SHARING IN THE WEAK LAMBDA-CALCULUS REVISITED

TOMASZ BLANC, JEAN-JACQUES LÉVY, AND LUC MARANGET

INRIA
e-mail address: tomasz.blanc@inria.fr

INRIA, Microsoft Research-INRIA Joint Centre
e-mail address: jean-jacques.levy@inria.fr

INRIA
e-mail address: luc.maranget@inria.fr

Abstract. In a previous paper [4] which appeared in the volume celebrating Klop's 60th anniversary, we presented a labeled lambda-calculus to characterize the dag implementation of the weak lambda-calculus as described in Wadsworth's dissertation [14]. In this paper, we simplify this calculus and present a simpler proof of the sharing property which allows the dag implementation. In order to avoid duplication of presentations, we mainly show here the modifications brought to the weak labeled lambda-calculus in [4]. The reader is therefore recommended to read first the companion article and later read our present paper. We are happy that this note can therefore be considered as establishing a new bridge between two friends and now senior colleagues, Jan Willem Klop and Henk Barendregt whom this work is dedicated to on the occasion of his 60th birthday.

Introduction

In the *strong* λ-calculus, the following ξ-rule is valid

$$(\xi)\ \frac{M \to N}{\lambda x.M \to \lambda x.N}$$

The *weak* calculus does not validate this rule and is therefore non confluent. Several confluent extensions of the weak λ-calculus exist [9, 5, 4]. Basically the ξ-rule is extended as follows:

$$(\xi')\ \frac{M \xrightarrow{R} N \quad x \notin R}{\lambda x.M \xrightarrow{R} \lambda x.N}$$

meaning that the ξ-rule is valid when the bound variable x is not free in the redex R contracted between M and N.

The goal of this note is to study the sharing property in the weak λ-calculus in the way defined by Wadsworth [14]. In section 1, we first quickly review the definitions and theorems of the weak λ-calculus. In section 2, we introduce a new weak labeled λ-calculus

2000 ACM Subject Classification: mathematical logic, formal languages.
Key words and phrases: lambda calculus, term rewriting systems, sharing, optimal reductions, dags.

LOGICAL METHODS
IN COMPUTER SCIENCE
DOI:10.2168/LMCS-???

which differs from the one in [4]; in section 3, we prove the sharing property, which validates the dag implementation of [14]. A more extensive study of this new weak labeled λ-calculus is produced in [3].

1. The weak λ-calculus

The set of λ-terms is the usual set, recursively defined by:

$$M, N ::= x \mid MN \mid \lambda x.M$$

We write $x \in M$ when x is a free variable in M. The reduction step relation $\rightarrow$ is defined by the following axiom and inference rules:

$$(\beta)\ \ R = (\lambda x.M)N \stackrel{R}{\rightarrow} M[\![x \backslash N]\!] \qquad (\nu)\ \frac{M \stackrel{R}{\rightarrow} M'}{MN \stackrel{R}{\rightarrow} M'N} \qquad (w)\ \frac{M \stackrel{R}{\rightarrow} N}{M \rightarrow N}$$

$$(\xi')\ \frac{M \stackrel{R}{\rightarrow} M' \quad x \notin R}{\lambda x.M \stackrel{R}{\rightarrow} \lambda x.M'} \qquad (\mu)\ \frac{N \stackrel{R}{\rightarrow} N'}{MN \stackrel{R}{\rightarrow} MN'}$$

with the classic definition of substitution (using implicit alpha renaming):

$$\begin{array}{rcl} x[\![x \backslash P]\!] & = & P \\ y[\![x \backslash P]\!] & = & y \quad (x \neq y) \\ (MN)[\![x \backslash P]\!] & = & M[\![x \backslash P]\!]\ N[\![x \backslash P]\!] \\ (\lambda y.M)[\![x \backslash P]\!] & = & \lambda y.M[\![x \backslash P]\!] \quad (x \neq y,\ y \notin P) \end{array}$$

The reduction step relation $\stackrel{R}{\rightarrow}$ is annotated with the contracted redex R. We follow Barendregt's notation and write $\twoheadrightarrow$ for the transitive and reflexive closure of $\rightarrow$. So $M \twoheadrightarrow N$ iff M can reduce in several steps (maybe none) to N. We notice that the (ξ')-rule is only valid when the bound variable x is not free in the contracted redex R. Therefore, in the weak λ-calculus, the redexes contracted in a given term cannot have a free variable bound outside.

Theorem 1.1. *The weak λ-calculus is confluent.*

Proof: The proof follows the standard Tait–Martin-Lof's method [2]. □

2. The weak labeled λ-calculus

The weak labeled λ-calculus is a calculus to study sharing. It mimics a calculus on dags, where labels represent the addresses of terms in these dags. Each subterm has its own label; subterms which are copied along reductions keep their labels since they are not copied in dags, but new subterms must have new labels. We want that the weak labeled λ-calculus is confluent, as sharing and reduction strategy are two independent concepts. Therefore an adequate labeling scheme should be invariant through permutations of reduction steps. To get confluence, labels are structured and new terms receive new addresses by composing old labels. Finally, the labeled calculus that we use here is different from the one used for labeling weak explicit substitutions [9], since we have no closures nor explicit substitutions, but here we have to take care of the variable binders.

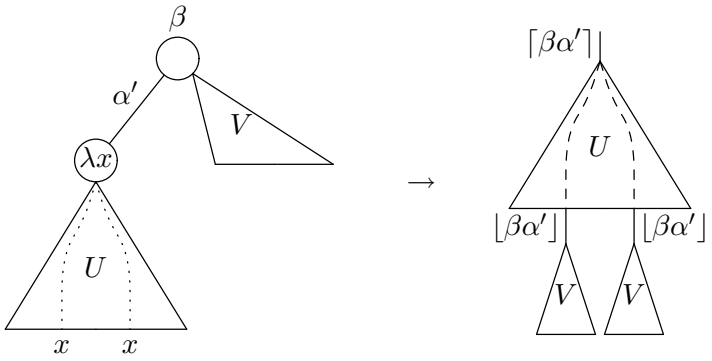

Figure 1: A reduction step in the weak labeled λ-calculus (dotted lines represent paths from the root of U to occurrences of the free variable x in U; dashed lines represent the paths on which diffusion operates).

We base our labeling scheme on the labeling for the strong λ-calculus [8]. The set of labeled terms is recursively defined by:

$$\begin{array}{rcll} U,V & ::= & \alpha:X & \text{labeled terms} \\ X,Y & ::= & S \mid U & \text{clipped or labeled terms} \\ S,T & ::= & x \mid UV \mid \lambda x.U & \text{clipped terms} \\ \alpha,\beta & ::= & a \mid \lceil\alpha'\rceil \mid \lfloor\alpha'\rfloor \mid [\alpha',\beta] & \text{labels} \\ \alpha',\beta' & ::= & \alpha_1\alpha_2\cdots\alpha_n \quad (n \geq 1) & \text{compound labels} \end{array}$$

The labeled term $\alpha:X$ is said to have label α. Labels can be stacked as in $\alpha_1:\alpha_2:\cdots\alpha_n:S$ $(n \geq 1)$. Compound labels are sequences of labels. An atomic label can be a simple letter, or formed by overlining $\lceil\alpha'\rceil$ and underlining $\lfloor\alpha'\rfloor$ the compound label α'; it also can be a pair $[\alpha',\beta]$ of the compound label α' and the label β. In the pair $[\alpha',\beta]$, we say that α' *tags* β.

The labeled reduction ℓ-rule is defined as

$$(\ell) \qquad R = \beta:((\alpha' \cdot \lambda x.U)V) \xrightarrow{R} \lceil\beta\alpha'\rceil:(\beta\alpha' \circledast U)[\![x \setminus \lfloor\beta\alpha'\rfloor:V]\!]$$

where

$$\alpha_1\alpha_2\cdots\alpha_n \cdot X = \alpha_1:\alpha_2:\cdots\alpha_n:X$$

The *name* of R is $\beta\alpha'$; we write $name(R) = \beta\alpha'$. We assume that substitution has a higher precedence than labeling. Hence $\lceil\beta\alpha'\rceil : U[\![x\backslash V]\!]$ is read as $\lceil\beta\alpha'\rceil : (U[\![x\backslash V]\!])$. As in the strong labeled λ-calculus, we sandwich the body of the function part of the redex with its name overlined and underlined as shown in figure 1. The diffusion $\beta\alpha' \circledast U$ creates new labels by tagging labels with $\beta\alpha'$ on the paths from the root of U to free occurrences of x, as illustrated in figure 1. Therefore new labels appeared for every subterm of U containing a free occurrence of x. Formally substitution and diffusion are defined as follows:

$$\begin{array}{rcl@{\qquad}rcl} x[\![x\backslash W]\!] & = & W & \alpha' \circledast X & = & X \text{ if } x \notin X \\ y[\![x\backslash W]\!] & = & y & \alpha' \circledast x & = & x \\ (UV)[\![x\backslash W]\!] & = & U[\![x\backslash W]\!]\, V[\![x\backslash W]\!] & \alpha' \circledast UV & = & (\alpha' \circledast U\ \ \alpha' \circledast V) \text{ if } x \in UV \\ (\lambda y.U)[\![x\backslash W]\!] & = & \lambda y.U[\![x\backslash W]\!] & \alpha' \circledast \lambda y.U & = & \lambda y.\, \alpha' \circledast U \text{ if } x \in \lambda y.U \\ (\beta:X)[\![x\backslash W]\!] & = & \beta:X[\![x\backslash W]\!] & \alpha' \circledast \beta:X & = & [\alpha',\beta]:\alpha' \circledast X \text{ if } x \in X \end{array}$$

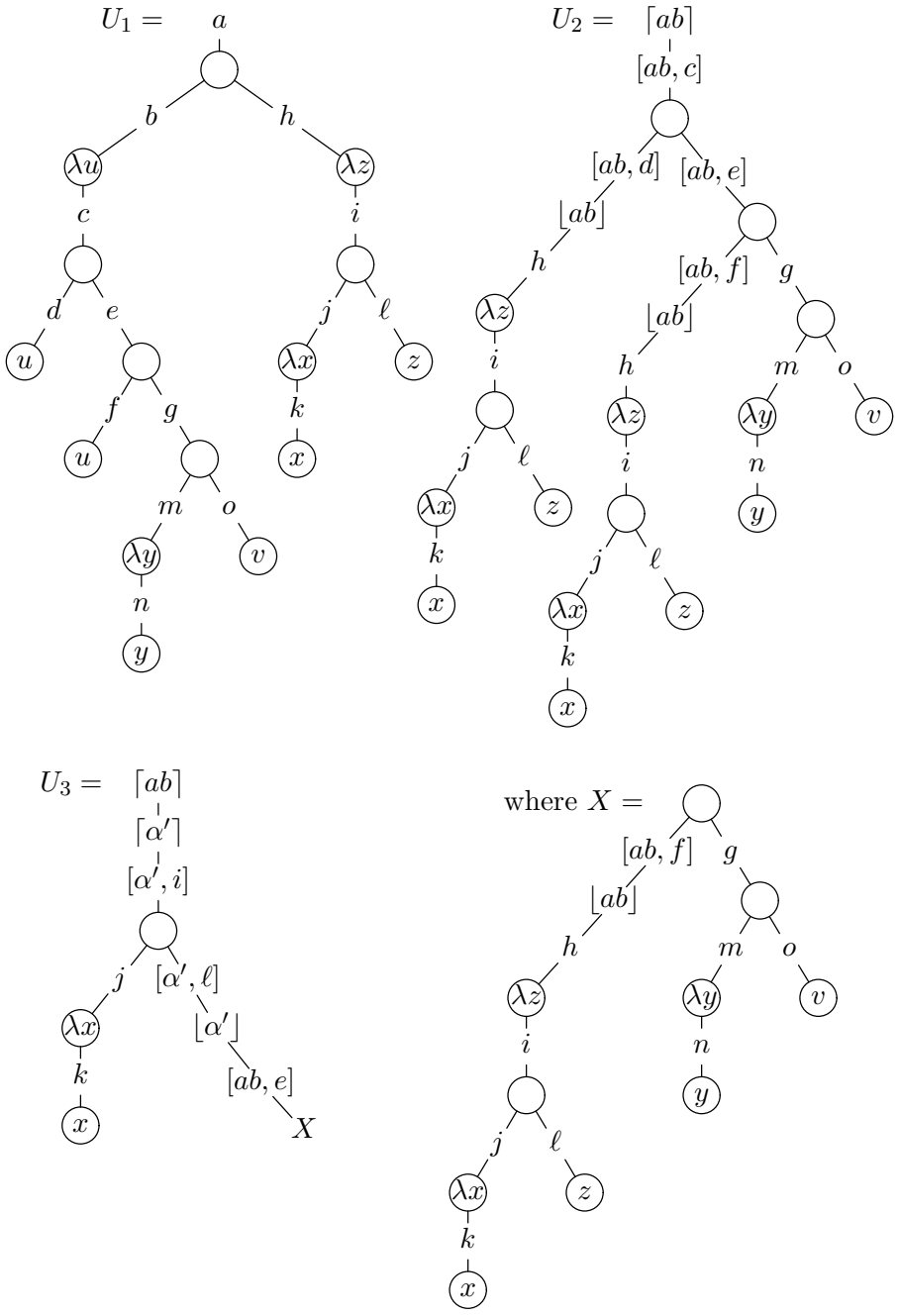

Figure 2: An example of a leftmost outermost reduction in the weak labeled λ-calculus where $U_1 \to U_2 \to U_3$ and $\alpha' = [ab, c][ab, d]\lfloor ab \rfloor h$. (The redex $(\lambda y.y)v$ could also be contracted in U_1, U_2 and U_3; however $(\lambda x.x)z$ with name ij is not a redex in U_1 and U_2; but its "residual" in U_3 is a created redex in U_3, therefore its name $[\alpha', i]j$ contains the name α' of the contracted redex.)

An example of reduction is illustrated in figure 2. The labeled λ-calculus differs from the calculus used in [4]. In this previous article, beta-conversion was simply defined by:

$$(\alpha' \cdot \lambda x.U)V \ \rightarrow \ \lceil \alpha' \rceil : (\alpha' \circledcirc U)[\![x \setminus \lfloor \alpha' \rfloor : V]\!]$$

with no inclusion of the (atomic) label of the application in the name of the contracted redex. This forced to consider a more complex diffusion operation with the so-called marked labels $\langle \alpha', \beta \rangle$ used on left of applications to keep trace of history for creations of K-redexes. In the present labeled λ-calculus, we guarantee that history of redexes is kept by including the atomic labels of applications in the names of redexes and by using a simpler definition of diffusion.

Hence the atomic label of a labeled subterm reflects its history. A simple letter stands for a labeled term existing in the initial term (empty history). Overlining $\lceil \alpha' \rceil$, underlining $\lfloor \alpha' \rfloor$ and tags $[\alpha', \beta]$ indicate past contraction of a redex with name α'.

Free variables are defined as in the unlabeled λ-calculus. The context rules are defined as follows for labeled reduction:

$$(\nu) \ \frac{U \stackrel{R}{\rightarrow} U'}{UV \stackrel{R}{\rightarrow} U'V} \qquad (\lambda) \ \frac{X \stackrel{R}{\rightarrow} X'}{\alpha : X \stackrel{R}{\rightarrow} \alpha : X'} \qquad (w) \ \frac{X \stackrel{R}{\rightarrow} X'}{\overline{X} \rightarrow \overline{X'}}$$

$$(\mu) \ \frac{V \stackrel{R}{\rightarrow} V'}{UV \stackrel{R}{\rightarrow} UV'} \qquad (\xi') \ \frac{U \stackrel{R}{\rightarrow} U' \quad x \notin R}{\lambda x.U \stackrel{R}{\rightarrow} \lambda x.U'}$$

We also write $\twoheadrightarrow$ for the transitive and reflexive closure of $\rightarrow$.

Lemma 2.1. *If $X \stackrel{R}{\rightarrow} X'$ and $x \notin R$, then $\alpha' \circledcirc X \rightarrow \alpha' \circledcirc X'$*

Proof: We first notice that we have $x \in X$ if and only if $x \in X'$. Clearly, $x \in X'$ implies $x \in X$ since the set of free variables of the contractum of a redex is a subset of the free variables of the contracted redex. Conversely, the free variable x disappears in the contractum iff the redex has all free occurrences of x in its argument and the contracted redex erases its argument. This case is not possible since $x \notin R$,

We now proceed by induction on the size of X:

(1) When $x \notin X$, we have $x \notin X'$ and the lemma is obvious, since $\alpha' \circledcirc X = X$ and $\alpha' \circledcirc X' = X'$.

(2) If $x \in X$, we have several cases according to the definition of diffusion:

   (a) Let $X = UV \rightarrow U'V = X'$ with R in U and $U \stackrel{R}{\rightarrow} U'$. By induction, we have $\alpha' \circledcirc U \rightarrow \alpha' \circledcirc U'$. Thus $\alpha' \circledcirc X \rightarrow \alpha' \circledcirc X'$.

   (b) Let $X = UV \rightarrow UV' = X'$ with R in V and $V \stackrel{R}{\rightarrow} V'$. This case is symmetric of the previous case.

   (c) Let $X = \lambda y.U \rightarrow \lambda y.U'$ with R in U and $U \stackrel{R}{\rightarrow} U'$. Again easy induction.

   (d) Let $X = \beta : Y \rightarrow \beta : Y' = X'$. Then R is in Y and $Y \stackrel{R}{\rightarrow} Y'$. Then by induction we get $\alpha' \circledcirc Y \rightarrow \alpha' \circledcirc Y'$ and $[\alpha', \beta] : \alpha' \circledcirc Y \rightarrow [\alpha', \beta] : \alpha' \circledcirc Y'$ by (λ), which means $\alpha' \circledcirc X \rightarrow \alpha' \circledcirc X'$ by the definition of diffusion.

□

Lemma 2.2. *If $U \rightarrow U'$ and $X \rightarrow X'$, then $X[\![x \backslash U]\!] \twoheadrightarrow X[\![x \backslash U']\!]$ and $X[\![x \backslash U]\!] \rightarrow X'[\![x \backslash U]\!]$.*

Proof: Straightforward by induction on the size of X. $\square$

Theorem 2.3. *The weak labeled λ-calculus is confluent.*

Proof: By the Tait–Martin-Lof method, see [2]. To illustrate confluence, we consider local confluence with the only interesting cases of the two following commuting diagrams, when $x \notin R$, and $U \stackrel{R}{\to} U'$, $V \to V'$:

$$\begin{array}{ccc}
\beta:((\alpha' \cdot \lambda x.U)V) & \xrightarrow{\quad R \quad} & \beta:((\alpha' \cdot \lambda x.U')V) \\
\downarrow & & \downarrow \\
\lceil\beta'\rceil:(\beta' \circledast U)[\![x \backslash \lfloor\beta'\rfloor : V]\!] & \longrightarrow & \lceil\beta'\rceil:(\beta' \circledast U')[\![x \backslash \lfloor\beta'\rfloor : V]\!]
\end{array}$$

$$\begin{array}{ccc}
\beta:((\alpha' \cdot \lambda x.U)V) & \longrightarrow & \beta:((\alpha' \cdot \lambda x.U)V') \\
\downarrow & & \downarrow \\
\lceil\beta'\rceil:(\beta' \circledast U)[\![x \backslash \lfloor\beta'\rfloor : V]\!] & \twoheadrightarrow & \lceil\beta'\rceil:(\beta' \circledast U)[\![x \backslash \lfloor\beta'\rfloor : V']\!]
\end{array}$$

where $\beta' = \beta\alpha'$. These cases are easily proved with the previous lemmas. $\square$

3. Sharing of subterms

The goal of this section is to prove that two subterms with the same label are equal in the weak labeled λ-calculus, provided that reductions start from an initial term with distinct letters on all its subterms. Thus two subterms labeled with a same label can be shared in a dag implementation of terms. To prove this sharing property, we introduce a containment relation on labels and prove a few invariants on labeled terms along reductions. Intuitively, the name of a redex records its history. When a redex S is the residual of an other redex R, their names are equal. If a redex S is created by the contraction of a redex R, the name of R is contained in the name of S.

The containment relation $\prec$ on labels is defined by

$$\alpha' \prec \lceil\alpha'\rceil \qquad \alpha' \prec \lfloor\alpha'\rfloor \qquad \alpha' \prec [\alpha', \beta]$$

$$\alpha' \prec \beta_i \;\Rightarrow\; \alpha' \prec \beta_1 \cdots \beta_n \qquad \alpha' \prec \beta' \prec \gamma' \;\Rightarrow\; \alpha' \prec \gamma'$$

The name α' of a redex is contained in any label where it is overlined or underlined. It is also contained in a pair of which it is the first component. Furthermore if it is contained in some β_i, it is also in any compound label of which β_i is a subcomponent. Finally, the containment relation is closed by transitivity. This relation is clearly a strict ordering. We show that it expresses our intuition, i.e. $\alpha' \prec \beta'$ when the contraction of a redex of name α' participates to the creation of β'. We recall that a reduction $X \twoheadrightarrow Y$ creates redex S in Y if S is not a residual (along this reduction) of a redex R in X.

Lemma 3.1. *If $X \stackrel{R}{\to} Y$ and a redex S in Y is created in this reduction step, then $name(R) \prec name(S)$.*

Proof: Straightforward by case inspection. $\square$

Invariant 1. $\mathcal{Q}(W)$ holds iff we have $\alpha' \not\prec \beta$ for every redex R with name α' and any subterm $\beta:X$ in W.

A *complete labeled reduction* step $U \stackrel{\alpha'}{\Longrightarrow} V$ is the finite development of all redexes with name α' in U. We first prove that invariant $\mathcal{Q}$ is preserved by complete labeled reduction steps. Intuitively, invariant $\mathcal{Q}(W)$ means that the names of redexes are maximal labels in W.

Lemma 3.2. *If* $\mathcal{Q}(W)$ *and* $W \stackrel{\gamma'}{\Longrightarrow} W'$, *then* $\mathcal{Q}(W')$.

Proof: Let R' be a redex with name α' in W'. Let $\beta : X'$ be a subterm of W' such that $\alpha' \prec \beta$.

(1) R' is a residual of a redex R of W. Then $name(R) = name(R') = \alpha'$.
  (a) β is a label in W. Impossible since $\mathcal{Q}(W)$.
  (b) β is new, since created by the reduction from W to W'.
    By cases on the label's creation:
      (i) $\beta = \lceil \gamma' \rceil$. Then $\alpha' \prec \lceil \gamma' \rceil$. There are again two cases:
        (A) $\alpha' = \gamma'$. Then R' cannot be residual of R with name γ' since all redexes with name γ' are contracted by $\stackrel{\gamma'}{\Longrightarrow}$.
        (B) $\alpha' \prec \gamma'$. If $\gamma' = \gamma_1\gamma_2\cdots\gamma_n$, there exists γ_i such that $\alpha' \prec \gamma_i$. Therefore there is a subterm $\gamma_i : X$ in W with $\alpha' \prec \gamma_i$. This contradicts $\mathcal{Q}(W)$.
      (ii) $\beta = \lfloor \gamma' \rfloor$ or $\beta = [\gamma', \beta_1]$. These cases are similar to the previous one.

(2) R' is created by the reduction from W to W'. Then $\gamma' \prec \alpha' \prec \beta$.
  (a) β is a label in W. We have $\gamma' \prec \beta$. Impossible by $\mathcal{Q}(W)$.
  (b) β is new, since created by the reduction from W to W'.
    By case on the label's creation:
      (i) $\beta = \lceil \gamma' \rceil$. Then we have $\gamma' \prec \alpha' \prec \lceil \gamma' \rceil$ meaning $\gamma' \prec \alpha' \preceq \gamma'$, which leads to the impossible $\gamma' \prec \gamma'$. Contradiction.
      (ii) $\beta = \lfloor \gamma' \rfloor$ or $\beta = [\gamma', \beta_1]$. These cases are similar to the previous one.

□

A second property consists in showing that redexes with same names are either none actual redexes of the weak λ-calculus or are all at same time redexes of the weak λ-calculus. This property relies on the following invariant related to variables with same labels.

Invariant 2. $\mathcal{R}(W)$ holds iff, for any pair of subterms $\alpha : x$ and $\alpha : y$ in W, we have x free in W iff y free in W.

Lemma 3.3. *If* $\mathcal{R}(W)$ *and* $W \to W'$, *then* $\mathcal{R}(W')$.

Proof: We first notice that the atomic label of a variable cannot change along reductions since when its label changes, the variable is substituted by the argument of the contracted redex. Indeed, let $R = \beta : ((\alpha' \cdot \lambda z.A)B)$ be the contracted redex. Then a variable occurrence $\alpha : x$ in W' comes either from an occurrence of x disjoint from R, either it comes from an occurrence in the body part A, or in the argument B. In first and third cases, the occurrence $\alpha : x$ has same (atomic) label α in W. In second case, we notice that the label of the variable x cannot be tagged since $x \neq z$. So, in all cases, the labeled variable $\alpha : x$ does exist in W.

Thanks to the lambda calculus, the binding of variables is preserved and a free variable cannot become bound and conversely. □

We can now state the sharing statement, meaning that two subterms with same atomic labels are equal and therefore can be be shared.

Invariant 3. $\mathcal{P}(W)$ holds iff, for any pair of subterms $\alpha:X$ and $\alpha:Y$ in W, we have $X = Y$.

Lemma 3.4. *If* $\mathcal{P}(W) \wedge \mathcal{Q}(W) \wedge \mathcal{R}(W)$ *and* $W \stackrel{\gamma'}{\Longrightarrow} W'$, *then* $\mathcal{P}(W')$.

Proof: By $\mathcal{P}(W)$, all redexes with name γ' are the same $R = \gamma:((\beta' \cdot \lambda x.A)B)$, $\gamma' = \gamma\beta'$, and their contractums are identical $R' = \lceil\gamma'\rceil:(\gamma' \textcircled{x} A)[\![x \backslash \lfloor\gamma'\rfloor : B]\!]$ in W'. As these redexes with name γ' are identical, they are all disjoint in W. We also notice that every subterm R is indeed a redex by $\mathcal{R}(W)$ since at least one R is a redex in W.

Let $U' = \alpha:X'$ and $V' = \alpha:Y'$ be two subterms of W'.

(1) Both U' and V' come from subterms $U = \alpha:X$ and $V = \alpha:Y$ in W, with same label as U' and V'. By $\mathcal{P}(W)$, we know that $X = Y$. There are two cases.
   (a) U' contains a contractum R'. Then either U contains x replaced by $\lfloor\gamma'\rfloor:B$ in U'. Either U contains R. The first case is not possible since then R would contain R in its argument. So U contains R. Since every R in X and Y is a redex in W, we get $X = Y$, $X \stackrel{\gamma'}{\Longrightarrow} X'$ and $Y \stackrel{\gamma'}{\Longrightarrow} Y'$. Thus $X' = Y'$.
   (b) U' does not contain a contractum R' of R.
      (i) U' is disjoint from the contractums. Thus U is a subterm of W disjoint from redexes R. Then $U = U'$.
      (ii) U' is in the argument part of a contractum. Thus, U is a subterm of W in the argument part of a redex R.
      (iii) U' is in the body part of the contractum. Thus, U is a subterm of W in the function body part of a redex R, but U does not contain the variable x bound in R, since otherwise its label α would not be equal to the one of U'.

   In any of these cases, $U' = \alpha:X' = U = \alpha:X$. Therefore $X' = X$. Moreover, as $X = Y$, then $V = \alpha:X$ cannot contain R (since X would do) or an occurrence of the bound variable of the function part in R, which means that $Y = Y'$. Thus $X' = Y'$.

(2) Both labels α are new, since both created in W'. Thus $\gamma' \prec \alpha$.
   (a) $\alpha = [\gamma', \alpha_1]$.
   There exist two labeled subterms $\alpha_1:X_1$ and $\alpha_1:Y_1$ in W such that $\alpha:X' = (\gamma' \textcircled{x} \alpha_1:X_1)[\![x \backslash \lfloor\gamma'\rfloor : B]\!]$ and $\alpha:Y' = (\gamma' \textcircled{x} \alpha_1:Y_1)[\![x \backslash \lfloor\gamma'\rfloor : B]\!]$. By $\mathcal{P}(W)$, we have $X_1 = Y_1$ which implies $X' = Y'$.
   (b) $\alpha = \lceil\gamma'\rceil$. By $\mathcal{Q}(W)$, this label is absent from W. It is created at the top of contractums of γ'-redexes, which are all equal by $\mathcal{P}(W)$.
   (c) $\alpha = \lfloor\gamma'\rfloor$. By $\mathcal{Q}(W)$, this label is absent from W. It is created at the top of copies of arguments of R in the contractums. These copies are all equal.

(3) α of $\alpha:X$ is created, but α of $\alpha:Y$ already exists in W. Thus $\gamma' \prec \alpha$, but by $\mathcal{Q}(W)$, we also know that $\gamma' \not\prec \alpha$. This case is impossible.

(4) $\alpha:X$ exists in W, but α of $\alpha:Y$ is created. Analogous to the previous case.

□

We can now state our sharing theorem, characterizing a dag implementation for evaluating terms in the weak λ-calculus. We need to have a notation for terms without sharing.

Notation 1. Let *Init*(U) holds when every subterm of U is labeled with a distinct letter.

Theorem 3.5. *Let* *Init*(U) *and* $U \Longrightarrow V$, *then* $\mathcal{P}(V)$.

Proof: We first notice that, if $Init(U)$, then $\mathcal{P}(U) \wedge \mathcal{Q}(U) \wedge \mathcal{R}(U)$. Thanks to previous lemmas, this invariant remains valid along reduction steps $\Longrightarrow$. □

4. Conclusion

The weak labeled λ-calculus provides a formal setting for a dag implementation of the weak λ-calculus. This calculus differs from the one in [4] since the name of a labeled redex is now not only considering the compound label between the application node and the abstraction subterm involved in a beta redex, but it also includes the atomic label of the application node. This allows a simplification of proofs for proving the sharing property. The proof is also simpler that the one used in [11, 10] which was dedicated to orthogonal term rewriting systems [13].

As also stated in [4], the theory of optimal reductions [8, 11, 3, 1, 7, 6] can be developed within this new setting. One might guess that the shared call-by-name strategy is optimal, which would correspond to the informal claim of Wadsworth within the weak labeled λ-calculus. Also the correspondence with the bottom-up strategy of Shivers and Wand [12] is to explore in a refinement of this calculus with a more explicit definition of substitution.

Finally, as demonstrated in [3], the strong labeled λ-calculus studied in [8] can also modified to include atomic labels of application nodes in the names of redexes. It is unclear whether this change simplifies sharing proofs.

References

[1] A. Asperti and S. Guerrini. *The Optimal Implementation of Functional Programming Languages*. Cambridge University Press, 1999.

[2] H. P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*. North-Holland, 1981.

[3] T. Blanc. *Propriétés de sécurité dans le lambda-calcul*. PhD thesis, Ecole polytechnique, Palaiseau, November 2006.

[4] T. Blanc, J.-J. Lévy, and L. Maranget. Sharing in the weak lambda-calculus. In *Processes, Terms and Cycles: Steps on the Road to Infinity. Essays dedicated to Jan Willem Klop*, volume 3838 of *LNCS*. Springer, 2005.

[5] N. Çağman and J. R. Hindley. Combinatory weak reduction in lambda calculus. *Theoretical Computer Science*, 198:239–249, 1998.

[6] Y. Lafont. Interaction nets. In *Principles of Programming Languages*, pages 95–108. ACM Press, 1990.

[7] J. Lamping. An algorithm for optimal lambda-calculus reduction. In *Proceedings of the 17th Annual ACM Symposium on Principles of Programming Languages*, pages 16–30, 1990.

[8] J.-J. Lévy. *Réductions correctes et optimales dans le lambda-calcul*. PhD thesis, Univ. of Paris 7, Paris, 1978.

[9] J.-J. Lévy and L. Maranget. Explicit substitutions and programming languages. In *Proc. 19th Conference on the Foundations of Software Technology and Theoretical Computer Science, Electronic Notes in Theoretical Computer Science*, volume 1738, pages 181–200, 1999.

[10] L. Maranget. Optimal derivations in orthogonal term rewriting systems and in weak lambda calculi. In *Proc. of the 18th Conference on Principles of Programming Languages*. ACM Press, 1991.

[11] L. Maranget. *La stratégie paresseuse*. PhD thesis, Univ. of Paris 7, Paris, 1992.

[12] O. Shivers and M. Wand. Bottom-up beta-substitution: uplinks and lambda-dags. Technical report, BRICS RS-04-38, DAIMI, Department of Computer Science, University of Århus, Århus, Denmark, 2004.

[13] Terese (M. Bezem, J. W. Klop, and R. de Vrijer eds). *Term Rewriting Systems*. Cambridge University Press, 2003.

[14] C. P. Wadsworth. *Semantics and pragmatics of the lambda-calculus*. PhD thesis, Oxford University, 1971.