Beyond β-Reduction in Church's $\lambda_{\rightarrow}$ *

Roel Bloo
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O.Box 513
5600 MB Eindhoven, the Netherlands
email: bloo@win.tue.nl

and

Fairouz Kamareddine †
Department of Computing Science
17 Lilybank Gardens
University of Glasgow
Glasgow G12 8QQ, Scotland
email: fairouz@dcs.glasgow.ac.uk
fax + 44 41 -3304913
and

Rob Nederpelt
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O.Box 513
5600 MB Eindhoven, the Netherlands
email: wsinrpn@win.tue.nl

October 23, 1996

*We are grateful for the discussions with Tijn Borghuis, and Erik Poll and for the helpful remarks received from them.

†Kamareddine is grateful to the Department of Mathematics and Computing Science, Eindhoven University of Technology, for their financial support and hospitality from October 1991 to September 1992, and during various short visits in 1993 and 1994.

Abstract

In this paper, we shall write $\lambda_{\rightarrow}$ using a notation, *item notation*, which enables one to make more redexes visible, and shall extend β-reduction to all visible redexes. We will prove that $\lambda_{\rightarrow}$ written in item notation and accommodated with extended reduction, satisfies all its original properties (such as Church Rosser, Subject Reduction and Strong Normalisation). The notation itself is very simple: if $\mathcal{I}$ translates classical terms to our notation, then $\mathcal{I}(t_1t_2) \equiv (\mathcal{I}(t_2)\delta)\mathcal{I}(t_1)$ and $\mathcal{I}(\lambda_{v:\rho}.t) \equiv (\rho\lambda_v)\mathcal{I}(t)$. For example, $t \equiv ((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\rightarrow X_2}.x_5x_4)x_3)x_2)x_1$, can be written in our item notation as $\mathcal{I}(t) \equiv (x_1\delta)(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})((X_1 \rightarrow X_2)\lambda_{x_5})(x_4\delta)x_5$ where the visible redexes are based on all the matching $\delta\lambda$-couples. So here, the redexes are based on $(x_2\delta)(X_4\lambda_{x_7})$, $(x_3\delta)(X_3\lambda_{x_6})$ and $(x_1\delta)((X_1 \rightarrow X_2)\lambda_{x_5})$. In classical notation however, only the redexes based on $(\lambda_{x_7:X_4}.\ -\ -)x_2$ and $(\lambda_{x_6:X_3}.\ -\ -)x_3$ are immediately visible. The third redex, $(\lambda_{x_5:(X_1\rightarrow X_2)}.\ -\ -)x_2$, only becomes visible when the first two redexes have been contracted. We extend β-reduction so that we can contract newly visible redexes even before other redexes have been contracted. So in our example above, $(x_1\delta)((X_1 \rightarrow X_2)\lambda_{x_5})$ can be contracted before $(x_2\delta)(X_4\lambda_{x_7})$ or $(x_3\delta)(X_3\lambda_{x_6})$. This refinement (which cannot be done in classical notation) is achieved by generalising the axiom β from $(t_1\delta)(\rho\lambda_v)t_2 \rightarrow_\beta t_2[v := t_1]$ to $(t_1\delta)\overline{s}(\rho\lambda_v)t_2 \rightsquigarrow_\beta \overline{s}(t_2[v := t_1])$ for $\overline{s}$ consisting of matching $\delta\lambda$-couples only. Hence, as $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})$ consists of matching $\delta\lambda$-couples, we get that $\mathcal{I}(t) \rightsquigarrow_\beta (x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(((x_4\delta)x_5)[x_5 := x_1])$.
Furthermore, with our item notation, it is possible to refine reduction by rewriting (or *reshuffling*) terms so that matching $\delta\lambda$-couples occur adjacent to each other. For example, we can rewrite $\mathcal{I}(t)$ above as $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(x_1\delta)((X_1 \rightarrow X_2)\lambda_{x_5})(x_4\delta)x_5$. We shall formalise term reshuffling and shall show that it is correct and preserves both β-reduction and typing.

Keywords: β-reduction, Church Rosser, Subject Reduction, Strong Normalisation.

Contents

1 Introduction

The notation of this paper, *item notation*, is a novel notation where the argument is given before the function, the type is given before the λ, and where the parentheses are grouped differently than those of the classical notation. So that, if $\mathcal{I}$ translates classical terms into our notation, then $\mathcal{I}(t_1 t_2)$ is written as $(\mathcal{I}(t_2)\delta)\mathcal{I}(t_1)$ and $\mathcal{I}(\lambda_{v:\rho}.t)$ is written as $(\rho\lambda_v)\mathcal{I}(t)$. Both $(t\delta)$ and $(\rho\lambda_v)$ are called **items**.

Example 1.1 $\mathcal{I}((\lambda_{x:X_1\to(X_2\to X_3)}.\lambda_{y:X_1}.xy)z) \equiv (z\delta)(X_1 \to (X_2 \to X_3)\lambda_x)(X_1\lambda_y)(y\delta)x$. The items are $(z\delta)$, $(X_1 \to (X_2 \to X_3)\lambda_x)$, $(X_1\lambda_y)$ and $(y\delta)$.

Before we discuss the calculus and the properties of typing, let us see why we want to extend the notion of a redex and to refine β-reduction.

In the classical λ-calculus, the notions of redex, and of β-reduction are described as follows:

Definition 1.2 *(Redexes and β-reduction in classical notation)*
In the classical notation of the λ-calculus, a redex is of the form $(\lambda_{v:\rho}.t)t'$. Moreover, one-step β-reduction $\to_\beta$ is the compatible relation generated out of the axiom β: $(\lambda_{v:\rho}.t)t' \to_\beta t[v := t']$. Many step β-reduction $\twoheadrightarrow_\beta$, is the reflexive transitive closure of $\to_\beta$.

With our item notation, classical redexes and β-reduction take the following form:

Definition 1.3 *(Classical redexes and β-reduction in item notation)*
*In the item notation of the λ-calculus, a classical redex is of the form $(t'\delta)(\rho\lambda_v)t$. We call the pair $(t'\delta)(\rho\lambda_v)$, a $\delta\lambda$-**pair**, or a $\delta\lambda$-**segment**. The classical β-reduction axiom is: $(t'\delta)(\rho\lambda_v)t \to_\beta t[v := t']$. One and many step β-reduction are defined as in Definition 1.2.*

Example 1.4 In the classical term $t \equiv ((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_2)x_1$, we have the following redexes (the fact that neither x_6 nor x_7 appear as free variables in their respective scopes does not matter here; this is just to keep the example simple and clear):

1. $(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3$
2. $(\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_2$

In item notation t becomes $(x_1\delta)(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$. Here, the two classical redexes correspond to $\delta\lambda$-pairs as follows:

1. $(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3$ corresponds to $(x_3\delta)(X_3\lambda_{x_6})$. $((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$ is ignored as it is easily retrievable in item notation. It is the maximal subterm of t to the right of $(X_3\lambda_{x_6})$.
2. $(\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_2$ corresponds to $(x_2\delta)(X_4\lambda_{x_7})$.
   Again $(x_3\delta)(X_3\lambda_{x_6})((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$ is ignored for the same reason as above.

There is however a third redex which is not immediately visible in the classical term; namely, $(\lambda_{x_5:X_1\to X_2}.x_5x_4)x_1$. Such a redex will only be visible after we have contracted the above two redexes (we will not discuss the order here). In fact, assume we contract the second redex in the first step, and the first redex in the second step. I.e.

$$\begin{array}{ll}((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_2)x_1 & \to_\beta \\ ((\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_1 & \to_\beta \\ (\lambda_{x_5:X_1\to X_2}.x_5x_4)x_1 & \to_\beta x_1x_4\end{array}$$

Now, even though all these three redexes are *needed* in order to get the normal form of t, only the first two were visible in the classical term at first sight. The third could only be seen once we had contracted the first two redexes. In item notation, the third redex $(\lambda_{x_5:X_1\to X_2}.x_5x_4)x_1$ is visible as it corresponds to the matching $(x_1\delta)((X_1 \to X_2)\lambda_{x_5})$ where $(x_1\delta)$ and $((X_1 \to X_2)\lambda_{x_5})$ are separated by the segment $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})$. Hence, by extending the notion of a redex from being a δ-item adjacent to a λ-item, to being a matching pair of δ- and λ-items, we can make more redexes visible. This extension furthermore is simple, as in $(t_1\delta)\overline{s}(\rho\lambda_v)$, we say that $(t_1\delta)$ and $(\rho\lambda_v)$ match if $\overline{s}$ has the same structure as a matching composite of opening and closing brackets, each δ-item corresponding to an opening bracket and each λ-item corresponding to a closing bracket. For example, in t above, $(x_1\delta)$ and $((X_1 \to X_2)\lambda_{x_5})$ match as $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})$ has the bracketing structure $[\,][\,]$ (see Figure 1 which is drawn ignoring types just for the sake of argument). With this

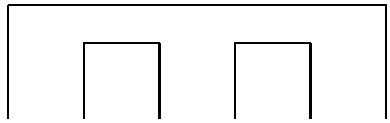


Figure 1: Redexes in item notation

extension of redexes, we refine β-reduction in two different ways:

1. By changing (β) from $(t_1\delta)(\rho\lambda_v)t_2 \to_\beta t_2[v := t_1]$ to $(t_1\delta)\overline{s}(\rho\lambda_v)t_2 \rightsquigarrow_\beta \overline{s}(t_2[v := t_1])$ if $(t_1\delta)$ and $(\rho\lambda_v)$ match.

2. By *reshuffling* terms so that matching δ's and λ's occur adjacently.

We start by showing that $\rightsquigarrow\!\!\!\!\!\to_\beta$ (the reflexive transitive closure of $\rightsquigarrow_\beta$) is a generalisation of $\twoheadrightarrow_\beta$ (Lemma 3.7). We will then show that $\lambda_\to$ with $\rightsquigarrow\!\!\!\!\!\to_\beta$ satisfies all the desirable typing properties. In particular, we will establish that $\lambda_\to$ extended with $\rightsquigarrow\!\!\!\!\!\to_\beta$ satisfies the following:

1. Church Rosser: this says that if a program is evaluated in two different ways, then the answer stays the same (Theorem 3.10).

2. Subject Reduction: this says that if a program P is well-typed then the program obtained from evaluating some steps in P is also well-typed (Theorem 3.13).

3. Unicity of Types: this says that a well-typed program has a unique type and that two equal programs have the same type (Lemma 3.15).

4. Strong Normalisation: this says that all ways of evaluating a well-typed program terminate (Theorem 3.21).

We will furthermore show that term reshuffling is correct. In particular, we shall show that $\lambda_\to$ accommodated with term reshuffling TS, satisfies the following:

1. Reshuffling a term, moves all δ's next to their matching λ's (Lemma 4.9).

2. Reshuffling terms preserves $\rightarrow_\beta$. That is, if $t \rightsquigarrow_\beta t'$ then there exists t'' such that $TS(t) \rightarrow_\beta t''$ and $TS(t') \equiv TS(t'')$ (Lemma 4.11).

3. Reshuffling terms preserves types. That is, if $\Gamma \vdash t : \rho$ then $\Gamma \vdash TS(t) : \rho$ (Lemma 4.13).

2 $\lambda_\rightarrow$ in item notation

In this section, we shall introduce the known $\lambda_\rightarrow$ (which uses the ordinary β-reduction $\twoheadrightarrow_\beta$), and its properties. We shall write $\lambda_\rightarrow$ immediately in item notation. That is, we assume a translation function $\mathcal{I}$ from terms in classical notation to terms in item notation such that:

$$\begin{array}{lcll} \mathcal{I}(v) & = & v & \text{if } v \text{ is a variable} \\ \mathcal{I}(\lambda_{v:\rho}.t) & = & (\rho\lambda_v)\mathcal{I}(t) & \\ \mathcal{I}(t_1 t_2) & = & (\mathcal{I}(t_2)\delta)\mathcal{I}(t_1) & \end{array}$$

2.1 The basic theory

In Church's system $\lambda_\rightarrow$, types and terms are defined as follows:

Definition 2.1 *(Types of $\lambda_\rightarrow$)*
The set of types $\mathcal{T}$ of $\lambda_\rightarrow$ is defined as follows:

$$\begin{array}{lcll} \mathcal{T} & ::= & \mathcal{V} \mid (\mathcal{T} \rightarrow \mathcal{T}) & \textit{Types} \\ \mathcal{V} & ::= & \{X_0, X_1, \cdots\} & \textit{Type variables} \end{array}$$

That is, types are either variables such as $X_0, X_1, X_2, \ldots$ or arrow types.

Definition 2.2 *(Terms of $\lambda_\rightarrow$)*
The set of terms $\Lambda_\mathcal{T}$ of $\lambda_\rightarrow$ is defined as follows:

$$\begin{array}{lcll} \Lambda_\mathcal{T} & ::= & V \mid (\mathcal{T}\lambda_V)\Lambda_\mathcal{T} \mid (\Lambda_\mathcal{T}\delta)\Lambda_\mathcal{T} & \textit{Terms} \\ V & ::= & \{x_0, x_1, \cdots\} & \textit{Variables} \end{array}$$

In other words, a term is either a variable $x_0, x_1, x_2, \ldots$, or an abstraction or an application.

Notation 2.3 We let $\rho, \rho', \rho_1, \rho_2, \ldots$ range over types, $\alpha, \alpha', \alpha_1, \alpha_2, \ldots$ range over type variables. Furthermore, we take $t, t', t_1, t_2, \ldots$ to range over terms and let $v, v', v_1, \ldots$ range over variables.

Parentheses moreover will be omitted when no confusion occurs.

We understand $\rho_1 \rightarrow \rho_2 \rightarrow \cdots \rho_n$ to mean $(\rho_1 \rightarrow (\rho_2 \rightarrow \cdots \rightarrow (\rho_{n-1} \rightarrow \rho_n)\cdots))$.

Bound and free variables in $\lambda_\rightarrow$ are defined as usual. We write $BV(t)$ and $FV(t)$ to represent the bound and free variables of t respectively. Substitution moreover, is defined in the usual way. Furthermore, we take terms to be equivalent up to variable renaming. For example, we take $(\rho\lambda_{x_0})x_0 \equiv (\rho\lambda_{x_1})x_1$. We assume moreover, the Barendregt variable convention which is formally stated as follows:

Definition 2.4 *(BC: Barendregt's Convention for $\lambda_\rightarrow$)*
Names of bound variables will always be chosen such that they differ from the free ones in a term. Hence, we will not have $(v\delta)(\rho\lambda_v)v$, but $(v\delta)(\rho\lambda_{v'})v'$ instead.

Definition 2.5 *(Compatibility)*
We say that a relation $\to$ on terms is compatible iff the following holds:

$$\frac{t \to t'}{(t\delta)t_1 \to (t'\delta)t_1} \qquad \frac{t \to t'}{(t_1\delta)t \to (t_1\delta)t'}$$

$$\frac{t \to t'}{(\rho\lambda_v)t \to (\rho\lambda_v)t'}$$

Basically compatibility means that if $t \to t'$ then $T[t] \to T[t']$ where $T[\,]$ is a "term with a hole in it".

Definition 2.6 *(β-reduction $\to_\beta$ in $\lambda_\to$)*
In $\lambda_\to$, β-reduction $\to_\beta$, is the least compatible relation generated out of the following axiom:

$$(\beta) \qquad (t_1\delta)(\rho\lambda_v)t \to_\beta t[v := t_1]$$

We take $\twoheadrightarrow_\beta$ to be the reflexive transitive closure of $\to_\beta$ and $=_\beta$ to be the least equivalence relation generated by $\twoheadrightarrow_\beta$.

Definition 2.7 *((main) items, (main, $\delta\lambda$-)segments, context, body, endvar, weight)*

- *If v is a variable, ρ is a type and t is a term then $(\rho\lambda_v)$ and $(t\delta)$ are items (the first is called λ-item, the second δ-item). We use $s, s_1, s_i, \ldots$ to range over items.*
- *A concatenation of zero or more items is a* **segment**. *We use $\overline{s}, \overline{s}_1, \overline{s}_i, \ldots$ as meta-variables for segments. We write $\emptyset$ for the empty segment.*
- *Each term t is the concatenation of zero or more items and a variable: $t \equiv s_1 s_2 \cdots s_n v$. These items $s_1, s_2, \cdots, s_n$ are called the* **main items** *of t.*
- *Analogously, a segment $\overline{s}$ is a concatenation of zero or more items: $\overline{s} \equiv s_1 s_2 \ldots s_n$; again, these items $s_1, s_2, \ldots, s_n$ (if any) are called the* **main items**, *this time of $\overline{s}$.*
- *A concatenation of adjacent main items (in t or $\overline{s}$), $s_m \cdots s_{m+k}$, is called a* **main segment** *(in t or $\overline{s}$).*
- *A* **context** *is a segment which consists of only λ-items. We use $\Gamma, \Gamma', \Gamma_1, \Gamma_2, \ldots$ to range over contexts.*
- *A* **$\delta\lambda$-segment** *is a δ-item immediately followed by a λ-item.*
- *Let $t \equiv \overline{s}v$ be a term. Then we call $\overline{s}$ the* **body** *of t, denoted $\mathtt{body}(t)$, and v the* **end variable** *of t, or $\mathtt{endvar}(t)$. It follows that $t \equiv \mathtt{body}(t)\ \mathtt{endvar}(t)$.*
- *The* **weight** *of a $\lambda_\to$-segment $\overline{s}$, $\mathtt{weight}(\overline{s})$, is the number of main items that compose the segment. Moreover, we define $\mathtt{weight}(t) = \mathtt{weight}(\mathtt{body}(t))$.*

Definition 2.8 *(Statements)*
A statement is of the form $t : \rho$, t and ρ are called the subject and the type of the statement respectively.

Convention 2.9 In a context, we never have two occurrences of λ_v (for the same v). Hence, contexts are what [Barendregt 92] calls bases.

We need the following definition over contexts:

Definition 2.10 *Let* $\Gamma = (\rho_1\lambda_{v_1})(\rho_2\lambda_{v_2})\cdots(\rho_k\lambda_{v_k})$ *be a context. Then*

1. $dom(\Gamma) = \{v_1, v_2, \ldots, v_k\}$
2. $(\rho\lambda_v) \in' \Gamma$ *iff* $(\rho\lambda_v)$ *is an item of* Γ. *If* Γ' *is a context such that all items of* Γ' *are also items of* Γ, *we write* $\Gamma' \subseteq' \Gamma$.
3. *Let* V_0 *be a set of term variables.* $\Gamma \restriction V_0$ *(the* **restriction** *of* Γ *to* V_0*) is the context which only contains the items* $(\rho\lambda_v) \in' \Gamma$ *for which* $v \in V_0$, *in the original order.*

Now for the formulation of the typing rules we can use the following definitions for the derivation of so-called *judgements* of the form $\Gamma \vdash t : \rho$.

Definition 2.11 *(Typing rules of* $\lambda_\to$*)*
A statement $t : \rho$ *is derivable in the context* Γ, *notation* $\Gamma \vdash t : \rho$, *if* $t : \rho$ *can be derived using the following rules:*

$(Axiom)$ $\quad \Gamma \vdash v : \rho \qquad$ *if* $(\rho\lambda_v) \in' \Gamma$

$(\to -elimination)$ $\quad \dfrac{\Gamma \vdash t : \rho \qquad \Gamma \vdash t' : (\rho \to \rho')}{\Gamma \vdash (t\delta)t' : \rho'}$

$(\to -introduction)$ $\quad \dfrac{\Gamma(\rho\lambda_v) \vdash t : \rho'}{\Gamma \vdash (\rho\lambda_v)t : (\rho \to \rho')}$

2.2 Properties of $\lambda_\to$

Here we list the properties of $\lambda_\to$ (that we will establish for extended reduction) without proofs. The reader can refer to [Barendregt 92] for details.

Theorem 2.12 *(The Church Rosser Theorem)*
If $t \twoheadrightarrow_\beta t_1$ *and* $t \twoheadrightarrow_\beta t_2$ *then there exists* t_3 *such that* $t_1 \twoheadrightarrow_\beta t_3$ *and* $t_2 \twoheadrightarrow_\beta t_3$ □

Lemma 2.13 *(Context lemma)*

1. $\forall_{\Gamma,\Gamma'}\forall_t\forall_\rho[\Gamma \subseteq' \Gamma' \wedge \Gamma \vdash t : \rho \Rightarrow \Gamma' \vdash t : \rho]$
2. $\forall_\Gamma\forall_t\forall_\rho[\Gamma \vdash t : \rho \Rightarrow FV(t) \subseteq dom(\Gamma)]$
3. $\forall_\Gamma\forall_t\forall_\rho[\Gamma \vdash t : \rho \Rightarrow \Gamma \restriction FV(t) \vdash t : \rho]$

Proof: *All by induction on the derivation* $\Gamma \vdash t : \rho$. □

Lemma 2.14 *(Generation lemma)*

1. $\forall_\Gamma\forall_v\forall_\rho[\Gamma \vdash v : \rho \Rightarrow (\rho\lambda_v) \in' \Gamma]$
2. $\forall_\Gamma\forall_{t,t'}\forall_\rho[\Gamma \vdash (t'\delta)t : \rho \Rightarrow \exists_{\rho'}[\Gamma \vdash t : (\rho' \to \rho) \wedge \Gamma \vdash t' : \rho']]$

3. $\forall_\Gamma \forall_v \forall_t \forall_{\rho,\rho'} [\Gamma \vdash (\rho' \lambda_v) t : \rho \Rightarrow \exists_{\rho''} [\rho \equiv \rho' \to \rho'' \wedge \Gamma(\rho' \lambda_v) \vdash t : \rho'']]$

Proof: *By induction on the derivation of* $\Gamma \vdash t : \rho$. □

Lemma 2.15 *(Subterm lemma)*
$\forall_\Gamma \forall_t \forall_\rho \forall_{t'} [(\Gamma \vdash t : \rho \wedge t' \text{ is a subterm of } t) \Rightarrow \exists_{\Gamma'} \exists_{\rho'} [\Gamma' \vdash t' : \rho']]$
Proof: *By induction on* t. □

Lemma 2.16 *(Substitution lemma)*

1. $\forall_\Gamma \forall_t \forall_{\rho,\rho'} \forall_{\alpha \in \mathcal{V}} [\Gamma \vdash t : \rho \Rightarrow \Gamma[\alpha := \rho'] \vdash t[\alpha := \rho'] : \rho[\alpha := \rho']]$
2. $\forall_\Gamma \forall_{t,t'} \forall_{\rho,\rho'} [\Gamma(\rho \lambda_v) \vdash t : \rho' \wedge \Gamma \vdash t' : \rho \Rightarrow \Gamma \vdash t[v := t'] : \rho']$

Proof: *1: by induction on* $\Gamma \vdash t : \rho$. *2: by induction on* $\Gamma(\rho \lambda_v) \vdash t : \rho'$. □

Theorem 2.17 *(Subject Reduction)*
$\forall_\Gamma \forall_t \forall_{t' | t \twoheadrightarrow_\beta t'} \forall_\rho [\Gamma \vdash t : \rho \Rightarrow \Gamma \vdash t' : \rho]$
Proof: *By induction on* $\twoheadrightarrow_\beta$ *using the Generation and Substitution lemmas for the basic case.* □

Lemma 2.18 *(Unicity of Types)*

1. $\forall_\Gamma \forall_t \forall_{\rho,\rho'} [\Gamma \vdash t : \rho \wedge \Gamma \vdash t : \rho' \Rightarrow \rho \equiv \rho']$
2. $\forall_\Gamma \forall_{t,t'} \forall_{\rho,\rho'} [\Gamma \vdash t : \rho \wedge \Gamma \vdash t' : \rho' \wedge t =_\beta t' \Rightarrow \rho \equiv \rho']$

Proof: *1 is by an easy induction on* t. *2 is by Church Rosser, Subject Reduction and 1.* □

Definition 2.19 *(Strongly Normalising terms with respect to* $\twoheadrightarrow_\beta$*)*
We say that a term t *is strongly normalising with respect to* $\twoheadrightarrow_\beta$ *iff every reduction path using* $\twoheadrightarrow_\beta$ *and starting at* t *terminates.*

Theorem 2.20 *(Strong Normalisation with respect to* $\twoheadrightarrow_\beta$*)*
If $\Gamma \vdash t : \rho$ *then* t *is strongly normalising with respect to* $\twoheadrightarrow_\beta$. □

3 Generalising reduction

In this section we shall extend the classical notions of redexes and β-reduction of $\lambda_\to$. We shall show that this extension of $\lambda_\to$ satisfies all the listed properties in Section 2.

3.1 Extending redexes and β-reduction

When one desires to start a β-reduction on the basis of a certain δ-item and a λ-item occurring in one segment, the *matching* of the δ and the λ in question is the important thing, even when the δ- and λ-items are separated by other items. I.e., the relevant question is whether they *may* together become a $\delta\lambda$-segment after a number of β-steps. This depends solely on the structure of the intermediate segment. If such an intermediate segment is well-balanced then the δ-item and the λ-item match and β-reduction based on these two items may take place. Here is the definition of well-balanced segments:

Definition 3.1 *(well-balanced segments in $\lambda_{\to}$)*

- *The empty segment $\emptyset$ is a well-balanced segment;*
- *If $\overline{s}$ is a well-balanced segment, then $(t'\delta)\overline{s}(\rho\lambda_v)$ is a well-balanced segment.*
- *The concatenation of well-balanced segments is a well-balanced segment;*

A well-balanced segment has the same structure as a matching composite of opening and closing brackets, each δ- (or λ-)item corresponding with an opening (resp. closing) bracket.

Now we can easily define what matching $\delta\lambda$-couples are, given a segment $\overline{s}$. Namely, they are a main δ-item and a main λ-item separated by a well-balanced segment. Such couples are reducible couples. The δ-item and λ-item of the $\delta\lambda$-couple are said to match and each of them is called a partner or a partnered item. The items in a segment that are not partnered are called bachelor items. The following definition summarizes all this:

Definition 3.2 *(match, $\delta\lambda$- or reducible couple, partner, partnered item, bachelor item, bachelor segment)*
Let t be a $\lambda_{\to}$-term. Let $\overline{s} \equiv s_1 \dots s_n$ be a segment occurring in t.

- *We say that s_i and s_j* **match**, *when $1 \leq i < j \leq n$, s_i is a δ-item, s_j is a λ-item, and the sequence $s_{i+1}, \dots, s_{j-1}$ forms a well-balanced segment.*
- *When s_i and s_j match, we call $s_i s_j$ a* **$\delta\lambda$-couple** *or* **reducible couple**.
- *When s_i and s_j match, we call both s_i and s_j the* **partners** *in the $\delta\lambda$-couple. We also say that s_i and s_j are* **partnered** *items.*
- *All λ- (or δ-)items s_k in t that are not partnered, are called* **bachelor** *λ- (resp. δ-)items.*
- *A segment consisting of bachelor items only, is called a* **bachelor segment**.
- *The segment $s_{i_1} \cdots s_{i_m}$ consisting of all bachelor main λ- (or δ-)items of $\overline{s}$ is called the* **bachelor** *λ- (or δ-)***segment** *of $\overline{s}$.*

Example 3.3 In $\overline{s} \equiv (\rho_1\lambda_{v_1})(\rho_2\lambda_{v_2})(t_1\delta)(\rho_3\lambda_{v_3})(\rho_4\lambda_{v_4})(t_2\delta)(t_3\delta)(t_4\delta)(\rho_5\lambda_{v_5})(\rho_6\lambda_{v_6})(t_5\delta)$:

- $(t_1\delta)$ matches with $(\rho_3\lambda_{v_3})$, $(t_4\delta)$ matches with $(\rho_5\lambda_{v_5})$ and $(t_3\delta)$ with $(\rho_6\lambda_{v_6})$. The segments $(t_1\delta)(\rho_3\lambda_{v_3})$ and $(t_4\delta)(\rho_5\lambda_{v_5})$ are $\delta\lambda$-segments (and $\delta\lambda$-couples). There is another $\delta\lambda$-couple in $\overline{s}$, viz. the couple of $(t_3\delta)$ and $(\rho_6\lambda_{v_6})$.
- $(t_1\delta)$, $(\rho_3\lambda_{v_3})$, $(t_3\delta)$, $(t_4\delta)$, $(\rho_5\lambda_{v_5})$ and $(\rho_6\lambda_{v_6})$, are the partnered main items of $\overline{s}$. $(\rho_1\lambda_{v_1})$, $(\rho_2\lambda_{v_2})$, $(\rho_4\lambda_{v_4})$, $(t_2\delta)$ and $(t_5\delta)$, are bachelor items.
- $(\rho_1\lambda_{v_1})(\rho_2\lambda_{v_2})$ and $(\rho_4\lambda_{v_4})(t_2\delta)$ are bachelor segments, whereas $(t_3\delta)(t_4\delta)(\rho_5\lambda_{v_5})$ and $(t_3\delta)(t_4\delta)(\rho_5\lambda_{v_5})(\rho_6\lambda_{v_6})$ are non-bachelor, the latter also being a well-balanced segment.

De Bruijn uses another terminology; see e.g. [de Bruijn 93]. In his phrasing, δ-items are *applicators* or A's, and λ-items are *abstractors* or T's. For $\delta\lambda$-segments he uses the word *AT-pair* and for $\delta\lambda$-couples he uses *AT-couples*. Void β-reduction (i.e.: the reduction $(t_1\delta)(\rho\lambda_v)t \to_\beta t$ if $v \notin FV(t)$), he calls *AT-removal*.

Having argued above that β-reduction should not be restricted to the $\delta\lambda$-segments but may take into account other candidates, we can extend our notion of β-reduction in this vein. That is to say, we may allow $\delta\lambda$-*couples* to have the same "reduction rights" as $\delta\lambda$-*segments*. In order to accomplish this, we change the β-reduction of Definition 2.6 to the following:

Definition 3.4 *(General β-reduction $\rightsquigarrow_\beta$ for $\lambda_\rightarrow$)*
General one-step β-reduction $\rightsquigarrow_\beta$, is the least compatible relation generated out of the following axiom:

$$(general\ \ \beta) \qquad (t_1\delta)\overline{s}(\rho\lambda_v)t \rightsquigarrow_\beta \overline{s}(t[v := t_1]) \qquad \text{if } \overline{s} \text{ is well-balanced}$$

General $\twoheadrightsquigarrow_\beta$ is the reflexive and transitive closure of $\rightsquigarrow_\beta$ and $\approx_\beta$ is the least equivalence relation generated by $\twoheadrightsquigarrow_\beta$.

Example 3.5 Take Example 1.4. As $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})$ is a well-balanced segment, then $(x_1\delta)((X_1 \rightarrow X_2)\lambda_{x_5})$ is a $\delta\lambda$-couple and

$$\begin{array}{ll} t \equiv (x_1\delta)(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})((X_1 \rightarrow X_2)\lambda_{x_5})(x_4\delta)x_5 & \rightsquigarrow_\beta \\ (x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(((x_4\delta)x_5)[x_5 := x_1]) & \equiv \\ (x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(x_4\delta)x_1 & \end{array}$$

The reducible *couple* $(x_1\delta)((X_1 \rightarrow X_2)\lambda_{x_5})$ also has a corresponding ("generalised") redex in the traditional notation, which will appear after two one-step β-reductions, leading to $(\lambda_{x_5:X_1\rightarrow X_2}.x_5x_4)x_1$. With $\rightsquigarrow_\beta$, we could reduce $((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\rightarrow X_2}.x_5x_4)x_3)x_2)x_1$ to $(\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.x_1x_4)x_3)x_2$. This reduction is difficult to carry out in the classical λ-calculus. The item notation enables a new and important sort of reduction which has not yet been studied in relation to the standard λ-calculus up to date. We believe that this generalised reduction (introduced in [Nederpelt 73]) can only be obtained tidily in a system formulated using some form of our item notation. In fact, one is to compare the bracketing structure of the classical term t of Example 1.4, with the bracketing structure of the corresponding term in item notation:

Example 3.6 The "bracketing structure" of $((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\rightarrow X_2}.--)x_3)x_2)x_1$ is compatible with '$[_1\ [_2\ [_3\ ]_2\ ]_1\ ]_3$', where '$[_i$' and '$]_i$' match. In item notation however, it has the bracketing structure $[[\,][\,]]$.

We strongly believe that it is the item notation which enables us to extend reduction smoothly beyond $\twoheadrightarrow_\beta$. Because a well-balanced segment may be empty, the general β-reduction rule presented above is really an extension of the classical β-reduction rule.

Lemma 3.7 *Let t_1, t_2 be $\lambda_\rightarrow$-terms. If $t_1 \rightarrow_\beta t_2$ in the sense of Definition 2.6, then $t_1 \rightsquigarrow_\beta t_2$ in the sense of Definition 3.4. Moreover, if $t_1 \rightsquigarrow_\beta t_2$ comes from contracting a $\delta\lambda$-segment then $t_1 \rightarrow_\beta t_2$.*

Proof: *Obvious as a $\delta\lambda$-segment is a $\delta\lambda$-couple.* □

3.2 Properties of $\lambda_\rightarrow$ with generalised reduction

If we look at Section 2.2, we see that Context, Generation, Subterm, Substitution and Unicity of Types (part 1) lemmas are not affected by our extension of Reduction. Hence, they all still hold for $\lambda_\rightarrow$ with $\twoheadrightsquigarrow_\beta$. The only three (and very important) properties that get affected

by $\rightsquigarrow\!\!\!\!\rightarrow_\beta$ are: Church Rosser (Theorem 2.12), Subject Reduction (Theorem 2.17), Unicity of Types (part 2, Lemma 2.18), and Strong Normalisation (Theorem 2.20). In this section, we shall show that these properties are preserved for $\rightsquigarrow\!\!\!\!\rightarrow_\beta$.

The proof of the generalised Church Rosser theorem is simple. The idea is to show that if $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t'$ then $t =_\beta t'$ and to use the Church Rosser property for $=_\beta$.

Lemma 3.8 *If $t \rightsquigarrow_\beta t'$ then $t =_\beta t'$.*

Proof: *It suffices to consider the case $t \equiv \overline{s_1}(t_1\delta)\overline{s}(\rho\lambda_v)t_2$ where the contracted redex is based on $(t_1\delta)(\rho\lambda_v)$, $t' \equiv \overline{s_1}\,\overline{s}(t_2[v := t_1])$, and $\overline{s}$ is well-balanced (hence $\mathtt{weight}(\overline{s})$ is even). We shall prove the lemma by induction on $\mathtt{weight}(\overline{s})$.*

- *Case $\mathtt{weight}(\overline{s}) = 0$ then obvious as $\rightsquigarrow_\beta$ coincides with $\rightarrow_\beta$ in this case.*
- *Assume the property holds when $\mathtt{weight}(\overline{s}) = 2n$. Take $\overline{s}$ such that $\mathtt{weight}(\overline{s}) = 2n+2$. Now, $\overline{s} \equiv (t_3\delta)\overline{s'}(\rho'\lambda_{v'})\overline{s''}$ where $\overline{s'}$, $\overline{s''}$ are well-balanced. Assume $v \not\equiv v'$ (if necessary, use renaming).*
  - *As $\overline{s}(t_2[v := t_1]) \rightsquigarrow_\beta \overline{s'}(\overline{s''}(t_2[v := t_1])[v' := t_3])$, we get by IH and compatibility that $t' =_\beta \overline{s_1}\,\overline{s'}(\overline{s''}(t_2[v := t_1])[v' := t_3]) \equiv \overline{s_1}\,\overline{s'}(\overline{s''}[v' := t_3])(t_2[v := t_1][v' := t_3]) \equiv t$.*
  - *Moreover, $t \equiv \overline{s_1}(t_1\delta)(t_3\delta)\overline{s'}(\rho'\lambda_{v'})\overline{s''}(\rho\lambda_v)t_2 \rightsquigarrow_\beta \overline{s_1}(t_1\delta)\overline{s'}(\overline{s''}(\rho\lambda_v)t_2[v' := t_3]) \equiv^{BC} \overline{s_1}(t_1\delta)\overline{s'}(\overline{s''}[v' := t_3])(\rho\lambda_v)(t_2[v' := t_3]) \equiv t''$. Hence by IH, $t =_\beta t''$.*
  - *Now, $t'' \rightsquigarrow_\beta \overline{s_1}\overline{s'}(\overline{s''}[v' := t_3])(t_2[v' := t_3][v := t_1])$. But by BC, v, $v' \notin FV(t_1) \cup FV(t_3)$. Hence, by IH and substitution, $t'' =_\beta \overline{s_1}\overline{s'}(\overline{s''}[v' := t_3])(t_2[v := t_1][v' := t_3]) \equiv t'''$.*

  Therefore, $t =_\beta t''$, $t'' =_\beta t'''$ and $t' =_\beta t'''$, hence $t =_\beta t'$. □

Corollary 3.9 *If $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t'$ then $t =_\beta t'$.* □

Theorem 3.10 *(The general Church Rosser theorem)*
If $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t_1$ and $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t_2$, then there exists t_3 such that $t_1 \rightsquigarrow\!\!\!\!\rightarrow_\beta t_3$ and $t_2 \rightsquigarrow\!\!\!\!\rightarrow_\beta t_3$.

Proof: *As $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t_1$ and $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t_2$ then by Corollary 3.9, $t =_\beta t_1$ and $t =_\beta t_2$. Hence, $t_1 =_\beta t_2$ and by the Church Rosser property for the classical lambda calculus, there exists t_3 such that $t_1 \twoheadrightarrow_\beta t_3$ and $t_2 \twoheadrightarrow_\beta t_3$. But, $t' \twoheadrightarrow_\beta t''$ implies $t' \rightsquigarrow\!\!\!\!\rightarrow_\beta t''$. Hence the Church-Rosser theorem holds for the general β-reduction.* □

For the proof of Subject Reduction, we need the following "shuffle lemma".

Lemma 3.11 $\Gamma \vdash \overline{s_1}(t_1\delta)\overline{s_2}t_2 : \rho \Leftrightarrow \Gamma \vdash \overline{s_1}\,\overline{s_2}(t_1\delta)t_2 : \rho$ *where $\overline{s_2}$ is well-balanced and the binding variables in $\overline{s_2}$ are not free in t_1.*

Proof: *By induction on $\mathtt{weight}(\overline{s_2})$.*

- *case $\mathtt{weight}(\overline{s_2}) = 0$ then nothing to prove.*
- *case $\mathtt{weight}(\overline{s_2}) = 2$, say $\overline{s_2} \equiv (t_3\delta)(\rho_1\lambda_v)$. We use induction on $\mathtt{weight}(\overline{s_1})$.*

– *Suppose* $\texttt{weight}(\overline{s_1}) = 0$.
$\Rightarrow$) *suppose* $\Gamma \vdash (t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho$.
Using the Generation lemma three times, we obtain:

$$\Gamma \vdash (t_3\delta)(\rho_1\lambda_v)t_2 : \rho' \to \rho$$
$$\Gamma \vdash t_1 : \rho' \quad (1)$$
$$\Gamma \vdash (\rho_1\lambda_v)t_2 : \rho_1 \to (\rho' \to \rho)$$
$$\Gamma \vdash t_3 : \rho_1 \quad (2)$$
$$\Gamma(\rho_1\lambda_v) \vdash t_2 : \rho' \to \rho \quad (3)$$

$$\textit{Hence} \quad \Gamma(\rho_1\lambda_v) \vdash (t_1\delta)t_2 : \rho \quad \textit{(Context, }\to\textit{-elimination, (1), (3))} \quad (4)$$
$$\textit{Now,} \quad \Gamma \vdash (\rho_1\lambda_v)(t_1\delta)t_2 : \rho_1 \to \rho \quad \textit{(}\to\textit{-introduction, (4))} \quad (5)$$
$$\textit{And so,} \quad \Gamma \vdash (t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho \quad \textit{(}\to\textit{-elimination, (5), (2))}$$

$\Leftarrow$) *Suppose* $\Gamma \vdash (t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho$.
Using the Generation lemma three times we obtain:

$$\Gamma \vdash (\rho_1\lambda_v)(t_1\delta)t_2 : \rho_1 \to \rho$$
$$\Gamma \vdash t_3 : \rho_1 \quad (6)$$
$$\Gamma(\rho_1\lambda_v) \vdash (t_1\delta)t_2 : \rho$$
$$\Gamma(\rho_1\lambda_v) \vdash t_2 : \rho' \to \rho \quad (7)$$
$$\Gamma(\rho_1\lambda_v) \vdash t_1 : \rho' \quad (8)$$

$$\textit{Hence,} \quad \Gamma \vdash (\rho_1\lambda_v)t_2 : \rho_1 \to (\rho' \to \rho) \quad \textit{(}\to\textit{-introduction, (7))} \quad (9)$$
$$\Gamma \vdash (t_3\delta)(\rho_1\lambda_v)t_2 : \rho' \to \rho \quad \textit{(}\to\textit{-elimination, (6), (9))} \quad (10)$$
$$\Gamma \vdash t_1 : \rho' \quad \textit{(context, (8), as } v \notin FV(t_1)\textit{)} \quad (11)$$
$$\Gamma \vdash (t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho \quad \textit{(}\to\textit{-elimination, (10), (11))}$$

– *Now suppose* $\texttt{weight}(\overline{s_1}) = n + 1$.

∗ *Case* $\overline{s_1} \equiv (t_4\delta)\overline{s_1'}$ *then*
$\Gamma \vdash (t_4\delta)\overline{s_1'}(t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho \Leftrightarrow^{Generation;\to-elimination}$
$\Gamma \vdash \overline{s_1'}(t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho' \to \rho \wedge \Gamma \vdash t_4 : \rho' \Leftrightarrow^{IH}$
$\Gamma \vdash \overline{s_1'}(t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho' \to \rho \wedge \Gamma \vdash t_4 : \rho' \Leftrightarrow^{\to-elimination;Generation}$
$\Gamma \vdash (t_4\delta)\overline{s_1'}(t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho$

∗ *Case* $\overline{s_1} \equiv (\rho_2\lambda_{v'})\overline{s_1'}$ *then*
$\Gamma \vdash (\rho_2\lambda_{v'})\overline{s_1'}(t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho \Leftrightarrow^{Generation;\to-introduction}$
$\Gamma(\rho_2\lambda_{v'}) \vdash \overline{s_1'}(t_1\delta)(t_3\delta)(\rho_1\lambda_v)t_2 : \rho_3 \wedge \rho \equiv \rho_2 \to \rho_3 \Leftrightarrow^{IH}$
$\Gamma(\rho_2\lambda_{v'}) \vdash \overline{s_1'}(t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho_3 \wedge \rho \equiv \rho_2 \to \rho_3 \Leftrightarrow^{\to-introduction;Generation}$
$\Gamma \vdash (\rho_2\lambda_{v'})\overline{s_1'}(t_3\delta)(\rho_1\lambda_v)(t_1\delta)t_2 : \rho.$

- *case* $\texttt{weight}(\overline{s_2}) = 2(n+1), n \geq 1$. *If* $\overline{s_2} \equiv (t_3\delta)\overline{s_3}(\rho_1\lambda_v)\overline{s_4}$ *where* $\overline{s_3}, \overline{s_4}$ *are well-balanced and IH holds for them, then:*
$\Gamma \vdash \overline{s_1}(t_1\delta)(t_3\delta)\overline{s_3}(\rho_1\lambda_v)\overline{s_4}t_2 : \rho \Leftrightarrow^{IH}$
$\Gamma \vdash \overline{s_1}(t_1\delta)\overline{s_3}(t_3\delta)(\rho_1\lambda_v)\overline{s_4}t_2 : \rho \Leftrightarrow^{IH}$
$\Gamma \vdash \overline{s_1}\,\overline{s_3}(t_1\delta)(t_3\delta)(\rho_1\lambda_v)\overline{s_4}t_2 : \rho \Leftrightarrow^{IH}$

$\Gamma \vdash \overline{s_1}\,\overline{s_3}(t_3\delta)(\rho_1\lambda_v)(t_1\delta)\overline{s_4}t_2 : \rho \Leftrightarrow^{IH}$
$\Gamma \vdash \overline{s_1}(t_3\delta)\overline{s_3}(\rho_1\lambda_v)(t_1\delta)\overline{s_4}t_2 : \rho \Leftrightarrow^{IH}$
$\Gamma \vdash \overline{s_1}(t_3\delta)\overline{s_3}(\rho_1\lambda_v)\overline{s_4}(t_1\delta)t_2 : \rho.$ □

Remark 3.12 Note that in Lemma 3.11 above, we insisted on the condition that the binding variables in $\overline{s_2}$ are not free in t_1 in order to avoid cases such as moving $(v\delta)$ in $(t_0\delta)(\rho\lambda_v)(v\delta)$ to the left of $(t_0\delta)(\rho\lambda_v)$.

Now we can prove Subject Reduction for generalised β-reduction.

Theorem 3.13 *(Generalised Subject Reduction)*
If $\Gamma \vdash t : \rho$ *and* $t \twoheadrightarrow_\beta t'$ *then* $\Gamma \vdash t' : \rho$.
Proof: *By induction on* $\twoheadrightarrow_\beta$.

- *Basic Case:* $(t_1\delta)\overline{s_2}(\rho_1\lambda_v)t_2 \rightsquigarrow_\beta \overline{s_2}(t_2[v := t_1])$ *and* $\Gamma \vdash (t_1\delta)\overline{s_2}(\rho_1\lambda_v)t_2 : \rho$.
  $\Gamma \vdash (t_1\delta)\overline{s_2}(\rho_1\lambda_v)t_2 : \rho \Rightarrow^{Lemma\ 3.11}$
  $\Gamma \vdash \overline{s_2}(t_1\delta)(\rho_1\lambda_v)t_2 : \rho \Rightarrow^{Lemma\ 2.17}$
  $\Gamma \vdash \overline{s_2}(t_2[v := t_1]) : \rho.$
- *The reflexivity, transitivity and compatibility cases are easy.* □

For Unicity of Types, we just need the following lemma:

Lemma 3.14 *If* $t \approx_\beta t'$ *then* $t =_\beta t'$.
Proof: *By induction on* $t \approx_\beta t'$ *using Corollary 3.9.* □

Lemma 3.15 *(Generalised Unicity of Types)*

1. $\forall_\Gamma \forall_t \forall_{\rho,\rho'}[\Gamma \vdash t : \rho \wedge \Gamma \vdash t : \rho' \Rightarrow \rho \equiv \rho']$
2. $\forall_\Gamma \forall_{t,t'} \forall_{\rho,\rho'}[\Gamma \vdash t : \rho \wedge \Gamma \vdash t' : \rho' \wedge t \approx_\beta t' \Rightarrow \rho \equiv \rho']$

Proof: *The proof of 1 is the same for Lemma 2.18. The proof of 2 is also carried from Lemma 2.18 using Lemma 3.14 above.* □

Now we come to the proof of Strong Normalisation. For this, we need the following definition:

Definition 3.16

- *We say that* $t \in \Lambda_\mathcal{T}$ *is* **strongly normalising** *with respect to* $\twoheadrightarrow_\beta$ *iff every reduction path (with respect to* $\twoheadrightarrow_\beta$*) starting at* t*, terminates.*
- *We define* $SN = \{t \in \Lambda_\mathcal{T} : t \text{ is strongly normalising with respect to } \twoheadrightarrow_\beta\}$.
- *For* $A, B \subseteq \Lambda_\mathcal{T}$ *we define* $A \longrightarrow B = \{t \in \Lambda_\mathcal{T} : \forall t' \in A[(t'\delta)t \in B]\}$.
- *We define* $[\![\ ]\!] : \mathcal{T} \longrightarrow$ *Power Set of* $\Lambda_\mathcal{T}$ *as follows:*

$$\begin{array}{lcl} [\![\alpha]\!] & = & SN \\ [\![\rho \to \rho']\!] & = & [\![\rho]\!] \longrightarrow [\![\rho']\!] \end{array}$$

- *We call* $X \subseteq SN$ **saturated** *iff:*

1. $\forall n \geq 0, t_1, \cdots t_n \in SN, v \in V[(t_1\delta)\cdots(t_n\delta)v \in X]$.
2. $\forall n \geq 0, t, t_1, \cdots, t_n \in SN, \rho \in \mathcal{T}, \overline{s}$ *well-balanced*, $t' \in \Lambda_{\mathcal{T}}$
   $[(t_1\delta)\cdots(t_n\delta)\overline{s}(t'[v := t]) \in X \Rightarrow (t_1\delta)\cdots(t_n\delta)(t\delta)\overline{s}(\rho\lambda_v)t' \in X]$.

- *We define* $SAT = \{X \subseteq \Lambda_{\mathcal{T}} : X$ *saturated*$\}$

Those familiar with the proof of Strong Normalisation of $\lambda_{\rightarrow}$, will notice that we have accommodated $\rightsquigarrow_\beta$ in the definition of SN and that in the second condition of a saturated set, we have accommodated extended redexes. The accommodation of saturated sets with extended redexes is not necessary, the proof can go without it. Furthermore, the following is the crucial lemma which highlights the difference between $\twoheadrightarrow_\beta$ and $\rightsquigarrow_\beta$. Once this lemma is established, the proof of Strong Normalisation proceeds similarly to that of ordinary $\twoheadrightarrow_\beta$.

Lemma 3.17

1. $SN \in SAT$.
2. $A, B \in SAT \Rightarrow A \longrightarrow B \in SAT$.
3. $\rho \in \mathcal{T} \Rightarrow [\rho] \in SAT$.

Proof:

1. $SN \subseteq SN$ *and if* $t_1, \cdots, t_n \in SN, v \in V$ *then similarly* $(t_1\delta)\cdots(t_n\delta)v \in SN$.
   Now, if $t, t_1, \cdots, t_n \in SN, \rho \in \mathcal{T}, \overline{s}$ *is well-balanced and* $t' \in \Lambda_{\mathcal{T}}$ *such that* $(t_1\delta)\cdots(t_n\delta)\overline{s}(t'[v := t]) \in SN$ *then also* $(t_1\delta)\cdots(t_n\delta)(t\delta)\overline{s}(\rho\lambda_v)t' \in SN$*:*
   - *Reductions inside* $t', t, \overline{s}$ *or one of the* t_i *must terminate since these terms are SN (subterms of SN-terms are themselves SN,* $t'[v := t]$ *is SN* $\Rightarrow t'$ *is SN).*
   - *A reduction path of* $(t_1\delta)\cdots(t_n\delta)(t\delta)\overline{s}(\rho\lambda_v)t'$ *goes to* $(t'_1\delta)\cdots(t'_n\delta)(t''\delta)\overline{s'}(\rho\lambda_v)t'''$ *with* $t' \rightsquigarrow_\beta t'''$ *etc. and then to* $(t'_1\delta)\cdots(t'_n\delta)\overline{s'}(t'''[v := t''])$*; since* $(t_1\delta)\cdots(t_n\delta)\overline{s}(t'[v := t]) \in SN$ *also* $(t'_1\delta)\cdots(t'_n\delta)\overline{s'}(t'''[v := t'']) \in SN$.
2. *Suppose* $A, B \in SAT$.
   - *As* $v \in A$ *for all* $v \in V$*, we see:* $t \in A \longrightarrow B \Rightarrow (v\delta)t \in B \Rightarrow (v\delta)t \in SN \Rightarrow t \in SN$*. So* $A \longrightarrow B \subseteq SN$.
   - *If* $t_1, \cdots, t_n \in SN, v \in V$ *then for all* $t \in A$*, as* $t \in SN$ *and* $B \in SAT$*, we get that* $(t\delta)(t_1\delta)\cdots(t_n\delta)v \in B$*. Hence* $(t_1\delta)\cdots(t_n\delta)v \in A \longrightarrow B$ *which proves condition* 1 *of saturation.*
   - *As to condition 2, suppose* $t, t_1, \cdots, t_n \in SN, t' \in \Lambda_{\mathcal{T}}, \overline{s}$ *is well-balanced,* ρ *a type and* $(t_1\delta)\cdots(t_n\delta)\overline{s}(t'[v := t]) \in A \longrightarrow B$.
     Let $t'' \in A$*. Then* $(t''\delta)(t_1\delta)\cdots(t_n\delta)\overline{s}(t'[v := t]) \in B$*, by definition of* $A \longrightarrow B$.
     Hence $(t''\delta)(t_1\delta)\cdots(t_n\delta)(t\delta)\overline{s}(\rho\lambda_v)t' \in B$ *since* $B \in SAT, t'' \in A \subseteq SN$.
     This means $(t_1\delta)\cdots(t_n\delta)(t\delta)\overline{s}(\rho\lambda_v)t' \in A \longrightarrow B$.
3. *Easy induction on the generation of* ρ *using 1 and 2.* □

Corollary 3.18 *For all $\rho \in \mathcal{T}$, we have $[\rho] \neq \emptyset$ and $[\rho] \subseteq SN$.*

Proof: *Note that no saturated set is empty (use $SN \neq \emptyset$ and condition 1 of saturated sets).* □

Definition 3.19

- *A* **valuation** *is a map $g : V \longrightarrow \Lambda_{\mathcal{T}}$*
- *If g is a valuation then $[\![\ ]\!]_g$ is defined inductively as follows:*

$$\begin{array}{lcl} [\![v]\!]_g & = & g(v) \\ [\![(t\delta)t']\!]_g & = & ([\![t]\!]_g\delta)[\![t']\!]_g \\ [\![(\rho\lambda_v)t]\!]_g & = & (\rho\lambda_v)[\![t]\!]_{g(v:=v)} \end{array}$$

  where $g(v := N)$ is the valuation that assigns $g(v')$ to $v' \not\equiv v$ and N to v. Note that $[\![t]\!]_g$ substitutes $g(v')$ for v' in t for all free variables v' of t. For example, $[\![(\rho\lambda_x)(y\delta)x]\!]_g = (\rho\lambda_x)(g(y)\delta)x$.

- *$\models$ is defined as follows:*

$$\begin{array}{lll} g & \models t : \rho & \textit{iff} \quad [\![t]\!]_g \in [\rho] \\ g & \models \Gamma & \textit{iff} \quad \textit{for all } (\rho\lambda_v) \in' \Gamma, \textit{ we have } g \models v : \rho \\ \Gamma & \models t : \rho & \textit{iff} \quad \textit{for all valuations } g, \textit{ if } g \models \Gamma \textit{ then } g \models t : \rho \end{array}$$

Lemma 3.20 *(Soundness)*
If $\Gamma \vdash t : \rho$ then $\Gamma \models t : \rho$.

Proof: *By a straightforward induction on the derivation of $\Gamma \vdash t : \rho$. We only treat the $\rightarrow$-introduction.*
Suppose $\Gamma \vdash (\rho\lambda_v)t : \rho \rightarrow \rho'$ out of $\Gamma(\rho\lambda_v) \vdash t : \rho'$.
Suppose $g \models \Gamma$ in order to show $g \models (\rho\lambda_v)t : \rho \rightarrow \rho'$ (i.e. for all $t' \in [\rho]$: $(t'\delta)[\![(\rho\lambda_v)t]\!]_g \in [\rho']$).
Let $t' \in [\rho]$. Then $g(v := t') \models \Gamma(\rho\lambda_v)$, so by the induction hypothesis $[\![t]\!]_{g(v:=t')} \in [\![\rho']\!]$.
Since $(t'\delta)[(\rho\lambda_v)t]\!]_g \equiv (t'\delta)(\rho\lambda_v)[\![t]\!]_{g(v:=v)} \rightarrow_\beta [\![t]\!]_{g(v:=v)}[v := t'] \equiv [\![t]\!]_{g(v:=t')}$, $t' \in [\![\rho]\!] \subseteq SN$ and $[\rho'] \in SAT$, also $(t'\delta)[(\rho\lambda_v)t]\!]_g \in [\rho']$. □

Theorem 3.21 *(Strong Normalisation with respect to $\twoheadrightarrow_\beta$)*
If $\Gamma \vdash t : \rho$ then t is strongly normalising with respect to $\twoheadrightarrow_\beta$.

Proof: *Suppose $\Gamma \vdash t : \rho$. Define $g(v) = v$. Then $g \models \Gamma$ (because $[\rho] \in SAT$, so $V \subseteq [\![\rho]\!]$). Hence by soundness $[\![t]\!]_g \in [\rho] \subseteq SN$. But $[\![t]\!]_g \equiv t$.* □

4 Term reshuffling

In this section we shall rewrite terms so that all the newly visible redexes (obtained as a result of our item notation), can be subject to the ordinary classical β-reduction $\twoheadrightarrow_\beta$. We shall show that this term rewriting is correct and preserves both reduction (be it only in a certain sense) and typing.

Let us go back to the definition of $\delta\lambda$-couples. Recall that if $\overline{s} \equiv s_1 \cdots s_m$ for $m > 1$ where $s_1 s_m$ is a $\delta\lambda$-couple then $s_2 \cdots s_{m-1}$ is a well-balanced segment, $s_1 \equiv (t_1\delta)$ is the δ-item of the $\delta\lambda$-couple and $s_m \equiv (\rho\lambda_v)$ is its λ-item. Now, we can move s_1 in $\overline{s}$ so that it occurs adjacently to s_m. That is, we may rewrite $\overline{s}$ as $s_2 \cdots s_{m-1} s_1 s_m$.

Example 4.1 The term $(x_1\delta)(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$ can be easily rewritten as $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(x_1\delta)((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$ by moving the item $(x_1\delta)$ to the right. Hence, we can rewrite (or *reshuffle*) a term so that all δ-items stand next to their matching λ-items. This means that we can keep the old β-axiom and we can contract redexes in any order. Such an action of reshuffling is not easy to describe in the classical notation. That is, it is difficult to describe how $((\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.\lambda_{x_5:X_1\to X_2}.x_5x_4)x_3)x_2)x_1$, is rewritten as $(\lambda_{x_7:X_4}.(\lambda_{x_6:X_3}.(\lambda_{x_5:X_1\to X_2}.x_5x_4)x_1)x_3)x_2$. This is another advantage of our item notation.

Note furthermore that the shuffling is not problematic because we use the Barendregt Convention which means that no free variable will become unnecessarily bound after reshuffling due to the fact that names of bound and free variables are distinct.

Lemma 4.2 *If v° is a free occurrence of v in $s\overline{s_1}\,t$, then v° is free in $\overline{s_1}s\,t$.*
Proof: *By BC as λ_v does not occur in $s\overline{s_1}t$.* □

Example 4.3 Note that in Example 4.1, reshuffling does not affect the "meaning" of the term. In fact, in $t \equiv (x_1\delta)(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$, the free variable x_1 cannot be captured by λ_{x_7} or λ_{x_6}. Moreover, t is equivalent, semantically and procedurally, to $(x_2\delta)(X_4\lambda_{x_7})(x_3\delta)(X_3\lambda_{x_6})(x_1\delta)((X_1 \to X_2)\lambda_{x_5})(x_4\delta)x_5$.

We call this process of moving δ-items of $\delta\lambda$-couples in a term to occupy positions adjacent to their λ-partners, *term reshuffling*. This term reshuffling should be such that *all* the δ-items of well-balanced segments in a term are shifted to the right until they meet their λ-partners. To do this however, we must study the classes of partnered and bachelor items in a term.

4.1 Partitioning the term into bachelor and well-balanced segments

With Definition 3.2, we may categorize the main items of a term t into different classes:

1. The "partnered" items (i.e. the δ- and λ-items which are partners, hence "coupled" to a matching one).
2. The "bachelors" (i.e. the bachelor λ-items and bachelor δ-items).

Lemma 4.4 *Let $\overline{s}$ be the body of a term t. Then the following holds:*

1. *Each bachelor main λ-item in $\overline{s}$ precedes each bachelor main δ-item in $\overline{s}$.*
2. *The removal from $\overline{s}$ of all bachelor main items, leaves behind a* well-balanced *segment.*
3. *The removal from $\overline{s}$ of all main $\delta\lambda$-couples, leaves behind a $\underbrace{\lambda\ldots\lambda}_{n}\underbrace{\delta\ldots\delta}_{m}$-segment, consisting of all bachelor main λ- and δ-items.*
4. *If $\overline{s} \equiv \overline{s_1}(t\delta)\overline{s_2}(\rho\lambda_v)\overline{s_3}$ where $(\rho\lambda_v)$ and $(t\delta)$ match, then $\overline{s_2}$ is well-balanced.*

Proof: *1 is by induction on* $\mathtt{weight}(\overline{s'})$ *for $\overline{s} \equiv \overline{s'}(\rho\lambda_v)\overline{s''}$ and $(\rho\lambda_v)$ bachelor in $\overline{s}$. 2 and 3 are by induction on* $\mathtt{weight}(\overline{s})$. *4 is by induction on* $\mathtt{weight}(\overline{s_2})$. □

Note that we have assumed $\emptyset$ well-balanced. We assume it moreover non-bachelor.

Corollary 4.5 *For each non-empty segment $\overline{s}$, there is a unique* partitioning *in segments $\overline{s_0}, \overline{s_1}, \ldots, \overline{s_n}$, such that*

1. $\overline{s} \equiv \overline{s_0}\,\overline{s_1} \cdots \overline{s_n}$,
2. *For all $0 \leq i \leq n$, $\overline{s_i}$ is well-balanced in $\overline{s}$ for even i and $\overline{s_i}$ is bachelor in $\overline{s}$ for odd i.*
3. *each bachelor λ-segment $\overline{s_j}$ precedes each bachelor δ-segment $\overline{s_k}$ in $\overline{s}$.*
4. $\overline{s_{2n}} \not\equiv \emptyset$ *for* $n > 0$. □

Example 4.6 $\overline{s} \equiv (\rho_1\lambda_{v_1})(\rho_2\lambda_{v_2})(t_1\delta)(\rho_3\lambda_{v_3})(\rho_4\lambda_{v_4})(t_2\delta)(t_3\delta)(t_4\delta)(\rho_5\lambda_{v_5})(\rho_6\lambda_{v_6})(t_5\delta)$ has the following partitioning:

- well-balanced segment $\overline{s_0} \equiv \emptyset$,
- bachelor segment $\overline{s_1} \equiv (\rho_1\lambda_{v_1})(\rho_2\lambda_{v_2})$,
- well-balanced segment $\overline{s_2} \equiv (t_1\delta)(\rho_3\lambda_{v_3})$,
- bachelor segment $\overline{s_3} \equiv (\rho_4\lambda_{v_4})(t_2\delta)$,
- well-balanced segment $\overline{s_4} \equiv (t_3\delta)(t_4\delta)(\rho_5\lambda_{v_5})(\rho_6\lambda_{v_6})$,
- bachelor segment $\overline{s_5} \equiv (t_5\delta)$.

4.2 The reshuffling procedure and its properties

In what follows, we use $\omega_1, \omega_2, \ldots$ to range over $\{\delta\} \cup \{\lambda_v; v \in V\}$, and we shall use $A_1, A_2, \ldots$ to range over both terms and types (i.e. over $\Lambda_{\mathcal{T}} \cup \mathcal{T}$).

Definition 4.7 *TS and T are defined mutually recursively such that:*

$TS(\rho)$	$=_{df}$	ρ	
$TS(\overline{s}v)$	$=_{df}$	$TS(\overline{s})v$	
$TS((A_1\omega_1)\cdots(A_n\omega_n))$	$=_{df}$	$(TS(A_1)\omega_1)\cdots(TS(A_n)\omega_n)$	*if $(A_1\omega_1)\cdots(A_n\omega_n)$ is bachelor*
$TS(\overline{s})$	$=_{df}$	$T(\emptyset, \overline{s})$	*if $\overline{s}$ is well-balanced*
$TS(\overline{s_0}\cdots\overline{s_n})$	$=_{df}$	$TS(\overline{s_0})\cdots TS(\overline{s_n})$	*If $\overline{s_0}\cdots\overline{s_n}$, is the unique partitioning of Corollary 4.5*
$T(\overline{s}(t\delta), (\rho\lambda_v)\overline{s'})$	$=_{df}$	$(t\delta)(\rho\lambda_v)T(\overline{s}, \overline{s'})$	
$T(\overline{s}, (t\delta)\overline{s'})$	$=_{df}$	$T(\overline{s}(TS(t)\delta), \overline{s'})$	
$T(\emptyset, \emptyset)$	$=_{df}$	$\emptyset$	

Note that in this definition, we use $\overline{s}$ bachelor to mean $\overline{s}$ bachelor in $\overline{s}$.

The following lemma will be needed in the proofs:

Lemma 4.8

1. *If $\overline{s}$ is well-balanced, then $T(\overline{s_1}, \overline{s}\,\overline{s_2}) \equiv TS(\overline{s})T(\overline{s_1}, \overline{s_2})$.*
2. *If $(t\delta)$ matches $(\rho\lambda_v)$ in $\overline{s'} \equiv (t\delta)\overline{s}(\rho\lambda_v)\overline{s''}$ then $TS(\overline{s'}) \equiv TS(\overline{s}(t\delta)(\rho\lambda_v)\overline{s''})$.*[1]

[1] Note here that, from BC, no binding variables of $\overline{s}$ are free in t.

3. *If $\overline{s}$ contains no items which are partnered in t then $TS(\overline{s}t) \equiv TS(\overline{s})TS(t)$.*

4. *If $\overline{s}$ is bachelor in $\overline{s}t$ or is well-balanced, then $TS(\overline{s}t) \equiv TS(\overline{s})TS(t)$.*

Proof: *1: by induction on* $\mathtt{weight}(\overline{s})$. *Case* $\mathtt{weight}(\overline{s}) = 0$ *then obvious. Case* $\overline{s} \equiv (t\delta)\overline{s'}(\rho\lambda_v)\overline{s''}$ *then*

$$\begin{array}{lcll} T(\overline{s_1}, (t\delta)\overline{s'}(\rho\lambda_v)\overline{s''}\,\overline{s_2}) & \equiv & T(\overline{s_1}(TS(t)\delta), \overline{s'}(\rho\lambda_v)\overline{s''}\,\overline{s_2}) & \equiv^{IH} \\ TS(\overline{s'})T(\overline{s_1}(TS(t)\delta), (\rho\lambda_v)\overline{s''}\,\overline{s_2}) & \equiv & TS(\overline{s'})(TS(t)\delta)(\rho\lambda_v)T(\overline{s_1}, \overline{s''}\,\overline{s_2}) & \equiv^{IH} \\ TS(\overline{s'})(TS(t)\delta)(\rho\lambda_v)TS(\overline{s''})T(\overline{s_1}, \overline{s_2}) & \equiv & TS(\overline{s'})T((TS(t)\delta), (\rho\lambda_v)\overline{s''})T(\overline{s_1}, \overline{s_2}) & \equiv^{IH} \\ T((TS(t)\delta), \overline{s'}(\rho\lambda_v)\overline{s''})T(\overline{s_1}, \overline{s_2}) & \equiv & TS((t\delta)\overline{s'}(\rho\lambda_v)\overline{s''})T(\overline{s_1}, \overline{s_2}) & \end{array}$$

2: using 1. 3: let $t \equiv \overline{s_0}\cdots\overline{s_n}v$ *and* $\overline{s} \equiv \overline{s'_0}\cdots\overline{s'_m}$ *be partitionings. Use cases on* $\overline{s_0}$ *being empty or not and on* $\overline{s'_m}$ *being bachelor or well-balanced. 4: This is a corollary of 3 above.* □

The following lemma shows that $TS(t)$ changes all $\delta\lambda$-couples of t to $\delta\lambda$-segments.

Lemma 4.9 *For every subterm t' of a term t, the following holds:*

1. *$TS(t')$ is well-defined.*

2. *If $\overline{s} \equiv (t''\delta)\overline{s'}(\rho\lambda_v)$ is a subsegment of t' where $\overline{s'}$ is well-balanced, then $TS(\overline{s}) \equiv TS(\overline{s'})(TS(t'')\delta)(\rho\lambda_v)$.*

3. *If $\overline{s} \equiv (A_1\omega_1)\cdots(A_n\omega_n)$ is bachelor in t', then $TS(\overline{s}) \equiv (TS(A_1)\omega_1)\cdots(TS(A_n)\omega_n)$ is bachelor in $TS(t')$.*

4. *If $\overline{s}$ is a subsegment of t' which is well-balanced, then $TS(\overline{s})$ is well-balanced.*

Proof: *By induction on t.*

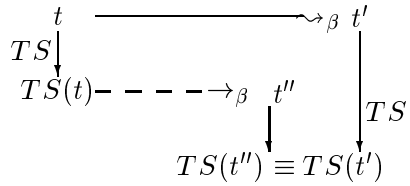
- *Case $t \equiv v$ then t is the unique subterm of t and all $1\cdots4$ hold.*
- *Assume $t \equiv (A\omega)t_2$ where IH holds for A if $A \equiv t_1$ and for t_2. Let t' be a subterm of t. If t' is a subterm of t_1 (for $A \equiv t_1$) or t_2 then use IH. If $t' \equiv t$ then:*
  - *Case $(A\omega)$ is bachelor then $TS(t) \equiv^{Lemma\ 4.8\,(4)} (TS(A)\omega)TS(t_2)$. Here all $1\cdots4$ hold by IH on A and t_2.*
  - *Case $A \equiv t_1 \wedge (t_1\delta)$ matches $(\rho\lambda_v)$ in t. I.e. $t \equiv (t_1\delta)\overline{s}(\rho\lambda_v)t_3$ then $TS(t) \equiv^{Lemma\ 4.8(1,3)} TS(\overline{s})(TS(t_1)\delta)(\rho\lambda_v)TS(t_3)$. Now use IH to show $1\cdots4$.* □

Lemma 4.10 *For all variables v and terms t, t' we have:*
$TS(t) \equiv TS(TS(t))$ and $TS(t[v := t']) \equiv TS(TS(t)[v := TS(t')])$.

Proof: *By induction on t we show that for all subterms t'' of t, $TS(t'') \equiv TS(TS(t''))$ and $TS(t''[v := t']) \equiv TS(TS(t'')[v := TS(t')])$.* □

Note that if $t \to_\beta t'$ and if all the $\delta\lambda$-couples in t are $\delta\lambda$-segments, then it is not necessary that all the $\delta\lambda$-couples of t' are $\delta\lambda$-segments. In other words, we can have $TS(t_1) \to_\beta t_2$ where $t_2 \not\equiv TS(t_2)$. For example, $(x_1\delta)(x_2\delta)(\rho\lambda_{x_3})((\rho'\lambda_{x_4})x_4\delta)(\rho''\lambda_{x_5})x_5 \to_\beta (x_1\delta)(x_2\delta)(\rho\lambda_{x_3})(\rho'\lambda_{x_4})x_4$. Following this remark, we show that in a sense, term reshuffling preserves β-reduction.

Lemma 4.11 *If $t, t' \in \lambda_\to$ and $t \rightsquigarrow_\beta t'$ then $(\exists t'')[(TS(t) \to_\beta t'') \wedge TS(t'') \equiv TS(t')]$. In other words, the following diagram commutes:*



Proof: *By induction on the general* $\rightsquigarrow_\beta$.

- *Case* $t \equiv \overline{s'}(t_1\delta)\overline{s}(\rho\lambda_v)t_3 \rightsquigarrow_\beta t' \equiv \overline{s'}\ \overline{s}(t_3[v := t_1])$, *we use induction on the number* n *of bachelor* δ*-items of* $\overline{s'}$ *that are partnered in* t_3. *Recall that* $\overline{s}$ *is well-balanced.*

  - *Case* $n = 0$ *then*

    $TS(\overline{s'}(t_1\delta)\overline{s}(\rho\lambda_v)t_3) \equiv^{Lemma\ 4.8\ (3,4)}$
    $TS(\overline{s'})TS((t_1\delta)\overline{s}(\rho\lambda_v))TS(t_3) \equiv^{Lemma\ 4.8\ (2,4)}$
    $TS(\overline{s'})TS(\overline{s})(TS(t_1)\delta)(\rho\lambda_v)TS(t_3) \rightarrow_\beta$
    $TS(\overline{s'})TS(\overline{s})(TS(t_3)[v := TS(t_1)]) \equiv t''.$
    $TS(t'') \equiv^{Lemmas\ 4.9,\ 4.10,\ 4.8\ (3,4)}$
    $TS(\overline{s'})TS(\overline{s})TS(TS(t_3)[v := TS(t_1)]) \equiv^{Lemma\ 4.10}$
    $TS(\overline{s'})TS(\overline{s})TS(t_3[v := t_1]) \equiv$
    $TS(\overline{s'}\ \overline{s}(t_3[v := t_1]))$

  - *Assume the property holds for* n *and let us show it for the case where* $\overline{s'}$ *contains* $n+1$ δ*-items which match* λ*-items of* t_3. *Let* $(t''\delta)$ *be the leftmost such* δ*-item of* $\overline{s'}$. *Take* $\overline{s'} \equiv \overline{s'_1}(t''\delta)\overline{s''_1}$ *and* $t_3 \equiv \overline{s'_2}(\rho'\lambda_{v'})t_2$ *where* $(t''\delta)$ *matches* $(\rho'\lambda_{v'})$. *By Lemma 4.4,* $(t''\delta)\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)\overline{s'_2}(\rho'\lambda_{v'})$ *is well-balanced. Moreover, no item of* $\overline{s'_1}$ *has a partner in* $(t''\delta)\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)t_3$.
    As $\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2 \rightsquigarrow_\beta \overline{s''_1}\overline{s}(\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2[v := t_1])$, *we find by IH,* t''' *such that*

    $$TS(\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2) \rightarrow_\beta t''' \wedge$$
    $$TS(t''') \equiv TS(\overline{s''_1}\overline{s}(\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2[v := t_1]))$$

    Now, $TS(\overline{s'_1})t'''$ *is the wanted term because:*
    $TS(t) \equiv^{Lemma\ 4.8\ (4)} TS(\overline{s'_1})TS((t''\delta)\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)\overline{s'_2}(\rho'\lambda_{v'})t_2) \equiv^{Lemma\ 4.8\ (2)}$
    $TS(\overline{s'_1})TS(\overline{s''_1}(t_1\delta)\overline{s}(\rho\lambda_v)\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2) \rightarrow_\beta TS(\overline{s'_1})t'''$
    and $TS(TS(\overline{s'_1})t''') \equiv^{Lemma\ 4.10}$
    $TS(\overline{s'_1})TS(\overline{s''_1}\overline{s}(\overline{s'_2}(t''\delta)(\rho'\lambda_{v'})t_2[v := t_1])) \equiv^{Lemma\ 4.8\ (2),\ BC}$
    $TS(\overline{s'_1})TS((t''\delta)\overline{s''_1}\overline{s}(\overline{s'_2}[v := t_1])(\rho'\lambda_{v'})(t_2[v := t_1])) \equiv^{Lemma\ 4.8\ (4),\ \overline{s''_1}\overline{s}\ \overline{s'_2}\ well-balanced}$
    $TS(\overline{s'_1}(t''\delta)\overline{s''_1}\overline{s}(\overline{s'_2}(\rho'\lambda_{v'})t_2[v := t_1])) \equiv TS(t').$

- *The proof of compatibility is technical. The difficult case is:* $t \equiv (t_1\delta)t_2$ *and* $t_2 \rightsquigarrow_\beta t'_2$. *Distinguish the cases:* $(t_1\delta)$ *is bachelor or non-bachelor in* t. □

Corollary 4.12 *If* $t \twoheadrightsquigarrow_\beta t'$ *then there exist* $t_0, t_1, \cdots t_n$ *such that*
$[(t \equiv t_0) \wedge (TS(t_0) \rightarrow_\beta t_1) \wedge (TS(t_1) \rightarrow_\beta t_2) \wedge \cdots \wedge (TS(t_{n-1}) \rightarrow_\beta t_n) \wedge (TS(t_n) \equiv TS(t'))]$

Proof: *By induction on* $\twoheadrightarrow_\beta$.

- *Case* $t \rightsquigarrow_\beta t'$ *use Lemma 4.11.*

- *Case* $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t$ *then obvious* $(n = 1 \wedge t_0 \equiv t \wedge t_1 \equiv TS(t))$.

- *Case* $t' \rightsquigarrow\!\!\!\!\rightarrow_\beta t'' \wedge t'' \rightsquigarrow\!\!\!\!\rightarrow_\beta t'''$, *then by IH, there exist* $t_0, t_1, \cdots, t_n, t'_0, t'_1, \cdots t'_m$ *such that* $(t' \equiv t_0) \wedge (TS(t_0) \rightarrow_\beta t_1) \wedge (TS(t_1) \rightarrow_\beta t_2) \wedge \cdots \wedge (TS(t_{n-1}) \rightarrow_\beta t_n) \wedge (TS(t_n) \equiv TS(t'')) \wedge (t'' \equiv t'_0) \wedge (TS(t'_0) \rightarrow_\beta t'_1) \wedge (TS(t'_1) \rightarrow_\beta t'_2) \wedge \cdots \wedge (TS(t'_{m-1}) \rightarrow_\beta t'_m) \wedge (TS(t'_m) \equiv TS(t'''))$. *Hence,* $(t' \equiv t_0) \wedge (TS(t_0) \rightarrow_\beta t_1) \wedge \cdots \wedge (TS(t_{n-1}) \rightarrow_\beta t_n) \wedge (TS(t_n) \rightarrow_\beta t'_1) \wedge \cdots \wedge (TS(t'_{m-1}) \rightarrow_\beta t'_m) \wedge (TS(t'_m) \equiv TS(t'''))$.

Note that for the basic and reflexive cases, $n = 1$ *for sure. For the transitive case, this may not be so. For example,* $t \equiv (\rho\lambda_{x_1})((\rho'\lambda_{x_2})(\rho''\lambda_{x_3})x_1\delta)(\rho'''\lambda_{x_4})(x_1\delta)(x_1\delta)x_4 \rightsquigarrow\!\!\!\!\rightarrow_\beta t' \equiv (\rho\lambda_{x_1})(x_1\delta)(\rho'\lambda_{x_2})x_1$ *yet* $t \rightsquigarrow\!\!\!\!\rightarrow_\beta t'$ *does not imply there exists* t'' *such that* $TS(t) \twoheadrightarrow_\beta t'' \wedge TS(t'') \equiv TS(t')$. *There is however* $t_1 \equiv (\rho\lambda_{x_1})(x_1\delta)(x_1\delta)(\rho'\lambda_{x_2})(\rho''\lambda_{x_3})x_1$ *and* $t_2 \equiv t'$ *such that* $TS(t) \rightarrow_\beta t_1 \wedge TS(t_1) \rightarrow_\beta t_2$ *and* $TS(t_2) \equiv TS(t')$. □

Finally, we show that term reshuffling preserves typing:

Lemma 4.13 *If* $\Gamma \vdash t : \rho$ *then* $\Gamma \vdash TS(t) : \rho$.

Proof: *By induction on* t.

- *Case* $t \equiv v$, *then nothing to prove.*

- *Case* $t \equiv (\rho'\lambda_v)t'$ *then*

$$\begin{array}{ll}
\Gamma \vdash (\rho'\lambda_v)t' : \rho & \Rightarrow^{Generation} \\
\Gamma(\rho'\lambda_v) \vdash t' : \rho'' \wedge \rho \equiv \rho' \rightarrow \rho'' & \Rightarrow^{IH} \\
\Gamma(\rho'\lambda_v) \vdash TS(t') : \rho'' \wedge \rho \equiv \rho' \rightarrow \rho'' & \Rightarrow^{\rightarrow-introduction,\ Lemma\ 4.8\ (3)} \\
\Gamma \vdash TS((\rho'\lambda_v)t') : \rho &
\end{array}$$

- *Case* $t \equiv (t'\delta)t''$ *then*

  - *Case* $(t'\delta)$ *is bachelor in* t *then*

$$\begin{array}{ll}
\Gamma \vdash (t'\delta)t'' : \rho & \Rightarrow^{Generation} \\
\Gamma \vdash t' : \rho' \wedge \Gamma \vdash t'' : \rho' \rightarrow \rho & \Rightarrow^{IH} \\
\Gamma \vdash TS(t') : \rho' \wedge \Gamma \vdash TS(t'') : \rho' \rightarrow \rho & \Rightarrow^{\rightarrow-elimination} \\
\Gamma \vdash (TS(t')\delta)TS(t'') : \rho & \Rightarrow^{Lemma\ 4.8\ (3)} \\
\Gamma \vdash TS((t'\delta)t'') : \rho &
\end{array}$$

  - *Case* $(t'\delta)$ *is partnered in* t, *then* $t \equiv (t'\delta)\overline{s}(\rho'\lambda_v)t_1$ *where* $\overline{s}$ *is well-balanced, and no binding variables of* $\overline{s}$ *are free in* t'.

$$\begin{array}{ll}
\Gamma \vdash (t'\delta)\overline{s}(\rho'\lambda_v)t_1 : \rho & \Rightarrow^{Generation} \\
\Gamma \vdash t' : \rho' \wedge \Gamma \vdash \overline{s}(\rho'\lambda_v)t_1 : \rho' \rightarrow \rho & \Rightarrow^{IH} \\
\Gamma \vdash TS(t') : \rho' \wedge \Gamma \vdash TS(\overline{s}(\rho'\lambda_v)t_1) : \rho' \rightarrow \rho & \Rightarrow^{\rightarrow-elimination} \\
\Gamma \vdash (TS(t')\delta)TS(\overline{s}(\rho'\lambda_v)t_1) : \rho & \Rightarrow^{Lemma\ 4.8\ (4)} \\
\Gamma \vdash (TS(t')\delta)TS(\overline{s})(\rho'\lambda_v)TS(t_1) : \rho & \Rightarrow^{Lemma\ 3.11} \\
\Gamma \vdash TS(\overline{s})(TS(t')\delta)(\rho'\lambda_v)TS(t_1) : \rho & \Rightarrow^{Lemma\ 4.8\ (4)} \\
\Gamma \vdash TS(\overline{s}(t'\delta)(\rho'\lambda_v)t_1) : \rho & \Rightarrow^{Lemma\ 4.8\ (2)} \\
\Gamma \vdash TS((t'\delta)\overline{s}(\rho'\lambda_v)t_1) : \rho & \Rightarrow \Gamma \vdash TS(t) : \rho
\end{array}$$

□

5 Conclusion

In this paper, we observed that if we change slightly the classical λ-notation, then we can make more redexes visible. This is useful and is in line with current research on the needed redexes (for normal forms) as in [BKKS 87]. Making more redexes visible will work to our advantage if we could also contract these redexes before other ones. For example, in lazy evaluation ([Launchbury 93]), some redexes get frozen while other ones are being contracted. Now, if we had the ability of choosing which redex to contract out of all visible redexes, rather than waiting for some redex to be evaluated before we can proceed with the rest, then we can say that we have achieved a flexible system where we have control over what to contract rather than letting reductions force themselves in some order. This may lead to some advantages concerning optimal reductions as in [Lévy 80].

With our notation, and our new β-reduction, we achieve this flexibility and freedom of choice. Moreover, we do not lose any of the original properties. We have shown in fact that what we provide is a more general β-reduction where more redexes are visible and where all the original properties (using ordinary classical reduction) still hold for our general reduction. We believe this to be an important breakthrough which may lead to new reduction strategies that may explain various programming principles (such as lazy evaluation) in an elegant way.

We have shown further that, using item notation (which makes more redexes visible), one is able to stick to the old β-reduction and just do a simple reshuffling so that these newly visible redexes can be contracted before other redexes. We have shown that this reshuffling (which is very simple and can only be enabled in our notation), is correct. In fact, reshuffling does really make all redexes subject to immediate contraction and preserves typing. So, if t has type ρ then the reshuffled version of t also has type ρ. It is moreover the case that if $t \twoheadrightarrow_\beta t'$ using our extended reduction, then $TS(t)$ can be transformed into $TS(t')$ using classical reduction and intermediate term reshuffling.

The work carried out in this paper will have many applications. We mentioned the semantics of lazy evaluation and the new reduction strategies which may lead to further optimal results. These points are under investigation. The new notation moreover deserves attention. [KN 93] and [NK 94] have shown many of its advantages for formulating and generalising type theory and for rendering substitution explicit in the λ-calculus. Further advantages are also studied in [KN 9z].

References

[Barendregt 92] Barendregt, H., Lambda calculi with types, *Handbook of Logic in Computer Science*, volume II, ed. Abramsky S., Gabbay D.M., Maibaum T.S.E., Oxford University Press, 1992.

[BKKS 87] Barendregt, H.P., Kennaway, J.R., Klop, J.W., and Sleep M.R., Needed reduction and spine strategies for the λ-calculus, *Information and Computation 75 (3)*, 1191-231, 1987.

[de Bruijn 93] Bruijn, N.G. de, Algorithmic definition of lambda-typed lambda calculus, in Huet, G. and Plotkin, G. eds. *Logical Environments*, 131-146, Cambridge University Press, 1993.

[KN 93] Kamareddine, F., and Nederpelt, R.P., On stepwise explicit substitution, *International Journal of Foundations of Computer Science 4 (3)*, 197-240, 1993.

[KN 9z] Kamareddine, F., and Nederpelt, R.P., *The beauty of the λ-calculus*, in preparation.

[Launchbury 93] Launchbury, J., A natural semantics of lazy evaluation, *ACM POPL 93*, 144-154, 1993.

[Lévy 80] Lévy, J.-J. Optimal reductions in the lambda calculus, in *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, J. Seldin and R. Hindley eds, Academic Press, 1980.

[Nederpelt 73] Nederpelt, R.P., *Strong normalisation in a typed lambda calculus with lambda structured types*, Ph.D. thesis, Eindhoven University of Technology, Department of Mathematics and Computer Science, 1973. To appear in Nederpelt, R.P., Geuvers, J.H. and de Vrijer, R.C., eds., *Selected Papaers on Automath*, North Holland, 1994.

[NK 94] Nederpelt, R.P., and Kamareddine, F., A unified approach to type theory through a refined λ-calculus, Proceedings of the 1992 conference on *Mathematical Foundations of Programming Semantics*, ed. M. Mislove et. al., 1994.