To appear in: *Math. Struct. in Comp. Science* (199?), *vol.* ?, *pp.* ?

Normalization and the Yoneda Embedding

DJORDJE ČUBRIĆ [1 †] , PETER DYBJER [2 ‡] and PHILIP SCOTT [3 §]

[1] *DPMMS, University of Cambridge, 16 Mill Lane, Cambridge CB2 1SB, UK.*
e-mail: cubric@triples.math.mcgill.ca

[2] *Department of Computing Science, Chalmers University of Technology, S-412 96 Göteborg, Sweden .* e-mail: peterd@cs.chalmers.se

[3] *Department of Mathematics, University of Ottawa, 585 King Edward, Ottawa, Ontario K1N 6N5 , Canada.* e-mail: phil@csi.uottawa.ca

Received

We show how to solve the word problem for simply typed $\lambda\beta\eta$-calculus by using a few well-known facts about categories of presheaves and the Yoneda embedding. The formal setting for these results is $\mathcal{P}$-category theory, a version of ordinary category theory where each hom-set is equipped with a partial equivalence relation. The part of $\mathcal{P}$-category theory we develop here is constructive and thus permits extraction of programs from proofs. It is important to stress that in our method, we make no use of traditional proof-theoretic or rewriting techniques. To show the robustness of our method, in the Appendix we give an extended treatment for more general λ-theories.

1. Introduction

In this paper we describe a new, categorical approach to normalization in typed λ-calculus and related theories. Traditionally, the operational semantics of λ-calculi have been based on rewriting theory or proof theory, e.g. normalization or cut-elimination, ordinal assignments, Church-Rosser, etc. Such techniques, e.g. the familiar Tait-Girard computability method (Girard, Lafont, Taylor 1989), or the method of logical relations (Statman 1985; Mitchell 1990) are often based on ingenuity and lack the explanatory power of a model-theoretic proof. At the same time, they introduce specialized syntactic notions intrinsic to the technique (e.g. *neutral terms*, *admissible logical relations*, etc.) but orthogonal to the problem.

We use categorical methods to model $\beta\eta$ convertibility in a presheaf category. This arises essentially from the fact that the Yoneda functor preserves cartesian closedness. This technique has intriguing analogues to the Joyal-Gordon-Power-Street techniques for

† Research supported by an NSERC postdoctoral fellowship at the Department of Pure Mathematics, Cambridge University.
‡ Research supported from ESPRIT Basic Research Action 6811 Categorical Logic in Computer Science (CLiCS-II) and from TFR, the Swedish Technical Research Council.
§ Research supported by an operating grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

proving coherence in various structured (bi-)categories and is also closely related to the (Berger and Schwichtenberg 1991) method for normalizing λ-terms. We discuss history and related work in more detail in section 5 below.

In a certain sense, our program is dual to Lambek's original goal of categorical proof theory (Lambek 1968), in which he used cut-elimination to study categorical coherence problems. Here, we use a method inspired from categorical coherence proofs to normalize lambda terms (and thus intuitionistic proofs.)

However to actually extract an algorithm from these observations requires us to constructively reinterpret the categorical setting, as explained below. It is this intuitionistic aspect of our work which is both novel and fundamental to extracting a normalization algorithm. Not only is this a non-trivial example of program extraction from a structured proof, but it also appears to illustrate a fundamental dictum of Martin-Löf, that one may understand normalization by direct semantic reflection.

1.1. *Categorical Normal Forms*

We can characterize a normal form function abstractly in the following way. Let $\mathcal{T}$ be a set of terms and $\sim$ a congruence relation on terms. One way to decide whether two terms are congruent is to find an *abstract normal form function*, by which we mean a computable function $\mathbf{nf} : \mathcal{T} \to \mathcal{T}$ satisfying the following conditions for some (finer) congruence relation $\equiv$:

NF1 $f \sim \mathbf{nf}(f)$
NF2 $f \sim g \Rightarrow \mathbf{nf}(f) \equiv \mathbf{nf}(g)$
NF3 $\equiv \subseteq \sim$
NF4 $\equiv$ is decidable.

From (NF1), (NF2) and (NF3) we see that $f \sim g \Leftrightarrow \mathbf{nf}(f) \equiv \mathbf{nf}(g)$. This clearly permits a decision procedure: to decide if two terms are equal, compute $\mathbf{nf}$ of each one, and see if they are $\equiv$ related, using (NF4). The normal form function $\mathbf{nf}$ essentially "reduces" the decision problem of $\sim$ to that of $\equiv$.

Here we will consider the example where $\mathcal{T}$ is a set of λ-terms of a given type, $\sim$ is $\beta\eta$-conversion, and $\equiv$ is α-congruence. Let us see heuristically how simply typed λ-calculus can be given a normal form function $\mathbf{nf}$, from categorical considerations.

Recall that λ-terms modulo $\beta\eta$-conversion can be organized as the arrows of a free ccc $\mathcal{F}_\mathcal{X}$ on the set of sorts (atoms) $\mathcal{X}$. The universal property of a free ccc is as follows: for any ccc $\mathcal{C}$, and any interpretation of the atoms $\mathcal{X}$ in $ob(\mathcal{C})$, there is a unique (up to iso) ccc-functor $[\![-]\!] : \mathcal{F}_\mathcal{X} \to \mathcal{C}$ freely extending this interpretation. Now let $\mathcal{C}$ be the presheaf category $Set^{\mathcal{F}_\mathcal{X}{}^{op}}$. There are two obvious ccc-functors: (i) the Yoneda embedding $\mathcal{Y} : \mathcal{F}_\mathcal{X} \to Set^{\mathcal{F}_\mathcal{X}{}^{op}}$ is a ccc-functor, (ii) if we interpret the atoms by Yoneda, there is also the free extension to the ccc-functor $[\![-]\!] : \mathcal{F}_\mathcal{X} \to Set^{\mathcal{F}_\mathcal{X}{}^{op}}$. By the universal property, there is a natural isomorphism $q : [\![-]\!] \to \mathcal{Y}$. Hence, by the Yoneda lemma, for each hom-set we can construct an inverse of the interpretation $[\![-]\!]$ (on this hom-set) according to the following commuting diagram

$$\begin{tikzcd}
\mathcal{F}_{\mathcal{X}}(A,B) \arrow[rr, "{[\![-]\!]}"] \arrow[dr, shift left, "\mathcal{Y}"] & & Set^{\mathcal{F}_{\mathcal{X}}^{op}}([\![A]\!],[\![B]\!]) \arrow[dl, "q_B \circ - \circ q_A^{-1}"] \\
& Set^{\mathcal{F}_{\mathcal{X}}^{op}}(\mathcal{Y}A,\mathcal{Y}B) \arrow[ul, shift left, "-_A(1_A)"] &
\end{tikzcd}$$

Hence, for any $f \in \mathcal{F}_{\mathcal{X}}(A,B)$, we obtain natural transformations $\mathcal{Y}A \xrightarrow{q_A^{-1}} [\![A]\!] \xrightarrow{[\![f]\!]} [\![B]\!] \xrightarrow{q_B} \mathcal{Y}B$. In particular, evaluating at A gives the following functions: $\mathcal{F}_{\mathcal{X}}(A,A) \xrightarrow{q_{A,A}^{-1}} [\![A]\!]A \xrightarrow{[\![f]\!]_A} [\![B]\!]A \xrightarrow{q_{B,A}} \mathcal{F}_{\mathcal{X}}(A,B)$.

Let us now define the function

$$\mathbf{nf}(f) = q_{B,A}\ ([\![f]\!]_A(q_{A,A}^{-1}(1_A)))$$

It immediately follows that

$$f = \mathbf{nf}(f) \quad .$$

Indeed, this is just a restatement of part of the Yoneda isomorphism. But what does it have to do with normalization? As it stands **nf** is nothing but the identity function on $\beta\eta$-convertibility classes of terms!

However, if we reinterpret this diagram in the setting of $\mathcal{P}$-category theory we shall show that **nf** indeed maps typed λ-terms to normal forms. In a $\mathcal{P}$-category each hom-set is equipped with a partial equivalence relation (per). We can thus construct a free $\mathcal{P}$-ccc $(\mathcal{F}_{\mathcal{X}},\sim)$, where the arrows are actual λ-terms and the per $\sim$ on arrows is $\beta\eta$-convertibility. In this setting **nf** will be a (per-preserving) function on terms (a $\mathcal{P}$-function), and not just on convertibility classes of terms. As above, it follows immediately that **nf** is an identity $\mathcal{P}$-function, that is, an identity *up to* $\sim$. But this is nothing but NF1:

$$f \sim \mathbf{nf}(f)$$

Moreover, the part of $\mathcal{P}$-category theory that we use is constructive in the sense that all functions we construct are algorithms. Therefore **nf** is computable.

It remains to prove NF2:

$$f \sim g \Rightarrow \mathbf{nf}(f) \equiv \mathbf{nf}(g) \quad .$$

This is the most subtle point. Here too the $\mathcal{P}$-version of a general categorical fact will help us: that the presheaf category $Set^{\mathcal{C}^{op}}$ is a ccc for any category $\mathcal{C}$. In particular, let $\mathcal{C}$ be the $\mathcal{P}$-category $(\mathcal{F}_{\mathcal{X}},\equiv)$ of sequences of λ-terms up to α-congruence $\equiv$. Note that this $\mathcal{P}$-category has the same objects and arrows as $(\mathcal{F}_{\mathcal{X}},\sim)$, but the pers on arrows are different. Because of the freeness of $(\mathcal{F}_{\mathcal{X}},\sim)$ we have another interpretation $\mathcal{P}$-functor

$$[\![-]\!]^{\equiv} : (\mathcal{F}_{\mathcal{X}},\sim) \rightarrow \mathcal{P}Set^{(\mathcal{F}_{\mathcal{X}},\equiv)^{op}}$$

($\mathcal{P}Set$ is the $\mathcal{P}$-version of the ordinary category Set.) This $\mathcal{P}$-functor has the same effect on objects and arrows as the previous interpretation $\mathcal{P}$-functor:

$$[\![-]\!] : (\mathcal{F}_{\mathcal{X}}, \sim) \to \mathcal{P}Set^{(\mathcal{F}_{\mathcal{X}}, \sim)^{op}}$$

Hence, we can conclude that $f \sim g$ implies $[\![f]\!] \equiv [\![g]\!]$ (here $\equiv$ refers to the per on arrows in $\mathcal{P}Set^{(\mathcal{F}_{\mathcal{X}}, \equiv)^{op}}$). Since we can show that $q_{B,A}$ and $q^{-1}_{B,A}$ preserve $\equiv$, it follows that $\mathbf{nf}(f) \equiv \mathbf{nf}(g)$.

This concludes the summary of our method. In the remainder of the paper we develop the relevant part of $\mathcal{P}$-category theory in detail and show how to construct the normal form function. Note that the development of $\mathcal{P}$-category theory is essentially nothing but the development of ordinary category theory, where the usual (set-theoretic) equality of arrows is everywhere replaced by an explicit per on arrows which is part of the structure of a $\mathcal{P}$-category.

1.2. *Plan of the Paper*

The rest of the paper is organized as follows:

Section 2 gives the basic definitions of $\mathcal{P}$-category theory. It finishes with the definition of free $\mathcal{P}$-cccs and shows how to extract the function **nf**. It is worth emphasizing that the definition of **nf** for which NF1 holds is done uniformly for any free $\mathcal{P}$-ccc.

Section 3 contains the proof that sequences of typed λ-terms form a free $\mathcal{P}$-ccc and instantiates the general normal form function **nf** to this case. We also prove that this **nf** satisfies NF2. It is worth remarking that the proof of NF2 (unlike the proof of NF1) depends on properties of the particular presentation. For example, NF2 fails if we instead construct **nf** from the proof of freeness of the $\mathcal{P}$-ccc of categorical combinators with syntactic identity as $\equiv$.

Section 4 shows that the normal forms returned in the case of typed λ-terms are long $\beta\eta$ normal forms in the ordinary λ-calculus sense.

Finally, in the Appendix we show how to apply our method to the word problem for typed λ-calculi with additional axioms and operations, i.e. to freely-generated ccc's modulo certain theories. This employs appropriate free $\mathcal{P}$-ccc's (over a $\mathcal{P}$-category, a $\mathcal{P}$-cartesian category, etc.) We introduce various notions of λ-theory, which are determined not only by a set of atomic types, but also by a set of basic typed constants as well as a set of equations between terms. Although our methods always yield an algorithm **nf**, it does not necessarily satisfy NF4 (the decidability of $\equiv$). What we obtain is a reduction of the word problems for such free ccc's to those of the underlying generating categories.

2. $\mathcal{P}$-Category Theory

2.1. *Basic Concepts*

As mentioned above, $\mathcal{P}$-category theory is like ordinary category theory, but equality is systematically replaced everywhere by partial equivalence relations (i.e. symmetric, transitive relations).

Definition 2.1. A *$\mathcal{P}$-set* is a pair $A = (|A|, \sim_A)$, where $|A|$ is a set and $\sim_A$ is a partial equivalence relation (per) on $|A|$.

If A is a $\mathcal{P}$-set, we say $|A|$ is its underlying set and refer to $\sim_A$ as *$\mathcal{P}$-equality* . Given a $\mathcal{P}$-set A, let dom_A, the *domain* of $\sim_A$, be the set of elements $a \in |A|$ such that $a \sim_A a$. Note that $\sim_A$ is an equivalence relation on dom_A.

Definition 2.2. A *$\mathcal{P}$-function* between the $\mathcal{P}$-sets $A = (|A|, \sim_A)$ and $B = (|B|, \sim_B)$ is a function $f : |A| \to |B|$ satisfying: $a \sim_A a'$ implies $f(a) \sim_B f(a')$, for all $a, a' \in |A|$.

Although $\mathcal{P}$-sets and $\mathcal{P}$-functions form an ordinary category, and even a cartesian closed category (ccc), we shall be interested in $\mathcal{P}$-analogues of these properties. To describe this, we first introduce some fundamental operations on $\mathcal{P}$-sets corresponding to the ccc structure.

Definition 2.3. We introduce the following $\mathcal{P}$-sets:

- $\mathbf{1}$ is the *one point* $\mathcal{P}$-set $(\{*\}, \sim)$, where $* \sim *$.
- Given two $\mathcal{P}$-sets A and B, their *cartesian product* $A \times B$ is obtained by taking the cartesian product of their underlying sets, with $\sim_{A\times B}$ defined pointwise.
- Given two $\mathcal{P}$-sets A and B, their *exponential* B^A is obtained by taking the exponential of their underlying sets, with $\mathcal{P}$-equality of functions defined as follows:

$$f \sim_{B^A} g \ \text{ iff for all } \ a, a' \in |A|,\, a \sim_A a' \ \text{ implies } \ f(a) \sim_B g(a').$$

It is easily verified that the above constructions of $\mathcal{P}$-sets are well-defined. Note that a $\mathcal{P}$-function from A to B is precisely an element in the domain of $\sim_{B^A}$.

Definition 2.4. A *$\mathcal{P}$-category* $\mathcal{C}$ is a set of objects $ob(\mathcal{C})$, an indexed family of $\mathcal{P}$-sets $(\mathcal{C}(A,B))_{A,B\in ob(\mathcal{C})}$ and two indexed families of $\mathcal{P}$-functions $1_A : \mathbf{1} \to \mathcal{C}(A,A)$, $c_{A,B,C} : \mathcal{C}(A,B) \times \mathcal{C}(B,C) \to \mathcal{C}(A,C)$ such that the appropriate diagrams from the definition of enriched category theory describe $\sim$-related arrows. Writing $c_{A,B,C}(f,g)$ as gf and all pers as $\sim$, we obtain the following explicit axioms:

(i) $f \sim f'$ implies $f' \sim f$,
(ii) $f \sim f'$ and $f' \sim f''$ implies $f \sim f''$,
(iii) $1_A \sim 1_A$,
(iv) $f \sim f'$ and $g \sim g'$ implies $fg \sim f'g'$,
(v) $f \sim f'$ implies $1f \sim f'$ and $f1 \sim f'$,
(vi) $f \sim f'$, $g \sim g'$ and $h \sim h'$ implies $(fg)h \sim f'(g'h')$.

Remark 2.5. A slightly different, but equivalent, presentation of a $\mathcal{P}$-category may be given by restricting functions to domains of pers; for example, in the definition above, (iii) says $1_A \in dom_{\mathcal{C}(A,A)}$, (v) says: for all $f \in dom_{\mathcal{C}(A,A)}$, $1f \sim f$ and $f1 \sim f$, and (vi) says for all f, g, h in domains of appropriate hom $\mathcal{P}$-sets, $(fg)h \sim f(gh)$.

We will also need the following notion:

Definition 2.6. Given a $\mathcal{P}$-category $\mathcal{C}$, we say $f \in \mathcal{C}(A,B)$ is a *$\mathcal{P}$-isomorphism* if there exists $f^{-1} \in \mathcal{C}(B,A)$ such that $f \sim f$, $f^{-1} \sim f^{-1}$, $ff^{-1} \sim 1_B$ and $f^{-1}f \sim 1_A$.

Observe that $\mathcal{P}$-category theory includes ordinary category theory by letting $\mathcal{P}$-equality be ordinary equality. But, in general, $\mathcal{P}$-categories are not categories, although one may obtain genuine categories from them by "sub-quotienting". Note that in $\mathcal{P}$-categories, as in ordinary categories, we have a *set* of objects, not a $\mathcal{P}$-set.

Notation We sometimes denote $\mathcal{P}$-categories by $(\mathcal{C}, \sim)$ if we wish to emphasize a particular per structure on the hom-sets, i.e. that each hom $\mathcal{P}$-set has the form $\mathcal{C}(A,B) = (|\mathcal{C}(A,B)|, \sim_{A,B})$. Of course there may be several $\mathcal{P}$-category structures on the same underlying data: $(\mathcal{C}, \sim)$, $(\mathcal{C}, \equiv)$, etc.

Proposition 2.7. *There is a $\mathcal{P}$-category $\mathcal{P}Set$, whose objects are $\mathcal{P}$-sets and where $\mathcal{P}Set(A,B) = B^A$, the $\mathcal{P}$-set of all functions between the underlying sets, under $\mathcal{P}$-equality of functions.*

Definition 2.8. Let $\mathcal{C}$ and $\mathcal{D}$ be two $\mathcal{P}$-categories. A *$\mathcal{P}$-functor* $F : \mathcal{C} \to \mathcal{D}$ is a function $F : ob(\mathcal{C}) \to ob(\mathcal{D})$ and an indexed family of $\mathcal{P}$-functions $F_{A,B} : \mathcal{C}(A,B) \to \mathcal{D}(FA, FB)$ such that all the expected diagrams from the definition of enriched functors specify related arrows. Explicitly, a $\mathcal{P}$-functor satisfies (omitting subscripts):

(i) $f \sim f'$ implies $Ff \sim Ff'$,
(ii) $F1 \sim 1$,
(iii) $f \sim f'$ and $g \sim g'$ implies $F(fg) \sim (Ff')(Fg')$.

Remark 2.9. As in Remark 2.5 we can equivalently restate some of these definitions using domains of pers. For example, (iii) would be reformulated as: for all $f \in dom_{\mathcal{C}(B,C)}, g \in dom_{\mathcal{C}(A,B)}, F(fg) \sim (Ff)(Fg)$

Definition 2.10. Let $F, G : \mathcal{C} \to \mathcal{D}$ be two $\mathcal{P}$-functors. A *$\mathcal{P}$-natural transformation* $\theta : F \to G$ is an indexed family of $\mathcal{P}$-functions $\theta_A : \mathbf{1} \to \mathcal{D}(FA, GA)$ such that all the expected diagrams from the definition of enriched natural transformation specify related arrows. Thus, a $\mathcal{P}$-natural transformation satisfies:

- $\theta_A \sim \theta_A$, i.e., $\theta_A \in dom_{\mathcal{D}(FA,GA)}$, for all A.
- $f \sim f' \in \mathcal{C}(A,B)$ implies $\theta_B(Ff) \sim (Gf')\theta_A$ (we call this *the $\mathcal{P}$-naturality condition*).

2.2. *$\mathcal{P}$-presheaves and $\mathcal{P}$-ccc's*

Definition 2.11. Let $\mathcal{C}$, $\mathcal{D}$ be two $\mathcal{P}$-categories. The *$\mathcal{P}$-functor category $\mathcal{D}^{\mathcal{C}}$* is the $\mathcal{P}$-category defined as follows. Objects are $\mathcal{P}$-functors from $\mathcal{C}$ to $\mathcal{D}$. Arrows between F and G are indexed families of arrows θ_A in $|\mathcal{D}(FA, GA)|$ with $\mathcal{P}$-equality of families defined as follows: $\theta \sim_{\mathcal{D}^{\mathcal{C}}(F,G)} \theta'$ iff

- θ and θ' satisfy the $\mathcal{P}$-naturality condition ,
- For all A, $\theta_A \sim_{\mathcal{D}(FA,GA)} \theta'_A$.

A $\mathcal{P}$-natural transformation from F to G is precisely an element in the domain of $\sim_{\mathcal{D}^{\mathcal{C}}(F,G)}$. Note that the arrows in a $\mathcal{P}$-functor category are *all* indexed families of arrows θ_A, not just the $\mathcal{P}$-natural ones.

For every object F we define $1_F : \mathbf{1} \to \mathcal{D}^{\mathcal{C}}(F,F)$ to be the indexed family defined by $(1_F)_D = 1_{FD}$. Likewise, we define $c_{F,G,H} : \mathcal{D}^{\mathcal{C}}(F,G) \times \mathcal{D}^{\mathcal{C}}(G,H) \to \mathcal{D}^{\mathcal{C}}(F,H)$ by $c_{F,G,H}(\theta,\psi)_A = c_{FA,GA,HA}(\theta_A,\psi_A)$.

Notation In a $\mathcal{P}$-functor category we denote the $\mathcal{P}$-set of arrows from F to G by $\mathcal{D}^{\mathcal{C}}(F,G)$ as usual, but sometimes also by $\mathcal{P}\mathcal{N}at(F,G)$, when the meaning is clear.

The opposite of a $\mathcal{P}$-category is again a $\mathcal{P}$-category. Therefore there is a $\mathcal{P}$-category $\mathcal{P}Set^{\mathcal{C}^{op}}$ whose objects are called *$\mathcal{P}$-presheaves.*

Definition 2.12. Let $\mathcal{C}$ be a $\mathcal{P}$-category. Define the $\mathcal{P}$-Yoneda functor $\mathcal{Y} : \mathcal{C} \longrightarrow \mathcal{P}Set^{\mathcal{C}^{op}}$ by $\mathcal{Y}B = \mathcal{C}(-,B)$.

The functor $\mathcal{Y}$ is a well-defined $\mathcal{P}$-functor, and we have the following $\mathcal{P}$-version of Yoneda's Lemma:

Lemma 2.13. ($\mathcal{P}$-Yoneda) *Let $\mathcal{C}$ be a $\mathcal{P}$-category.*

(i) For every $\mathcal{P}$-presheaf $F : \mathcal{C}^{op} \to \mathcal{P}Set$, $\mathcal{P}Nat(\mathcal{Y}-,F)$ is a $\mathcal{P}$-presheaf.

(ii) There is a $\mathcal{P}$-natural isomorphism of $\mathcal{P}$-presheaves

$$\theta : F \to \mathcal{P}Nat(\mathcal{Y}-,F)$$

given by: for any $c \in F(C)$, $f \in \mathcal{C}(D,C)$, $(\theta_C(c))_D(f) = (Ff)(c)$ and $\theta_C^{-1}(\eta) = \eta_C(1_C)$.

Corollary 2.14. *For a $\mathcal{P}$-category $\mathcal{C}$*

$$\theta_{B,A} : \mathcal{C}(A,B) \to \mathcal{P}Nat(\mathcal{Y}A,\mathcal{Y}B)$$

is a $\mathcal{P}$-isomorphism of $\mathcal{P}$-sets.

We now introduce the $\mathcal{P}$-version of a cartesian closed category.

Definition 2.15. A *$\mathcal{P}$-ccc* is a $\mathcal{P}$-category with the following distinguished arrows and arrow-forming operations:

Products $\quad A \xrightarrow{O_A} \top \;,\; A \times B \xrightarrow{\pi_{A,B}} A \;,\; A \times B \xrightarrow{\pi'_{A,B}} B \;,\; \dfrac{C \xrightarrow{f} A \quad C \xrightarrow{g} B}{C \xrightarrow{\langle f,g\rangle} A \times B}$

Exponentials $\quad B^A \times A \xrightarrow{\varepsilon_{A,B}} B \;,\; \dfrac{C \times A \xrightarrow{f} B}{C \xrightarrow{f^*} B^A}$

such that the following rules are satisfied:

- $\mathcal{P}$-Products

$$0_A \sim 0_A \;,\; \pi_{A,B} \sim \pi_{A,B} \;,\; \pi'_{A,B} \sim \pi'_{A,B}$$

$$\text{if } f \sim f' \text{ then } f \sim 0_A \quad (\text{for } f, f' : A \to \top)$$

$$\text{if } f \sim f', g \sim g' \text{ then } \langle f,g\rangle \sim \langle f',g'\rangle$$

$$\text{if } f \sim f', g \sim g' \text{ then } \pi_{A,B}\langle f,g\rangle \sim f'$$

$$\text{if } f \sim f', g \sim g' \text{ then } \pi'_{A,B}\langle f,g\rangle \sim g'$$

$$\text{if } k \sim k' \text{ then } \langle \pi_{A,B}k, \pi'_{A,B}k\rangle \sim k'$$

• $\mathcal{P}$-Exponentials

$$\varepsilon_{A,B} \sim \varepsilon_{A,B}$$
$$\text{if } h \sim h' \text{ then } h^* \sim {h'}^*$$
$$\text{if } h \sim h' \text{ then } \varepsilon_{A,B}\langle h^* \pi_{C,A}, \pi'_{C,A}\rangle \sim h'$$
$$\text{if } l \sim l' \text{ then } (\varepsilon_{A,B}\langle l\pi_{C,A}, \pi'_{C,A}\rangle)^* \sim l'.$$

Proposition 2.16. *$\mathcal{P}Set$ is a $\mathcal{P}$-ccc.*

Proof. The $\mathcal{P}$-ccc structure of $\mathcal{P}Set$ was given in Example 2.3, with the $\mathcal{P}$-ccc structure on arrows inherited from Set. □

Theorem 2.17. *If $\mathcal{C}$ is a $\mathcal{P}$-category, then $\mathcal{P}Set^{\mathcal{C}^{op}}$ is a $\mathcal{P}$-ccc.*

Proof. The cartesian closed structure of $\mathcal{P}Set^{\mathcal{C}^{op}}$ is given as follows:

(i) $\mathbf{1}A = \mathbf{1}$.
(ii) $(F \times G)A = FA \times GA$
(iii) $G^F A = \mathcal{P}Nat(\mathcal{Y}A \times F, G)$.

On arrows $f \in \mathcal{C}(B, A)$ define:

(iv) $\mathbf{1}f = id_{\mathbf{1}}$
(v) $(F \times G)f = (Ff) \times (Gf) : FA \times GA \to FB \times GB$
(vi) $G^F f : \mathcal{P}Nat(\mathcal{C}(-, A) \times F, G) \to \mathcal{P}Nat(\mathcal{C}(-, B) \times F, G)$ is defined as follows: for $\theta \in \mathcal{P}Nat(\mathcal{C}(-, A) \times F, G)$, $C \in ob(\mathcal{C})$, and $(g, c) \in \mathcal{C}(C, B) \times FC$:

$$(G^F f)(\theta)_C(g, c) \quad = \quad \theta_C(fg, c) \tag{1}$$

The $\mathcal{P}$-ccc structure on arrows of $\mathcal{P}Set^{\mathcal{C}^{op}}$ is defined by analogy with ordinary presheaves as follows:

(vii) The evaluation $\varepsilon : G^F \times F \to G$ is given by:

$$\varepsilon_A(\theta, a) \quad = \quad \theta_A(1_A, a) \tag{2}$$

for all $\theta \in G^F A$ and $a \in FA$

(viii) The exponential transpose $\theta^* : H \to G^F$ of $\theta : H \times F \to G$ is given by: for all $x \in HA$, $h : \mathcal{C}(B, A)$, $b \in FB$,

$$(\theta^*_A(x))_B(h, b) \quad = \quad \theta_B((Hh)(x), b) \tag{3}$$

It is easy to see that all transformations above are $\mathcal{P}$-natural. □

We are interested in $\mathcal{P}$-functors that preserve $\mathcal{P}$-ccc structure up to $\mathcal{P}$-isomorphism. We shall use the following explicit notion:

Definition 2.18. Let $\mathcal{C}, \mathcal{D}$ be $\mathcal{P}$-ccc's. A *$\mathcal{P}$-ccc functor* is a $\mathcal{P}$-functor $H : \mathcal{C} \to \mathcal{D}$ equipped with specified $\mathcal{P}$-isomorphisms $r_\top : \top \longrightarrow H\top$, $r_{A\times B} : HA \times HB \to H(A \times B)$, $r_{B^A} : (HB)^{HA} \longrightarrow H(B^A)$, whose inverses are as follows:

(i) $r_\top^{-1} = 0_{H\top} : H\top \longrightarrow \top$.
(ii) $r_{A\times B}^{-1} = \langle H\pi, H\pi'\rangle : H(A \times B) \longrightarrow HA \times HB$.

(iii)$r^{-1}_{B^A} = ((H\varepsilon) r_{B^A \times A})^* : H(B^A) \longrightarrow (HB)^{HA}$.

Remark 2.19. The above r-notation is slightly ambiguous. A more precise notation would be to introduce three families $x : \top \longrightarrow H(\top)$, $y_{A,B} : HA \times HB \to H(A \times B)$, $z_{A,B} : H(B^A) \longrightarrow (HB)^{HA}$, along with their inverses. Note that y and z are $\mathcal{P}$-natural in both arguments. However, we shall keep the notation $r_\top = x$, $r_{A\times B} = y_{A,B}$ and $r_{B^A} = z_{A,B}$ in the sequel.

Proposition 2.20. *Let $\mathcal{C}$ be a $\mathcal{P}$-ccc. The $\mathcal{P}$-Yoneda functor $\mathcal{Y} : \mathcal{C} \longrightarrow \mathcal{P}Set^{\mathcal{C}^{op}}$ is a $\mathcal{P}$-ccc functor.*

Proof. We shall examine the explicit structure referred to in Definition 2.18. In each case, r^{-1} comes from that definition. So, for every object $C \in \mathcal{C}$ we define $(r_\top)_C : \mathbf{1}C \to \mathcal{C}(C, \top)$, $(r_\top^{-1})_C : \mathcal{C}(C,\top) \to \mathbf{1}C$, $(r_{A\times B})_C : \mathcal{C}(C,A) \times \mathcal{C}(C,B) \to \mathcal{C}(C, A\times B)$, $(r^{-1}_{A\times B})_C : \mathcal{C}(C, A \times B) \to \mathcal{C}(C,A) \times \mathcal{C}(C,B)$, $(r_{B^A})_C : \mathcal{P}Nat(\mathcal{C}(-,C) \times \mathcal{C}(-,A), \mathcal{C}(-,B)) \to \mathcal{C}(C, B^A)$ and $(r^{-1}_{B^A})_C : \mathcal{C}(C, B^A) \to \mathcal{P}Nat(\mathcal{C}(-,C) \times \mathcal{C}(-,A), \mathcal{C}(-,B))$ as follows:

$$(r_\top)_C(*) = 0_C, \quad (r_\top^{-1})_C(l) = *$$

$$(r_{A\times B})_C(f,g) = \langle f, g\rangle, \quad (r^{-1}_{A\times B})_C(h) = (\pi\, h, \pi'\, h)$$

$$(r_{B^A})_C(\theta) = (\theta_{C\times A}(\pi, \pi'))^*, \quad (r^{-1}_{B^A})_C(k)_D(c,a) = \varepsilon\langle kc, a\rangle$$

for every $l : C \to \top$, $f : C \to A$, $g : C \to B$, $h : C \to A \times B$, $k : C \to B^A$, $c : D \to C$, $a : D \to A$ and $\theta \in \mathcal{P}Nat(\mathcal{C}(-,C) \times \mathcal{C}(-,A), \mathcal{C}(-,B))$.

It is easy to see that the above are indeed components of $\mathcal{P}$-natural isomorphisms which satisfy Definition 2.18. $\square$

2.3. *Free $\mathcal{P}$-ccc's*

We work with the following definition of freeness (for some bicategorical aspects, see Remark 3.17 below).

Definition 2.21. Let $\mathcal{X}$ be a set. $\mathcal{F}_\mathcal{X}$ *is a free $\mathcal{P}$-ccc on* $\mathcal{X}$ provided there exists a function $I : \mathcal{X} \to ob(\mathcal{F}_\mathcal{X})$ such that for any $\mathcal{P}$-ccc $\mathcal{C}$ and any function $J : \mathcal{X} \to ob(\mathcal{C})$, there is a $\mathcal{P}$-ccc functor $[\![-]\!] : \mathcal{F}_\mathcal{X} \to \mathcal{C}$ making the following diagram commute:

$$\begin{tikzcd} & \mathcal{F}_\mathcal{X} \arrow[dr, "{[\![-]\!]}"] & \\ \mathcal{X} \arrow[ur, "I"] \arrow[rr, "J"] & & \mathcal{C} \end{tikzcd}$$

and such that for any other $\mathcal{P}$-ccc functor $G : \mathcal{F}_\mathcal{X} \to \mathcal{C}$ satisfying the same property as $[\![-]\!]$, there is a $\mathcal{P}$-natural isomorphism $q : [\![-]\!] \to G$.

It follows that if $\mathcal{F}_\mathcal{X}$ is a free $\mathcal{P}$-ccc as above, then there is a $\mathcal{P}$-ccc functor $[\![-]\!]$: $\mathcal{F}_\mathcal{X} \to \mathcal{P}Set^{\mathcal{F}_\mathcal{X}{}^{op}}$ which agrees on generators with Yoneda, together with a $\mathcal{P}$-natural isomorphism $q : [\![-]\!] \to \mathcal{Y}$.

We have now developed the $\mathcal{P}$-category theory needed for defining the function $\mathbf{nf}$: $\mathcal{C}(A,B) \to \mathcal{C}(A,B)$ for an arbitrary free $\mathcal{P}$-ccc $\mathcal{C}$, as outlined in the introduction:

Definition 2.22. For $f \in \mathcal{F}_\mathcal{X}(A,B)$, define

$$\mathbf{nf}\ (f) = q_{B,A}\ ([\![f]\!]_A\ (q^{-1}_{A,A}\ (1_A)))$$

Hence, as in the discussion in the introduction, we immediately conclude axiom NF1

Corollary 2.23. $f \sim \mathbf{nf}(f)$.

3. A Normal Form Algorithm for the Typed λ-Calculus

There is a well-known correspondence between cartesian closed categories and typed λ-calculi *with* product and terminal types (see (Lambek and Scott 1986)). In this section, from our $\mathcal{P}$-perspective, we expand on the remark of A. Pitts ((Pitts 199-), Section 4.2) that a similar and quite natural connection can be established between ccc's and typed λ-calculi *without* product and terminal types. We choose Pitts' connection, because our $\mathbf{nf}$ then satisfies $NF1 - NF4$ with $\equiv$ as ordinary α-congruence; this is not the case for Lambek-Scott's terms up to α-congruence, nor for categorical combinators up to syntactic identity (cf. also Remark 3.10). We first develop the connection between free $\mathcal{P}$-ccc's and typed λ-calculi with no free arrows or additional theories. The extension to the more general case is discussed in Remark 3.10.

3.1. *Typed λ-Calculus*

We briefly recall the typed λ-calculus, as presented in (Barendregt 1984; Girard, Lafont, Taylor 1989; Lambek and Scott 1986; Mitchell and Scott 1989).

Definition 3.1. (Typed λ-calculus) Let Sorts be a set of sorts (or atomic types). The *typed λ-calculus generated by Sorts* is a formal system consisting of Types, Terms and Equations between terms, as follows:

Types This is the set inductively generated from the set of Sorts using the following rules: Sorts are Types and if A and B are types then so is B^A.

Terms To every type A we assign a denumerable set of typed variables denoted x : $A, \ldots$ A *context* is a finite (possibly empty) sequence of different typed variables $x_1 : A_1, \ldots, x_n : A_n$, often denoted Σ. Terms (in context) are inductively generated using the following rules:

$$\frac{x : A \in \Sigma}{\Sigma \rhd x : A} \qquad \frac{\Sigma, x : A \rhd t : B}{\Sigma \rhd \lambda x^A.t : B^A} \qquad \frac{\Sigma \rhd s : B^A \quad \Sigma \rhd t : A}{\Sigma \rhd st : B}$$

Note, we write $\lambda x^A.t$, rather than $\lambda x : A.t$.

Equations between terms Equations between terms (in context) of the same type are inductively generated using the following axioms and rules:

$$\Sigma \rhd t = t : B \qquad \frac{\Sigma \rhd s = t : B}{\Sigma \rhd t = s : B} \qquad \frac{\Sigma \rhd s = t : B \quad \Sigma \rhd t = r : B}{\Sigma \rhd s = r : B}$$

$$\frac{\Sigma \rhd s_1 = t_1 : B^A \quad \Sigma \rhd s_2 = t_2 : A}{\Sigma \rhd s_1 s_2 = t_1 t_2 : B} \qquad \frac{\Sigma, x : A \rhd s = t : B}{\Sigma \rhd \lambda x^A.s = \lambda x^A.t : B^A}$$

$$\Sigma \rhd (\lambda x^A.t)s = t(s/x) : B \qquad (\beta)$$

$$\Sigma \rhd t = \lambda x^A.tx : B^A, \text{ if } \; x : A \notin \Sigma \qquad (\eta)$$

In the above, we write $t(s/x)$ to denote the substitution in $\Sigma, x : A \rhd t : B$ of the term $\Sigma \rhd s : A$ for the variable $x : A$. It is assumed that we have taken care of clashes of variables for (β) as usual.

Remark 3.2.

(i) In what follows, when we speak of "a λ-calculus", we mean the typed λ-calculus generated by some set of Sorts, unless otherwise specified.

(ii) It is easy to see that a general substitution rule is admissible in the above system. We will use a still more general "simultaneous substitution" rule:

$$\frac{\Sigma \rhd t_1 : A_1 \cdots \Sigma \rhd t_n : A_n \quad x_1 : A_1, \cdots, x_n : A_n \rhd s : B}{\Sigma \rhd s(t_1/x_1, \ldots t_n/x_n) : B}$$

which is also admissible in the above system.

(iii) We have chosen to work with ordinary λ-terms in this paper. The decidable equivalence relation $\equiv$ (referred to in NF2 - NF4) is α-congruence, where terms are α-congruent if they only differ with respect to names of bound variables. Alternatively, we could have used terms à la de Bruijn and let $\equiv$ be syntactic identity.

(iv) We adopt familiar λ-calculus notation, e.g. uvw denotes $(uv)w$ and $\lambda x.\lambda y.s$ denotes $\lambda x.(\lambda y.s)$. Sometimes we say a well-formed equation $\Sigma \rhd t = t' : B$ "holds" if it can be proved using the previous inductive rules.

3.2. *Constructing a Free $\mathcal{P}$-ccc from λ-terms*

Definition 3.3. Given a set of sorts $\mathcal{X}$, a $\mathcal{P}$-ccc $\mathcal{C}$ and a function $J : \mathcal{X} \to ob(\mathcal{C})$, the *interpretation of the λ-calculus generated by $\mathcal{X}$* is the following recursively defined function $[\![-]\!]$ which maps types to objects and maps a term in context $x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \rhd t : A$ to an arrow $(\ldots([\![A_1]\!] \times [\![A_2]\!]) \times \ldots) \times [\![A_n]\!] \to [\![A]\!]$ (for $n = 0$ we get an arrow $\top \to [\![A]\!]$, and for $n = 1$ we get an arrow $[\![A_1]\!] \to [\![A]\!]$), as follows:

Objects:
- $[\![X]\!] = J(X)$, if $X \in \mathcal{X}$.
- $[\![B^A]\!] = [\![B]\!]^{[\![A]\!]}$.

Arrows:
- $[\![x_1 : A_1, \ldots, x_n : A_n \rhd x_i : A_i]\!] = \pi_i : (\ldots([\![A_1]\!] \times [\![A_2]\!]) \times \ldots) \times [\![A_n]\!] \to [\![A_i]\!]$.
- if $\Sigma \neq \langle\,\rangle$ then $[\![\Sigma \rhd \lambda x^A.t : B^A]\!] = [\![\Sigma, x : A \rhd t : B]\!]^*$.
- if $\Sigma = \langle\,\rangle$ then $[\![\ \rhd \lambda x^A.t : B^A]\!] = ([\![x : A \rhd t : B]\!]\pi'_{\top,[\![A]\!]})^*$.
- $[\![\Sigma \rhd ts]\!] = \varepsilon\langle[\![\Sigma \rhd t]\!], [\![\Sigma \rhd s]\!]\rangle$.

The following is standard and easy to prove:

Proposition 3.4. (Soundness) *Given a set of sorts $\mathcal{X}$, a $\mathcal{P}$-ccc $\mathcal{C}$ and a function $J : \mathcal{X} \to ob(\mathcal{C})$, the above induced interpretation $[\![-]\!]$ satisfies: if $\Sigma \triangleright s = t : B$ then $[\![\Sigma \triangleright s : B]\!] \sim [\![\Sigma \triangleright t : B]\!]$ in $\mathcal{C}$.*

Fact: To prove Soundness, and only in the case of β, one needs to show by induction on u that $[\![x_1 : A_1, \ldots, x_n : A_n \triangleright u : B]\!]\langle[\![\Sigma \triangleright s_1 : A_1]\!], \ldots, [\![\Sigma \triangleright s_n : A_n]\!]\rangle \sim [\![\Sigma \triangleright u(s_1/x_1, s_2/x_2, \ldots, s_n/x_n) : B]\!]$.

To show completeness we first define a syntactic $\mathcal{P}$-ccc. We use the following notation: bold face letters denote finite sequences *e.g.* $\mathbf{A}$ denotes a finite sequence $A_1, \ldots, A_n$ of types and $\mathbf{t}$ denotes a finite sequence $t_1, \ldots, t_m$ of terms with the same context . The empty sequence of types will be denoted by $\langle\rangle$ and the empty sequence of terms in a context Σ is denoted by $\Sigma \triangleright$. Concatenation of two sequences will be denoted by a comma; for example, the concatenation of $\mathbf{A}$ and $\mathbf{B}$ will be denoted by $\mathbf{A}, \mathbf{B}$.

Definition 3.5. Given a λ-calculus generated by a set of sorts $\mathcal{X}$, the $\mathcal{P}$-category $(\mathcal{F}_{\mathcal{X}}, \sim)$ is defined as follows:

Objects are finite sequences of types.

An *arrow* between $\mathbf{A}$ and $\mathbf{B} = B_1, \ldots, B_m$ is a finite sequence of terms each in context $\mathbf{x} : \mathbf{A}$, say $\mathbf{x} : \mathbf{A} \triangleright t_1 : B_1, \ldots, \mathbf{x} : \mathbf{A} \triangleright t_m : B_m$. Such an arrow will often be written $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B})$, where $\mathbf{t} = t_1, \ldots, t_m$.

Two such arrows are $\mathcal{P}$-equivalent, denoted $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B}) \sim (\mathbf{y} : \mathbf{A} \triangleright \mathbf{s} : \mathbf{B})$, if for every i, $\mathbf{z} : \mathbf{A} \triangleright t_i(\mathbf{z}/\mathbf{x}) = s_i(\mathbf{z}/\mathbf{y})$ holds. Obviously, the above relation is an equivalence relation on sequences of terms.

The *identity* on $\mathbf{A}$ is $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{x} : \mathbf{A})$. *Composition* is defined by simultaneous componentwise term substitution.

Proposition 3.6. *$(\mathcal{F}_{\mathcal{X}}, \sim)$ is a $\mathcal{P}$-ccc.*

Proof. It is an easy observation that this is a category with finite products: the empty sequence is the terminal object, with the terminal arrow from $\mathbf{A}$ being $(\mathbf{x} : \mathbf{A} \triangleright\ )$. The product of $\mathbf{A}$ and $\mathbf{B}$ is given by $\mathbf{A}, \mathbf{B}$, i.e. by concatenation of sequences. The projections $\pi : \mathbf{A}, \mathbf{B} \to \mathbf{A}$, $\pi' : \mathbf{A}, \mathbf{B} \to \mathbf{B}$ are defined to be $(\mathbf{x} : \mathbf{A}, \mathbf{y} : \mathbf{B} \triangleright \mathbf{x} : \mathbf{A})$ and $(\mathbf{x} : \mathbf{A}, \mathbf{y} : \mathbf{B} \triangleright \mathbf{y} : \mathbf{B})$, respectively. Pairing $\langle(\mathbf{z} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B}), (\mathbf{y} : \mathbf{A} \triangleright \mathbf{s} : \mathbf{C})\rangle$ is the concatenation $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{t}(\mathbf{x}/\mathbf{z}) : \mathbf{B}, \mathbf{s}(\mathbf{x}/\mathbf{y}) : \mathbf{C})$.

We introduce some abbreviations: $B^{A_1,\ldots,A_n}$ denotes $(\cdots(B^{A_n})\cdots)^{A_1}$. In particular, $B^{\langle\,\rangle} = B$. We define exponents: $(B_1, \ldots, B_m)^{\mathbf{A}} = {B_1}^{\mathbf{A}}, \ldots, {B_m}^{\mathbf{A}}$. In particular, $\langle\,\rangle^{\mathbf{A}} = \langle\,\rangle$.

The arrow $\varepsilon : \mathbf{B}^{\mathbf{A}} \times \mathbf{A} \to \mathbf{B}$ is given by

$$(x_1 : {B_1}^{\mathbf{A}}, \ldots, x_m : {B_m}^{\mathbf{A}}, \mathbf{y} : \mathbf{A} \triangleright x_i y_1 \ldots y_n : B_i)_{i=1,\ldots,m}$$

For an arrow $(\mathbf{z} : \mathbf{C}, \mathbf{x} : \mathbf{A} \triangleright \mathbf{h} : \mathbf{B}) : \mathbf{C} \times \mathbf{A} \to \mathbf{B}$ we define

$$(\mathbf{z} : \mathbf{C}, \mathbf{x} : \mathbf{A} \triangleright \mathbf{h} : \mathbf{B})^* : \mathbf{C} \to \mathbf{B}^{\mathbf{A}}$$

to be

$$(\mathbf{z} : \mathbf{C} \triangleright \lambda x_1^{A_1}.\ldots.\lambda x_n^{A_n}.h_i : B_i^{\mathbf{A}}\ )_{i=1,\cdots,m}$$

It is easy to see that the above gives a $\mathcal{P}$-ccc. $\square$

Remark 3.7. There are various syntactic presentations of $(\mathcal{F}_{\mathcal{X}}, \sim)$, each with their own advantages and disadvantages. For example, another way (cf. (Pitts 199-)) involves taking objects = contexts, and arrows = lists of terms in context.

A familiar Lindenbaum-Tarski style argument shows that completeness follows "by definition". For the record:

Proposition 3.8. (Completeness) *The canonical interpretation of the typed λ-calculus generated by $\mathcal{X}$ is obtained by letting $\mathcal{C} = (\mathcal{F}_{\mathcal{X}}, \sim)$ and $J(X) = X$ (the singleton sequence) in Definition 3.3. For this interpretation $[\![\Sigma \triangleright s : B]\!] \sim [\![\Sigma \triangleright t : B]\!]$ implies $\Sigma \triangleright s = t : B$.*

Proposition 3.9. *$(\mathcal{F}_{\mathcal{X}}, \sim)$ is a free $\mathcal{P}$-ccc over the set of generators $\mathcal{X}$, with $I : \mathcal{X} \to ob(\mathcal{F}_{\mathcal{X}})$ the function $I(X) = X$, qua singleton sequence.*

Suppose that we are given a function $J : \mathcal{X} \to ob(\mathcal{D})$ where $\mathcal{D}$ is a $\mathcal{P}$-ccc. Such a J determines an interpretation $[\![-]\!]$ of the simply typed λ-calculus into $\mathcal{D}$, as in definition 3.3. We now extend $[\![-]\!]$ to a $\mathcal{P}$-ccc functor also denoted $[\![-]\!] : (\mathcal{F}_{\mathcal{X}}, \sim) \to \mathcal{D}$. The definition is by primitive list recursion, but with two base cases (one for the empty list and one for the singleton list).

On objects we let

$$\begin{array}{rcl} [\![\langle\rangle]\!] &=& \top \\ [\![A]\!] &=& \text{as in Definition 3.3} \\ [\![\mathbf{A}, B]\!] &=& [\![\mathbf{A}]\!] \times [\![B]\!] \end{array}$$

where in the last clause $\mathbf{A} \neq \langle\,\rangle$.

On arrows we let

$$\begin{array}{rcl} [\![\mathbf{x} : \mathbf{A} \triangleright \ ]\!] &=& 0_{[\![\mathbf{A}]\!]} \\ [\![\mathbf{x} : \mathbf{A} \triangleright t : A]\!] &=& \text{as in Definition 3.3} \\ [\![\mathbf{x} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B}, s : C]\!] &=& \langle [\![\mathbf{x} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B}]\!], [\![\mathbf{x} : \mathbf{A} \triangleright s : C]\!] \rangle \end{array}$$

where in the last clause $\mathbf{t} \neq \langle\,\rangle$.

As required by Definition 2.18 there are obvious $\mathcal{P}$-natural isomorphisms between $[\![\mathbf{A}]\!] \times [\![\mathbf{B}]\!]$ and $[\![\mathbf{A}, \mathbf{B}]\!]$ and between $[\![\mathbf{B}]\!]^{[\![\mathbf{A}]\!]}$ and $[\![\mathbf{B}^{\mathbf{A}}]\!]$ and hence it follows that $[\![-]\!]$ is a $\mathcal{P}$-ccc functor. Also, obviously $[\![-]\!] I = J$.

Assume now that $H : \mathcal{F}_{\mathcal{X}} \to \mathcal{D}$ is another $\mathcal{P}$-ccc functor such that $HI = J$. We want to find a $\mathcal{P}$-natural isomorphism $q : [\![-]\!] \to H$. Recall that for H to be a $\mathcal{P}$-ccc functor means that there exist $\mathcal{P}$-natural isomorphisms $r_\top$, $r_{\mathbf{A} \times \mathbf{B}}$, $r_{\mathbf{B}^{\mathbf{A}}}$ as in Definition 2.18. We will define $q_{\mathbf{A}}$ and $q_{\mathbf{A}}^{-1}$ by induction on the complexity of $\mathbf{A}$ using these r's:

$q_{\langle\rangle} = r_{\langle\rangle}$, $q_{\langle\rangle}^{-1} = 0_{H(\langle\rangle)}$, $q_X = q_X^{-1} = 1_{J(X)}$,

$q_{B^A} = r_{B^A}(q_B \varepsilon \langle \pi, q_A^{-1} \pi' \rangle)^*$,

$q_{B^A}^{-1} = (q_B^{-1} H(x : B^A, y : A \triangleright xy : B) r_{B^A \times A} \langle \pi, q_A \pi' \rangle)^*$,

and for a nonempty sequence $\mathbf{A}$ we have

$q_{(\mathbf{A},B)} = r_{\mathbf{A}\times B}\langle q_{\mathbf{A}}\pi, q_B\pi'\rangle$ and
$q^{-1}_{(\mathbf{A},B)} = \langle q_{\mathbf{A}}^{-1}H(\mathbf{x}:\mathbf{A}, y:B \rhd \mathbf{x}:\mathbf{A}), q_B^{-1}H(\mathbf{x}:\mathbf{A}, y:B \rhd y:B)\rangle.$

Although tedious, the proof that the above defined q's are $\mathcal{P}$-natural isomorphisms is easy.

Remark 3.10. It is not difficult to see that we can extend the discussion above to more general notions of free $\mathcal{P}$-cccs (over a $\mathcal{P}$-category, a $\mathcal{P}$-cartesian category, etc.). Our method then applies to solving the word problem for free ccc's generated by a category, a cartesian category, etc. To this end, in the Appendix we introduce various notions of λ-theory which are determined not only by a set of atomic types, but also by a set of basic typed constants as well as a set of equations between terms. Then the relative soundness holds as well as completeness with respect to a "canonical model" $I : \mathcal{T} \to \mathcal{F}_{\mathcal{T}}$ where $\mathcal{F}_{\mathcal{T}}$ is built similarly to $\mathcal{F}_{\mathcal{X}}$ except that the role of the per on arrows is played by provability from the theory. Moreover, one can show that I is the "initial" model in the following sense: for every model $J : \mathcal{T} \to \mathcal{C}$ of a λ-theory $\mathcal{T}$ in a $\mathcal{P}$-ccc $\mathcal{C}$ there is a unique (up to a $\mathcal{P}$-iso) $\mathcal{P}$-ccc functor $[\![-]\!] : \mathcal{F}_{\mathcal{T}} \to \mathcal{C}$ such that $[\![-]\!]I \cong J$ where $\cong$ is a naturally defined notion of isomorphism of models. (Actually, the "right" notion of initiality here includes further coherence conditions but we are not going to use them: cf. also Remark 3.17.)

3.3. *The Normal Form Algorithm*

Recall from Section 2.3 that for an *arbitrary* free $\mathcal{P}$-ccc, we can define a normal form function such that NF1 holds. We now use the specific syntactic constructions in the above proof of freeness of $(\mathcal{F}_{\mathcal{X}}, \sim)$ to get a normal form algorithm for the typed λ-calculus.

First, from Definition 2.22 applied to $(\mathcal{F}_{\mathcal{X}}, \sim)$ above, we define the normal form function to be

$$\mathbf{nf}(t) = q_{B,\mathbf{A}}([\![t]\!]_{\mathbf{A}}(q^{-1}_{\mathbf{A},\mathbf{A}}(\mathbf{x}:\mathbf{A} \rhd \mathbf{x}:\mathbf{A}))) \tag{4}$$

Here $[\![-]\!]$ in (4) is obtained by instantiating the interpretation $\mathcal{P}$-functor in the proof of freeness (Proposition 3.9) to the case where the target $\mathcal{P}$-category $\mathcal{D}$ is the $\mathcal{P}$-presheaf category $\mathcal{P}Set^{(\mathcal{F}_{\mathcal{X}},\sim)}$, and where $J(X) = \mathcal{F}_{\mathcal{X}}(-, X)$. Moreover, q and q^{-1} in (4) are obtained by instantiating the q and q^{-1} in the freeness proof to the case where the $\mathcal{P}$-natural transformations $r_\top$, $r_{\mathbf{A}\times\mathbf{B}}$, $r_{\mathbf{B}^\mathbf{A}}$ refer to the constructions used for proving that Yoneda preserves the distinguished $\mathcal{P}$-ccc structure (see Theorem 2.17 and Proposition 2.20, with $\mathcal{C} = (\mathcal{F}_{\mathcal{X}}, \sim)$). We now give the details of this instantiation.

We start with $[\![-]\!]$. On an object A, $[\![A]\!]$ is a $\mathcal{P}$-presheaf, so we have to show its effects both on objects and arrows. We define it recursively as follows.

On objects:

$$[\![\langle\rangle]\!]\mathbf{C} = \mathbf{1} \tag{5}$$

$$[\![X]\!]\mathbf{C} = \mathcal{F}_{\mathcal{X}}(\mathbf{C}, X) \tag{6}$$

$$[\![B^A]\!]\mathbf{C} = \mathcal{P}Nat(\mathcal{F}_{\mathcal{X}}(-, \mathbf{C}) \times [\![A]\!], [\![B]\!]) \tag{7}$$

$$[\![\mathbf{A}, B]\!]\mathbf{C} = [\![\mathbf{A}]\!]\mathbf{C} \times [\![B]\!]\mathbf{C} \quad \mathbf{A} \neq \langle\,\rangle \tag{8}$$

On arrows:

$$[\![\langle\rangle]\!]\mathbf{h}(*) = * \tag{9}$$

$$[\![X]\!](\mathbf{w} : \mathbf{E} \rhd \mathbf{h} : \mathbf{C})(\mathbf{z} : \mathbf{C} \rhd g : X) = (\mathbf{w} : \mathbf{E} \rhd g(\mathbf{h}/\mathbf{z}) : X) \tag{10}$$

$$\begin{aligned}&(([\![B^A]\!](\mathbf{w} : \mathbf{E} \rhd \mathbf{h} : \mathbf{C}))\theta)_{\mathbf{D}}((\mathbf{k} : \mathbf{D} \rhd \mathbf{g}' : \mathbf{E}), a) \\ &= \theta_{\mathbf{D}}((\mathbf{k} : \mathbf{D} \rhd \mathbf{h}(\mathbf{g}'/\mathbf{w}) : C), a)\end{aligned} \tag{11}$$

$$([\![\mathbf{A}, B]\!]\mathbf{h})(\mathbf{a}, b) = (([\![\mathbf{A}]\!]\mathbf{h})\mathbf{a}, ([\![B]\!]\mathbf{h})b) \quad \mathbf{A} \neq \langle\,\rangle \tag{12}$$

On arrows $[\![-]\!]$ is defined recursively as follows:

$$[\![\mathbf{x} : \mathbf{A} \rhd \quad]\!]_{\mathbf{C}}(\mathbf{a}) = * \tag{13}$$

$$[\![\mathbf{x} : \mathbf{A} \rhd x_i : A_i]\!]_{\mathbf{C}}(\mathbf{a}) = a_i \tag{14}$$

$$\begin{aligned}&([\![\mathbf{x} : \mathbf{A} \rhd \lambda x^A.h : B^A]\!]_{\mathbf{C}}(\mathbf{a}))_{\mathbf{D}}((\mathbf{w} : \mathbf{D} \rhd \mathbf{g} : \mathbf{C}), a') = \\ &[\![\mathbf{x} : \mathbf{A}, x : A \rhd h : B]\!]_{\mathbf{D}}([\![\mathbf{A}]\!](\mathbf{w} : \mathbf{D} \rhd \mathbf{g} : \mathbf{C})(\mathbf{a}), a')\end{aligned} \tag{15}$$

$$\begin{aligned}&[\![\mathbf{x} : \mathbf{A} \rhd st : B]\!]_{\mathbf{C}}(\mathbf{a}) = \\ &([\![\mathbf{x} : \mathbf{A} \rhd s : B^A]\!]_{\mathbf{C}}(\mathbf{a}))_{\mathbf{C}}((\mathbf{z} : \mathbf{C} \rhd \mathbf{z} : \mathbf{C}), [\![\mathbf{x} : \mathbf{A} \rhd t : A]\!]_{\mathbf{C}}(\mathbf{a}))\end{aligned} \tag{16}$$

For a nonempty $\mathbf{t}$

$$[\![\mathbf{x} : \mathbf{A} \rhd \mathbf{t} : \mathbf{B}, s : D]\!]_{\mathbf{C}}(\mathbf{a}) = ([\![\mathbf{x} : \mathbf{A} \rhd \mathbf{t} : \mathbf{B}]\!]_{\mathbf{C}}(\mathbf{a}), [\![\mathbf{x} : \mathbf{A} \rhd s : D]\!]_{\mathbf{C}}(\mathbf{a})) \tag{17}$$

The specific instances of q are the following:

$$q_{\langle\rangle,\mathbf{C}}(*) = (\mathbf{x} : \mathbf{C} \rhd\ ) \tag{18}$$

$$q_{X,\mathbf{C}}(\mathbf{z} : \mathbf{C} \rhd t : X) = (\mathbf{z} : \mathbf{C} \rhd t : X) \tag{19}$$

$$\begin{aligned}&q_{B^A,\mathbf{C}}(\theta) = \\ &\lambda x^A.q_{B,(\mathbf{C},A)}(\theta_{(\mathbf{C},A)}((\mathbf{x} : \mathbf{C}, x : A \rhd \mathbf{x} : \mathbf{C}), q^{-1}_{A,(\mathbf{C},A)}(\mathbf{x} : \mathbf{C}, x : A \rhd x : A)))\end{aligned} \tag{20}$$

$$q_{(\mathbf{A},B),\mathbf{C}}(a, b) = (q_{\mathbf{A},\mathbf{C}}(a), q_{B,\mathbf{C}}(b)) \quad \mathbf{A} \neq \langle\,\rangle \tag{21}$$

(The right hand side of the above equation is a pairing *i.e.* concatenation of the two sequences with previously equated contexts.)

$$q^{-1}_{\langle\rangle,\mathbf{C}}(\mathbf{z} : \mathbf{C} \rhd\ ) = * \tag{22}$$

$$q^{-1}_{X,\mathbf{C}}(\mathbf{z} : \mathbf{C} \rhd t : X) = (\mathbf{z} : \mathbf{C} \rhd t : X) \tag{23}$$

$$(q^{-1}_{B^A,\mathbf{C}}(\mathbf{z} : \mathbf{C} \triangleright t : B^A))_{\mathbf{D}}((\mathbf{w} : \mathbf{D} \triangleright \mathbf{s} : \mathbf{C}), a') \; = \qquad (24)$$
$$q^{-1}_{B,\mathbf{D}}(\mathbf{w} : \mathbf{D} \triangleright t(\mathbf{s}/\mathbf{z})q_{A,\mathbf{D}}(a') : B)$$

$$q^{-1}_{(\mathbf{A},B),\mathbf{C}}(\mathbf{z} : \mathbf{C} \triangleright \mathbf{t} : \mathbf{A}, s : B) \; = \qquad (25)$$
$$(q^{-1}_{\mathbf{A},\mathbf{C}}(\mathbf{z} : \mathbf{C} \triangleright \mathbf{t} : \mathbf{A}), q^{-1}_{B,\mathbf{C}}(\mathbf{z} : \mathbf{C} \triangleright s : B)) \quad \mathbf{A} \neq \langle\,\rangle$$

This completes the algorithm. Note the similarity between this algorithm and the algorithm by (Berger and Schwichtenberg 1991, p. 203). Our $[\![-]\!]$ corresponds to their "evaluation functional" into *Sets*. Our q corresponds to their functional "procedure $\to$ expression" $p \to e$; and our q^{-1} corresponds to their "make self evaluating" *mse*.

3.4. *Uniqueness of Normal Forms*

Following the discussion in the Introduction, we shall now prove NF2, i.e. that the normal forms returned by **nf** are unique up to α-congruence $\equiv$: if $f \sim g$ then $\mathbf{nf}(f) \equiv \mathbf{nf}(g)$. Clearly, α-congruence is decidable (NF3) and is a subrelation of $\sim$ (NF4).

Recall the free $\mathcal{P}$-ccc $(\mathcal{F}_{\mathcal{X}}, \sim)$ from Definition 3.5. We shall now move to a $\mathcal{P}$-category given by α-conversion:

Definition 3.11. Consider the $\mathcal{P}$-category $(\mathcal{F}_{\mathcal{X}}, \equiv)$ which has the same objects and arrows as $(\mathcal{F}_{\mathcal{X}}, \sim)$, but where the per $\equiv$ is defined from α-congruence as follows: $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{t} : \mathbf{B}) \equiv (\mathbf{y} : \mathbf{A} \triangleright \mathbf{s} : \mathbf{B})$, if for every i, the terms $\mathbf{z} : \mathbf{A} \triangleright t_i(\mathbf{z}/\mathbf{x})$ and $\mathbf{z} : \mathbf{A} \triangleright s_i(\mathbf{z}/\mathbf{y})$ are α-congruent.

To prove that $(\mathcal{F}_{\mathcal{X}}, \equiv)$ is indeed a $\mathcal{P}$-category, we use that substitution of the identity arrow $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{x} : \mathbf{A})$ for the free variables in a term returns an α-congruent term and that termwise simultaneous substitution is associative up to α-congruence.

Remark 3.12. What categorical properties does $(\mathcal{F}_{\mathcal{X}}, \equiv)$ enjoy? More generally, recalling Remark 3.10, given a λ-theory $\mathcal{T}$ what are the categorical properties of $(\mathcal{F}_{\mathcal{T}}, \equiv)$ where $\equiv$ is a per obtained without β and η but using the rest of the rules as well as the equations from the theory? It is not difficult to show that such a ($\mathcal{P}$-)category has finite ($\mathcal{P}$-)products, and for every two objects A and B there is an object B^A which satisfies the following: there is an arrow $\varepsilon : B^A \times A \to B$, and for every arrow $f : C \times A \to B$ there exists an arrow $f^* : C \to B^A$ such that for any $g : D \to C$, $f^*g \equiv (f\langle g\pi, \pi'\rangle)^*$. This identity is a $\equiv$-version of a familiar ccc-identity (cf. (Lambek and Scott 1986), (3.2), p. 54). Furthermore, this is a complete characterization since for every ($\mathcal{P}$-)category $\mathcal{C}$ with these properties, using the internal language, we can find a λ-theory $\mathcal{T}_{\mathcal{C}}$ such that the above construction gives $\mathcal{F}_{\mathcal{T}_{\mathcal{C}}} \cong \mathcal{C}$ where $\cong$ denotes a ($\mathcal{P}$-)equivalence of ($\mathcal{P}$-)categories. Coming back to our $(\mathcal{F}_{\mathcal{X}}, \equiv)$ we can say that this is a free object in the category of such (structured) categories.

Notation: Recall that the underlying set of a per A is denoted by $|A|$. Also recall that a $\mathcal{P}$-function f between $\mathcal{P}$-sets is a function between their underlying sets. We sometimes write $|f|$ when we want to emphasize that we consider f as an underlying function rather than a $\mathcal{P}$-function. Now, we extend this notation. For a $\mathcal{P}$-presheaf $F : (\mathcal{C}, \sim)^{op} \to \mathcal{P}Set$ we let $|F|$ denote the function $\mathcal{C} \to Set$ defined as follows: for objects C, $|F|(C) = |F(C)|$ and for arrows f, $|F|(f) = F(f)$. Finally, for every arrow $\theta : F \to G$ in $\mathcal{P}$-presheaves, $|\theta|$ is used to emphasize that we consider it as a family of underlying functions $|\theta|_C = \theta_C : |F|C \to |G|C$. Obviously, $|\theta_1\theta_2| = |\theta_1||\theta_2|$ and if $|F| = |G|$ then $|1_F| = |1_G|$.

We can now state the key lemma for proving uniqueness of normal forms:

Lemma 3.13.
Let $\mathcal{C}_1 = (\mathcal{C}, \sim_1)$ and $\mathcal{C}_2 = (\mathcal{C}, \sim_2)$ be two $\mathcal{P}$-categories with the same "underlying" $\mathcal{C}$. Then, their $\mathcal{P}$-presheaf categories have the same underlying $\mathcal{P}$-ccc structure in the following sense: $|\top_1| = |\top_2|$, if $F_i, G_i \in \mathcal{C}_i^{op} \to \mathcal{P}Set$ and $|F_1| = |F_2|$ and $|G_1| = |G_2|$ then: $|F_1 \times G_1| = |F_2 \times G_2|$ and $|F_1^{G_1}| = |F_2^{G_2}|$. Also, under the same assumptions $|0_{F_1}| = |0_{F_2}|$, $|\pi_{F_1,G_1}| = |\pi_{F_2,G_2}|$, $|\pi'_{F_1,G_1}| = |\pi'_{F_2,G_2}|$, if $|\theta_1| = |\theta_2|$ and $|\theta'_1| = |\theta'_2|$ then $|\langle\theta_1, \theta_2\rangle| = |\langle\theta'_1, \theta'_2\rangle|$, $|\varepsilon_1| = |\varepsilon_2|$ and if $|\theta_1| = |\theta_2|$ then $|\theta_1^*| = |\theta_2^*|$.

Proof. By definition of the canonical $\mathcal{P}$-ccc structure of a $\mathcal{P}$-presheaf category. □

So far we have considered $\mathcal{P}$-presheaves over $\mathcal{F}_1 = (\mathcal{F}_\mathcal{X}, \sim)$. Let us now also consider $\mathcal{P}$-presheaves over $\mathcal{F}_2 = (\mathcal{F}_\mathcal{X}, \equiv)$. Since these form a $\mathcal{P}$-ccc and $\mathcal{F}_1$ is a free $\mathcal{P}$-ccc, Proposition 3.9 shows how to define a $\mathcal{P}$-ccc functor $[\![-]\!]^\equiv : \mathcal{F}_1 \to \mathcal{P}Set^{\mathcal{F}_2}$ with $[\![X]\!]^\equiv = \mathcal{F}_2(-, X)$ for each sort $X \in \mathcal{X}$. Since it is a $\mathcal{P}$-functor it satisfies:

$$f \sim g \text{ implies } [\![f]\!]^\equiv \equiv [\![g]\!]^\equiv \tag{26}$$

Lemma 3.14. *For every object C and every arrow f, the following holds: $|[\![C]\!]| = |[\![C]\!]^\equiv|$ and $|[\![f]\!]| = |[\![f]\!]^\equiv|$.*

Proof. By induction, using the previous lemma and the fact that $|\mathcal{F}_1(-, X)| = |\mathcal{F}_2(-, X)|$. □

Consider the $\mathcal{P}$-natural transformations q_A and q_A^{-1} defined in subsection 3.3. Using the above lemma (and the fact that $|\mathcal{F}_1(-, A)| = |\mathcal{F}_2(-, A)|$ for every object A, not only a generator) we know their underlying functions are maps $|q_A| : |[\![A]\!]^\equiv| \to |\mathcal{F}_2(-, A)|$ and $|q_A^{-1}| : |\mathcal{F}_2(-, A)| \to |[\![A]\!]^\equiv|$. To prove uniqueness of normal forms we need to know that these preserve $\equiv$, that is, that they are $\mathcal{P}$-functions $q_{A,B} : [\![A]\!]^\equiv B \to \mathcal{F}_2(B, A)$ and $q_{A,B}^{-1} : \mathcal{F}_2(B, A) \to [\![A]\!]^\equiv B$. To this end we prove the following stronger property :

Lemma 3.15. *The above transformations $q_A : [\![A]\!]^\equiv \to \mathcal{F}_2(-, A)$ and $q_A^{-1} : \mathcal{F}_2(-, A) \to [\![A]\!]^\equiv$ are $\equiv$-natural, i.e. $\mathcal{P}$-arrows in $\mathcal{P}Set^{\mathcal{F}_2{}^{op}}$. In particular, for every A and B, $q_{A,B} : [\![A]\!]^\equiv B \to \mathcal{F}_2(B, A)$ and $q_{A,B}^{-1} : \mathcal{F}_2(B, A) \to [\![A]\!]^\equiv B$ are $\mathcal{P}$-functions.*

Proof. By induction on A. Only the exponential case is non-trivial. Here we show that $q_{B^A} : [\![B^A]\!]^\equiv \to \mathcal{F}_2(-, B^A)$ is $\equiv$-natural. The proof that $q_{B^A}^{-1} : \mathcal{F}_2(-, B^A) \to [\![B^A]\!]^\equiv$ is $\equiv$-natural leads to similar calculations.

We have to show that for every $\mathbf{C}$, $\mathbf{D}$, $\theta \sim \theta' \in [\![B^A]\!]\mathbf{C}$, $(\mathbf{w} : \mathbf{D} \rhd \mathbf{s} : \mathbf{C}) \equiv (\mathbf{w}' : \mathbf{D} \rhd \mathbf{s}' : \mathbf{C})$

(i) $q_{B^A,\mathbf{C}}(\theta) \equiv q_{B^A,\mathbf{C}}(\theta')$

(ii) $q_{B^A,\mathbf{D}}([\![B^A]\!](\mathbf{s})(\theta)) \equiv \mathcal{F}_2(\mathbf{s}', B^A)(q_{B^A,\mathbf{C}}(\theta'))$.

(i) is immediate. Let us see just (ii). By definition of q_{B^A} (cf. equation 20) this is equivalent to

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\llbracket B^A \rrbracket(\mathbf{s})(\theta))_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{w}:\mathbf{D}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A)) \equiv$$

$$(\lambda x^A.q_{B,(\mathbf{C},A)}(\theta'_{(\mathbf{C},A)}((\mathbf{z}:\mathbf{C},x:A \triangleright \mathbf{z}:\mathbf{C}),q^{-1}_{A,(\mathbf{C},A)}(\mathbf{z}:\mathbf{C},x:A \triangleright x:A))))(\mathbf{s}'/\mathbf{x}).$$

By equation (11) this is the same as:

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))) \equiv$$

$$(\lambda x^A.q_{B,(\mathbf{C},A)}(\theta'_{(\mathbf{C},A)}((\mathbf{z}:\mathbf{C},x:A \triangleright \mathbf{z}:\mathbf{C}),q^{-1}_{A,(\mathbf{C},A)}(\mathbf{z}:\mathbf{C},x:A \triangleright x:A))))(\mathbf{s}'/\mathbf{x})).$$

By definition of substitution, since $x:A \notin \mathbf{w}:\mathbf{D}$ the above is the same as:

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))) \equiv$$

$$\lambda x^A.(q_{B,(\mathbf{C},A)}(\theta'_{(\mathbf{C},A)}((\mathbf{z}:\mathbf{C},x:A \triangleright \mathbf{z}:\mathbf{C}),q^{-1}_{A,(\mathbf{C},A)}(\mathbf{z}:\mathbf{C},x:A \triangleright x:A)))(\mathbf{s}'/\mathbf{x})).$$

By the induction hypothesis, q_B is $\mathcal{P}$-natural. We use its naturality with respect to $(\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C},x:A) \equiv (\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}':\mathbf{C},x:A)$ to transform the above into the following equivalent expression:

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))) \equiv$$

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\llbracket B \rrbracket(\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C},x:A)(\theta'_{(\mathbf{C},A)}((\mathbf{z}:\mathbf{C},x:A \triangleright \mathbf{z}:\mathbf{C}),q^{-1}_{A,(\mathbf{C},A)}(\mathbf{z}:\mathbf{C}, x:A \triangleright x:A))).$$

Now, by $\theta \sim \theta'$ (so therefore both of them are $\mathcal{P}$-natural), the above is equivalent to:

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))) \equiv$$

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{C},x:A \triangleright \mathbf{s}:\mathbf{C}),\llbracket A \rrbracket(\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C},x:A)(q^{-1}_{A,(\mathbf{C},A)}(\mathbf{z}:\mathbf{C},x:A \triangleright x:A)))).$$

Finally, by the induction hypothesis, q_A^{-1} is $\mathcal{P}$-natural with respect to $(\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C},x:A) \equiv (\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C},x:A)$ and therefore the above is equivalent to:

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{D},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))) \equiv$$

$$\lambda x^A.q_{B,(\mathbf{D},A)}(\theta_{(\mathbf{D},A)}((\mathbf{w}:\mathbf{C},x:A \triangleright \mathbf{s}:\mathbf{C}),q^{-1}_{A,(\mathbf{D},A)}(\mathbf{w}:\mathbf{D},x:A \triangleright x:A))))$$

and this is true. □

Assume now that $f \sim g : A \to B$. Implication (26) gives $\llbracket f \rrbracket^{\equiv} \equiv \llbracket g \rrbracket^{\equiv}$. By Lemma 3.15 it holds that $q_{B,A}\llbracket f \rrbracket^{\equiv}_{\overline{A}} q^{-1}_{A,A}(1_A) \equiv q_{B,A}\llbracket g \rrbracket^{\equiv}_{\overline{A}} q^{-1}_{A,A}(1_A)$. By Lemma 3.14 $q_{B,A}\llbracket f \rrbracket^{\equiv}_{\overline{A}} q^{-1}_{A,A}(1_A) = q_{B,A}\llbracket f \rrbracket_A q^{-1}_{A,A}(1_A)$ and $q_{B,A}\llbracket g \rrbracket^{\equiv}_{\overline{A}} q^{-1}_{A,A}(1_A) = q_{B,A}\llbracket g \rrbracket_A q^{-1}_{A,A}(1_A)$. Therefore $\mathbf{nf}(f) \equiv \mathbf{nf}(g)$. This completes the proof of $NF2$ and since the other properties were proved before, our algorithm **nf** indeed gives a normal form for terms in the typed λ-calculus. For the record:

Theorem 3.16. *The above specified* **nf** *is indeed an algorithm which satisfies the properties $NF1 - NF4$ from the Introduction, where $\sim$ is $\beta\eta$-equality of terms in simply typed λ-calculus and $\equiv$ is α-congruence.*

Before we end the section, let us remark on an alternative definition of freeness.

Remark 3.17. The reader may wonder why we did not define $[\![-]\!]$ by ordinary primitive recursion on lists, which has only one base case (the empty list), rather than by the variant here which has another base case for the singleton list. We could instead try to define $[\![-]\!]$ by $[\![\langle\,\rangle]\!] = \top$ and $[\![\mathbf{A}, B]\!] = [\![\mathbf{A}]\!] \times [\![B]\!]$. But then, it may seem that we have a problem since for the object which is just a singleton sequence, *e.g.* a basic type X, we would have $[\![X]\!] = [\![\langle\,\rangle]\!] \times [\![X]\!] = \top \times J(X)$ and this does not have to be *equal* to $J(X)$. And therefore, this more elegant definition does not satisfy our definition of freeness. But, it does satisfy another "bicategorical" definition of freeness which is anyway the preferred one when dealing with "categories" of categories. In our case this definition goes as follows: $\mathcal{F}_{\mathcal{X}}$ is a free ($\mathcal{P}$-)ccc over $I : \mathcal{X} \to ob(\mathcal{F}_{\mathcal{X}})$ if for every ($\mathcal{P}$-)ccc $\mathcal{D}$ and every $J : \mathcal{X} \to ob(\mathcal{D})$ there exists a ($\mathcal{P}$-)ccc functor $[\![-]\!] : \mathcal{F}_{\mathcal{X}} \to \mathcal{D}$ such that there is a $\mathcal{P}$-iso $\theta : [\![I(-)]\!] \Rightarrow J$. Moreover, for any other $J' : \mathcal{X} \to ob(\mathcal{D})$, $[\![-]\!]' : \mathcal{F}_{\mathcal{X}} \to \mathcal{D}$ and $\theta' : [\![I(-)]\!]' \Rightarrow J'$ as above, and for any ($\mathcal{P}$-)iso $\sigma : J \Rightarrow J'$ there exists a ($\mathcal{P}$-)unique ($\mathcal{P}$-)natural iso $\rho : [\![-]\!] \Rightarrow [\![-]\!]'$ such that $\theta'(\rho \circ I) \sim \sigma\theta$ (in the "ordinary" (non $\mathcal{P}$) world, the last $\sim$ is actually equality). $\mathcal{P}$-uniqueness for $\mathcal{P}$-natural transformation ρ meant that for any other ρ' satisfying the above condition $\rho \sim \rho'$. Let us also mention here that the "coherence conditions" from Remark 3.10 are same as the above ones.

Another advantage of the above definition of freeness is that it does not use equality on objects and this is one of the basic features of "pure" $\mathcal{P}$-category theory as well. We find it interesting that the direct inductive definitions in our study gives rise to the categorically prefered notion of freeness. Yet, to make our paper more accessible to non-category theorists we decided to define freeness as in Definition 2.21 which then implied the more extensive "case handling".

4. Characterizing Normal Forms in λ-Calculus

In this section we characterize our categorical normal forms $\mathbf{nf}(t)$ as being exactly the *long $\beta\eta$-normal forms* familiar from λ-calculus (Huet 1976) (see also (Čubrić 199-; DiCosmo 1995)). Observe that we have obtained our normal forms purely categorically–without reference to rewriting techniques in λ-calculus. It is only now, when we must prove that our normal forms coincide with the usual ones, that we employ familiar syntactic methods.

4.1. *Normal Forms in λ-Calculus*

We now generate (by mutual induction) two collections of terms, *neutral* and *long-$\beta\eta$ normal*, and we prove that neutral terms are closed under substitution. From now on, the word "normal" means "long-$\beta\eta$ normal".

Definition 4.1. *Variables of arbitrary type are neutral. Neutral terms of atomic type are normal as well. If u is normal and v is neutral then vu is neutral and $\lambda x.u$ is normal.*

At this point we want to show that normal terms are irreducible and that neutral terms are irreducible "inside". For that we need a few definitions.

Notation: By a *substitution context* $C[\ ]$ we mean a term with a hole in it (Barendregt 1984). These play a fundamental role in the rewriting theory of λ-calculi (both typed and untyped). The most important fact about contexts is that one can plug in a term (of the right type) into the hole without paying attention to possible clashes of variables. This permits a smooth discussion of local reduction rules, as we see below. Substitution into a context $C[\ ]$ is just formal plugging-in, whereas whenever we write $u(v/x)$ we mean genuine substitution, that is, we rename bound variables by α-conversion to prevent clashes.

Let us now define the notion of (long) $\beta\eta$-reduction with respect to a substitution context $C[\ ]$ by two clauses:

$$C[(\lambda x.u)v] \xrightarrow{\beta} C[u(v/x)],$$

$$C[u^{A\to B}] \xrightarrow{\eta} C[\lambda x.(ux)]\ ,\ x \notin FV(u)$$

$$\text{provided neither } \ u \equiv \lambda y.w \ \text{ nor } \ C[u] \equiv D[(uw)]$$

for any term w and any substitution context $D[\ ]$. The η rule used here is sometimes called "η-expansion" in the literature.

The term $(\lambda x.u)v$ in (β) and the term u in (η) on which reduction is possible are called (long $\beta\eta$-)*redexes*. A term which contains no redexes is called *irreducible*.

Lemma 4.2. *Normal terms are irreducible and neutral terms don't have proper subterms which are redexes. Also, irreducible terms are normal.*

Proof. The first sentence is proved by simultaneous induction on the structure of normal and neutral terms. The second sentence is then obviously true. □

From now on we will use the fact that normal and irreducible mean the same thing without explicitly stating it.

Lemma 4.3. *If u and v are neutral terms then $u(v/x)$ is neutral and if u is normal and v is neutral then $u(v/x)$ is normal.*

Proof. This is proved by induction on u using the previous lemma. □

Remark 4.4. In what follows, we simplify notation by suppressing the context Σ when no ambiguity arises. Thus we speak simply of "terms".

4.2. *The Characterization Theorem*

In this section we shall show that for a term $\mathbf{x} : \mathbf{A} \triangleright t : B$ our categorical normal form $\mathbf{nf}(t)$ is actually the long normal form. We shall prove this by a computability argument (Girard, Lafont, Taylor 1989; Lambek and Scott 1986); this is related to the glueing argument in (Altenkirch, Hofmann, Streicher 1995).

We start with our free $\mathcal{P}$-ccc $(\mathcal{F}_{\mathcal{X}}, \sim)$. Recall that the objects and arrows were finite sequences of types and terms, respectively. First we extend the notions of neutral and normal to arbitrary arrows: for an arrow $\mathbf{t} \in \mathcal{F}_{\mathcal{X}}(\mathbf{A}, \mathbf{B})$ we say $\mathbf{t} \in \mathcal{NE}(\mathbf{A}, \mathbf{B})$ if for every

i all the terms t_i are neutral; similarly, we say $\mathbf{t} \in \mathcal{NF}(\mathbf{A}, \mathbf{B})$ if for every i all the terms t_i are normal. In particular, empty sequences of terms in context are neutral and normal.

We shall define computability predicates $Comp_{\mathbf{B},\mathbf{C}} \subseteq [\![\mathbf{B}]\!]\mathbf{C}$ by induction on the complexity of $\mathbf{B}$.

Definition 4.5.

(a) $Comp_{\langle\rangle,\mathbf{C}} = \{*\}$
(b) $Comp_{X,\mathbf{C}} = \mathcal{NE}(\mathbf{C}, X)$ for X atomic,
(c) $Comp_{B^A,\mathbf{C}} = \{\theta : \mathcal{F}_{\mathcal{X}}(-, \mathbf{C}) \times [\![A]\!] \longrightarrow [\![B]\!] \mid \forall \mathbf{D} \forall \mathbf{t} \in \mathcal{NE}(\mathbf{D}, \mathbf{C})$ $\forall u \in Comp_{A,\mathbf{D}}\ \theta_{\mathbf{D}}(\mathbf{t}, u) \in Comp_{B,\mathbf{D}}\}$, where θ is $\mathcal{P}$-natural.
(d) $Comp_{(\mathbf{B},A),\mathbf{C}} = Comp_{\mathbf{B},\mathbf{C}} \times Comp_{A,\mathbf{C}}$, where $\mathbf{B} \neq \langle\rangle$.

Lemma 4.6. *For every object* $\mathbf{A}$, *if* $\mathbf{s} \in \mathcal{NE}(\mathbf{D}, \mathbf{C})$ *and* $\mathbf{a} \in Comp_{\mathbf{A},\mathbf{C}}$ *then* $[\![\mathbf{A}]\!](\mathbf{s})(\mathbf{a}) \in Comp_{\mathbf{A},\mathbf{D}}$.

Proof. The proof is by induction on $\mathbf{A}$ using the fact that neutral terms are closed under substitution (Lemma 4.3). The only interesting case is when $\mathbf{A}$ is a single type A. Recall that for $s : D \to C$, $[\![A]\!](s) : [\![A]\!]C \to [\![A]\!]D$.

For A atomic, since $[\![A]\!] = \mathcal{Y}(A)$, the action of $[\![A]\!](s)$ is given by composition, i.e. we have $[\![A]\!](\mathbf{s})(a) = a(\mathbf{s}/\mathbf{z})$ which is neutral by the lemma, and for atomic objects this is the same as computable.

For A an exponent, say $A_1^{A_2}$, we have to check that $[\![{A_1}^{A_2}]\!](\mathbf{s})(a) \in Comp_{A_1^{A_2},\mathbf{D}}$. Using the definition of computability at an exponent, the above is equivalent to: $\forall \mathbf{E} \forall \mathbf{r} \in \mathcal{NE}(\mathbf{E}, \mathbf{C}) \forall u \in Comp_{A_2,\mathbf{E}}\ [\![{A_1}^{A_2}]\!](\mathbf{s})(a)_{\mathbf{E}}(\mathbf{r}, u) \in Comp_{A_1,\mathbf{E}}$. By equation (11) this is the same as $a_{\mathbf{E}}(\mathbf{s}(\mathbf{r}), u) \in Comp_{A_1,\mathbf{E}}$. This holds by the assumption of computability of a and u, and neutrality of $\mathbf{s}(\mathbf{r})$ by Lemma 4.3.

□

Theorem 4.7.

(i) $\theta \in Comp_{\mathbf{A},\mathbf{C}}$ *implies* $q_{\mathbf{A},\mathbf{C}}(\theta) \in \mathcal{NF}(\mathbf{C}, \mathbf{A})$;
(ii) $\mathbf{f} \in \mathcal{NE}(\mathbf{C}, \mathbf{A})$ *implies* $q^{-1}_{\mathbf{A},\mathbf{C}}(\mathbf{f}) \in Comp_{\mathbf{A},\mathbf{C}}$;
(iii) Let $\mathbf{t} \in \mathcal{F}_{\mathcal{X}}(\mathbf{A}, \mathbf{B})$. *Then for all* $\mathbf{a} \in Comp_{\mathbf{A},\mathbf{C}}$ $[\![\mathbf{t}]\!]_{\mathbf{C}}(\mathbf{a}) \in Comp_{\mathbf{B},\mathbf{C}}$.

Proof. (1) and (2) are proved simultaneously by induction on $\mathbf{A}$. The only interesting case is when $\mathbf{A}$ is a single exponential type $A_1^{A_2}$ Then, by equation (20) (and using π, π' as the obvious meta-notation) we have $q_{A_1^{A_2},\mathbf{C}}(\theta) = \lambda y^{A_2}.q_{A_1,(\mathbf{C},A_2)}(\theta_{(\mathbf{C},A_2)}(\pi, q^{-1}_{A_2,(\mathbf{C},A_2)}(\pi')))$. By induction hypothesis $q^{-1}_{A_2,(\mathbf{C},A_2)}(\pi')$ is computable since projections/variables are normal; also by definition of computability at an exponent $\theta_{\mathbf{C},A_2}(\pi, q^{-1}_{A_2,(\mathbf{C},A_2)}(\pi'))$ is then computable. By the induction hypothesis applied to $q_{A_1,(\mathbf{C},A_2)}$ we have that $q_{A_1,(\mathbf{C},A_2)}(\theta_{\mathbf{C},A_2}(\pi, q^{-1}_{A_2,(\mathbf{C},A_2)}(\pi')))$ is normal. Finally, λ-abstraction of a normal term remains normal.

Let us now prove that $q^{-1}_{A_1^{A_2},\mathbf{C}}(f) \in Comp_{A_1^{A_2},\mathbf{C}}$ for a normal f. For that we have to show that $\forall \mathbf{D}\, \forall \mathbf{t} \in \mathcal{NE}(\mathbf{D}, \mathbf{C})\, \forall u \in Comp_{A_2,\mathbf{D}}$,

$$(q^{-1}_{A_1^{A_2},\mathbf{C}}(f))_{\mathbf{D}}(\mathbf{t}, u) \in Comp_{A_1,\mathbf{D}} \quad .$$

Using equation (25) we get $(q^{-1}_{A_1^{A_2},\mathbf{C}}(f))_{\mathbf{D}}(\mathbf{t}, u) = q^{-1}_{A_1,\mathbf{D}}(f(\mathbf{t})q_{A_2,\mathbf{D}}(u))$. By the induction hypothesis $q_{A_2,\mathbf{D}}(u)$ is normal, by lemma 4.3 $f(\mathbf{t})$ is neutral, neutral applied to normal is neutral by the definition of normal terms, and finally, by the induction hypothesis applied to $q^{-1}_{A_1,\mathbf{D}}$, the whole expression is computable.

(3) is proved by induction on $\mathbf{t}$. The only interesting case is when the sequence is a single term. Also, if the term is just a variable we are again done. It remains to check two cases:

1. $t = uv : B$. Observe that by equation (17) $[\![uv]\!]_{\mathbf{C}}\mathbf{a} = [\![u]\!]_{\mathbf{C}}(\mathbf{a})(id_{\mathbf{A}}, [\![v]\!]_{\mathbf{C}}(\mathbf{a}))$. We must show $[\![u]\!]_{\mathbf{C}}(\mathbf{a})(id_{\mathbf{A}}, [\![v]\!]_{\mathbf{C}}(\mathbf{a}))$ is computable. This follows by the inductive hypothesis applied to u, neutrality of $id_{\mathbf{A}}$ and the computability of $[\![v]\!]_{\mathbf{C}}(\mathbf{a})$ (again by the inductive hypothesis), using the definition of computability at an exponent.
2. $t = \lambda x^{B_1}.h$. We must show $[\![\lambda x^{B_1}.h]\!]_{\mathbf{C}}(\mathbf{a}) \in Comp_{B_2^{B_1},\mathbf{C}}$ where $\mathbf{a} \in [\![\mathbf{A}]\!]_{\mathbf{C}}$ is computable. To show this, we must show: $\forall \mathbf{D}, \mathbf{r} \in \mathcal{NE}(\mathbf{D}, \mathbf{C}), u \in Comp_{B_1,\mathbf{D}}$ it holds that $[\![\lambda x^{B_1}.h]\!]_{\mathbf{C}}(\mathbf{a})_{\mathbf{D}}(\mathbf{r}, u) \in Comp_{B_2,\mathbf{D}}$. But $[\![\lambda x^{B_1}.h]\!]_{\mathbf{C}}(\mathbf{a})_{\mathbf{D}}(\mathbf{r}, u) = [\![h]\!]_{\mathbf{D}}([\![A]\!](\mathbf{r})(\mathbf{a}), u)$ (equation (16)). The right hand side is indeed computable by the inductive hypothesis applied to h and Lemma 4.6.

□

Corollary 4.8.

$$\mathbf{nf}(t) = q_{B,\mathbf{A}}([\![t]\!]_{\mathbf{A}}((q^{-1}_{\mathbf{A}\mathbf{A}})(\mathbf{x} : \mathbf{A} \triangleright \mathbf{x} : \mathbf{A})))$$

is an element of $\mathcal{NF}(\mathbf{A}, B)$.

Proof. Observe that $(\mathbf{x} : \mathbf{A} \triangleright \mathbf{x} : \mathbf{A})$ is a finite sequence of variables, hence computable. The result follows by using (in order) statements (ii), (iii), (i) of the theorem above. □

Observe that until now, we have only used computability techniques. However it is worth remarking that if we appeal to the Church-Rosser theorem, we can prove the following result:

Corollary 4.9. *Every element of* $\mathcal{NF}(\mathbf{A}, \mathbf{B})$ *is* $\mathbf{nf}(g)$*, for some* g*.*

5. Historical Background and Related Work

Our method is an example of normalization by intuitionistic model construction, a method going back to (Martin-Löf 1975a; Martin-Löf 1975b). The general idea is to prove normalization by first interpreting a term in a suitable model and then map this interpretation back to the normal form of the term. By working in an intuitionistic framework one ensures that the normalization function thus obtained is an algorithm.

Martin-Löf's approach was investigated further by (T. Coquand and P. Dybjer 1997). They formulated the abstract conditions NF1 and NF2 (for the special case that $\equiv$ is syntactic equality of terms, so that NF3 and NF4 are trivially satisfied). Moreover, they focussed on algebraic aspects and used the fact that syntax modulo *conversion* is a free model and hence has a unique homomorphic interpretation into any other model. They also related Martin-Löf's construction to the glueing construction from category theory, especially as used by (Lafont 1988).

In (Martin-Löf 1975a; Martin-Löf 1975b), he introduced the technique for normalization in typed combinatory logic and weak λ-calculus. The same general technique was used to construct a normalization algorithm for simply typed $\lambda\beta\eta$-calculus by (Berger and Schwichtenberg 1991). They inverted the interpretation function into the set-theoretic model. They were also able to generalize Friedman's completeness theorem (Friedman 1975).

Another normalization algorithm for the simply typed λ-calculus (actually, a variant with explicit substitutions) was presented by (C. Coquand 1993). Her algorithm is similar to Berger and Schwichtenberg's but algebraically cleaner. It is obtained by inverting an interpretation function into a Kripke model.

Then (Altenkirch, Hofmann, Streicher 1995) gave a "categorical reconstruction" of the work of Berger and Schwichtenberg and C. Coquand in terms of inversion of an interpretation functor into the category of presheaves. The fact that the Yoneda functor preserves ccc-structure plays a key role in their reconstruction.

The present paper has exploited and reformulated the insights of Altenkirch, Hofmann, and Streicher. The crucial difference is due to our use of $\mathcal{P}$-category theory, which enables us to solve the word problem for cccs by more purely categorical means. In particular we do not need to introduce syntactic notions such as the sets of normal and neutral terms as in their proof and in Girard's computability methods (cf. (Girard, Lafont, Taylor 1989)). Altenkirch, Hofmann, and Streicher also used a "twisted" variant of the glueing construction, which is unnecessary here. Although for us it is a subsidiary concern, we have included a proof that our normal form function actually returns long $\beta\eta$-normal forms. This proof uses syntactic techniques related to traditional computability/glueing arguments.

$\mathcal{P}$-category theory may be of interest in its own right. An interesting feature is that a $\mathcal{P}$-category only comes equipped with a notion of equality of arrows but not with an equality of objects. It may also be the appropriate way to develop category theory inside a constructive framework such as Martin-Löf type theory. As such it provides an alternative to $\mathcal{E}$-category theory (category theory where each hom-set is equipped with an equivalence relation) as studied by (Aczel 1993), (Huet and Saibi 1995), and (Duval and Reynaud 1994). $\mathcal{E}$-categories were also studied abstractly by (Lack 1995), who shows them to be bicategories enriched over a monoidal bicategory.

The fact that the $\mathcal{P}$-category theory we use can be formalized (programmed) in Martin-Löf type theory is one way of ensuring that our normalization function is indeed an algorithm. Of course, $\mathcal{P}$-category theory can equally well be understood in an ordinary (set-theoretic) way, but then we are left with the problem of showing that our normalization function is computable. Although ultimately all our constructs are given inductively (and therefore have an informal constructive character) to show that they are algorithms in a precise recursion-theoretic sense would require some encoding. We can use here, for example, the result that Martin-Löf type theory has a recursion-theoretic model, see (Beeson 1985).

Our $\mathcal{P}$-categorical setting also highlights the connection between our technique and the abstract approach to coherence problems in ordinary category theory described for example by (Joyal and Street 1993), (Power 1989), and (Gordon, Power, and Street 1996). As in these references, it is an essential feature of our approach that we use a $\mathcal{P}$-version of Yoneda to embed our free $\mathcal{P}$-ccc in a "stricter" $\mathcal{P}$-ccc of $\mathcal{P}$-presheaves.

The paper by (Beylin and Dybjer 1996) shows how to go from a reduction-free normalization proof for monoids to a proof of coherence for monoidal categories. They first solve the word problem for monoids using a simple version of the technique employed in the present paper. Then they show coherence for monoidal categories by examining the formal proof objects (obtained from a formalization in Martin-Löf type theory) of this proof of normalization. The proof obtained in this way is much like Joyal and Street's.

A preliminary version of the results reported in this paper was obtained in December 1995 and presented in seminars in Göteborg, Montreal, and Cambridge during February and March, and at the Workshop on Constructive Programming in Kyoto in April 1996. This version was based on $\mathcal{E}$-category theory. But it then became clear to us that it would be more elegant to rework the result in the context of $\mathcal{P}$-category theory. The reason is that $\mathcal{P}$-category theory makes it possible to identify the underlying "data parts" of the ccc-structures on presheaves over terms modulo $\beta\eta$-convertibility and terms modulo α-congruence respectively. This new version was presented at the Peripatetic Seminar on Sheaves and Logic in Dunquerque in July, at the Seventh Scandinavian Logic Symposium in Uppsala in August, and at the LOGSEM Workshop in Birmingham in September, 1996.

6. Acknowledgements

The work on this paper has been done while D. Čubrić was an NSERC postdoctoral fellow at the Department of Pure Mathematics of Cambridge University, associated to Martin Hyland. He would like to thank them all. Part of the work on this paper was done at the Newton Institute while the second and third author participated in the programme on Semantics of Computation during the autumn of 1995. We gratefully acknowledge the support of the Newton Institute. P. Dybjer also wishes to acknowledge support from ESPRIT Basic Research Action 6811 Categorical Logic in Computer Science (CLiCS-II) and from TFR, the Swedish Technical Research Council. P. Scott was also partially supported by an operating grant from the Natural Sciences and Engineering Research Council of Canada (NSERC). The authors would like to thank the anonymous referee for careful reading and detailed comments.

References

P. Aczel, Galois: a theory development project, in: A report on work in progress for the Turin meeting on the *Representation of Logical Frameworks*, 1993.

T. Altenkirch, M. Hofmann, T. Streicher, Categorical reconstruction of a reduction-free normalisation proof, *Proc. CTCS '95* Springer Lecture Notes in Computer Science, Vol. 953, pp. 182-199.

H. P. Barendregt. *The Lambda Calculus*, Studies in Logic, Vol. 103, North- Holland, 1984.

H. P. Barendregt, Lambda Calculus with Types, *Handbook of Logic in Computer Science*, Vol. 2, ed. by S. Abramsky, D. Gabbay, T. Maibaum. Oxford U. Press, 1992.

M. Beeson. *Foundations of Constructive Mathematics*, Springer-Verlag, 1985.

U. Berger and H. Schwichtenberg, An inverse to the evaluation functional for typed λ-calculus, *Proc. of the 6th Annual IEEE Symposium of Logic in Computer Science*, 1991, pp. 203-211.

I. Beylin and P. Dybjer, Extracting a proof of coherence for monoidal categories from a proof of normalization for monoids, in *Types for Proofs and Programs* (= TYPES '95), Springer Lecture Notes in Computer Science 1158, Stefano Berardi and Mario Coppo, eds, 1996, pp.47-61

C. Coquand, From Semantics to Rules: a Machine Assisted Analysis, in : *Proceedings of CSL '93*, Egon Börger, Yuri Gurevich and Karl Meinke, eds. Springer Lecture Notes in Computer Science, 832 (1993).

T. Coquand and P. Dybjer, Intuitionistic Model Constructions and Normalization Proofs , *Math. Structures in Computer Science*, Vol.7, No.1 (1997), pp. 75-94.

R. Crole. *Categories for Types*, Cambridge Mathematical Textbooks, 1993

D. Čubrić, Embedding of a free cartesian closed category into the category of Sets , to appear in *J. Pure and Applied algebra.*

R. Di Cosmo. *Isomorphism of Types: from λ-calculus to information retrieval and language design*, Birkhäuser, 1995.

D. Duval and J-C Reynaud, Sketches and computation I: basic definitions and static evaluation, *Mathematical Structures in Computer Science*, Vol. 4, No. 2 (1994), pp. 185-238 .

H. Friedman, Equality between functionals, *Logic Colloquium '73*, Springer Lecture Notes in Mathematics, Vol. 453 1975, pp. 22-37.

J.Y. Girard. *Proof Theory and Logical Complexity*, Vol. 1, Bibliopolis, (1987).

J.Y. Girard, Y. Lafont, P.Taylor. *Proofs and Types*, Cambridge Tracts in Theoretical Computer Science 7, (1989).

R. Gordon, A. Power, and R. Street. *Coherence for tricategories*, Memoirs of the American Mathematical Society, 1996.

V. Harnik and M. Makkai, Lambek's categorical proof theory and Läuchli's abstract realizability, *J. Symbolic Logic* 57 (1992), pp. 200-230.

G. Huet. Résolution d'équations dans les langages d'ordre $1, 2, \ldots, \omega$. Thèse d'Etat, U. Paris VII, 1976.

G. Huet and A. Saibi , Constructive Category Theory, in: *Proceedings of the Joint CLICS-TYPES Workshop on Categories and Type Theory* (Göteborg), Jan. 1995.

A. Joyal, R. Street, Braided tensor categories , *Advances in Mathematics*, vol.102. no. 1 Nov.1993, pp. 20-79.

J-P. Jouannaud and M. Okada, A Computation Model for Executable Higher-Order Algebraic Specification Languages, LICS 6 (sixth Annual IEEE Symp. on Logic in Computer Science), 1991, pp. 350-361.

G. M. Kelly. *Basic Concepts of Enriched Category Theory*, London Math. Soc. Lecture Notes 64, Camb. Univ. Press, 1982.

S. G. Lack. *The algebra of distributive and extensive categories.* PhD thesis, University of Cambridge, 1995.

Y. Lafont. Logiques, catégories et machines. Thèse de doctorat, Université Paris VII, 1988.

J. Lambek, Deductive Systems and Categories I, *J. Math. Systems Theory* **2**, pp. 278-318.

J. Lambek and P. J. Scott. *Introduction to Higher Order Categorical Logic*, Cambridge Studies in Advanced Mathematics **7**, Cambridge University Press, 1986.

S. Mac Lane. *Categories for the Working Mathematician*, Graduate Texts in Mathematics 5, Springer-Verlag, 1971.

P. Martin-Löf, An Intuitionistic Theory of Types: Predicative Part, in *Logic Colloquium '73*, H. E. Rose and J. C. Shepherdson, eds., North-Holland, 1975, pp. 73-118.

P. Martin-Löf , About Models for Intuitionistic Type Theories and The Notion of Definitional Equality , in: *Proceedings of the 3rd Scandinavian Logic Symposium*, S. Kanger, ed., North-Holland, 1975, pp. 81-109.

J. C. Mitchell, Type Systems for Programming Languages, in: *Handbook of Theoretical Computer Science*, Vol.B, (*Formal Models and Semantics*), J. Van Leeuwen, ed., North-Holland, 1990 pp. 365-458.

J. C. Mitchell and P. J. Scott,Typed Lambda Models and Cartesian Closed Categories, in *Contemp. Math.: Categories in Computer Science and Logic* (J. Gray and A. Scedrov, eds), **92** (1989), pp. 301-316.

A. Pitts, Categorical Logic , in *Handbook of Logic in Computer Science*, S. Abramsky and D. M. Gabbay and T. S. E. Maibaum, eds. Oxford University Press, 1997, Vol. 6 (to appear)

J. Power, A general coherence result , *J. Pure and Applied Algebra* 57, 165-173, 1989.

R. Statman, Logical Relations and the Typed Lambda Calculus, *Inf. and Control* **65** (1985), pp. 85-97.

A. S. Troelstra. *Metamathematical investigations of intuitionistic arithmetic and analysis*, Springer LNM 344, 1973.

7. Appendix: λ-Theories

In this Appendix we apply our methods to the word problem for typed λ-calculus with additional operations and axioms, i.e. to λ-*theories*. This shows the robustness of our method: the extraction of the normalization algorithm **nf** from a proof of freeness is done in essentially the same way as before (and NF1 follows for the same general reasons). What changes is that we work with new notions of freeness which correspond to the respective notions of theory. Moreover, the proof of NF2 is also essentially unchanged. NF3 is (as always) trivial. But proving NF4, the decidability of $\equiv$, involves additional syntactical considerations. From the examples which follow (see also Remark 7.10), it should be clear that our method gives an algorithm **nf** for *any* λ-theory. However, in general, this **nf** does not have to satisfy $NF4$. The examples below will satisfy $NF4$ relative to the initial theory.

We consider several theories:

(i) A λ-calculus T^λ_f with additional operations but with no additional axioms. This corresponds to a free ccc with free arrows (Harnik and Makkai 1992; Lambek and Scott 1986; Čubrić 199-).

(ii) A λ-calculus T^λ_{cat} with additional operations and axioms such that all the types occurring in them are sorts and moreover their contexts contain exactly one variable typed by a sort. This corresponds to the free ccc generated by a category.

(iii) A λ-calculus T^λ_{cart} with additional operations and axioms but in which all types are sorts. This corresponds to the free ccc generated by a cartesian category.

In each case we will obtain a solution to a word problem by constructing our normal form algorithm. In cases (ii) and (iii) this solution will be relative to the solution of the word problem for the generating category (cartesian category, respectively). Our (relative) proofs of NF4, however, are syntactical. The main technical result of this Appendix, Theorem 7.5, resembles Cut-Elimination and is used to reduce the number of cases examined. Of course to even apply our methods, we require appropriate notions of $\mathcal{P}$-category, $\mathcal{P}$-freeness, etc. *modulo* a λ-theory.

We now introduce the appropriate machinery.

7.1. *λ-theories*

In order to deal with additional operations and axioms we slightly extend our definition of λ-calculus (cf. Definition 3.1).

To *Terms* we add the following: (a) to every sequence of types $\mathbf{A}$ and every type B we associate (a possibly empty) set of operations $Op(\mathbf{A};B)$. (b) We also add the following term-formation rule:

$$(Op)\quad \frac{\Sigma \rhd t_1 : A_1 \ldots \Sigma \rhd t_n : A_n \quad c \in Op(A_1,\ldots,A_n;B)}{\Sigma \rhd c(t_1,\ldots,t_n) : B}$$

For a given Σ and B the set of all terms with context Σ and type B is denoted $Term(\Sigma, B)$.

To the *Equations between terms* we add a specified set of *basic equations* $BEq(\Sigma, B) \subseteq Term(\Sigma, B) \times Term(\Sigma, B)$, for every Σ, B. We also add the following two rules

$$(SubOp)\quad \frac{c \in Op(A_1,\ldots,A_n;B) \quad \Sigma \rhd s_1 = u_1 : A_1 \;\ldots\; \Sigma \rhd s_n = u_n : A_n}{\Sigma \rhd c(s_1,\ldots,s_n) = c(u_1,\ldots,u_n) : B}$$

$$(Beq)\quad \frac{(t,s) \in BEq(x_1^{A_1},\ldots,x_n^{A_n};B) \quad \Sigma \rhd u_1 = v_1 : A_1 \;\ldots\; \Sigma \rhd u_n = v_n : A_n}{\Sigma \rhd t(u_1/x_1,\ldots,u_n/x_n) = s(v_1/x_1,\ldots,v_n/x_n) : B}$$

The pair $Sig = (Sorts, Operations)$ we call a *signature* of a λ-calculus, the pair $\mathcal{L} = (Types, Terms)$ we call its *language*, the basic equations BEq we call its *axioms* and, finally, a language together with its set of provable equations we call a *λ-theory*.

(i) T_f^λ is now defined as a λ-theory such that for every Σ and B, $BEq(\Sigma, B) = \emptyset$.

(ii) T_{cat}^λ is now defined as a λ-theory such that for every $\mathbf{A}$ and B, $Op(\mathbf{A}, B) = \emptyset$ unless B is a sort and $\mathbf{A}$ is a single sort. Moreover, we require that no basic equation $\Sigma \rhd s = t : B$ can contain exponent types, that is, all subterms of s and t are typed by sorts and that Σ is a single variable typed by a sort.

(iii) T_{cart}^λ is the following λ-theory: for every $\mathbf{A}$ and every B, $Op(\mathbf{A}, B) = \emptyset$ unless B is a sort and $\mathbf{A}$ is a sequence of sorts. Also we require that no basic equation $\mathbf{x} : \mathbf{A} \rhd s = t : B$ can contain exponent types (here, we do not require $\mathbf{x} : \mathbf{A}$ to be a single variable but only that $\mathbf{A}$ is a finite sequence of sorts).

Obviously, T_{cat}^λ is a special case of T_{cart}^λ but T_f^λ is not a special case of either of them.

7.2. **nf** *and NF1 for theories*

To each theory T above, we associate a $\mathcal{P}$-category just as we did in Definition 3.5. The key difference is that we have to include the additional operations in the construction of terms and that in the second and third case we have to include provability from the basic equations. Therefore we define the relation $\sim_T$ in $(\mathcal{F}_T, \sim_T)$ as follows:

$$(\mathbf{x} : \mathbf{A} \;\rhd\; \mathbf{t} : \mathbf{B}) \sim_T (\mathbf{y} : \mathbf{A} \;\rhd\; \mathbf{s} : \mathbf{B}) \quad \text{iff for every } i \quad T \vdash \mathbf{z} : \mathbf{A} \;\rhd\; t_i(\mathbf{z}/\mathbf{x}) = s_i(\mathbf{z}/\mathbf{y})$$

where T stands for any of T_f^λ, T_{cat}^λ or T_{cart}^λ and $T \vdash \;\cdots$ denotes provability from T. In each case we get a $\mathcal{P}$-ccc (cf. also Remark 3.10).

As before, we will construct the normalization algorithm **nf** from a proof of $\mathcal{P}$-freeness (in the appropriate sense) and the proof that Yoneda preserves $\mathcal{P}$-ccc structure. This amounts to defining $[\![-]\!]$ and q, q^{-1} as before, except that we now must define the effect of $[\![-]\!]$ on terms of the form $c(\mathbf{t})$ too: this is the only change in the previous explicit definition of $[\![-]\!]$. Therefore we add to the equations (13-17) the following. For every $c \in Op(\mathbf{A}, B)$ and every $(\mathbf{w} : \mathbf{D} \rhd \mathbf{t} : \mathbf{A})$

$$[\![\mathbf{w} : \mathbf{D} \rhd c(\mathbf{t}) : B]\!] = [\![c]\!][\![\mathbf{w} : \mathbf{D} \rhd \mathbf{t} : \mathbf{A}]\!] \tag{27}$$

where $[\![c]\!] : [\![\mathbf{A}]\!] \to [\![B]\!]$ is defined as:

$$[\![c]\!]_{\mathbf{C}}(\mathbf{a}) = q^{-1}_{B,\mathbf{C}}(c(q_{\mathbf{A},\mathbf{C}}(\mathbf{a}))) \tag{28}$$

In categorical terminology, we map the generating arrow $c : \mathbf{A} \to B$ to $q_B^{-1}\mathcal{Y}(c)q_{\mathbf{A}} : [\![\mathbf{A}]\!] \to [\![B]\!]$.

Therefore, just as before, we have our algorithm $\mathbf{nf}$. Also, as before, we easily have $NF1$, i.e. for every $\mathbf{t}$, $\mathbf{nf}(\mathbf{t}) \sim_T \mathbf{t}$.

7.3. *NF2 and NF3 for theories*

To prove the other properties we first have to define a subrelation $\equiv_T$ of $\sim_T$, where T stands for any of T^λ_f, T^λ_{cat}, T^λ_{cart}. As in our previous work, $\equiv_T$ is essentially "provability without $\beta\eta$, up to α-congruence". More precisely, for each λ-theory T above, we first define a "restricted" theory T^α (i.e. T^α_f, T^α_{cat}, T^α_{cart}) which is the same as T except that β and η are not included.

As in Definition 3.11 we define (in each of the above cases) the $\mathcal{P}$-category $(\mathcal{F}_T, \equiv_T)$ to have the same objects and arrows as $(\mathcal{F}_T, \sim_T)$ but the relation on arrows is defined as follows:

$$(\mathbf{x} : \mathbf{A} \rhd \mathbf{t} : \mathbf{B}) \equiv_T (\mathbf{y} : \mathbf{A} \rhd \mathbf{s} : \mathbf{B}) \text{ iff for every } i \;\; T^\alpha \vdash \mathbf{z} : \mathbf{A} \rhd t_i(\mathbf{z}/\mathbf{x}) = s_i(\mathbf{z}/\mathbf{y}) \; .$$

Obviously, in all three cases $\equiv_T \subseteq \sim_T$ *i.e.* condition $NF3$ is satisfied. Let us now show that $NF2$ holds as well.

As before, we would now like to show that in each of the above cases we have a $\mathcal{P}$-ccc functor $[\![-]\!]^\alpha : (\mathcal{F}_T, \sim_T) \to \mathcal{P}Set^{(\mathcal{F}_T, \equiv_T)^{op}}$. The first thing which we want to ensure is the following:

Lemma 7.1. *For every theory T and $c \in Op(\mathbf{A}, B)$, $[\![c]\!]^\alpha$ defined using the same formula as in Definition 28 is a $\mathcal{P}$-arrow in $\mathcal{P}Set^{(\mathcal{F}_T, \equiv_T)^{op}}$.*

Proof. By examining the proof of Lemma 3.15 we could notice that $q_{\mathbf{A}}$ and q_B^{-1} are $\equiv_T$-natural here as well. For the rest we use the rule $(SubOp)$, or in other words that $\mathcal{Y}c$ is $\equiv_T$-natural as well. □

The next thing which we have to ensure is that $[\![-]\!]^\alpha$ preserves the axioms. But for the case of $T = T^\lambda_f$ (the "completely free" case) we are done since there are no axioms. Let us consider the two other cases.

Lemma 7.2. *Let T be either T^λ_{cat} or T^λ_{cart}. Let $\mathbf{x} : \mathbf{A} \rhd s = t : B$ be a basic equation in T . Then, $[\![s]\!]^\alpha \equiv_T [\![t]\!]^\alpha$ in $\mathcal{P}Set^{(\mathcal{F}_T, \equiv_T)^{op}}$.*

Proof. First we should notice the following easy

Fact: For every "T-restricted" $\mathbf{A}$ we have $q_{\mathbf{A}} q_{\mathbf{A}}^{-1} \equiv_T 1_{\mathcal{F}_T(-,\mathbf{A})}$ where "T^{λ}_{cat}-restricted" means that $\mathbf{A}$ is a single sort and "T^{λ}_{cart}-restricted" means that $\mathbf{A}$ is a finite sequence of sorts.

By induction, and using the above fact, we can show that for every T-restricted arrow we have $[\![\,\mathbf{t}\,]\!]^{\alpha} \equiv_T q_{\mathbf{B}}^{-1}(\mathcal{Y}\mathbf{t})q_{\mathbf{A}}$.

To prove that $[\![\,s\,]\!]^{\alpha} \equiv_T [\![\,t\,]\!]^{\alpha}$ it is enough to show that $\mathcal{Y}s \equiv_T \mathcal{Y}t$. This follows from the fact that rule Beq is included in T^{α}. □

Therefore, by the appropriate freeness of $(\mathcal{F}_T, \sim_T)$ we indeed have a $\mathcal{P}$-ccc functor $[\![-]\!]^{\alpha} : (\mathcal{F}_T, \sim_T) \to \mathcal{P}Set^{(\mathcal{F}_T,\equiv_T)^{op}}$ defined as above.

Using exactly the same reasoning as in Section 3.4 we can now show the property $NF2$ for all three cases of T, *i.e.* if $s \sim_T t$ then $\mathbf{nf}(s) \equiv_T \mathbf{nf}(t)$. The only remaining thing now is to see whether $\equiv_T$ is decidable.

7.4. *NF4 for theories*

To investigate $\equiv_T$ is the same as to investigate provability in T^{α}, for each theory $T = T^{\lambda}_f, T^{\lambda}_{cat}, T^{\lambda}_{cart}$. As before, ordinary α-congruence among terms is denoted by $\equiv$.

First, we separate the easiest case T^{α}_f. As in Section 3.4, provability here is exactly the same as α-congruence and therefore NF4 holds. Now we shall prove NF4 for the other two cases. We begin with some syntax.

Recall the notion of a substitution context, i.e. "term with a hole". Here we need to consider several holes at once; we denote a term with precisely n distinct holes by $C[\sqcup_1, \ldots, \sqcup_n]$ (strictly speaking, the indexing of the holes is redundant.) Our holes are typed (although we try to avoid explicitly writing types); that means that we can plug into the holes only terms of the appropriate type.

A term is called *simple* if it contains neither exponents nor λ nor application. It is easy to see that every term typed by a sort arises by substitution from a simple term with holes, i.e. for any term typed by a sort $\Sigma \triangleright g : Z$ there is a simple term with holes $C[\sqcup_1, \ldots, \sqcup_n]$ containing no variables and a sequence of terms $\Sigma \triangleright u_i : X_i$ such that $g \equiv C[u_1, \ldots u_n]$ and where each u_i is either an application or a sorted variable. Such a term with holes $C[\sqcup_1, \ldots, \sqcup_n]$ is called the *maximal simple head* of $\Sigma \triangleright g : Z$. For any given term $g : Z$, there is an effective method to find its maximal simple head $C[\ldots]$ and the associated terms u_i satisfying the above conditions. Moreover, they are uniquely determined. For example, assuming c and d are operation symbols typed by sorts, $c(\sqcup_1, \sqcup_2, d(\sqcup_3))$ is the maximal simple head of both $x : X, y : Y \triangleright c(x, x, d(uv)) : Z$ and $x : X, y : Y \triangleright c(x, y, d(uv)) : Z$. From now on, the notation $F[u_1, \ldots, u_l]$ always implies that $F[\ldots]$ is the appropriate maximal simple head.

We now introduce two algebraic theories T_{alg} and T_{un}, where T_{alg} refines T^{λ}_{cart} and T_{un} refines T_{alg} :

- T_{alg}: Types are only the sorts of T^{λ}_{cart} while the terms are those terms of T^{λ}_{cart} that do not contain exponents. The equations of T_{alg} are formed by only those rules from T^{λ}_{cart} which do not contain exponents.

- T_{un}: T_{un} is a further restriction of T_{alg} as well as of T^{λ}_{cat} such that the contexts appearing can only have a single sorted variable.

It is folklore that T_{alg} corresponds precisely to the internal theories of cartesian categories and that T_{un} corresponds to the internal theories of categories.

Remark 7.3. Since we work in $\mathcal{P}$-category theory it is important to realize that the corresponding $\mathcal{P}$-statements hold: for example, T_{alg} corresponds precisely to the internal theories of $\mathcal{P}$-cartesian categories as well.

Let $\mathcal{E}q\{\Sigma \triangleright u_i : A \,|\, i = 1, \ldots m\}$ denote a finite set of *equations* among u's such that for every i and j, $\Sigma \triangleright u_i = u_j : A$ is in the equivalence closure of $\mathcal{E}q\{\Sigma \triangleright u_i : A \,|\, i = 1, \ldots m\}$. Although, the above definition would be sufficient for the development below, we will further specialize equations to be in *circular form*, i.e. we assume our set of equations is written:

$$\mathcal{E}q\{\Sigma \triangleright u_i : A \,|\, i = 1, \ldots m\} = \{\Sigma \triangleright u_1 = u_2 : A, \Sigma \triangleright u_2 = u_3 : A, \ldots, \Sigma \triangleright u_m = u_1\}$$

(altogether m equations). From now on whenever we write $\mathcal{E}q\{...\}$ we assume that the equations are in the above special form.

Next we notice that the following rule is admissible in T^{λ}_{cart}:

$$(ASub) \quad \frac{\mathcal{E}q\{\Sigma \triangleright u_i : X_k \,|\, i \in \varphi^{-1}(k)\}_{k=1,\ldots,n}}{\Sigma \triangleright F[u_1, \ldots, u_l] = G[u_{l+1}, \ldots, u_{l+l'}] : Y}$$

where $\varphi : \{1, \ldots, l + l'\} \to \{1, \ldots, n\}$ with the additional restrictions: $T_{alg} \vdash x_1 : X_1, \ldots x_n : X_n \triangleright F[x_{\varphi(1)}, \ldots, x_{\varphi(l)}] = G[x_{\varphi(l+1)}, \ldots, x_{\varphi(l+l')}] : Y$, each u_i is either a sorted variable or an application and all the variables of $F[x_{\varphi(1)}, \ldots, x_{\varphi(l)}]$ and $G[x_{\varphi(l+1)}, \ldots, x_{\varphi(l+l')}]$ are exactly $x_{\varphi(1)}, \ldots, x_{\varphi(l+l')}$. (All this implies that $F[...]$ and $G[...]$ are maximal simple heads of the appropriate terms.) Moreover, in case that $k \notin Image(\varphi)$ we assume that $\mathcal{E}q\{\Sigma \triangleright u_i : X_k \,|\, i \in \varphi^{-1}(k)\}$ denotes a single equation $\Sigma \triangleright u = u : X_k$. Since such a term u is not going to cause any new occurrences in the conclusion, it is not important which term it is—the only important thing is that such a term has to exist.

Similarly, the rule below is admissible in T^{λ}_{cat}:

$$(USub) \quad \frac{\Sigma \triangleright u = v : X}{\Sigma \triangleright f(u/x) = g(v/x)}$$

with the additional restrictions: $T_{un} \vdash x : X \triangleright f = g : Y$ and u and v are either sorted variables or applications.

Lemma 7.4. *Let T'^{λ}_{cart} denote the same theory as T^{λ}_{cart} but such that the rules $(SubOp)$ and (Beq) are replaced by $(ASub)$. Then, both theories have the same equational consequences. Similarly, let T'^{λ}_{cat} denote the same theory as T^{λ}_{cat} but such that the rules $(SubOp)$ and (Beq) are replaced by $(USub)$. Then, both theories have the same equational consequences. Moreover (and even more important for us) both statements hold for T^{α}_{cart} and T^{α}_{cat}.*

Proof. We leave to the reader, as an easy exercise, to show how to replace each occurrence of $(SubOp)$ and (Beq) by a combination of the other rules plus the new rules.

To prove the converse we use that $F[...]$ and $G[...]$ in the rule $(Asub)$ are maximal simple heads. □

Now we want to further reduce the number of cases which will appear in our proofs.

Theorem 7.5. *Let T'^{α} denote either T'^{α}_{cart} or T'^{α}_{cat}. Then $(Transitivity)$ and $(Symmetry)$ are admissible rules in T'^{α}.*

Both the statement and the proof of Theorem 7.5 closely resemble cut elimination. Moreover, the purpose is basically the same: to obtain a kind of subterm/subformula property. Since the proof is rather long and technical, we leave it to Subsection 7.5 below.

In the following proposition we reduce equality in T^{α} to the equality of simple terms.

Proposition 7.6. *Let T be T^{λ}_{cat} or T^{λ}_{cart} and let T^{-} denote T_{un} and T_{alg} respectively.*

(a) *If $T^{\alpha} \vdash \Sigma \triangleright x = g : B^A$ then $g \equiv x$.*

(b) *If $T^{\alpha} \vdash \Sigma \triangleright \lambda x^A.f = g : B^A$ then there exists h such that $g \equiv \lambda x^A.h$ and $T^{\alpha} \vdash \Sigma, x : A \triangleright f = h : B$.*

(c) *If $T^{\alpha} \vdash \Sigma \triangleright u_1 v_1 = u_2 v_2 : B$ then there are the following two possibilities: either $T^{\alpha} \vdash \Sigma, x : B, y : B \triangleright x = y : B$ and B is a sort, or $T^{\alpha} \vdash \Sigma \triangleright u_1 = u_2 : B^A$ and $T^{\alpha} \vdash \Sigma \triangleright v_1 = v_2 : A$.*

(d) *If $T^{\alpha} \vdash \Sigma \triangleright x = uv : X$ where X is a sort then $T^{\alpha} \vdash \Sigma, y : X \triangleright x = y : X$.*

(e) *If $T^{\alpha} \vdash \Sigma \triangleright C[u_1, \ldots, u_l] = D[u_{l+1}, \ldots u_{l+l'}] : Z$ such that $C[\sqcup_1, \ldots, \sqcup_l]$ and $D[\sqcup_{l+1}, \ldots, \sqcup_{l+l''}]$ are the appropriate maximal simple heads and such that this case is not covered by (c) nor (d). Then, there exists a function $\varphi : \{1, \ldots, l + l'\} \to \{1, \ldots, n\}$ and a sequence of sorts $X_1, \cdots, X_n$ so that $T^{-} \vdash x_1 : X_1, \ldots x_n : X_n \triangleright C[x_{\varphi(1)}, \ldots, x_{\varphi(l)}] = D[x_{\varphi(l+1)}, \ldots x_{\varphi(l+l')}] : Z$ where for every i and j if $\varphi(i) = \varphi(j)$ then $T^{\alpha} \vdash \Sigma \triangleright u_i = u_j : X_i$ and moreover, for every $k \notin Image(\varphi)$ there is a term in context Σ of sort X_k. In the case when T is T^{λ}_{cat} (and $T^{-} = T_{un}$) $n = 1$.*

Let us first observe that the case (e) would hold even if we were not to exclude its overlapping with (c) and (d) but this exclusion is helpful when extracting an algorithm out of the proposition.

Proof. By Theorem 7.5, we may assume that proofs do not contain $(Transitivity)$ or $(Symmetry)$. We then proceed by induction on the height of a proof by a straightforward analysis of the last rules (for the precise definition of the *height* of a proof , see Definition 7.12 in Section 7.5 below.) □

To construct an algorithm from the above proposition, we would have to be able to determine the sequence $X_1, \ldots, X_n$ (in case (e)) not from the proof of $\Sigma \triangleright f = g : Z$ but only from the fact that $\Sigma \triangleright f : Z$ and $\Sigma \triangleright g : Z$ are well formed terms. In the case of T^{α}_{cat} (and T_{un}) it is trivial. However, in the case of T^{α}_{cart} (and T_{alg}) this is a difficult question, and in this case we will introduce a new assumption. But before we do that, let us prove a lemma.

Lemma 7.7. *For a type C we define its "right sort" $R(C)$ as follows: $R(A^B) = R(A)$ and $R(X) = X$. Then* *(i)* *If* $T'^{\alpha} \vdash x : X, y : X, y_1 : A_1, \ldots, y_n : A_n \triangleright x = y : X$ *then* $T_{alg} \vdash x : X, y : X, x_1 : R(A_1), \ldots, x_n : R(A_n) \triangleright x = y : X$.

(ii) *If* $T_{alg} \vdash x_1 : X_1, \ldots, x_n : X_n \triangleright x = y : X_1$ *then* $T'^{\alpha} \vdash x_1 : X_1, \ldots, x_n : X_n \triangleright x = y : X_1$.

Proof. The second statement is trivial. To prove the first one we observe that with several lambda abstractions we can always produce a term in context $x_i : R(A_i)$ of the type A_i. Substituting these terms in $x : X, y : X, y_1 : A_1, \ldots, y_n : A_n \triangleright x = y : X$ we obtain $T'^{\alpha} \vdash x : X, y : X, x_1 : R(A_1), \ldots, x_n : R(A_n) \triangleright x = y : X$. This latter equality is between two simple terms, so it can be shown that it had to hold in T_{alg}, as follows: the proof is by induction on the length of the derivation. By inspection, the only possible last rule is (Beq), for which the result follows by induction assumption. $\square$

Now we can introduce our additional hypothesis:

Assume: that the theory T_{alg} satisfies the "non-empty" rule *i.e.*

$$\frac{\mathbf{x} : \mathbf{X}, y : Y \triangleright s = t : Z \quad y : Y \notin FV(s,t)}{\mathbf{x} : \mathbf{X} \triangleright s = t : Z}$$

(where $FV(s,t)$ denotes the variables occurring in s or t). The appropriate variant of the assumption is always trivially satisfied for T_{un}.

With this assumption the above lemma becomes an equivalence and can be expressed as $T'_{\alpha} \vdash \Sigma \triangleright x = y : X$ iff $T_{alg} \vdash x : X, y : X \triangleright x = y : X$.

Also, the sequence $x_1 : X_1, \ldots, x_n : X_n$ in case (e) of Proposition 7.6 is reduced to variables which actually appear. Since the converse of the above proposition obviously holds, the decidability of equality in T^{α}_{cart} has been reduced to the decidability of equality in T_{alg}. In that sense we have also reduced $NF4$ for $\equiv_T$ to the existence of these decision procedures.

As we mentioned above T_{un} and T_{alg} correspond to the internal theories of categories and cartesian categories respectively. Hence we have the following:

Corollary 7.8. (Reduction of Word Problems) *In the case of* T_{un}*, our method reduces the word problem for the ccc generated by a category to the word problem of the starting category. In the case of* T_{alg}*, we reduce the word problem for the ccc generated by a cartesian category to the word problem of the starting cartesian category but only provided that in the starting cartesian category all projections are epi.*

Let us summarize the algorithm: to test whether $T^{\lambda} \vdash \Sigma \triangleright u = v : A$ we first normalize u and v and then check $T^{\alpha} \vdash \Sigma \triangleright \mathbf{nf}(u) = \mathbf{nf}(v) : A$. For this we use Proposition 7.6 and do case analysis on $\mathbf{nf}(u) : A$. First, if A is exponential, we get recursive calls corresponding to the appropriate parts of (a), (b), and (c). If A is a sort we determine the maximal simple head of $\mathbf{nf}(u)$. In case this is just a hole we get recursive calls according to the appropriate parts of (c) and (d). Otherwise we use (e), but note that in the case of T^{α}_{cart} it is a non-trivial question to determine the sequence $X_1, \ldots, X_n$ from the sorts of the holes. With the non-emptiness assumption we can take the sequence of sorts determined by the holes and try all of the finitely many φ's. Without the non-emptiness assumption we would have had to consider sequences of sorts of unbounded length.

Remark 7.9. There are several cases when the above assumption is satisfied: *e.g.* when all the sorts have a closed term. Also, if in all the axioms all the variables appear on both sides of equations.

Remark 7.10. What would happen if we were to work with an arbitrary λ-theory (i.e. arbitrary operations and basic equations allowed). Using exactly the same reasoning as in Section 7.2 we would obtain an algorithm which would satisfy NF1. But, in proving NF2, we would have to establish that $q_{AB}q_{AB}^{-1} \equiv 1$ (see the Fact in the proof of Lemma 7.2). But to establish this, we would have to include β and η into $\equiv$. In other words, we would not reduce the problem of decidability of $\sim$ to anything else but itself!

We now present an example which illustrates the need for the additional assumption on "non-emptiness".

Proposition 7.11. There is a cartesian category with a decidable word problem but such that its free ccc closure has an undecidable word problem.

Proof. The cartesian category is the one obtained from the following algebraic theory. The sorts are Y, Z and X_i, $i = 1, 2, \ldots n, \ldots$ The generating arrows are: two constants $a, b : Z$ and the following family of constants $c_i : X_{f(i)}$ where f is a total recursive function whose range is not a recursive set. The only postulated equation is $y : Y \triangleright a = b : Z$. (The rest are provable equations in a cc).

Claim 1. Equality is decidable in this cartesian category.

Proof (sketch): all the arrows are expressible as finite sequences of terms in context, and here we can generate only variables or constants. So, we only have to decide whether $u = v$ in context, where u, v are either variables or constants. In general they should be different, unless they are identical or they are $a = b$ and the context includes Y.

Consider the free ccc closure of the above theory.

Claim 2. It is undecidable in this ccc closure whether

$$w : Y^{X_j} \triangleright a = b : Z \quad .$$

Proof: we will show that $\{j \mid w : Y^{X_j} \triangleright a = b : Z\} = Range(f)$ and this will be sufficient. Obviously, if $j \in Range(f)$, say $j = f(i)$ for some i, then the constant c_i determines an arrow $1 \to X_j$. We then have an arrow $Y^{X_j} \to Y$. Using the fact that $y : Y \triangleright a = b : Z$, then $w : Y^{X_j} \triangleright a = b : Z$. For the converse, assume that $w : Y^{X_j} \triangleright a = b : Z$. Then by Proposition 7.6 (e) we have $T_{alg} \vdash x_1 : X'_1, \ldots, x_n : X'_n \triangleright a = b : Z$ and for every X'_k $k = 1, \ldots, n$ there exists a term/arrow in the ccc closure $Y^{X_j} \to X'_k$. By the proof of Claim 1 we know that one of X'_k must be Y. So, the above assumption $w : Y^{X_j} \triangleright a = b : Z$ is equivalent to the existence of an arrow in the ccc closure of the form $Y^{X_j} \to Y$. This is so iff there is an arrow in the ccc closure of the form $1 \to X_j$. The last equivalence is obvious from right to left. For the other direction one needs to examine the normal form of a term of sort Y with a free variable of type Y^{X_j}. It must be an application (it cannot be a λ, a variable, or a basic constant.) Let the application be uv. If u were not a variable it would have to be another application, etc. The leftmost term of this chain of applications would have to be a variable of higher type then Y^{X_j}. This is impossible, so u must be a variable of type Y^{X_j}. Now, v has to be of type X_j. If v is a constant we are done. Otherwise, v has to be an application but this time such a chain of applications would have to stop with the leftmost term being a variable of a "very wrong type" i.e. having the rightmost type X_j which is no good. A gluing argument following Lafont (cf. (Lafont 1988), also reproduced in (Crole 1993)) shows the fullness of the canonical embedding of a cartesian category into the free ccc generated by it. So, such an arrow $1 \to X_j$ has to be in the cartesian part. But the arrow must then arise from constants $c_i : X_j$, for some i such that $f(i) = j$; that is, $j \in \mathit{Range}(f)$. □

7.5. *Proof of Theorem 7.5*

Although the main thing is the elimination of $(\mathit{Transitivity})$, we use the elimination of $(\mathit{Symmetry})$ to simplify this proof. The following exposition is modelled on the cut-elimination proof in Girard (Girard 1987). First we introduce some definitions :

Definition 7.12.

(i) A *quasiterm* is a typed term without its explicit context.

(ii) For every quasiterm t its *degree* $d(t)$ is defined as follows: for a variable x, $d(x) = 1$; if $F[...]$ is not a single hole then $d(F[u_1, \ldots, u_l]) = \max\{d(u_i) + 1 \mid i = 1, \ldots, l\}$; $d(uv) = \max\{d(u), d(v)\} + 1$; $d(\lambda x.u) = d(u) + 1$.

(iii) The *degree* $d(P)$ of a proof P of $\Sigma \triangleright f = g : A$ in T'_α is

$$\mathit{sup}\{d(t) \mid t \text{ is a "middle" term of a } (\mathit{Transitivity}) \text{ rule in } P\}$$

and let $d(P) = 0$ if P does not contain any $(\mathit{Transitivity})$ rules.

(iv) The *height* $h(P)$ of a proof P in T'_α is defined inductively as follows: the height of an axiom (and of reflexivity) is 0; for all the other rules the height is defined as the maximum of the heights of the proofs of its assumptions increased by 1.

(v) The *width* $\gamma(t)$ of a term is defined to be 1 if the term is not typed by a sort; otherwise, $\gamma(t) = k$ where k is the number of holes in its maximal simple head (so, $\gamma(x) = \gamma(uv) = 1$).

(vi) The *width* $\gamma(P)$ of a proof is $\max\{\gamma(t) \mid t \text{ appears in } P \text{ as a subterm}\}$.

Now we can eliminate $(\mathit{Symmetry})$. We will prove the following:

Claim 1: For every proof in T'^α of $\Sigma \triangleright s = t : A$ which does not contain any $(\mathit{Symmetry})$ there is a proof in T'^α of $\Sigma \triangleright t = s : A$ of exactly the same height which does not contain the $(\mathit{Symmetry})$ rule either.

Proof of claim 1: By induction on the height of the proof observing that the equational consequences of T_{alg} and T_{un} are closed under $(\mathit{Symmetry})$. In other words: reverse each equality in the proof.

Claim 2: For every proof in T'^α of $\Sigma \triangleright s = t : A$ in which the rule $(\mathit{Symmetry})$ was used exactly once at the end, there is a shorter proof of $\Sigma \triangleright s = t : A$ which does not use any $(\mathit{Symmetry})$ rule.

Proof of claim 2 : Just use the previous claim.

Corollary 7.13. (a) If P is a proof in T'^α of $\Sigma \triangleright s = t : A$ there is a proof P' in T'^α of $\Sigma \triangleright s = t : A$ such that P' does not contain any symmetry and such that $d(P) = d(P')$, $h(P') \leq h(P)$ and $\gamma(P) = \gamma(P')$.

(b) If P is a proof in T'^{α} of $\Sigma \triangleright s = t : A$ there is a proof P' in T'^{α} of $\Sigma \triangleright t = s : A$ such that P' does not contain any symmetry and such that $d(P) = d(P')$, $h(P') \leq h(P)$ and $\gamma(P) = \gamma(P')$.

Proof. Use the previous claims. It is important to observe that in the second part of the above statement, if P did not contain a symmetry then P' can be obtained by reversing every equation in P. □

From now on assume that our T'^{α} does not contain the $(Symmetry)$ rule. We want to show that $(Transitivity)$ is also an admissible rule (in T'^{α}).

Lemma 7.14. *Given a proof P_1 of $\Sigma \triangleright f = g : A$ and a proof P_2 of $\Sigma \triangleright g = h : A$ in T'^{α} such that $d(P_1), d(P_2) < d(g) = n$ then we can construct a proof P of $\Sigma \triangleright f = h : A$ in T'^{α} such that $d(P) < n$ and $h(P) \leq \gamma(g)(h(P_1) + h(P_2))$.*

Proof. By induction on $\gamma(g)(h(P_1) + h(P_2))$. (This is a modification of Girard's approach(Girard 1987)). Recall that our rules are: $(Reflexivity)$, $(Transitivity)$ (App), (ξ) and $(ASub)$ or $(USub)$ respectively. Let us start with the easy cases.

If one of the last rules is reflexivity just take the other proof to represent P. Obviously, it satisfies all the requirements. We have 12 more cases.

Suppose now that the last rule in P_1 is $(Transitivity)$ *i.e.* suppose that P_1 is :

$$\frac{\begin{array}{cc} P_{11} & P_{12} \\ f = b & b = g \end{array}}{f = g} \; (Tran)$$

and suppose that P_2 proves $g = h$. We can apply the induction hypothesis on P_{12} and P_2 and so we obtain a proof P_2' of $b = h$ of degree $< d(g)$ and $h(P_2') \leq \gamma(g)(h(P_{12}) + h(P_2))$. Applying $(Transitivity)$ to the proof P_{11} of $f = b$ and to the previous proof of $b = h$ we obtain a proof of $f = h$ whose degree is again $< d(g)$ (since by assumption, $d(b) < d(g)$) and whose height is $\leq \max\{h(P_{11}), \gamma(g)(h(P_{12}) + h(P_2))\} + 1$. It is not difficult to see that this is $\leq \gamma(g)(h(P_1) + h(P_2))$.

A similar analysis holds in the case when P_2 ends with a transitivity.

Now we have 5 cases left. The cases when both proofs end with (App) or with (ξ) are quite easy: just use transitivity on their assumptions.

In the remaining 3 cases at least one rule ends up with $(ASub)$ (respectively $(USub)$). Assume now that P_1 ends with

$$\frac{\mathcal{E}q\{\Sigma \triangleright u_i : X_k \,|\, i \in \varphi^{-1}(k)\}_{k=1,\ldots,n}}{\Sigma \triangleright F[u_1, \ldots, u_l] = v_1 w_1 : Y}$$

where $T_{alg} \vdash x_1 : X_1, \ldots x_n : X_n \triangleright F[x_{\varphi(1)}, \ldots, x_{\varphi(l)}] = x_{\varphi(l+1)} : Y$ and where $u_{l+1} \equiv v_1 w_1$; and assume also that P_2 is

$$\frac{\begin{array}{cc} P_{21} & P_{22} \\ v_1 = v_2 & w_1 = w_2 \end{array}}{v_1 w_1 = v_2 w_2} \; (App)$$

By the proof of the above corollary, part (b), we know that we also have a proof P_2'

$$\frac{\begin{array}{cc} P_{21}' & P_{22}' \\ v_2 = v_1 & w_2 = w_1 \end{array}}{v_2 w_2 = v_1 w_1} \; (App)$$

such that P_2', P_{21}' and P_{22}' have the same characteristics as P_2, P_{21} and P_{22} respectively.

Let us now concentrate on $\mathcal{E}q\{\Sigma \rhd u_i : X_k \,|\, i \in \varphi^{-1}(l+1)\} = \{\Sigma \rhd u_{i_1} = u_{i_2}, \ldots, \Sigma \rhd u_{i_{j-1}} = v_1 w_1, \Sigma \rhd v_1 w_1 = u_{i_{j+1}}, \ldots, \Sigma \rhd u_{i_r} = u_{i_1}\}$ in circular form. In case this circle has only one equation, i.e. $\Sigma \rhd v_1 w_1 = v_1 w_1$ we could replace the proof of this equation by the reflexivity $\Sigma \rhd v_2 w_2 = v_2 w_2$. Obviously, the new proof would satisfy all the requirements.

Assume now that the above circle of equations has at least two equations. By assumption, each of these equations is proved by a proof whose degree is $< d(v_1 w_1)$ and whose height is $< h(P_1)$. In particular it holds for the proof P_{11} of $u_{i_{j-1}} = v_1 w_1$ and for the proof P_{12} of $v_1 w_1 = u_{i_{j+1}}$. Now, we apply the induction hypothesis to P_{11} and P_2 and also to P_2' and P_{12}. In that way we obtain a proof P_3 of $u_{i_{j-1}} = v_2 w_2$ and a proof P_4 of $v_2 w_2 = u_{i_{j+1}}$ and such that their degrees are $< d(v_1 w_1)$ and $h(P_3) \leq 1(h(P_{11}) + h(P_2))$, $h(P_4) \leq 1(h(P_{12}) + h(P_2'))$. All the other equations from the above $\mathcal{E}q$ stay intact. By one application of the rule $(ASub)$ we obtain $F[u_1, \ldots, u_l] = v_2 w_2$. This proof indeed has degree $< n$. Its height is

$$\leq \max(\{h(P_i)+1 \,|\, P_i \text{ is a proof of an "intact" equation}\} \cup \{h(P_{11})+h(P_2)+1, h(P_{12})+h(P_2')+1\}).$$

We have to show that this height is $\leq 1(h(P_1) + h(P_2))$. We know that

$$h(P_1) = \max(\{h(P_i)+1 \,|\, P_i \text{ is a proof of an "intact" equation}\} \cup \{h(P_{11})+1, h(P_{12})+1\}).$$

This indeed holds since $h(P_2) = h(P_2')$.

The only remaining case is when both proofs end with $(ASub)$. So we have

$$\frac{\mathcal{E}q\{\Sigma \rhd u_i : X_k \,|\, i \in \varphi^{-1}(k)\}_{k=1,\ldots,m}}{\Sigma \rhd F[u_1, \ldots, u_l] = G[u_{l+1}, \ldots, u_{l+l'}] : Y}$$

where $T_{alg} \vdash x_1 : X_1, \ldots, x_m : X_m \rhd F[x_{\varphi(1)}, \ldots, x_{\varphi(l)}] = G[x_{\varphi(l+1)}, \ldots, x_{\varphi(l+l')}] : Y$ and $\varphi : \{1, \ldots, l+l'\} \to \{1, \ldots m\})$ as the last rule of P_1 and

$$\frac{\mathcal{E}q\{\Sigma \rhd u_i : X_k' \,|\, i \in {\varphi'}^{-1}(k)\}_{k=1,\ldots,m'}}{\Sigma \rhd G[u_{l+1}, \ldots, u_{l+l'}] = H[u_{l+l'+1}, \ldots, u_{l+l'+l''}] : Y}$$

where $T_{alg} \vdash x_1' : X_1', \ldots, x_{m'}' : X_{m'}' \rhd G[x'_{\varphi'(l+1)}, \ldots, x'_{\varphi'(l+l')}] = H[x'_{\varphi'(l+l'+1)}, \ldots, x'_{\varphi(l+l'+l'')}] : Y$ and with conditions similar to above, e.g. $\varphi' : \{l+1, \ldots, l+l'+l''\} \to \{1, \ldots, m'\})$ as the last rule of P_2.

The fact that for $j = l+1, \ldots, l+l'$ $x_{\varphi(j)}$ and $x'_{\varphi'(j)}$ take the same places in $G[\ldots]$ implies that $X_{\varphi(j)} = X'_{\varphi'(j)}$. We want to prove $\Sigma \rhd F[u_1, \ldots, u_l] = H[u_{l+l'+1}, \ldots, u_{l+l'+l''}] : Y$ with a proof whose degree is $< d(G[u_{l+1}, \ldots, \ldots, u_{l+l'}]) = \max\{d(u_j) + 1 \,|\, j = l+1, \ldots l+l'\}$ or $d(u_{l+1})$ depending whether $G[...]$ is single hole or not and with height $\leq \gamma(g)(h(P_1) + h(P_2)) = l'(h(P_1) + h(P_2))$.

Let us consider a pushout in *Sets* over $\varphi \cdot incl_2 : \{l+1, \ldots, l+l'\} \to \{1, \ldots, m\}$ and $\varphi' \cdot incl_1' : \{l+1 \ldots, l+l'\} \to \{1, \ldots, m'\}$. Denote the two newly obtained functions by $I : \{1, \ldots, m\} \to \{1, \ldots, m''\}$ and $I' : \{1, \ldots, m'\} \to \{1, \ldots, m''\}$. Recall that we can think of $\{1, \ldots, m''\}$ as a set of equivalence classes on the disjoint union $\{1, \ldots, m\} \sqcup \{1, \ldots, m'\}$ generated by the relation: $\{\varphi(i) \sim \varphi'(i) \,|\, i = l+1, \ldots, l+l'\}$. Since φ and φ' preserve the typing, it follows that the variables in the same equivalence class have the same sort and that I and I' preserve the sorts. For every such sort we take a different variable and in that way we obtain a sequence $y_1 : Y_1, \ldots, y_{m''} : Y_{m''}$.

First we observe that if $T_{alg} \vdash (x_1 : X_1, \ldots, x_m : X_m \triangleright F[x_{\varphi(1)}, \ldots, x_{\varphi(l)}] = G[x_{\varphi(l+1)}, \ldots, x_{\varphi(l+l')}] : Y)$ then also $T_{alg} \vdash (y_1 : Y_1, \ldots, y_{m''} : Y_{m''} \triangleright F[y_{I(\varphi(1))}, \ldots, y_{I(\varphi(l))}] = G[y_{I(\varphi(l+1))}, \ldots, y_{I(\varphi(l+l'))}] : Y)$ since just some variables are identified, renamed, and new variables are added. Similarly if $T_{alg} \vdash (x'_1 : X'_1, \ldots, x'_{m'} : X'_{m'} \triangleright G[x'_{\varphi'(l+1)}, \ldots, x'_{\varphi'(l+l')}] = H[x'_{\varphi'(l+l'+1)}, \ldots, x'_{\varphi(l+l'+l'')}] : Y)$ then also $T_{alg} \vdash (y_1 : Y_1, \ldots, y_{m''} : Y_{m''} \triangleright G[y_{I'(\varphi'(l+1))}, \ldots, y_{I'(\varphi'(l+l'))}] = H[y_{I'(\varphi'(l+l'+1))}, \ldots, y_{I'(\varphi(l+l'+l''))}] : Y)$. We also know that $G[y_{I(\varphi(l+1))}, \ldots, y_{I(\varphi(l+l'))}] \equiv G[y_{I'(\varphi'(l+1))}, \ldots, y_{I'(\varphi'(l+l'))}]$ by the definition of the above pushout. Since T_{alg} is closed under transitivity, we have $T_{alg} \vdash (y_1 : Y_1, \ldots, y_{m''} : Y_{m''} \triangleright F[y_{I(\varphi(1))}, \ldots, y_{I(\varphi(l))}] = H[y_{I'(\varphi'(l+l'+1))}, \ldots, y_{I'(\varphi'(l+l'+l''))}] : Y)$. (Here, we are not actually using the kind of transitivity which we want to eliminate!)

We also know that $\{1, \ldots, l+l'+l''\}$ is a pushout of $\{1, \ldots, l+l'\}$ and $\{l+1, \ldots, l+l'+l''\}$ over $\{l+1, \ldots, l+l'\}$, therefore there exists a unique function $\psi_0 : \{1, \ldots, l+l'+l''\} \to \{1, \ldots, m''\}$ satisfying the universal property. In other words, ψ_0 can be defined as follows:

$$\psi_0(i) = \begin{cases} I\varphi(i) & \text{if } \ i \in \{1, \ldots l\} \\ I\varphi(i) = I'\varphi'(i) & \text{if } i \in \{l+1, \ldots, l+l'\} \\ I'\varphi'(i) & \text{if } i \in \{l+l'+1, \ldots, l+l'+l''\} \end{cases}$$

Let $\psi : \{1, \ldots, l\} \sqcup \{l+l'+1, \ldots, l+l'+l''\} \to \{1, \ldots, m''\}$ denote the appropriate restriction of ψ_0. With this notation, the above equation reads $y_1 : Y_1, \ldots, y_{m''} : Y_{m''} \triangleright F[y_{\psi(1)}, \ldots, y_{\psi(l)}] = H[y_{\psi(l+l'+1)}, \ldots, y_{\psi(l+l'+l'')}] : Y$. Recall that our goal is to prove $\Sigma \triangleright F[u_1, \ldots, u_l] = H[u_{l+l'+1}, \ldots, u_{l+l'+l''}] : Y$ with a proof whose degree is $< d(G[u_{l+1}, \ldots, u_{l+l'}])$ and whose height $\leq \gamma(g)(h(P_1) + h(P_2)) = l'(h(P_1) + h(P_2))$. Having the above equality in T_{alg} and using the rule $(ASub)$ it would be enough to show that we can obtain $\mathcal{E}q\{\Sigma \triangleright u_i : Y_k \,|\, i \in \psi^{-1}(k)\}$ for every $k = 1, \ldots, m''$ in the "proper" way *i.e.* if $\{u_{i_1}, \ldots, u_{i_r}\} = \psi^{-1}(k)$ then we can prove each $\Sigma \triangleright u_{i_1} = u_{i_2}$,... ,$\Sigma \triangleright u_{i_r} = u_{i_1}$ with a proof Q whose degree is $< d(G[u_{l+1}, \ldots, \ldots, u_{l+l'}])$ and whose height is $< l'(h(P_1) + h(P_2))$.

First of all notice that $\psi^{-1}(k) = \psi_0^{-1}(k) \cap (\{1, \ldots l\} \sqcup \{l+l'+1 \ldots, l+l'+l''\})$. In order to maintain our assumption on $\psi^{-1}(k)$ when k is not in the image of ψ we need the following additional definitions. For $k \notin Image(\psi)$ define $\varphi_0^{-1}(k)$ to be $\varphi^{-1}(k)$ or ${\varphi'}^{-1}(k)$ depending whether k is in the image of I or I'; this definition is correct since I is just inclusion on $\{1, \ldots, m\} - Image(\varphi)$ and similarly for I'. In addition, I and I' do not intersect when restricted to those sets. Recall that this means, that $\psi_0^{-1}(k)$ is a single term in context Σ and of type Y_k. We take this as the definition of $\psi^{-1}(k)$ as well for such a k. There is one more case to be considered: $k \in Image(\psi_0) - Image(\psi)$. That means that all the u's which got substituted instead of the k-th variable appeared only in $G[u_{l+1}, \ldots, u_{l+l'}]$. We choose any such term u to represent $\psi^{-1}(k)$.

It is useful to visualize the situation as follows: let us call $u_1, \ldots, u_l$ "black" terms, $u_{l+1}, \ldots, u_{l+l'}$ "gray" and $u_{l+l'+1}, \ldots, u_{l+l'+l''}$ "white". For each $k = 1, \ldots, m$ we call $\mathcal{E}q\{\Sigma \triangleright u_i : X_k \,|\, i \in \varphi^{-1}(k)\}$ a "black circle" (although it could be entirely made of gray elements!), and for each $k = 1, \ldots, m'$ we call $\mathcal{E}q\{\Sigma \triangleright u_i : X'_k \,|\, i \in {\varphi'}^{-1}(k)\}$ a "white circle" (again a white circle could be entirely gray). Each $\psi_0^{-1}(s)$ can be considered as a union of black and white circles. Obviously, black circles are disjoint from each other as are the white ones (therefore, each gray element appears in exactly one black and one white circle). One black and one white circle can have an empty intersection or they can have several common gray elements. By the definition of the above pushout, we have that each $\psi_0^{-1}(s)$ is a maximal "chain" of circles which are of interchanged colour (a chain is a union of circles C_1,...,C_a such that C_i and C_{i+1} have common elements). From each such chain we first want to produce a circle which may have perhaps fewer gray elements but the black and white ones must remain the same. Let us first consider a couple of special cases which come from our convention on $\psi^{-1}(k)$ when k is not in the image of ψ. As we said before, if $k \notin Image(\psi_0)$ one reflexivity will constitute the "circle" $\mathcal{E}q\{\Sigma \triangleright u : X_k\}$ and if $k \in Image(\psi_0) - Image(\psi)$ then the above chain is entirely gray and we choose an arbitrary element and reflexivity to represent this $\mathcal{E}q$ circle. Observe also that if a circle in a chain consists only of one element then there are two cases: if this element is not gray then the whole chain is actually just this "singleton" circle; if this element is gray it is a member of another circle or again we have a chain with just one singleton circle. Moreover, this chain can be considered as a single circle already (since otherwise, we would have an intersection of two circles of the same colour). Now we are left with the less trivial case that the chain is not entirely gray and that each circle in the chain has at least two elements.

Before we analyse that let us recall that each "edge" in a circle is actually one equation which, by the inductive assumption, is proved with a proof whose degree is $< n$ and whose height is $\leq \max\{h(P_1)-1, h(P_2)-1\}$. So, suppose that we have circles C_1, ... ,C_a. We claim that we can make such a chain into a circle where each edge will again have a proof of degree $< n$ and of height $\leq 2\max\{h(P_1)-1, h(P_2)-1\}$. Let us start just with two circles. Suppose that the first circle consists of $a_1 = a_2$,...,$a_{i-1} = c$, $c = a_{i+1}$,...,$a_n = a_1$ and the second consists of $b_1 = b_2$,...,$b_{j-1} = c$, $c = b_{j+1}$,...,$b_m = b_1$ where c denotes a common element. The new circle is going to be $a_1 = a_2$,...,$a_{i-1} = b_{j+1}$,...,$b_m = b_1$, $b_1 = b_2$,...,$b_{j-1} = a_{i+1}$,...,$a_n = a_1$. To form this new circle we needed to use $a_{i-1} = c$ and $c = b_{j+1}$ to obtain $a_{i-1} = b_{j+1}$ and similarly, $b_{j-1} = c$ and $c = a_{i+1}$ to obtain $b_{j-1} = a_{i+1}$. All the other equations are left as they were. We consider two cases:

(i) $G[\ldots]$ is just a single hole. In that case the middle term of transitivity is u_{l+1} and its degree is by assumption n. This has to be our c since this is the only possible common term (and in this case the maximal chain has at most two circles). By the induction hypothesis we can obtain the new equations with proofs whose degrees are again $< n$ and whose heights are $\leq 1(h(P') + h(P''))$ where P' is the proof of $a_{i-1} = c$ and P'' is a proof $c = b_{j+1}$ (similarly for b's). Recall that $\gamma(c) = 1$ since c is an application or a variable. Therefore, all the equations in the new circle have proofs of degree $< n$ and heights $\leq h(P_1) - 1 + h(P_2) - 1$.

(ii) Assume now that $G[\ldots]$ is not a single hole and that it has l' holes (still, l' could be 1). We can obtain the above two new equations using the two transitivity rules (and not the induction hypothesis!) whose degrees are $< d(G[u_{l+1}, \ldots, u_{l+l'}] = n$ and the heights of the new proofs are $\max\{h(P'), h(P'')\} + 1$ where P' and P'' are as in the previous case. These heights are $\leq \max\{h(P_1), h(P_2)\}$.

If we have more than 2 circles then we do not have to consider the first case. On the other hand we would have to repeat the step (ii) $a-1$ times. In the final circle, every equation would have been proved with a proof whose degree is $< n$ and the height $\leq \max\{h(P_1), h(P_2)\} + a - 2$. Notice also that each application of the step (ii) reduces the number of gray elements by 1.

After such a procedure we have ended up with a single circle which possibly contains some gray elements. If this new circle does not contain gray elements we are done. Also, we are done if such a circle has exactly one, even if gray, element.

If it does have at least 2 elements and at least one of them is gray, we again separate two cases:

(a) if $G[\ldots]$ was a single hole then the gray element could have "survived" the above step (i) only if the step (i) was not even applied *i.e.* if one of the circles consisted of only one gray element and in this case we considered the chain to be a single circle. So, this circle can be represented as $a_1 = a_2$,...,$a_{i-1} = c$, $c = a_{i+1}$,...,$a_n = a_1$ where c is the single gray element. We can apply the inductive assumption on $a_{i-1} = c$ and $c = a_{i+1}$ to get $a_{i-1} = a_{i+1}$ (the other equations stay intact). This proof of $a_{i-1} = a_{i+1}$ again has degree $< d(c) = n$ and height $\leq 1(h(P') + h(P''))$ where P' and P'' are the proofs of $a_{i-1} = c$ and $c = a_{i+1}$ respectively.

(b) if $G[u_{l+1}, \ldots, u_{l+l'}]$ is not a single hole then again we would have a circle with at most $l' - a + 1$ gray elements such that each equation has been proved with a proof whose degree is $< n$ and whose height is $\leq \max\{h(P_1), h(P_2)\} + a - 2$. Also, the degree of each term in the circle is $< n$. Now applying at most $l' - a + 1$ transitivity rules we can get rid of all the gray elements and the proofs of final equations will be of degree $< n$ and of height

$$\leq (\max\{h(P_1), h(P_2)\} + a - 2) + (l' - a + 1) = \max\{h(P_1), h(P_2)\} + l' - 1.$$

In that manner, we have shown that for each $k = 1, \ldots, m''$ $\mathcal{E}q\{\Sigma \triangleright u_i : Y_k \mid i \in \psi^{-1}(k)\}$ can be arranged in a circle of equations whose proofs have degrees $< n$ and whose heights are $< l'(h(P_1) + h(P_2))$.

The other cases are much simpler. □

Remark 7.15. If we were to work "in parallel" the above $\gamma(g)$ in the estimate for height could have been lowered to $\lceil log_2(l') \rceil + 1$ *i.e.* the number of digits needed to write l' in base 2.

Lemma 7.16. *If P is a proof of $\Sigma \triangleright a = b$ in T'_α and $d(P) > 0$ then there is a proof P' of $\Sigma \triangleright a = b$ in T'_α such that $d(P') < d(P)$ and $h(P') \leq h(P)^{\gamma(P)}$.*

Proof. By induction on $h(P)$ again examining all the possible last rules. □

We now restate and prove the elimination of transitivity theorem with some estimates.

Theorem 7.17. *If P is a proof of $\Sigma \triangleright a = b$ in T'_α then there exists a transitivity free proof P' of $\Sigma \triangleright a = b$ in T'_α such that $h(P') \leq (2\gamma)_{deg(P)}(h(P))$ where $(2\gamma)_0(h(P)) = h(P)$ and $(2\gamma)_{n+1}(h(P)) = (2\gamma)^{(2\gamma)_n(h(P))}$.*

Proof. By induction on $d(P)$ and the previous lemma. □