Delayed updating and renaming in beta reduction of lambda terms

draft version

Rob Nederpelt, Ferruccio Guidi

January 17, 2022

Abstract

In this paper we focus on

1 Name-carrying lambda trees

1.1 Lambda-trees

We use a tree representation for lambda terms in (typed) lambda calculus. See Figure 1. The obtained tree is an undirected rooted tree with labels attached to the *edges* of the tree. We prefer labelling the edges instead of the vertices for reasons to be explained below.

Label L represents a 'lambda' and A stands for an 'application'. Moreover, we use label S for 'subordination'; this extra label enables us to discriminate between main branches and subbranches, which is essential when considering different pathes in a tree.

NOTE: When considering typed lambda calculi containing Π-terms: the tree representation of $\Pi x : M \,.\, N$ is similar to that of $\lambda x : M \,.\, N$, with P_x instead of L_x. In some typed calculi, $*$ occurs as a constant.

Definition 1.1. *tree*: connected acyclic undirected graph, consisting of vertices and edges.

labels: one of the symbols L_x (for each variable x), A, S, or a variable name. (L_x and A represent λ-binding of x, and application, respectively. Label S represents the beginning of a subexpression in λ-calculus.)

Note: In some typed calculi: also P_x is a label (for each variable x). (It represents Π-binding of x.) Also $*$ can be a label.

binders: L_x and P_x are binders. These symbols are intended to bind free variables (e.g., x), in the usual manner. We omit the details.

edge-labelled tree: a tree in which the leaf vertices are omitted and in which each edge has a label.

Meta-variables for trees: $\mathbf{t}$, $\mathbf{t}'$, $\mathbf{t}_1$, ...

Metavariables for labels: $\ell, \dots$

Metavariables for binders: B_x, B'_x, $\dots$

Convention 1.2. (*i*) *From now on, all trees and paths will be edge-labelled.*

(*ii*) *Each* path *in an edge-labelled tree will be* identified *with its string of labels.*

Metavariables for paths: p, q, $\dots$.

Definition 1.3. Let p and q be paths. Then $p \preceq q$ if there is a (possibly empty) path p' obeying $q \equiv p\,p'$, and $p \prec q$ if there is a non-empty path p' with that property.

Definition 1.4. A-cell, L-cell, P-cell, var-cell, $*$-cell: see Figure 1.

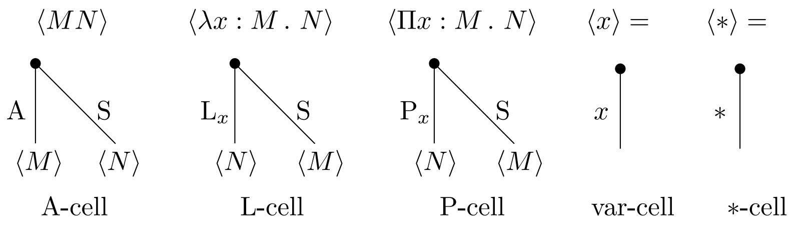

Figure 1: cells of lambda trees

A-*,* L- *and* P-*cells* are binary; they have *open ends* on the lower sides. They can be downwards connected to any other cells at both open ends.

Var-cells and $*$-cells are unary; they can be (upwards) connected to A-, L- and P-cells, but they can not be downwards connected to any other cell: they have *no open ends*.

Note 1.5. *In* untyped *lambda-calculus,* L-*cells and* P-*cells have only one open end: the edge marked* L_x *or* P_x*, because the types* M *are not given in those calculi.*

Definition 1.6. A λ-*tree* is a non-empty, rooted and edge-labelled tree, built by an arbitrary connection of L-cells, A-cells, P-cells, var-cells and $*$-cells, such that there are no open ends. (So var-cells and $*$-cells appear at all leaves, and only there.)

Each λ-tree apparently has a top-cell. The *root* of a λ-tree is the leftmost label of the top-cell of the tree.

Example 1.7. In Figure 2 (1) we represent the λ-tree of the following term from typed lambda calculus:

$\lambda\alpha : * \,.\, \lambda\beta : * \,.\, \lambda x : \alpha \,.\, \lambda y : \alpha \to \beta \,.\, y\,x$.

The term marked (1) is written in the *name-carrying* notation, using the *variable names* x, y, α and β. The notation $\alpha \to \beta$ is treated as an abbreviation of $\Pi u : \alpha \,.\, \beta$. The symbol $*$ is a *constant* of typed λ-calculus.

(The term marked (2) will be discussed later.)

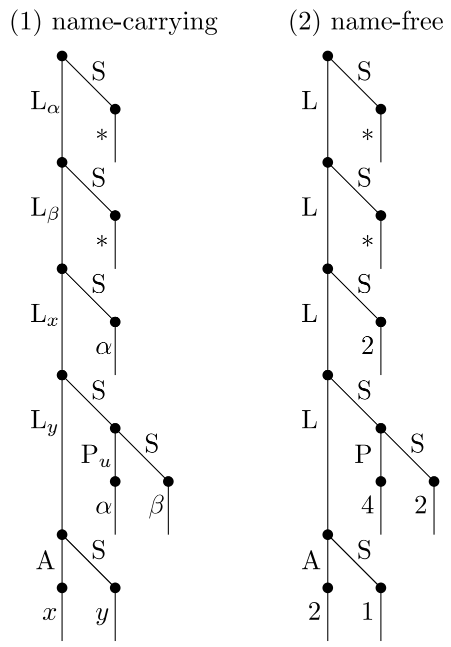

Figure 2: Lambda trees in name-carrying and name-free form

Definition 1.8. Let $\mathbf{t}$ be a λ-tree.

We write $p \in \mathbf{t}$ if path p is a (coherent) part of $\mathbf{t}$.

A *root path* p in $\mathbf{t}$ is a (non-empty) path starting in the root. Notation: $p \in^{\wedge} \mathbf{t}$.

A *leaf path* in $\mathbf{t}$ is a (non-empty) path with a leaf as final label. Notation: $p \in_{\vee} \mathbf{t}$.

A *complete path* in $\mathbf{t}$ is a path that is both a root path and a leaf path. Notation: $p \in^{\wedge}_{\vee} \mathbf{t}$.

Example 1.9. In the λ-tree of Example 2 (1), the path $\mathrm{L}_\alpha\,\mathrm{L}_\beta\,\mathrm{L}_x\,\mathrm{S}\,\mathrm{P}_u\,\alpha$ is a complete path.

For the sake of convenience, we assume that the binding variables in a λ-tree always differ from each other. This can be expressed as follows.

Convention 1.10. *Let* $\mathbf{t}$ *be a* λ*-tree, let* B_x *and* B'_y *be binders and let the paths* $p\,B_x$ *and* $q\,B'_y \in^{\wedge} \mathbf{t}$*. Then* $x \equiv y \;\Rightarrow\; (p \equiv q \wedge B \equiv B')$.

Hence, in particular: If B_x is a binder in $\mathbf{t}$, then there is no other binder with subscript x in $\mathbf{t}$.

Definition 1.11. (i) Let $p_1, \ldots, p_n$ be all complete paths in a λ-tree $\mathbf{t}$. Then we identify $\mathbf{t}$ with the set $\{p_1, \ldots, p_n\}$.

(ii) Let $\mathbf{t} \equiv \{p_1, \ldots, p_m\}$ and $\mathbf{t}' \equiv \{q_1, \ldots, q_n\}$ be λ-trees. Then $\mathbf{t}$ is a *subtree* of $\mathbf{t}'$, or $\mathbf{t} \subseteq \mathbf{t}'$, if for each $p_i \in^{\wedge}_{\vee} \mathbf{t}$ there exists a path $q_j \in^{\wedge}_{\vee} \mathbf{t}'$ such that $p_i \preceq q_j$.

Remark 1.12. *We consider here the typed version of λ-calculus, since that version is widespread in modern applications. Most of the material presented here and below, however, also applies to the* untyped *version of λ-calculus, with some adaptations.*

A λ-tree can be characterized as follows.

Lemma 1.13. *Let $\mathbf{s} \equiv p_1, \ldots, p_n$ be a non-empty set of non-empty strings of labels.*

$\mathbf{s}$ *is a λ-tree iff the following condition holds:*

For all p_i and all (possibly empty) paths $p \prec p_i$, either (i) or (ii) holds:
(i) *$p\,\mathrm{S} \preceq p_i$ and $p\,\ell \prec p_i$ for a* unique *label ℓ, where ℓ is not* S, $*$ *or some variable x (i.e., if also $p\,\ell' \preceq p_i$ for $\ell' \not\equiv \mathrm{S}$, then $\ell' \equiv \ell$).*
(ii) *$p_i \equiv p\,x$ or $p_i \equiv p\,*$.*

1.2 β-reduction on λ-trees

The relation β-reduction between λ-terms in typed lambda calculi is defined as the compatible relation generated by

$(\lambda x : K\,.\, M)P \to_\beta M[x := P]$.

Here $M[x := P]$ is the result of substituting P for all free x's in M. (We do not give the necessary conditions on the names of variables which prevent undesired bindings in the 'process' of β-reduction.)

We give another description of β-reduction, suited to the *tree format* sketched previously and facilitating the alternative β-reductions described below. We first give some definitions, and a lemma on binding.

Definition 1.14. Let $\mathbf{t}$ be a λ-tree and $p \in^{\wedge} \mathbf{t}$.

Consider the set $S \subseteq \mathbf{t}$ of all complete paths $p\,q\,x \in^{\wedge}_{\vee} \mathbf{t}$, so the paths beginning with p, ending in some leaf x and having some path q in between. The set of all these $q\,x$ is itself a tree, called *tree*(p).

We call the set S the *grafted tree* of p in $\mathbf{t}$. We denote this grafted tree by $p\,\mathbf{t}'$, if $\mathbf{t}'$ is *tree*(p).

Definition 1.15. Let $\mathbf{t}$ and $\mathbf{t}'$ be λ-trees.

(1) Let $p\,y \in^{\wedge}_{\vee} \mathbf{t}$. Then $(p\,y)[x := \mathbf{t}']$ is defined as the grafted tree $p\,\mathbf{t}'$ if $x \equiv y$, and as $p\,y$ if $x \not\equiv y$.

(2) We define $\mathbf{t}[x := \mathbf{t}']$ as $\{q[x := \mathbf{t}'] \mid q \in^{\wedge}_{\vee} \mathbf{t}\}$.

(3) Let $p \in^{\wedge} \mathbf{t}$ but $p \not\in^{\wedge}_{\vee} \mathbf{t}$. Then $\mathbf{t}[\mathit{tree}(p) := \mathbf{t}']$ is the tree obtained from $\mathbf{t}$ by replacing the grafted tree $p\,\mathit{tree}(p)$ by the grafted tree $p\,\mathbf{t}'$.

Lemma 1.16. *Let $\mathbf{t}$ be a λ-tree.*

(i) *Assume that $p\,x \in^{\wedge}_{\vee} \mathbf{t}$. If x is a variable in $\mathbf{t}$ that is bound in the original λ-term by a λ, then there is exactly one $\mathrm{L}_x \in \mathbf{t}$ binding this x, and $p \equiv p_1\,\mathrm{L}_x\,p_2$ for some paths p_1 and p_2.*

(*ii*) *Assume that* $p\,\mathrm{L}_x \in^{\wedge} \mathbf{t}$. *If* $p\,\mathrm{L}_x\,q\,x \in^{\wedge}_{\vee} \mathbf{t}$, *then* x *is bound by* L_x. *In a* bound *term, all* x*'s are bound by this* L_x*, so there are no* x*'s 'outside'* $\mathit{tree}(p\,\mathrm{L}_x)$.

Hence, the binder of a variable x in $\mathbf{t}$ can be found on the path leading backwards from x to the root of $\mathbf{t}$. (A similar lemma holds for variables bound by a Π in the original λ-term.)

Moreover, an L_x in $\mathbf{t}$ binds all (free) x's that occur in $\mathbf{t}$.

Definition 1.17. (*i*) Let $\mathbf{t}$ be a λ-tree and let $p\,\mathrm{L}_x\,q\,x \in^{\wedge}_{\vee} \mathbf{t}$, such that L_x binds x. Then the path $\mathrm{L}_x\,q\,x$ is called the L-*block of (this occurrence of)* x.

(*ii*) A λ-tree is called *closed* if all leaf variables x are bound by an L_x.

Obviously, in a closed λ-tree, every leaf variable x corresponds to exactly one L-block ending in that x.

We can describe *β-reduction* of a λ-tree $\mathbf{t}$, with relation symbol $\rightarrow_\beta$, as follows.

Definition 1.18. Let $\mathbf{t}$ be a λ-tree and assume that $p\,\mathrm{A}\,\mathrm{L}_x \in^{\wedge} \mathbf{t}$.

Now $\mathbf{t} \;\rightarrow_\beta\; \mathbf{t}[\mathit{tree}(p) := \mathit{tree}(p\,\mathrm{A}\,\mathrm{L}_x)[x := \mathit{tree}(p\,\mathrm{S})]]$.

Again, we disregard some necessary conditions on the variable names. Note: if $p\,\mathrm{A}\,\mathrm{L}_x \in^{\wedge} \mathbf{t}$, then also $p\,\mathrm{S} \in^{\wedge} \mathbf{t}$.

1.3 Some variants of beta-reduction

1.3.1 Balanced beta-reduction

There is a variant of β-reduction that is interesting for certain purposes. We call it *balanced β-reduction*. In the literature, it originally appeared under the name β_1 (Nederpelt, 1973). For details, see the more recent literature about the Linear Substitution Calculus (cf., Accattoli and Kesner, 2010) and Accattoli and Kesner, 2012, in which it is called *distant beta*, symbol $\rightarrow_{dB}$. See also Barenbaum and Bonelli, 2017.

Firstly, we give the following definition.

Definition 1.19. A path p in a λ-tree $\mathbf{t}$ is called *balanced*, denoted $\mathit{bal}(p)$, if it is constructed by means of the following inductive rules:

(*i*) $\mathit{bal}(\varepsilon)$, i.e., the empty string is balanced;

(*ii*) if $\mathit{bal}(p)$, then $\mathit{bal}(\mathrm{A}\,p\,\mathrm{L}_x)$, for every variable x;

(*iii*) if $\mathit{bal}(p)$ and $\mathit{bal}(q)$, then $\mathit{bal}(p\,q)$.

In case (*ii*), we say that the mentioned A *matches* the mentioned L.

Examples of balanced paths: ε, A L, A A L L, A L A L, A A L A A L L L.

Note the close correspondence between nested paths and (consecutive) nested pairs of parentheses. Only A- and L-labels occur in balanced paths, so there is no other label involved, such as S.

We define *balanced β*-reduction, with symbol $\rightarrow_b$, as follows.

Definition 1.20. Let $\mathbf{t}$ be a λ-tree, let $b \in \mathbf{t}$ be a balanced path and assume that $p\,\mathrm{A}\,b\,\mathrm{L}_x \in^{\wedge} \mathbf{t}$ and there is at least one path $p\,\mathrm{A}\,b\,\mathrm{L}_x\,q\,x \in^{\wedge}_{\vee} \mathbf{t}$.

Then $\mathbf{t} \ \to_b \ \mathbf{t}[\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x) := \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x)[x := \mathit{tree}(p\,S)]]$.

Compare this with Definition 1.18.

Consider two λ-trees $\mathbf{t}$ and $\mathbf{t}'$ such that $\mathbf{t} \to_b \mathbf{t}'$ as described in Definition 1.20, so each x has been replaced by $\mathit{tree}(p\,S)$ in $\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x)$. Now we have that $\mathbf{t}$ is a subtree of $\mathbf{t}'$, provided that we omit all var-labels x in $\mathbf{t}$. So, balanced β-reduction has the property that it *extends* the original underlying tree $\mathbf{t}$ if all instances of variable x are skipped in $\mathbf{t}$. (In the 'tree-like' image of $\mathbf{t}$, we have to skip the edges of the x-labels, as well.)

Definition 1.21. Let $\mathbf{t}$ be a λ-tree, $b \in \mathbf{t}$ a balanced path and assume that $\mathrm{A}\,b\,\mathrm{L}_x \in \mathbf{t}$. Let $\mathrm{A}\,b\,\mathrm{L}_x\,p\,x$ be a path in $\mathbf{t}$.

(*i*) The path $\mathrm{A}\,b\,\mathrm{L}_x\,p\,x \in \mathbf{t}$ (where L_x binds x) is called the A-*block of (this occurrence of) x*.

(*ii*) An A-block $\mathrm{A}\,b\,\mathrm{L}_x\,p\,x \in \mathbf{t}$ is a *front extension* of the L-block $\mathrm{L}_x\,p\,x$. This A-block and L-block are called *corresponding*.

(*iii*) The path $\mathrm{A}\,b\,\mathrm{L}_x$ is called the *redex block* or r-block of x.

Lemma 1.22. (*i*) *The* A-*block of a certain $x \in \mathbf{t}$, if it exists, is unique, just as the* L-*block of x and the* r-*block of x.*

(*ii*) *In a closed term, each var-label x corresponds to exactly one* L-*block; but even when the term is closed, not every* L-*block has a corresponding* A-*block.*

(*iii*) *An* A-*block of a certain $x \in \mathbf{t}$ is a join of exactly one* r-*block and exactly one* L-*block, overlapping at the L_x binding x.*

1.3.2 Focused beta-reduction

Sometimes there is a need for another form of β-reduction. One occasion is when β-reduction is invoked to model *definition unfolding*: then a defined notion occurring in M, say x, is replaced by the definiens, say P. Such an action generally occurs for only one instance of the definiendum x. So instead of replacing *all* occurrences of x in M, one aims at *precisely one* occurrence.

When adapting β-reduction to this situation, there are several things to be considered:

(*i*) The substitution $[x := P]$ should act on exactly *one* free x in M.

(*ii*) Hence, x must occur free in M.

(*iii*) The redex $(\lambda x : K \,.\, M)P$ should remain active after the intended reduction, since there may be other free x's in M which still need a type annotation (i.e., K); moreover, there must remain a possibility to substitute P for one or more of these x's in a later stage of the process.

All this is covered in the following definition of *focused* β-reduction, for which we use the symbol $\to_f$.

Definition 1.23. Let $\mathbf{t}$ be a λ-tree, let $b \in \mathbf{t}$ be a balanced path and assume that $r\,x \in^{\wedge}_{\vee} \mathbf{t}$ is a complete path in $\mathbf{t}$ with $r \equiv p\,\mathrm{A}\,b\,\mathrm{L}_x\,q$. We call the balanced reduction identified by p and focussed on the path $q\,x$, a (p,q)-*reduction*. It is defined by

$\mathbf{t} \ \to_f \ \mathbf{t}[r\,x := r\,\mathit{tree}(p\,\mathrm{S})]$.

The possibility of having *balanced* β-reduction is necessary to be able to deal with redexes which otherwise would be forbidden by the maintenance of the redex $(\lambda x : K\,.\,M)P$ after β-reduction, as described in requirement (iii). See the following example.

Example 1.24. We have, in λ-calculus with normal untyped β-reduction:
$(\lambda x\,.\,((\lambda y\,.\,M)Q))P \to_\beta (\lambda x\,.\,(M[y := Q]))P \to_\beta M[y := Q][x := P]$.
In *focused* β-reduction, this becomes:
$(\lambda x\,.\,((\lambda y\,.\,M)Q))P \to_f (\lambda x\,.\,(\lambda y\,.\,M[y_0 := Q])Q)P \to_f$
$(\lambda x\,.\,((\lambda y\,.\,M[y_0 := Q])Q)[x_0 := P])P$.
Here y_0 and x_0 are selected instances of the free y's and x's in M, respectively.
The second of the two one-step, focused, reductions could not be executed without the possibility to have a balanced λ-term $(\lambda y\,.\,M[y_0 := Q])Q$ between the λx and the P.

1.3.3 Erasing reduction

After having applied balanced or focused β-reduction, one also desires a reduction that gets rid of the 'remains', i.e., the A and the L_x in grafted trees $p\,\mathrm{A}\,b\,\mathrm{L}_x\,\mathbf{t}'$ where $x \notin FV(\mathbf{t}')$. Such pairs $\mathrm{A}\ldots\mathrm{L}_x$ are superfluous, since $\mathbf{L}_x$ has no x that is bound to it. We call the corresponding reduction *erasing* β-reduction and use symbol $\to_e$ for it. (This reduction is also referred to as 'garbage collection' in the literature.)

Definition 1.25. Let $\mathbf{t}$ be a λ-tree, let $b \in \mathbf{t}$ be a balanced path and assume that $p\,\mathrm{A}\,b\,\mathrm{L}_x\,\mathbf{t}' \in_{\vee}^{\wedge} \mathbf{t}$. Assume moreover that $x \notin FV(\mathbf{t}')$.
Then $\mathbf{t} \ \to_e \ \mathbf{t}[\mathit{tree}(p) := \mathit{tree}(p\,\mathrm{A})[\mathit{tree}(p\,\mathrm{A}\,b) := \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x)]]$.

We denote the transitive closure of a reduction $\to_i$ by $\twoheadrightarrow_i$. An arbitrary sequence of reductions $\to_i$ and $\to_j$ is denoted $\twoheadrightarrow_{i,j}$.

Theorem 1.26. *Let* $\mathbf{t}$ *and* $\mathbf{t}'$ *be* λ*-trees.*
(i) $\mathbf{t} \twoheadrightarrow_\beta \mathbf{t}' \Rightarrow \mathbf{t} \twoheadrightarrow_{b,e} \mathbf{t}'$.
(ii) $\mathbf{t} \to_b \mathbf{t}' \Rightarrow \mathbf{t} \twoheadrightarrow_f \mathbf{t}'$
(iii) (Postponement of $\to_e$ after $\to_b$) *If* $\mathbf{t} \twoheadrightarrow_{b,e} \mathbf{t}'$, *then there is* $\mathbf{t}''$ *such that* $\mathbf{t} \twoheadrightarrow_b \mathbf{t}'' \twoheadrightarrow_e \mathbf{t}'$.
(iv) (Postponement of $\to_e$ after $\to_f$) *If* $\mathbf{t} \twoheadrightarrow_{f,e} \mathbf{t}'$, *then there is* $\mathbf{t}''$ *such that* $\mathbf{t} \twoheadrightarrow_f \mathbf{t}'' \twoheadrightarrow_e \mathbf{t}'$.

Proof (iii) Nederpelt, 1973, p. 48, Theorem 6.19.
(iv) Similarly. □

Theorem 1.27. $\to_b$, $\to_f$ *and* $\to_e$ *are confluent.*

Proof For $\to_b$ and $\to_e$, see Nederpelt, 1973, Theorems 6.38 and 6.42. For $\to_f$, see Accattoli and Kesner, 2012.

2 Name-free lambda-terms

2.1 The binding of variables

A recurrent nuisance in the formalization of lambda calculus is the *naming* of variables, which plays a dominant role in the establishment of binding. Let's consider some λ-term in which λx binds variable x. Then one may replace both mentioned x's by a y, on the condition that this is done consistently through the term, and that one prevents that the variable renaming does lead to a name-clash. (This relation is called α-reduction.) For example, the mentioned renaming is forbidden in the term $\lambda x \,.\, \lambda y \,.\, x$, for obvious reasons.

Another cause of worry is that *beta*-reduction has a spreading effect on all kinds of variables, so that it is sometimes a precarious matter to ensure that no 'undesired' binding between a λ and a variable arises.

To prevent these matters, N.G. de Bruijn invented a *name-free* version for terms in λ-calculus (de Bruijn, 1972). Instead of using names such as x to record bindings in terms, he employed *natural numbers*. The idea is to see the λ-term as a tree, comparable to the way we introduced λ-trees in the previous chapter. The principle is, that a variable with number n is bound to the λ which can be found by following the root path ending in this n, and choosing the n-th λ along this path as the binder. In a λ-tree, we call such an end label n a *numerical variable* or a *num-label*.

This approach is known as: 'bound variables references by depth' (de Bruijn, 1978a). Another way of identifying the binder, also discussed in that paper, is to count the number of λ's from the root to the intended one ('reference by level').

In Example 2 (2), the name-carrying λ-tree of Example 2 (1) has been exhibited, as an example of a name-free tree.

This *name-free* representation of λ-calculus, straightforward as it seems, is not so simple as it appears. A pleasant feature of it is that α-reduction is no longer required. But on the other hand, when applying β-reduction, a lot of updating is necessary. This updating is not very easy and it may require extra calculations that complicate matters.

Example 2.1. Consider the term $t_1 \equiv (\lambda x \,.\, \lambda y \,.\, (\lambda z \,.\, y)x)$ in untyped lambda calculus. This term β-reduces to $t_2 \equiv \lambda x \,.\, \lambda y \,.\, y$.

In name-free notation, $t_1 \equiv \lambda\, \lambda\, (\lambda\, 2)\, 2$. (Note that the third λ is not on the root path of the free x; so, the x in t_1 becomes not 3, but 2 in the name-free version.)

The name-free version of t_2 is $\lambda\, \lambda\, 1$: the first number 2 in t_1 must be updated after the β-reduction: it returns as the number 1 in t_2. (The second 2 in t_1, together with the third λ, vanish by the β-reduction.)

Note 2.2. *There is another version of name-free trees for λ-terms, in which the labels are situated not at the edges – as in our proposal – but at the nodes (see de Bruijn, 1978a). Moreover, the labels* S *that we use, are omitted. This works just as well, but for two details. We show this with the same Example 2 (2).*

Imagine the same tree, but with all labels 'raised' to the closest node. So, for example, the root vertex is now labelled L *and the* S*'s have vanished.*

(1) When we consider the tree as not being embedded in the plane, then it is unclear in this case which is the main term and which is the subterm for a given branching. But this can be easily solved by defining trees as planar.

(2) A more serious matter is that an extra provision is necessary for determining the binder of a variable. For example, in the tree of Example 2 (2) with all labels raised, the root path of the variable numbered 4 becomes one L *longer, viz.* L L L L P*. Now observe that the fourth* L *is never meant to bind a variable on this path. So one has to neglect that fourth* L *when counting backwards from 4 to 0 in the process of finding its binder. (Simply changing variable number 4 into 5 is not the proper way to solve this problem; for example, this makes Lemma 1.16, (2), untrue, thus undermining the intended binding structure.)*

Definition 2.3. We use the symbol $\mathcal{T}^{car}$ for the set of *name-carrying*, closed λ-trees. The symbol $\mathcal{T}^{fre}$ means the set of *name-free*, closed λ-trees.

The following sections discuss β-reduction in the name-free case. Since reduction itself is independent of typing, we do not distinguish between typed or untyped version of λ-calculus. So when discussing the sets $\mathcal{T}^{car}$ and $\mathcal{T}^{fre}$ we do not bother whether the terms are typed or not.

2.2 Name-free reductions with instant updating

In the present section we consider name-free β-reduction and some of its variants and each time we describe the updating that is required. We consider β-reduction and its variants, all with *instant updating*. That is, each one-step reduction ends in a λ-tree in which the num-labels have immediately been updated. In Section 3, we try to simplify the name-free β-reduction of λ-terms, by postponing the updates (*delayed updating*).

Note 2.4. *Below, we use the same symbols for the various name-free reductions, as in the* named *case of Section 1.3 (viz., $\to_\beta$, $\to_b$ and $\to_f$). There will be no confusion, since we use letters like* $\mathbf{t}$ *for paths in the name-carrying cases, and* $\mathbf{u}$ *in the name-free cases.*

2.2.1 Beta-reduction

We give a description of β-reduction with instant updating in Definition 2.6.

Notation 2.5. (*i*) *We use the notation $[x := A; y := B]$ for simultaneous substitution.*

(*ii*) *The* lenght $|p|$ *of a path p is the number of labels (including num-labels) in p.*

(*iii*) *The* L-length $\|p\|$ *of a path p is the number of binders (i.e.,* P *or* L*) occurring in p.*

Definition 2.6. Let $\mathbf{u} \in \mathcal{T}^{fre}$ and assume that $p\,\mathrm{A\,L} \in^{\wedge} \mathbf{u}$. Then the β-reduction based on the redex identified by p, also called *p-reduction*, is

$$\mathbf{u} \to_\beta \mathbf{u}\,[\mathit{tree}(p) := \{q\,n \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{A\,L})\,[\mathit{upd}_1\ ;\ \mathit{upd}_2]\}], \text{ where}$$

$$\begin{cases} \mathit{upd}_1 \equiv n := n-1 \ \mathit{if}\ n > \|q\|{+}1, \\ \mathit{upd}_2 \equiv n := \{r\,l \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{S})\,[\mathit{upd}_3]\}\ \mathit{if}\ n = \|q\|{+}1 \end{cases}$$

$$\text{and } \mathit{upd}_3 \equiv l := l + \|q\|\ \mathit{if}\ l > \|r\|$$

We now explain the contents of this definition.

Just as in Definition 1.18, we consider two subtrees of the original λ-tree $\mathbf{u}$: $\mathit{tree}(p\,\mathrm{A\,L})$ and $\mathit{tree}(p\,\mathrm{S})$. Both trees need updating because of the performed β-reduction. In the first tree, we have two simultaneous updatings, upd_1 and upd_2. They act on *all* complete paths $q\,n$ in $\mathit{tree}(p\,\mathrm{A\,L})$, and the choice depends on the value of the leaf n in the path considered. We discern two cases: $n > \|q\|{+}1$ and $n = \|q\|{+}1$. (It is understood that in the case not mentioned, viz. $n < \|q\|{+}1$, the *identical* update is meant, so $n := n$.) See Figure 3 for a visual explanation.

Note that, in the case $n = \|q\|{+}1$, the n is replaced by a copy of the full $\mathit{tree}(p\,S)$, updated by upd_3 if $l > \|r\|$. (Also here, the intention is that an identical update $l := l$ is applied on $r\,l \in^{\wedge}_{\vee} \mathit{tree}(p\,S)$, in the missing case $l \leq \|r\|$.) If $n > \|q\| + 1$, one has to compensate (n becoming $n-1$) for the missing L.

In all three cases, not only the relevant A-L-pair, but also the grafted tree S $\mathit{tree}(p\,\mathrm{S})$ has vanished.

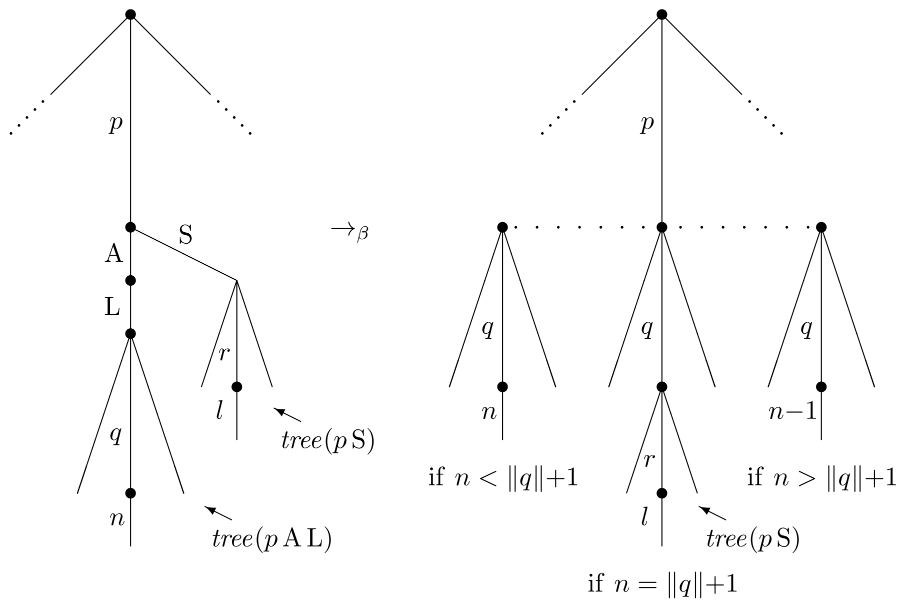

Figure 3: A picture of name-free β-reduction with updating

2.2.2 Balanced beta-reduction

For balanced β-reduction, the updates are somewhat different, due to the non-vanishing of the A-L-couple involved, and the remaining of the original 'argument' $\mathit{tree}(p\,S)$.

Definition 2.7. Let $\mathbf{u} \in \mathcal{T}^{\mathit{fre}}$, let $b \in \mathbf{u}$ be a balanced path and assume that $p\,\mathrm{A}\,b\,\mathrm{L} \in^{\wedge} \mathbf{u}$. Then the balanced reduction based on the redex identified by p (again called p-reduction) is

$$\begin{aligned} \mathbf{u} \to_b \mathbf{u}\,[\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}) &:= \{q\,n \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})\,[\mathit{upd}_1]\}], \text{ where} \\ \mathit{upd}_1 &\equiv n := \{r\,l \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{S})[\mathit{upd}_2]\} \textit{ if } n = \|q\|{+}1 \\ &\text{and } \mathit{upd}_2 \equiv l := l + \|q\|{+}1 + \|b\| \textit{ if } l > \|r\| \end{aligned}$$

As in Section 2.2.1, an identical substitution (i.e., nothing changes) applies in the missing cases for n and l.

2.2.3 Focused beta-reduction

In the case of focused β-reduction, a simple adaptation of Section 2.2.2 is required. This leads to the following.

Definition 2.8. Let $\mathbf{u} \in \mathcal{T}^{\mathit{fre}}$, let $b \in \mathbf{u}$ be a balanced path, assume that $p\,\mathrm{A}\,b\,\mathrm{L} \in^{\wedge} \mathbf{u}$ and that $q\,n$ is a fixed, complete path in $\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})$.

Consider a β-reduction identified by p. When focusing on q as the *focus path*, we call this a *(p, q)-reduction* defined by

$$\begin{aligned} \mathbf{u} \to_f \mathbf{u}\,[\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}) &:= \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})[\mathit{upd}_1]], \text{ where} \\ \mathit{upd}_1 &\equiv q\,n := q\,\{r\,l \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{S})[\mathit{upd}_2]\} \textit{ if } n = \|q\|{+}1 \\ &\text{and } \mathit{upd}_2 \equiv l := l + \|q\|{+}1 + \|b\| \textit{ if } l > \|r\| \end{aligned}$$

2.2.4 Erasing reduction

For erasing reduction, the following definition applies.

Definition 2.9. Let $\mathbf{u} \in \mathcal{T}^{\mathit{fre}}$, let $b \in \mathbf{u}$ be a balanced path, assume that $p\,\mathrm{A}\,b\,\mathrm{L} \in^{\wedge} \mathbf{u}$ and that no numerical variable in $\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})$ is bound by the mentioned L. (Otherwise said: for no $q\,n \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})$ we have that $n = \|q\|{+}1$.)

Then $\mathbf{u} \to_e \mathbf{u}[\mathit{tree}(p) := \mathit{tree}(p\,\mathrm{A})[\mathit{tree}(p\,\mathrm{A}\,b) := \{q\,n \in^{\wedge}_{\vee} \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})\}[\mathit{upd}]]$, where

$$\mathit{upd} \equiv n := n - 1 \textit{ if } n > \|q\|$$

3 Delayed updating

Another possibility is to make reduction easy, by not considering the updates of the numerical variables, until required. In this case we *delay all updates*. In the text below, we restrict this case to the *focused* balanced β-reduction described above (Section 2.2.3), but it can easily be extended to the more general balanced version of β-reduction discussed in Section 2.2.2.

There have been many proposals for describing the necessary updating in a formal way. The fist one has been described in de Bruijn, 1978a: an update function θ is added to a term constructor φ in order to update the num-labels. See also, e.g., Ventura *et al.*, 2015.

3.1 Focused beta-reduction with delayed updating

We use the symbol '$\rightarrow_{df}$' for the delayed, focused β-reduction.

Definition 3.1. Let $\mathbf{u} \in \mathcal{T}^{fre}$, let $b \in \mathbf{u}$ be a balanced path, assume that $p\,\mathrm{A}\,b\,\mathrm{L} \in^{\wedge} \mathbf{u}$ and that $q\,n$ is a fixed, complete path in $\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})$, where n is bound by L.

Then $\mathbf{u} \rightarrow_{df} \mathbf{u}\,[\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}) := \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L})[q\,n := q\,n\,\mathit{tree}(p\,\mathrm{S})]]$.

Note that the edge labelled n stays where it is, and $\mathit{tree}(p\,\mathrm{S})$ is simply attached to it in $\rightarrow_{df}$-reduction, whereas this edge is *replaced* by an updated $\mathit{tree}(p\,\mathrm{S})$ in the original focused reduction. The remaining presence of the label n is to enable updating at a later stage.

For a pictorial representation, see Figure 4. Note: if $\mathit{tree}(p\,\mathrm{S})$ consists of a single edge only, labelled with a num-variable, then this edge is just attached to the open end of the edge labelled n, see Section 3.4 for an example.

The above definition implies that we have to revise our definition of λ-trees, since *var-cells now have open ends*: they may have connections to other cells at their bottom end. So num-variables no longer need to be end-labels in a path. Num-variables not being end-labels, we call *inner num-labels*. To distinguish them from the (still present) num-labels that *are* end-labels, we call the latter *outer* num-labels (or leafs).

Consequently, other definitions should be extended. For example, the definition of a *balanced path* (Definition 1.19) must be adapted such that it allows inner num-labels *inside* the string of A's and L's.

In the remainder of this paper, we assume that these definition revisions have been made.

Definition 3.2. The symbol $\mathcal{T}^{fre}$ concerns the set of name-free, closed trees *without* inner variables. The set of these trees where also inner variables are permitted, we denote by $\mathcal{T}^{fre+}$.

We define what *inclusion* of λ-trees means.

Definition 3.3. Let $\mathbf{u}$ and $\mathbf{u}'$ be λ-trees. Then $\mathbf{u} \subseteq \mathbf{u}'$ iff $p \in^{\wedge}_{\vee} \mathbf{u} \Rightarrow p \in^{\wedge} \mathbf{u}'$.

In Definition 3.1, right hand side, the copy of $\mathit{tree}(p\,\mathrm{S})$ attached to the edge labelled n, is not updated. Moreover, the complete tree $\mathbf{u}$ (including the edge labelled n) remains intact as integral part of $\mathbf{u}'$. Hence, we have the following theorem.

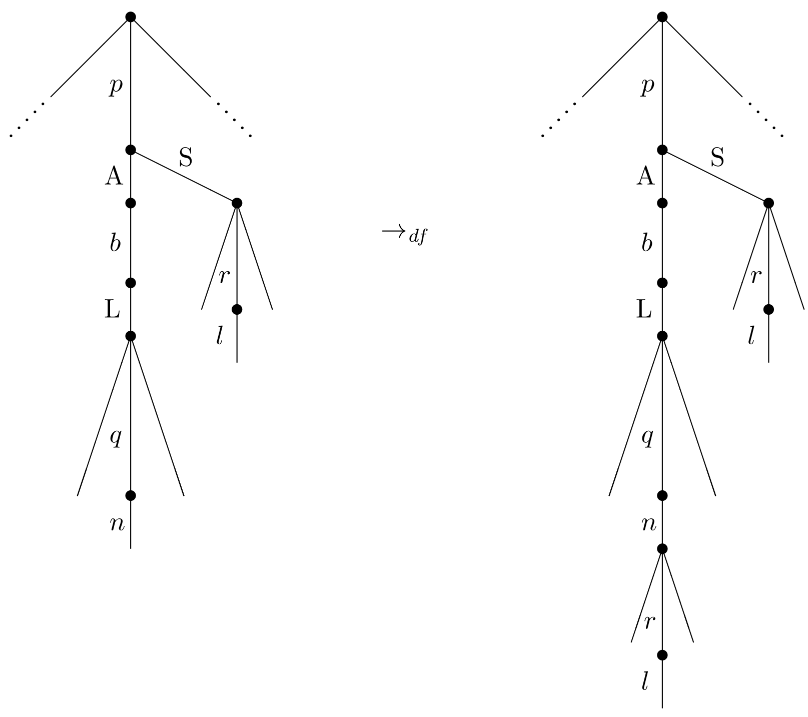

Figure 4: A picture of name-free β-reduction with delayed updating

Theorem 3.4. *Let* $\mathbf{u}, \mathbf{u}' \in \mathcal{T}^{fre+}$. *Then* $\mathbf{u} \to_{df} \mathbf{u}'$ *implies* $\mathbf{u} \subset \mathbf{u}'$.

The latter fact is clearly an advantage. But it comes with a price: when we wish to establish the relation between a leaf-variable of the copied *tree*$(p\,\mathrm{S})$ and its binder, we have to do more work. We discuss this in the following subsection.

3.2 Tracing the binder in reduction with delayed updating

Let $\mathbf{u} \in \mathcal{T}^{fre}$ and $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u}'$, so $\mathbf{u}'$ is the result of a series of *df*-reductions. These reductions may introduce inner variables, so it is not immediately clear what the binders are for (inner or outer) variables. In this section we investigate how one can determine the binder of a variable in $\mathbf{u}'$.

Let $p\,n \in^{\wedge} \mathbf{u}'$, so $p\,n$ is a root path. Here n can be an inner or an outer num-label. We describe a pushdown automaton $\mathcal{P}_{fre}$ to find:

(i) the L-**binder** of n, i.e., the label $\mathrm{L} \in p$ that binds n (this label always exists, since $\mathcal{T}^{fre}$ only contains closed terms),

(ii) or the A-**binder** of n, i.e., the label $\mathrm{A} \in p$ that matches the L-binder of n, *if such an* A *exists* (this needs not to be the case).

In case (i), the initial pair is $(n, 0)$; in case (ii) it is $(n, 1)$.

The **use of the algorithm** is as follows. Let $\mathbf{u} \in \mathcal{T}^{fre+}$ and $p\,n \in^{\wedge} \mathbf{u}$. Assume that we desire to apply algorithm $\mathcal{P}_{fre}$ to find the L-binder or the A-binder of n. Then transform $p\,n$ into $p\,(n,i)\,n$ i being 0 in the first case and 1 in the second, and *start* the algorithm.

In algorithm $\mathcal{P}_{fre}$, we employ *states* that are pairs of natural numbers: (k,l). We start with the insertion of such a pair in the tail of the string $p\,n$, *between* p and n. The automaton moves the pair *to the left* through p, one step at a time, successively passing the labels in p and meanwhile adapting the numbers in the pair. The automaton stops when the desired label (either the binding L or the A matching that L) has been found.

The automaton has an outside *stack* that will contain certain states that are *pushed* on the top of the stack; a state on top of the stack can also be *popped back*, i.e., inserted into the path p, again.

The *transitions* are described in Definition 3.5. A possible one-step transition is denoted by the symbol $\to$ (the reflexive, transitive closure of this relation is denoted $\twoheadrightarrow$). The procedure may be complicated by several recursive calls (see Example 3.9).

The *action* of $\mathcal{P}_{fre}$ is described in Definition 3.6 and in Note 3.7.

Definition 3.5. The algorithm $\mathcal{P}_{fre}$ is specified by the following rules:

1. *first step: stack* $= \emptyset$
2. p L (m,k) q $\to$ p $(m{-}1,k)$ L q, if $m > 0$
3. p A (m,k) q $\to$ p (m,k) A q, if $m > 0$
4. p S (m,k) q $\to$ p (m,k) S q, if $m > 0$
5. p j (m,k) q $\to$ p $(j,1)$ j q, if $m > 0$; *push* (m,k)
6. p L $(0,l)$ q $\to$ p $(0,l{+}1)$ L q, if $l > 0$
7. p A $(0,l)$ q $\to$ p $(0,l{-}1)$ A q, if $l > 0$
8. p j $(0,l)$ q $\to$ p $(0,l)$ j q, if $l > 0$

9*a*. p $(0,0)$ q $\to$ p *pop* q, if *stack* $\neq \emptyset$

9*b*. p $(0,0)$ q $\to$ *stop*, if *stack* $= \emptyset$.

Definition 3.6. Let $\mathbf{u} \in \mathcal{T}^{fre+}$ and $p\,n \in^{\wedge} \mathbf{u}$.

(i) If $\mathcal{P}_{fre}$ is applied to $p\,(n,0)\,n$ and stops in $p'\,(0,0)\,q\,n$, then $q\,n$ is called the L-*block* of n.

(ii) If $\mathcal{P}_{fre}$ is applied to $p\,(n,1)\,n$ and stops in $p'\,(0,0)\,q\,n$, then $q\,n$ is called the A-*block* of n.

After the start step, the procedure $\mathcal{P}_{fre}$ has two, possibly recursive, *rounds* with different actions. We explain this below.

Note 3.7. (i)*: Round (1) (steps (2) to (4)), gives a count-down of the* L*-labels in the path leftwards from an (inner or outer) num-label n; in this round,* A*-labels and* S*-labels have no effect. When the count-down results in $m = 0$, then we just passed the* L *binding the original n. So we found the* L*-block of n. If the stack is empty, so $\mathcal{P}_{fre}$ is not in a recursion, then the* L*-block of the original variable has been found (step (9b)).*

(ii)*: When an inner variable j is met in Round (1) (step (5)), then $\mathcal{P}_{fre}$ starts a recursive round, in which the* A*-block of that j is determined. The count-down of Round (1) resumes immediately left of that* A*-block (step (9a)).*

(iii) *The* A*-block of an inner variable j is found by executing Round (1), giving the* L*-block of j, immediately followed by Round (2) (steps (6) to (8)) determining the* r*-block connected with j. The latter is done by counting upwards for* L*'s and downwards for* A*'s. The* L*-block and the* r*-block combine in the desired* A*-block of j.*

(iv) *Inner variables met in the search of an* r*-block, are skipped (step (8)). So no recursion is started inside an* r*-block.*

(v) *When determining an* r*-block, no label S may be encountered, as can be seen in steps (6) to (8), where S is missing. (This corresponds with Definition 1.19.)*

Lemma 3.8. (i)*: If $q\,n$ is the* L*-block of n, then $q\,n \equiv \mathrm{L}\,q'\,n$ and the* L *is the* L*-binder of n.*

(ii)*: If $q\,n$ is the* A*-block of n, then $q\,n \equiv \mathrm{A}\,q'\,n$ and the* A *is the* A*-binder of n.*

3.3 An example of the action of algorithm $\mathcal{P}_{fre}$

We give an example of the execution of algorithm $\mathcal{P}_{fre}$.

Example 3.9. Firstly, we look for the L-binder of the final num-label – i.e., 3 – in the path A A L L A A L L L A A L A L L L A 2 S L 4 L L S 3.

So we take $i = 0$, and start $\mathcal{P}_{fre}$:

A A L L A A L L L A A L A L L L A 2 S L 4 L L S $(\mathbf{3},\mathbf{0})$ 3 ($stack = \emptyset$) $\rightarrow$
A A L L A A L L L A A L A L L L A 2 S L 4 L L $(\mathbf{3},\mathbf{0})$ S 3 $\rightarrow$
A A L L A A L L L A A L A L L L A 2 S L 4 L $(\mathbf{2},\mathbf{0})$ L S 3 $\rightarrow$
A A L L A A L L L A A L A L L L A 2 S L 4 $(\mathbf{1},\mathbf{0})$ L L S 3 (*push* $(\mathbf{1},\mathbf{0})$) $\rightarrow$
A A L L A A L L L A A L A L L L A 2 S L $(\mathbf{4},\mathbf{1})$ 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A L L L A 2 S $(\mathbf{3},\mathbf{1})$ L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A L L L A 2 $(\mathbf{3},\mathbf{1})$ S L 4 L L S 3 (*push* $(\mathbf{3},\mathbf{1})$) $\rightarrow$
A A L L A A L L L A A L A L L L A $(\mathbf{2},\mathbf{1})$ 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A L L L $(\mathbf{2},\mathbf{1})$ A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A L L $(\mathbf{1},\mathbf{1})$ L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A L $(\mathbf{0},\mathbf{1})$ $\underbrace{\mathrm{L\,L\,A\,2}}_{\text{L-block of 2}}$ S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L A $(\mathbf{0},\mathbf{2})$ L L L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A L $(\mathbf{0},\mathbf{1})$ A L L L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A A $(\mathbf{0},\mathbf{2})$ L A L L L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L A $(\mathbf{0},\mathbf{1})$ A L A L L L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L L $(\mathbf{0},\mathbf{0})$ $\underbrace{\mathrm{A\,A\,L\,A\,L\,L\,L\,A\,2}}_{\text{A-block of 2}}$ S L 4 L L S 3 (*pop* $(\mathbf{3},\mathbf{1})$) $\rightarrow$
A A L L A A L L L $(\mathbf{3},\mathbf{1})$ A A L A L L L A 2 S L 4 L L S 3 $\rightarrow$
A A L L A A L L $(\mathbf{2},\mathbf{1})$ L A A L A L L L A 2 S L 4 L L S 3 $\rightarrow$

$$\mathrm{A\,A\,L\,L\,A\,A\,L}\,(\mathbf{1},\mathbf{1})\,\mathrm{L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4\,L\,L\,S\,3} \rightarrow$$
$$\mathrm{A\,A\,L\,L\,A\,A}\,(\mathbf{0},\mathbf{1})\,\underset{\text{L-block of 4}}{\underline{\mathrm{L\,L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4}}}\,\mathrm{L\,L\,S\,3} \rightarrow$$
$$\mathrm{A\,A\,L\,L\,A}\,(\mathbf{0},\mathbf{0})\,\underset{\text{A-block of 4}}{\underline{\mathrm{A\,L\,L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4}}}\,\mathrm{L\,L\,S\,3}\ \ (\mathit{pop}\ (\mathbf{1},\mathbf{0})) \rightarrow$$
$$\mathrm{A\,A\,L\,L\,A}\,(\mathbf{1},\mathbf{0})\,\mathrm{A\,L\,L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4\,L\,L\,S\,3} \rightarrow$$
$$\mathrm{A\,A\,L\,L}\,(\mathbf{1},\mathbf{0})\,\mathrm{A\,A\,L\,L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4\,L\,L\,S\,3} \rightarrow$$
$$\mathrm{A\,A\,L}\,(\mathbf{0},\mathbf{0})\,\underset{\text{L-block of 3}}{\underline{\mathrm{L\,A\,A\,L\,L\,L\,A\,A\,L\,A\,L\,L\,L\,A\,2\,S\,L\,4\,L\,L\,S\,3}}}\ \ (\mathit{stack} = \emptyset, \text{ hence } \mathit{stop})$$

Example 3.10. (*i*) In the case of Example 3.9, we can derive the A-block of the final num-label 3 by starting with $i = 1$. The derivation then needs three more steps.

(*ii*) Notice that several L-blocks and A-blocks arise during the execution of the procedure, and that these can, on their own, be derived by means of a sub-calculation of the one above.

Example 3.9 shows that during the execution of the algorithm $\mathcal{P}_{fre}$, several L-blocks and A-blocks are 'generated'. Moreover, the procedure stops when the application of $\mathcal{P}_{fre}$ to a path $p\,n \in^{\wedge} \mathbf{u}$ results in the conclusion that a certain sub-path $\mathrm{L}\,q\,n$ (ending in the mentioned n) is an L-block. Then we have found that the displayed L binds the final n.

Lemma 3.11. *Let* $\mathbf{u} \in \mathcal{T}^{fre+}$ *and assume that* $\mathcal{P}_{fre}$ *has been applied to* $p\,n \in^{\wedge} \mathbf{u}$ *with conclusion that* $\mathrm{L}\,q\,n$ *is an* L-*block.*

Then each inner variable of q *is the end-variable of exactly one generated* A-*block and exactly one – corresponding – generated* L-*block.*

The generated A-blocks occur inside the path q in a *balanced* pattern:

Lemma 3.12. *Let* $p\,\mathrm{L}\,q\,n \in^{\wedge} \mathbf{u} \in \mathcal{T}^{fre+}$, $\mathrm{L}\,q\,n$ *being an* L-*block, and assume that* a_1 *and* a_2 *are different* A-*blocks inside* q *generated by* $\mathcal{P}_{fre}$.

(*i*) *Then either* $a_1 \subset a_2$, *or* $a_2 \subset a_1$, *or* a_1 *and* a_2 *do not overlap in* q.

(*ii*) *If* $a_2 \subset a_1$, *then also* $a_2 \subset l_1$, *where* l_1 *is the* L-*block corresponding to* a_1.

3.4 An example of beta-reduction with delayed updating

We consider the following untyped λ-term (in the usual notation) and three of its delayed, focused β-reducts. The underlining is meant to mark the redex parts, the dot above the variable (e.g., $\dot{y}$) marks the focus of the reduction. Transitions (*i*) $\rightarrow_f$ (*ii*) and (*ii*) $\rightarrow_f$ (*iii*) are one step reductions, (*iii*) $\twoheadrightarrow_f$ (*iv*) is two-step.

(*i*) $((\lambda x\,.\,(\underline{\lambda y}\,.\,\lambda z\,.\,\dot{y}\,z)\underline{\lambda u\,.\,x})x)\lambda v\,.\,v\ \rightarrow_f$

(*ii*) $((\lambda x\,.\,(\lambda y\,.\,\underline{\lambda z}\,.\,(\lambda u\,.\,x)\dot{z})\lambda u\,.\,x)\underline{x})\lambda v\,.\,v\ \rightarrow_f$

(*iii*) $((\underline{\lambda x}\,.\,(\lambda y\,.\,\lambda z\,.\,(\lambda u\,.\,x)\dot{x})\lambda u\,.\,\dot{x})x)\underline{\lambda v\,.\,v}\ \twoheadrightarrow_f$

(*iv*) $((\lambda x\,.\,(\lambda y\,.\,\lambda z\,.\,(\lambda u\,.\,x)\lambda v\,.\,v)\lambda u\,.\,\lambda v\,.\,v)x)\lambda v\,.\,v.$

In Figure 3.4 we represent these four λ-terms as trees. We mark the A-L-couples and the arguments in boxes. In the picture, we use a short arrow to mark the focus(ses).

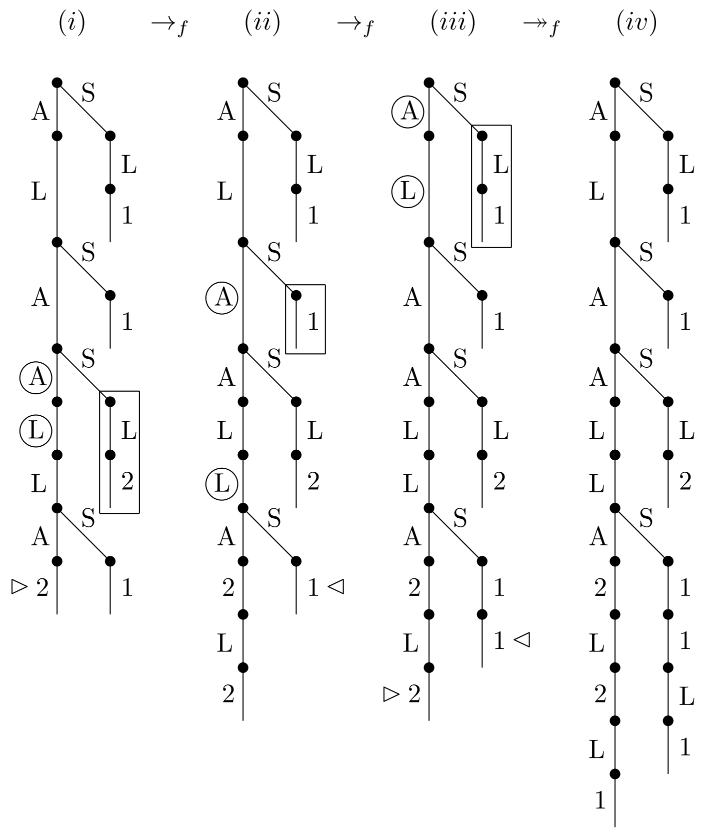

Figure 5: Examples of delayed, focused β-reduction

Note 3.13. *Figure 5, (iii), shows that it may occur that two (or more) num-labels occur adjacently. See the right-most part of the figure, at the end of the complete path* A L A A L L S 1 1*. The first (or top-) label* 1 *is an inner num-variable, the other* 1 *is outer. Since the* A*-block of the first* 1 *is* A A L L S 1*, the* L*-block of the second* 1 *is* L A A L L S 1 1*, so the second* 1 *is bound to the leftmost, uppermost* L.

In Figure 5, (iv), both mentioned labels have become inner ones.

3.5 From name-carrying to name-free lambda-terms, and vice versa

In this section we compare $\mathcal{T}^{car}$ with $\mathcal{T}^{fre+}$ and define mappings between them. It turns out that there is a natural relation between L-blocks in $\mathcal{T}^{car}$

and L-blocks in $\mathcal{T}^{fre+}$, and between A-blocks in $\mathcal{T}^{car}$ and in $\mathcal{T}^{fre+}$ (cf. Definitions 1.17 (*i*), 1.21 and 3.6).

We only consider *closed* λ-trees, i.e., λ-trees in which all variables have been bound. We only consider λ-abstraction, so there are no P_x's or P's in the sets of terms studied below. (Hence, we limit ourselves to the set $\lambda_{\underline{\omega}}$ in Barendregt's cube; cf. Barendregt, 1992.) But Π-abstraction behaves similarly.

Definition 3.14. (*i*) Let $\mathbf{t} \in \mathcal{T}^{car}$. The *structure* of $\mathbf{t}$, or $str(\mathbf{t})$, is $\mathbf{t}$ in which all variables have been omitted. This concerns the variables at the leaves, but also the subscripts of labels L_x, for any x.

(*ii*) Let $\mathbf{u} \in \mathcal{T}^{fre+}$. The *structure* of $\mathbf{u}$, or $str(\mathbf{u})$, is (again) $\mathbf{u}$ in which all variables have been omitted. This concerns both the outer and the inner variables. In the trees representing a structure, the vertices are all rendered as identical dots. This also applies to the vertices with multiple num-labels.

Definition 3.15. We define a mapping from $\mathcal{T}^{car}$ to $\mathcal{T}^{fre}$. Let $\mathbf{t} \in \mathcal{T}^{car}$. Then $[\mathbf{t}] \in \mathcal{T}^{fre}$ is the tree with the same structure as $\mathbf{t}$ and with numbers n instead of the variable names at the leaves of $\mathbf{t}$. These numbers are chosen such that they *respect* the bindings. That is, if $p\,\mathrm{L}_x\,q\,x$ is a path in $\mathbf{t}$, then the corresponding path $p\,\mathrm{L}\,q\,n$ in $[\mathbf{t}]$ has the property that $n = \|q\|+1$.

The reverse mapping is slightly more complicated. We include λ-trees with inner variables, so we describe a mapping from $\mathcal{T}^{fre+}$ to $\mathcal{T}^{car}$. This mapping consists of three stages.

Let $\mathbf{u}$ be a λ-tree in $\mathcal{T}^{fre+}$.

(*i*) *Add names* to L-labels: provide all L-labels in $\mathbf{u}$ with a name, such that $p\,\mathrm{L}_x,\ q\,\mathrm{L}_y \in^\wedge \mathbf{u} \Rightarrow (x \equiv y \Rightarrow p \equiv q)$.

By this process, the various L-labels in $\mathbf{u}$ get names that are *different* from each other. We call the paths thus obtained *enriched paths*.

(*ii*) *Replace* the outer num-labels by appropriate names: consider all enriched complete paths $p\,\mathrm{L}_x\,q\,n$ where $\mathrm{L}\,q\,n$ is an L-block in the original tree $\mathbf{u}$. In every such path, replace n by x.

(*iii*) *Erase* all inner num-labels and their edges. (So the tree changes, it becomes more 'compact'.)

Definition 3.16. Let $\mathbf{u} \in \mathcal{T}^{fre+}$. The combined procedure (*i*), followed by (*ii*), followed by (*iii*), applied to $\mathbf{u}$, gives a mapping from $\mathcal{T}^{fre+}$ to $\mathcal{T}^{car}$. We denote the resulting tree as $\langle\mathbf{u}\rangle$.

Note 3.17. *A possibility for speeding up the procedure in stage* (*ii*)*, is to use the method of* backtracking*, applied to the whole tree, instead of doing the job path by path.*

Lemma 3.18. (*i*) *Let* $\mathbf{t} \in \mathcal{T}^{car}$*. Then* $\langle[\mathbf{t}]\rangle \equiv_\alpha \mathbf{t}$*.*

(*ii*) *Let* $\mathbf{u} \in \mathcal{T}^{fre}$*. Then* $[\langle\mathbf{u}\rangle] \equiv_\alpha \mathbf{u}$*.*

(*iii*) *Let* $\mathbf{u} \in \mathcal{T}^{fre+}$*. Then* $\langle[\langle\mathbf{u}\rangle]\rangle \equiv_\alpha \langle\mathbf{u}\rangle$*.*

3.6 The behaviour of name-free lambda-trees under beta-reduction

In this section, we compare delayed, focused reduction on λ-terms in $\mathcal{T}^{fre+}$ with focused reduction for $\mathcal{T}^{car}$. We choose to study *focused* β-reduction only, since the behaviour of the usual β-reduction and of balanced β-reduction is more or less similar. We discuss erasing reduction later in this chapter.

For that purpose, it is necessary that we can refer to redexes in both $\mathcal{T}^{car}$ and $\mathcal{T}^{fre+}$.

Definition 3.19. Let $r \equiv p\,\mathrm{A}\,b\,\mathrm{L}_x\,q\,x \in^{\wedge}_{\vee} \mathbf{t} \in \mathcal{T}^{car}$, with balanced path b; then p identifies a redex and the pair $\rho \equiv (p, q)$ identifies a *focused* reduction. (Cf. Definition 2.8.)

We write $\mathbf{t} \to^{\rho}_{f} \mathbf{t}'$ for the β-reduction generated by ρ.

Now look at $\mathcal{T}^{fre+}$. Assume that $\mathbf{u} \in \mathcal{T}^{fre+}$ has the same structure as $\mathbf{t}$. There exists a path $r' \equiv p'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,k \in^{\wedge}_{\vee} \mathbf{u}$ with the same structure as r. That is, adding appropriate variable names and striking out all inner variables in r', results in r. Then the pair $\rho' \equiv (p', q')$ identifies a focused reduction in $\mathcal{T}^{fre+}$.

We say that the pairs ρ and ρ' *correspond* to each other and we write $[\rho]$ for ρ'.

Lemma 3.20. *Let* $\mathbf{t} \in \mathcal{T}^{car}$ *and assume that* ρ *identifies a focused* β*-reduction such that* $\mathbf{t} \to^{\rho}_{f} \mathbf{t}'$*.*

Further, let $\mathbf{u} \in \mathcal{T}^{fre+}$ *and* $str(\mathbf{u}) \equiv str(\mathbf{t})$*. Then* $\mathbf{u} \to^{[\rho]}_{df} \mathbf{u}'$ *and* $str(\mathbf{u}') \equiv str(\mathbf{t}')$*.*

In the following lemma, we compare a sequence of focused β-reductions in $\mathcal{T}^{car}$ with a corresponding sequence of delayed, focused β-reductions in $\mathcal{T}^{fre+}$.

Lemma 3.21. *Let* $\mathbf{t}_1 \in \mathcal{T}^{car}$ *and assume that* $\mathbf{t}_1 \to^{\rho_1}_{f} \mathbf{t}_2 \to^{\rho_2}_{f} \ldots \to^{\rho_{n-1}}_{f} \mathbf{t}_n$*, where* $\rho_1 \ldots \rho_{n-1}$ *identify the chosen one-step focused reductions.*
Let $\mathbf{u}_1 \equiv [\mathbf{t}_1] \in \mathcal{T}^{fre}$*. Then there is a delayed, focused reduction:*
$\mathbf{u}_1 \to^{[\rho_1]}_{df} \mathbf{u}_2 \to^{[\rho_2]}_{df} \ldots \to^{[\rho_{n-1}]}_{df} \mathbf{u}_n$*, where* $\mathbf{u}_i \in \mathcal{T}^{fre+} (2 \leq i \leq n)$*.*
Moreover, $str(\mathbf{t}_i) \equiv str(\mathbf{u}_i)$ *for all* $1 \leq i \leq n$*.*

Proof Induction. $\mathbf{t}_1$ and $\mathbf{u}_1$ have the same structure, by Definition 3.15. Use Lemma 3.20. □

Note that, although $\mathbf{u}_i \in \mathcal{T}^{fre+}$, it will in general *not* be the case that $\mathbf{u}_i \equiv [\mathbf{t}_i]$ for $2 \leq i \leq n$.

Lemma 3.22. *Let* $\mathbf{u} \in \mathcal{T}^{fre+}$ *and* $p\,S\,q\,n \in^{\wedge}_{\vee} \mathbf{u}$*. Then the leaf* n *is bound by a certain* L *in some position in either* p *or* q*.*

Let a *be any* A*-block in* $\mathbf{u}$*. Then leaf* n *is bound by an* L *in* $p\,a\,q\,n$ *at the same position in* p *or in* q*.*

Proof If $\mathrm{L} \in q$, then trivial.

Assume $\mathrm{L} \in p$, say $p \equiv p_1\,\mathrm{L}\,p_2$. Apply algorithm $\mathcal{P}_{fre}$:

(i) $p\,\mathrm{S}\,q\,(n,0)\,n \twoheadrightarrow^{(1)} p\,\mathrm{S}\,(m,0)\,q\,n \twoheadrightarrow p\,(m,0)\,\mathrm{S}\,q\,n \equiv p_1\,\mathrm{L}\,p_2\,(m,0)\,\mathrm{S}\,q\,n \twoheadrightarrow^{(2)}$ $p_1\,(0,0)\,\mathrm{L}\,p_2\,\mathrm{S}\,q\,n$.

(ii) Let $a \equiv a'\,k$. Then: $p\,a\,q\,(n,0)\,n \twoheadrightarrow^{\text{see }(1)} p\,a'\,k\,(m,0)\,q\,n \twoheadrightarrow^{\text{push }(m,0)}$ $p\,a'\,(k,1)\,k\,q\,n \twoheadrightarrow^{\text{Def. 3.6,}(ii)} p\,(0,0)\,a'\,k\,q\,n \twoheadrightarrow^{\text{pop }(m,0)} p\,(m,0)\,a\,q\,n \equiv$ $p_1\,\mathrm{L}\,p_2\,(m,0)\,a\,q\,n \twoheadrightarrow^{\text{see }(2)} p_1\,(0,0)\,\mathrm{L}\,p_2\,a\,q\,n$.

So $\mathrm{L}\,p_2\,\mathrm{S}\,q\,n$ and $\mathrm{L}\,p_2\,a\,q\,n$ are L-blocks by Definition 3.6,(i), and n is bound by the same L in both cases. □

Note 3.23. *At transition* (1) *in* (i)*, the reached state must be* $(m,0)$ *for some* m*, and not* (m,l) *for some* $l > 0$*, since the latter case only occurs when the displayed* S *is part of an* r*-block. This would contradict what we said in Note 3.7,* (iv)*.*

Theorem 3.24. *Let* $\mathbf{t}_1 \in \mathcal{T}^{car}$ *and assume that* $\mathbf{t}_1 \to_f^{\rho_1} \ldots \to_f^{\rho_{n-1}} \mathbf{t}_n$.

Let $\mathbf{u}_1 \equiv [\mathbf{t}_1] \in \mathcal{T}^{fre}$ *and* $\mathbf{u}_1 \to_{df}^{[\rho_1]} \ldots \to_{df}^{[\rho_{n-1}]} \mathbf{u}_n$ *(cf. Lemma 3.21).*

Then for all $1 \leq i \leq n$*:* $\langle \mathbf{u}_i \rangle \equiv_\alpha \mathbf{t}_i$.

Proof Induction.

(i) Let $i = 1$. Then $\langle \mathbf{u}_1 \rangle \equiv \langle [\mathbf{t}_1] \rangle \equiv_\alpha \mathbf{t}_1$ by Lemma 3.18.

(ii) Assume that $\langle \mathbf{u}_i \rangle \equiv_\alpha \mathbf{t}_i$. Now $str(\mathbf{u}_i) \equiv str(\mathbf{t}_i)$ by Lemma 3.21, (i).

Since $\mathbf{t}_i \to_f^{\rho_i} \mathbf{t}_{i+1}$, also $\mathbf{u}_i \to_{df}^{[\rho_i]} \mathbf{u}_{i+1}$ and $str(\mathbf{u}_{i+1}) \equiv str(\mathbf{t}_{i+1})$ (Lemma 3.20).

Let ρ_i be an (p,q)-reduction, so there is a path $p\,\mathrm{A}\,b\,\mathrm{L}_x\,q\,x \in \mathbf{t}_i$, with b a balanced path. The corresponding path in $\mathbf{u}_i$ is $p'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,n$, with $str(p') \equiv str(p)$, and similarly for b' and q'. By the reduction $[\rho_i]$, this path is transformed into the grafted tree $p'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,n\,tree(p'\,\mathrm{S})$ in $\mathbf{u}_{i+1}$.

Note that the paths in the mentioned grafted tree are the only paths that we have to consider, since all other paths remain unchanged, both in the transition from $\mathbf{t}_i$ to $\mathbf{t}_{i+1}$ as from $\mathbf{u}_i$ to $\mathbf{u}_{i+1}$.

So let $p'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,n\,r'\,m$ be a complete path in $p'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,n\,tree(p\,\mathrm{S})$. Then $r'\,m \in_{\vee}^{\wedge} tree(p\,\mathrm{S})$. We have to check whether m is bound and if so, to locate the binding L and compare this with the situation in $\mathbf{t}_{i+1}$.

Since the mentioned $tree(p\,\mathrm{S})$ is a copy of the $tree(p\,\mathrm{S})$ that follows $p\,\mathrm{S}$ in $\mathbf{u}_{i+1}$, we also find a path $r'\,m$ in the latter tree, occurring in both $\mathbf{u}_{i+1}$ and $\mathbf{u}_i$. Since by induction $\langle \mathbf{u}_i \rangle \equiv_\alpha \mathbf{t}_i$, we have that there is a path $p\,\mathrm{S}\,r\,y \in \mathbf{t}_i$ for some r with $str(r) \equiv str(r')$.

The final y in this path $p\,\mathrm{S}\,r\,y$ is bound by an L_y in either p or r.

(i) Assume y is bound in p. Then $p \equiv p_1\,\mathrm{L}_y\,p_2$. Also, $p' \equiv p_1'\,\mathrm{L}\,p_2'$ and the final m in $p_1'\,\mathrm{L}\,p_2'\,\mathrm{S}\,r'\,m$ is bound by the mentioned L.

Now note that the path $a \equiv \mathrm{A}\,b'\,\mathrm{L}\,q'\,n$ is an A-block in $\mathbf{u}_i$, so also in $\mathbf{u}_{i+1}$. It follows from Lemma 3.22 that the final m in $p_1'\,\mathrm{L}\,p_2'\,\mathrm{A}\,b'\,\mathrm{L}\,q'\,n\,r'\,m$ is bound by the L following p_1'.

Correspondingly, in $\mathbf{t}_{i+1}$ we have that the final variable z in $p_1\mathrm{L}_y\,p_2\,\mathrm{A}\,b\,\mathrm{L}_x\,q\,r\,z$ must be $z \equiv y$. So the two last-mentioned paths are matching.

(ii) Assume y is bound in r. Then $r \equiv r_1\,\mathrm{L}_y\,r_2$. Also, $r' \equiv r_1'\,\mathrm{L}\,r_2'$ and the final m in $p'\,\mathrm{S}\,r_1'\,\mathrm{L}\,r_2'\,m$ is bound by the mentioned L.

So also the final m in $p' \,\mathrm{A}\, b' \,\mathrm{L}\, q' \, n \, r'_1 \,\mathrm{L}\, r'_2 \, m \in \mathbf{u}_{i+1}$ is bound by the L following r'_1.

In $\mathbf{t}_{i+1}$ we have that the corresponding path is $p \,\mathrm{A}\, b \,\mathrm{L}_x \, q \, r_1 \,\mathrm{L}_z \, r_2 \, z$ for some z. (Note that all binding variables in $\mathbf{t}_{i+1}$ must be different, so α-reduction should change $\ldots \mathrm{L}_y \ldots y$ into something like $\ldots \mathrm{L}_z \ldots z$ in the copy of *tree*$(p\,\mathrm{S})$ that pops up in $\mathbf{t}_{i+1}$ by the $\to_f$-reduction.) Clearly, the bindings of m and z are matching.

It follows that all bindings in $\mathbf{u}_{i+1}$ correspond one-to-one to the bindings in $\mathbf{t}_{i+1}$. This is enough to conclude that $\langle \mathbf{u}_{i+1} \rangle \equiv_\alpha \mathbf{t}_{i+1}$. $\Box$

3.7 Theorems on delayed updating

We single out the λ-trees in $\mathcal{T}^{fre+}$ that 'originate' from $\mathcal{T}^{fre}$ and give a number of theorems and lemmas concerning these trees.

We start with a lemma on inner and outer variables. (See also Lemma 1.22.)

Lemma 3.25. *Let* $\mathbf{u} \in \mathcal{T}^{fre}$ *and* $\mathbf{u} \to_{df} \mathbf{u}'$. *Assume that* $p \in^{\wedge}_{\vee} \mathbf{u}'$ *such that* $p \equiv p_1 \, n \, p_2$.

(i) *Let* p_2 *be non-empty (so* n *is an* inner *variable of* $\mathbf{u}'$*). Then* n *is the final variable of a corresponding* A*-block enclosing a corresponding* L*-block, both being subpaths of* p_1. *Moreover, the front-*L *of the* L*-block binds* n.

(ii) *Let* p_2 *be empty (so* n *is an* outer *variable of* $\mathbf{u}'$*). Then* n *is the final variable of a corresponding* L*-block being a subpath of* p_1. *Moreover, the front-*L *of the* L*-block binds* n. *(Note: there may also be a corresponding* A*-block, enclosing the* L*-block, but this is not necessarily so.)*

Next, we give a general theorem for reductions in $\mathcal{T}^{fre}$ and $\mathcal{T}^{fre+}$ (cf. Theorem 1.27).

Theorem 3.26. (i) *Each of the reductions* $\to_b$, $\to_f$ *and* $\to_e$ *is confluent for name-free paths.*

(ii) *The reduction* $\to_{df}$ *is confluent.*

Proof of (ii) Use Theorem 3.24. $\Box$

Definition 3.27. If $\mathbf{u}' \in \mathcal{T}^{fre+}$ such that there is an $\mathbf{u} \in \mathcal{T}^{fre}$ with $\mathbf{u} \to_{df} \mathbf{u}'$, then $\mathbf{u}'$ is called *legal*. Such a $\mathbf{u}$ is unique and is called the *origin* of $\mathbf{u}'$.

Lemma 3.28. *Let* $\mathbf{u}' \in \mathcal{T}^{fre+}$ *be legal. Let* $\mathbf{u}$ *be defined as the set of all paths* $p\,n \in^{\wedge} \mathbf{u}'$ *such that* p *has no inner variables. Then* $\mathbf{u}$ *is the origin of* $\mathbf{u}'$.

We now discuss some important consequences of Theorem 3.4.

Definition 3.29. Let $\mathbf{u} \in \mathcal{T}^{fre+}$ and $p \in^{\wedge} \mathbf{u}$. The *trail* of p, or *trail*(p), is the non-empty sequence of edges obtained from p by omitting all labels, but preserving the 'directions' of the edges; a *direction* being `l` ('left') or `r` ('right'). This is done such that labels A and L are reflected as `l` in *trail*(p) and label S as `r`. For the (unary) num-labels (inner and outer), we choose `l`.

Example 3.30. Let p be $\mathrm{L\,L\,A\,2\,S\,L\,S\,1\,L}$, then $\mathit{trail}(p) \equiv \mathtt{l\,l\,l\,l\,r\,l\,r\,l\,l}$.

The following lemma is a consequence of the construction principles for trees.

Lemma 3.31. *Let $\mathbf{u} \in \mathcal{T}^{fre+}$ and $p, p' \in^{\wedge} \mathbf{u}$. If $\mathit{trail}(p) \equiv \mathit{trail}(p')$, then $p \equiv p'$.*

Theorem 3.32. *Let $\mathbf{u} \in \mathcal{T}^{fre}$ and assume $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u_1}$ and $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u_2}$. Assume that $p_1 \in^{\wedge} \mathbf{u_1}$ and $p_2 \in^{\wedge} \mathbf{u_2}$ such that $\mathit{trail}(p_1) \equiv \mathit{trail}(p_2)$. Then $p_1 \equiv p_2$.*

Proof By 'confluence' (Theorem 3.26), there exists $\mathbf{u_3} \in \mathcal{T}^{fre+}$ such that $\mathbf{u_1} \twoheadrightarrow_{df} \mathbf{u_3}$ and $\mathbf{u_2} \twoheadrightarrow_{df} \mathbf{u_3}$. Then by Theorem 3.4, $\mathbf{u_1} \subset \mathbf{u_3}$ and $\mathbf{u_2} \subset \mathbf{u_3}$. Hence $p_1, p_2 \in^{\wedge} \mathbf{u_3}$, and by Lemma 3.31, $p_1 \equiv p_2$. $\square$

Definition 3.33. (i) By $\mathcal{T}_{bin}$ we denote the infinite binary tree that is the graphic representation of the (infinite) set of all infinite lists composed of $\mathtt{l}$'s and $\mathtt{r}$'s.

(ii) Each trail $\mathtt{b}$ is a non-empty, finite initial part of some element of $\mathcal{T}_{bin}$. We denote this correspondence by $\mathtt{b} \in^{\wedge} \mathcal{T}_{bin}$.

(iii) An edge in $\mathcal{T}_{bin}$ is called a *position* in $\mathcal{T}_{bin}$.

Definition 3.34. (i) Let $\mathtt{b} \in^{\wedge} \mathcal{T}_{bin}$ and assume that the final element of $\mathtt{b}$ corresponds to position ϵ in $\mathcal{T}_{bin}$. Then we say that $\mathtt{b}$ *marks* ϵ.

(ii) Let $\mathbf{u} \in \mathcal{T}^{fre+}$ and assume that there is a non-empty $p \in^{\wedge} \mathbf{u}$ with final label ℓ, such that $\mathit{trail}(p)$ marks position ϵ. Then we say that $\mathbf{u}$ *covers* position ϵ *with label* ℓ.

Now we can show that, given a λ-tree $\mathbf{u} \in \mathcal{T}^{fre}$ and some $\rightarrow_{df}$-reduct $\mathbf{u}'$ of $\mathbf{u}$ (i.e., a $\mathbf{u}' \in \mathcal{T}^{fre+}$ with $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u}'$) that covers position ϵ with label ℓ, then this label is *unique* – whatever the $\mathbf{u}'$ is.

Theorem 3.35. *Let $\mathbf{u} \in \mathcal{T}^{fre}$ and let ϵ be a position in $\mathcal{T}_{bin}$. Then:*

– either *there is no $\rightarrow_{df}$-reduct of $\mathbf{u}$ that covers ϵ,*

– or *there is such an $\mathbf{u}'$; assume that this $\mathbf{u}'$ covers ϵ with label ℓ; then* all *$\rightarrow_{df}$-reducts of $\mathbf{u}$ covering ϵ, cover ϵ with the same label ℓ.*

Proof Use Theorem 3.32. $\square$

The following theorem states that weak normalization (WN) implies strong normalization (SN) in $\mathcal{T}^{fre+}$. We firstly give a definition.

Definition 3.36. Let $\mathbf{u} \in \mathcal{T}^{fre}$ and $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u}'$.

(i) The *border* of $\mathbf{u}'$, or $\mathit{border}(\mathbf{u}')$, is the set of all outer num-variables of $\mathbf{u}'$.

(ii) Let also $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u}''$. A path $p \in^{\wedge} \mathbf{u}''$ *crosses* the border of $\mathbf{u}'$ *at position* $p_1\, n$ if $p \equiv p_1\, n\, p_2$ where $n \in \mathit{border}(\mathbf{u}')$ and p_2 is not-empty.

Theorem 3.37. *Let $\mathbf{u} \in \mathcal{T}^{fre}$ be weakly normalizing. Then $\mathbf{u}$ is also strongly normalizing.*

Proof By WN, there exists a $\mathbf{u}' \in \mathcal{T}^{fre+}$ and an $\twoheadrightarrow_{df}$-reduction path such that $\mathbf{u} \twoheadrightarrow_{df} \mathbf{u}'$, while there is no $\mathbf{u}''$ such that $\mathbf{u}' \rightarrow_{df} \mathbf{u}''$. (So $\mathbf{u}'$ is a *normal form*.)

Assume that there also exists an *infinite* reduction path $\mathbf{u} \rightarrow_{df} \mathbf{u_1} \rightarrow_{df} \mathbf{u_2} \rightarrow_{df} \ldots$. Since $\mathbf{u} \subset \mathbf{u_1} \subset \mathbf{u_2} \subset \ldots$, there must be an $\mathbf{u_i}$ that is not included in $\mathbf{u}'$. Hence, there is a path $p \in^{\wedge} \mathbf{u_i}$ that crosses $\mathit{border}(\mathbf{u}')$ at position, say, $p_1\, n$. So $p \equiv p_1\, n\, p_2$ for some non-empty p_2.

This n is an inner variable of p, so by Lemma 3.25, (i), num-variable n corresponds to an A-L-couple positioned on p_1. This A-L-couple generates a (p_1, p_2)-reduction in $\mathbf{u}'$ that extends $\mathbf{u}'$ outside its borders. So $\mathbf{u}'$ is not normal. Contradiction. □

4 Delayed renaming

In the previous Section 3 we considered beta-reduction in the *namefree* notation and discussed the possibility to delay the updating of the num-variables that is needed because the count to find the binder of a num-variable has been disturbed in some instances by the beta-reduction. We introduced a procedure to restore the bond between variable and binder.

The present section deals with similar observations for the *name-carrying* notation. Also in that case, the renaming of variables – often a cumbersome task accompanying the beta-reduction – may be postponed. And there is a similar procedure to determine the binding between a variable and its binder. We describe these matters below.

As before, we concentrate on *focused*, *balanced* beta-reduction.

4.1 Focused beta-reduction with delayed renaming

As a start, we rephrase Definition 3.1 for this case. We use symbol $\rightarrow_{dc}$ for delayed, focused β-reduction in $\mathcal{T}^{car}$. See the definition below.

Definition 4.1. Let $\mathbf{t} \in \mathcal{T}^{car}$, let $b \in \mathbf{t}$ be a balanced path, assume that $p\,\mathrm{A}\,b\,\mathrm{L}_x \in^{\wedge} \mathbf{t}$ and that $q\,x$ is a fixed, complete path in $\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x)$, so x is bound by the final L_x in $p\,\mathrm{A}\,b\,\mathrm{L}_x$.

Then $\mathbf{t} \rightarrow_{dc} \mathbf{t}\,[\mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x) := \mathit{tree}(p\,\mathrm{A}\,b\,\mathrm{L}_x)[q\,x := q\,x\,\mathit{tree}(p\,\mathrm{S})]]$.

Hence, we leave the variable x in the tree and copy $\mathit{tree}(p\,\mathrm{S})$ right behind it.

Similarly to Section 3.1, we distinguish *inner* and *outer* variables in trees of $\mathcal{T}^{car}$. (The x in $q\,x$ is clearly an outer variable, the x in $q\,x\,\mathit{tree}(p\,\mathrm{S})$ an inner one.)

The following definition reflects Definition 3.2.

Definition 4.2. The symbol $\mathcal{T}^{car}$ concerns the set of name-carrying, closed trees *without* inner variables. The set of these trees where also inner variables are permitted, we denote by $\mathcal{T}^{car+}$.

Also for name-carrying trees and $\rightarrow_{dc}$-reduction, there is a procedure $\mathcal{P}_{car}$ to locate a binder. This procedure is similar to the one for name-free trees (cf. Definition 3.5), although there is one major difference: the first element of the pair representing a state is not a natural number, but either a variable or the symbol #. This # pops up when the binding L_x for x has been found and the name of variable x does no longer play a role. The second element of a state is a natural number, as earlier.

The **use of the algorithm** $\mathcal{P}_{car}$ is as follows. Let $\mathbf{t} \in \mathcal{T}^{car+}$ and let $p\,x \in^{\wedge} \mathbf{t}$. Assume that we desire to apply algorithm $\mathcal{P}_{car}$. Then transform $p\,x$ into $p\,(x,k)\,x$ for k either 0 or 1, and *start* the algorithm. The k decides whether it delivers an L-block or an A-block (cf. Definition 4.4).

Definition 4.3. The algorithm $\mathcal{P}_{car}$ is specified by the following rules:

1. *first step: stack* $= \emptyset$
2a. $p\ \mathrm{L}_y\ (x,k)\ q\ \rightarrow\ p\ (x,k)\ \mathrm{L}_y\ q$, if $x \not\equiv y$
2b. $p\ \mathrm{L}_x\ (x,k)\ q\ \rightarrow\ p\ (\#,k)\ \mathrm{L}_x\ q$
3. $p\ \mathrm{A}\ (x,k)\ q\ \rightarrow\ p\ (x,k)\ \mathrm{A}\ q$
4. $p\ \mathrm{S}\ (x,k)\ q\ \rightarrow\ p\ (x,k)\ \mathrm{S}\ q$
5. $p\ y\ (x,k)\ q\ \rightarrow\ p\ (y,1)\ y\ q$; *push* (x,k)
6. $p\ \mathrm{L}_y\ (\#,k)\ q\ \rightarrow\ p\ (\#,k{+}1)\ \mathrm{L}_y\ q$, if $k > 0$
7. $p\ \mathrm{A}\ (\#,k)\ q\ \rightarrow\ p\ (\#,k{-}1)\ \mathrm{A}\ q$, if $k > 0$
8. $p\ y\ (\#,k)\ q\ \rightarrow\ p\ (\#,k)\ y\ q$, if $k > 0$
9a. $p\ (\#,0)\ q\ \rightarrow\ p\ \mathit{pop}\ q$, if $\mathit{stack} \neq \emptyset$
9b. $p\ (\#,0)\ q\ \rightarrow \mathit{stop}$, if $\mathit{stack} = \emptyset$.

Definition 4.4. Let $\mathbf{t} \in \mathcal{T}^{car}$ and $p\,x \in^{\wedge} \mathbf{t}$.

(i) If $\mathcal{P}_{car}$ is applied to $p\,(x,0)\,x$ and stops in $p'\,(\#,0)\,q\,x$, then $q\,x$ is called the L-*block* of x.

(ii) If $\mathcal{P}_{car}$ is applied to $p\,(x,1)\,x$ and stops in $p'\,(\#,0)\,q\,x$, then $q\,x$ is called the A-*block* of x.

This algorithm for the name-carrying version has a similar behaviour as that for the name-free case (cf. Lemma's 3.21 and 3.22 and Theorem 3.24). There are also theorems similar to the ones in Section 3.7 that hold in the name-carrying situation.

References

Accattoli, B. and Kesner, D., The structural lambda-calculus. In Dawar, A. and and Veith, H. eds, *CSL 2010*, Vol. 6247 of LNCS, 381-395, Springer, 2010.

Accattoli, B. and Kesner, D., The permutative lambda calculus, *18th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning - LPAR-18*, Merida, Venezuela, 2012

Barenbaum, P. and Bonelli, E., Optimality and the Linear Substitution Calculus. In: Miller, D., ed., *2nd International Conference on Formal Structures* *for Computation and Deduction (FSCD 2017)*, 9:1-9:16, Leibniz International Proceedings in Informatics, 2017.

Barendregt, H.P., Lambda calculi with types. In Abramsky, S., Gabbay, D.M. and Maibaum, T., eds, *Handbook of Logic in Computer Science*, Vol. 2, 117–309, Oxford, 1992.

de Bruijn, N.G., Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem, *Indagationes Math.* 34 (1972), 381–392. Also in Nederpelt *et al.* (1994).

de Bruijn, N.G., *A namefree lambda calculus with facilities for internal definitions of expressions and segments*, Eindhoven University of Technology, EUT-report 78-WSK-03, 1978. (See also The Automath Archive AUT 050, www.win.tue.nl>Automath.)

de Bruijn, N.G., Lambda calculus notation with namefree formulas involving symbols that represent reference transforming mappings, *Indagationes Math.* 40 (1978), 348–356. Also in Nederpelt *et al.* (1994). (See also The Automath Archive AUT 055, www.win.tue.nl>Automath.)

Nederpelt, R.P., *Strong normalisation in a typed lambda-calculus with lambda-structured types.* Ph.D. thesis, Eindhoven University of Technology, 1973. Also in Nederpelt *et al.* (1994).

Nederpelt, R.P., Geuvers, J.H. and de Vrijer, R.C., eds, 1994: *Selected Papers on Automath*, North-Holland, Elsevier.

Ventura, D. L., Kamareddine, F. and Ayala-Rincon, M., Explicit substitution calculi with de Bruijn indices and intersection type systems, *The Logic Journal of the Interest Group of Pure and Applied Logic*, Vol. 23, issue 2, 295–340, Oxford 2015.