&\quad=_{\mathrm{N}} \ulcorner P\urcorner_{\mathrm{N}}\langle\lambda x.\mu!.[\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,(x)^{\star})\cdot\sigma/\alpha\rangle\\
&\quad=_{\mathrm{N}} [\omega](\mu\alpha.\ulcorner P\urcorner_{\mathrm{N}}\,\lambda x.\mu!.[\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{N}}\,(x)^{\star}))\\
&\quad\triangleq \ulcorner P\widehat{\alpha}\not\dagger\widehat{x}Q\urcorner_{\mathrm{N}}
\end{aligned}$$

(dL) : Let $F \triangleq \lambda x.\mu!.[\omega](\lambda x.\mu!.\ulcorner P\urcorner_{\mathrm{N}}\,(x)^{\star})$ in

$$\begin{aligned}
&\ulcorner\langle y.\alpha\rangle\widehat{\alpha}\not\dagger\widehat{x}P\urcorner_{\mathrm{N}}\\
&\quad\triangleq [\omega](\mu\alpha.[\omega](y\,\lambda v_0.\mu!.[\alpha](v_0)^{\star})\,F)\\
&\quad=_{\mathrm{N}} [\omega](y\,\lambda v_0.\mu!.[\omega]((v_0)^{\star}\,F))\\
&\quad=_{\mathrm{N}} [\omega](\mu\alpha.[\omega](y\,\lambda v_0.\mu!.[\alpha](v_0)^{\star})\,F)\\
&\quad=_{\mathrm{N}} [\omega]((F)^{\star}\,\mu\alpha.[\omega](y\,\lambda v_0.\mu!.[\alpha](v_0)^{\star}))\\
&\quad\triangleq \ulcorner\langle y.\alpha\rangle\widehat{\alpha}\dagger\widehat{x}P\urcorner_{\mathrm{N}}
\end{aligned}$$

(*exp-outs*$\not\dagger$) : Take Γ fresh.
Let $M \triangleq \mu!.[\omega](\lambda x.\mu!.\ulcorner P\urcorner_{\mathrm{N}}\,(x)^{\star})$ in

$$\begin{aligned}
&\ulcorner(\widehat{y}Q\widehat{\beta}\cdot\alpha)\widehat{\alpha}\not\dagger\widehat{x}P\urcorner_{\mathrm{N}}\\
&\quad\triangleq [\omega](\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{N}})^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{N}} [\omega]((\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{N}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{N}} [\omega]((\lambda x.M)\,\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{N}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)\\
&\quad=_{\mathrm{N}} [\omega]((\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{N}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{N}} [\omega](\mu\sigma.[\omega](\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{N}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{N}} [\omega](\mu\sigma.[\omega](\lambda y.\mu\beta.[\omega](\mu\alpha.\ulcorner Q\urcorner_{\mathrm{N}}\,\lambda x.M))^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{N}} [\omega]((\lambda x.M)^{\star}\,\mu\sigma.[\omega](\lambda y.\mu\beta.[\omega](\mu\alpha.\ulcorner Q\urcorner_{\mathrm{N}}\,\lambda x.M))^{\star})\\
&\quad\triangleq \ulcorner(\widehat{y}(Q\widehat{\alpha}\not\dagger\widehat{x}P)\widehat{\beta}\cdot\gamma)\widehat{\gamma}\dagger\widehat{x}P\urcorner_{\mathrm{N}}
\end{aligned}$$

Theorem 6.11 *For all $\mathcal{X}$-circuits P, P', if $P \rightarrow_{\mathrm{V}}{}^{*} P'$, then* $\ulcorner P\urcorner^{\Omega}_{\mathrm{V}} =_{\mathrm{V}} \ulcorner P'\urcorner^{\Omega}_{\mathrm{V}}$.

Proof: By induction on the length of reduction sequences; we only consider the base case, for which we show the more interesting cases.

(*exp-imp*) : Let $M \triangleq \mu!.\ulcorner R\urcorner_{\mathrm{V}}$.

$$\begin{aligned}
&\ulcorner(\widehat{y}P\widehat{\beta}\cdot\alpha)\widehat{\alpha}\dagger\widehat{x}(Q\widehat{\gamma}\,[x]\,\widehat{z}R)\urcorner_{\mathrm{V}}\\
&\quad\triangleq [\omega]\,(\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}})^{\star}\\
&\qquad\qquad \lambda x.\mu!.[\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda v_0.\mu!.[\omega](x\,v_0\,\lambda z.M)))\\
&\quad=_{\mathrm{V}} [\omega]\,((\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}})^{\star}\\
&\qquad\qquad \lambda x.\mu!.[\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda v_0.\mu!.[\omega](x\,v_0\,\lambda z.M)))\\
&\quad=_{\mathrm{V}} [\omega]\,(\lambda x.\mu!.[\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda v_0.\mu!.[\omega](x\,v_0\,\lambda z.M))\\
&\qquad\qquad \lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}})\\
&\quad=_{\mathrm{V}} [\omega]\mu!.[\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda v_0.\mu!.[\omega](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}}\,v_0\,\lambda z.M))\\
&\quad=_{\mathrm{V}} [\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda v_0.\mu!.[\omega](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}}\,v_0\,\lambda z.M))\\
&\quad=_{\mathrm{V}} \ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda v_0.\mu!.[\omega](\lambda y.\mu\beta.\ulcorner P\urcorner_{\mathrm{V}}\,v_0\,\lambda z.M)\cdot\sigma/\gamma\rangle\\
&\quad=_{\mathrm{V}} \ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda z.M\cdot\sigma/\beta\rangle\langle\lambda y.\mu!.\ulcorner P\urcorner_{\mathrm{V}}\langle\lambda z.M\cdot\sigma/\beta\rangle\cdot\sigma/\gamma\rangle\\
&\quad=_{\mathrm{V}} [\omega](\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda y.\mu!.\ulcorner P\urcorner_{\mathrm{V}}\cdot\sigma/\gamma\rangle\,\lambda z.M)\\
&\quad=_{\mathrm{V}} [\omega](\mu\beta.[\omega](\mu\gamma.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda y.\mu!.\ulcorner P\urcorner_{\mathrm{V}})\,\lambda z.M)\\
&\quad\triangleq \ulcorner(Q\widehat{\gamma}\dagger\widehat{y}P)\widehat{\beta}\dagger\widehat{z}R\urcorner_{\mathrm{V}}
\end{aligned}$$

(*exp-outs*$\not\dagger$) : Let $M \triangleq \mu!.\ulcorner P\urcorner_{\mathrm{V}}$.

$$\begin{aligned}
&\ulcorner(\widehat{y}Q\widehat{\beta}\cdot\alpha)\widehat{\alpha}\not\dagger\widehat{x}P\urcorner_{\mathrm{V}}\\
&\quad\triangleq [\omega]\mu\alpha.[\alpha](\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}})^{\star}\,\lambda x.M\\
&\quad=_{\mathrm{V}} [\omega]((\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{V}} [\omega]((\lambda x.M)\,\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)\\
&\quad=_{\mathrm{V}} [\omega]\mu!.(\ulcorner P\urcorner_{\mathrm{V}})\,\langle\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle/x\rangle\\
&\quad=_{\mathrm{V}} (\ulcorner P\urcorner_{\mathrm{V}})\,\langle\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle/x\rangle\\
&\quad=_{\mathrm{V}} [\omega]\mu!.(\ulcorner P\urcorner_{\mathrm{V}})\,\langle\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle/x\rangle\\
&\quad=_{\mathrm{V}} [\omega]((\lambda x.M)\,\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)\\
&\quad=_{\mathrm{V}} [\omega]((\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{V}} [\omega](\mu\sigma.[\omega](\lambda y.\mu\beta.\ulcorner Q\urcorner_{\mathrm{V}}\langle\lambda x.M\cdot\sigma/\alpha\rangle)^{\star}\,\lambda x.M)\\
&\quad=_{\mathrm{V}} [\omega](\mu\sigma.[\omega](\lambda y.\mu\beta.[\omega](\mu\alpha.\ulcorner Q\urcorner_{\mathrm{V}}\,\lambda x.M))^{\star}\,\lambda x.M)\\
&\quad\triangleq \ulcorner(\widehat{y}(Q\widehat{\alpha}\not\dagger\widehat{x}P)\widehat{\beta}\cdot\sigma)\widehat{\sigma}\dagger\widehat{x}P\urcorner_{\mathrm{V}}
\end{aligned}$$

$(\textit{act}\text{-R}):\ \ulcorner P\widehat{\alpha} \dagger \widehat{x}Q\urcorner_{\mathrm{V}}$

$$\begin{array}{rl}
\triangleq & [\omega](((\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-})^{\star}\, \lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}) \\
=_{\mathrm{V}} & [\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}\, (\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}) \\
=_{\mathrm{V}} & [\omega]\mu!.(\ulcorner Q\urcorner_{\mathrm{V}})\, \langle(\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}/x\rangle \\
=_{\mathrm{V}} & (\ulcorner Q\urcorner_{\mathrm{V}})\, \langle(\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}/x\rangle \\
=_{\mathrm{V}} & [\omega]\mu!.(\ulcorner Q\urcorner_{\mathrm{V}})\, \langle(\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}/x\rangle \\
\triangleq & [\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}\, (\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}) \\
\triangleq & \ulcorner P\widehat{\alpha} \rotatebox[origin=c]{30}{$\dagger$} \widehat{x}Q\urcorner_{\mathrm{V}}
\end{array}$$

$(d\mathrm{R}):$ Let $M \triangleq (\mu\alpha.\ulcorner P\urcorner_{\mathrm{V}})^{-}$:

$$\begin{array}{rcl}
\ulcorner P\widehat{\alpha} \rotatebox[origin=c]{30}{$\dagger$} \widehat{x}Q\urcorner_{\mathrm{V}} & \triangleq & [\omega](\lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}\, M) \\
& =_{\mathrm{V}} & [\omega]((M)^{\star}\, \lambda x.\mu!.\ulcorner Q\urcorner_{\mathrm{V}}) \\
& \triangleq & \ulcorner P\widehat{\alpha} \rotatebox[origin=c]{-30}{$\dagger$} \widehat{x}Q\urcorner_{\mathrm{V}}
\end{array}$$

$\square$