Every unsolvable lambda-terme has a decoration

René David

Every unsolvable λ term has a decoration

René DAVID

Laboratoire de Mathématiques. Université de Savoie F 73376 Le Bourget du Lac.
email : david@univ-savoie.fr

Abstract. I give a proof of the conjecture stated in [2] by R.Kerth : Every unsolvable λ term has a decoration.

1 Introduction

In this paper I give a proof of the conjecture stated in [2] by R. Kerth : Every unsolvable λ term has a decoration.

Let t be unsolvable. Denote by t_k the term obtained from t after k many steps of head reduction and by $(d\ \overrightarrow{u})$ the term d applied to the sequence $\overrightarrow{u}$ of arguments. If t reduces to t', say that a subterm d' of t' is a descendent (cf. definition 9) of a subterm d of t if it is a "copy" of d.

A sequence $(d_k)_{k\in N}$ of λ terms is a decoration for (the computation of) t if there is a strictly increasing function f from N to N such that for every k :

1. $t_{f(k)} = \overrightarrow{\lambda}(d_k\ \overrightarrow{u_k})$ for some finite (non empty) sequence $\overrightarrow{u_k}$ of λ terms.
2. d_k is solvable and d_{k+1} is a descendent of some element of $\overrightarrow{u_k}$.

Comments, notations and examples

1. The definition of a decoration given above is exactly the one of [2] but, in fact, the hypothesis "d_k is solvable" is useless since it is a consequence of the other hypothesis (cf the corollary 2)
2. Let $\delta = \lambda x\ (x\ x)$, $I = \lambda x\ x, B = \lambda b\ \lambda f\ (f\ (b\ b\ f))$ and $Y = (B\ B)$. Y is the Turing fixed point operator.
3. Let $t = (\delta\ \delta)$. Then the constant sequence (δ) is a decoration for t since t reduces by head reduction to $t' = (\delta\ \delta)$ and the first δ in t' is a descendent of the second δ in t.
4. Let $t = (B\ B\ I)$. Then the constant sequence (B) is a decoration for t since t reduces to itself (in 3 steps) and the first occurrence of B in this reduct is a descendent of the second occurence of B in t.
5. Let $w_1 = \lambda xyz\ (z\ x\ y)$, $w_2 = \lambda xyz\ (y\ (x\ (z\ x)))$, $R = (w_1\ I\ w_2)$ and $w_3 = (w_2\ R)$. Then,
   - $t = (w_2\ R\ I\ w_2) \twoheadrightarrow (R\ w_3)$ (in 4 steps)
   - $(R\ w_3) \twoheadrightarrow (w_3\ I\ w_2) = t'$ (in 3 steps)
   - $(w_3\ I\ w_2) \twoheadrightarrow (w_2\ R\ I\ w_2) = t$ (in 7 steps)

It is easy to check that w_2, w_3 and R are solvable and that the descendent condition is satisfied. Thus the sequence $[w_2,\ R,\ w_3,\ w_2,\ R,\ w_3,\ w_2, ...]$ is a decoration for t. Note that t' is equal to t but t is written as w_2 applied to 3 arguments whereas t' is written as w_3 applied to 2 arguments and thus the R in t' is not seen as an argument of the head term.

6. Other examples can be found in [1].

The motivation (see [2]) of this conjecture is the following : A model of λ calculus is said to be sensible if all the unsolvable terms are equal in this model. It is not easy, in general, to check whether a given model of λ calculus is sensible or not. In [1] , [3] R Kerth built an uncountable number of graph models with different equational theories but he was unable to prove they were sensible, because the usual argument of reducibility did not work in his models. He was able to show that his models had no critical sequences (a semantical notion he introduced) and he showed that a graph model without critical sequences is sensible ... if his conjecture is true.

Thus, the constructions in [1] , [3] and the present paper show that there are uncountably many sensible distinct equational theories of continuous models (and similarly for the stable and strongly stable semantics).

Acknowledgements Rainer Kerth has read very carefully the first versions of this paper and suggested many improvements. Thanks, Rainer.

2 The idea of the proof

R. Kerth defines a decoration only for the head reduction of unsolvable terms, i.e. terms whose Böhm tree is $\perp$. I define below a decoration for the computation (by left reduction) of *any* branch of a term t. A branch in t is either an infinite branch of its Böhm tree or a finite one finishing with $\perp$, i.e. a branch in t corresponds to an infinite computation. I prove a more general result (The computation of any branch in any λ term admits a decoration. cf. Theorem 1) but this general notion of decoration is necessary for the proof of even the restricted case. The idea of the proof is the following.

1) Let a be a branch of t and b be a branch of a subterm u of t. I say that b is (t, a) useful if, intuitively (see the definition 10) the computation of the branch a of t "uses" all the nodes of addresses $b \restriction i$ $(i < lg(b))$ of the Böhm tree of u. I first show that (cf the proposition 5) if a branch b of u is (t, a) useful and there is a decoration for (u, b), then there is a decoration for (t, a). This is the reason for which it is necessary to extend the notion of decoration to solvable terms. The decoration of an unsolvable term t may "come from" a decoration of a solvable subterm u of t.

2) Let $t = (u\ r_1\ ...\ r_n)$ and a be a branch in t. Say that a is created by the application of u to $r_1\ ...\ r_n$ if neither in u nor in any r_i there is a branch that is (t, a) useful. I also show (this is the key point of the proof, see the proposition 6) that if the branch a in $t = (u\ r_1\ ...\ r_n)$ is created by the application of u to $r_1\ ...\ r_n$, then t reduces to some $t' = \overrightarrow{\lambda}(r_i\ s_1\ ...\ s_m)$ for some $s_1\ ...\ s_m$ and

- the occurrence of r_i in t' is a descendent of the one in t.
- the branch a in t' still is created by the application of r_i to $s_1 \ldots s_m$.

Actually the proposition 6 is a bit more complicated because we have to deal with possible substitutions of the free variables.

3) The theorem 1 is then proved by induction on the complexity of t. If t is in head normal form the result follows immediately from the induction hypothesis. Otherwise $t = \overrightarrow{\lambda}\ (u\ r_1\ \ldots\ r_p)$ for some $p \geq 1$. If the branch a is not created by the application of u to $r_1\ \ldots\ r_n$, i.e. either in u or in some r_i there is a branch that is (t, a) useful, the result follows from the induction hypothesis and the first point above. Otherwise, we get a decoration by using repeatedly the second point above.

3 Definitions

Definition 1. *1. Let A be the set of finite or infinite lists of elements of* $N^* = N - \{0\}$. *A finite list is called an address.*

2. *Let a, a' be in A. a* $\leq$ *a' means that a is an initial segment of a'. For i* $<lg(a)$, $a \restriction i$ *denotes the restriction of a to its first i elements.*
3. *The list a with i added at the beginning (resp at the end) will be denoted by [i :: a] (resp [a :: i]). The empty list is denoted by nil.*

To be able to prove results on substitutions I need some extension of Λ. This is closely related (and a bit more general) to the directed λ calculus introduced in [4].

Definition 2. *1.* Λ *denotes the set of* λ *terms.*

2. *The set* Λ' *of terms is defined by the following grammar :*
   $\Lambda' = V \mid \bot \mid c(a, \sigma) \mid \ \lambda x\ \Lambda' \mid (\Lambda'\ \Lambda')$
   where
   (a) V *is the set of variables*
   (b) *a substitution is a function from* V *to* Λ' *that is the identity except for a finite set (called its domain) of variables.*
   (c) *for every address a and every substitution,* $c(a, \sigma)$ *is a constant.*
3. *A Böhm function is a partial function* $f : A \rightsquigarrow \{\bot\} \cup \{(E, x, p)\ /\ E \subset V, E$ *finite,* $x \in V, p \in N\}$ *which satisfies :*
   (a) *f(nil) is defined.*
   (b) *f([a :: i]) is defined iff f(a) = (E, x, p) and* $i \leq p$.
   (c) *If f(a) = (E, x, p), f(a') = (E', x', p') and* $a \neq a'$ *then* $E \cap E' = \emptyset$.

Notations, conventions and comments

- I adopt the Barendregt convention that variables are always named in such a way that there is no undesired capture and no confusion between different names.
- $\overrightarrow{\lambda}$ denotes a sequence (possibly empty) of abstractions and $(t\ \overrightarrow{r})$ represents the term t applied to a sequence (possibly empty) of arguments.

- $c(a, \sigma)$ represents the subterm (at the address a, in the environment given by σ) of the Böhm tree of some term u that will be substituted later on.
- A Böhm function codes a Böhm tree in the following way : $f(a) = (\{x_1, ..., x_k\}, x, p)$ (resp $\perp$) means that the node at the address a in the Böhm tree coded by f is $\lambda x_1 ... \lambda x_k\ (x\ t_1 ... t_p)$ for some terms $t_1, ..., t_p$ (resp $\perp$).

Definition 3. *Let σ, σ' be substitutions and t be in Λ'.*

1. *The free variables of t are defined by the usual rules and*
   - *$\perp$ has no free variables*
   - *x is a free variable of $c(a, \sigma)$ iff x is a free variable of $\sigma(y)$ for some y in the domain of σ.*
2. *The substitution $\sigma(t)$ is defined by the usual rules and*
   - *$\sigma(c(a, \tau)) = c(a, \sigma \circ \tau)$ for every τ and a.*
   - *$\sigma(\perp) = \perp$.*

Lemma 1. *Every term in Λ' can be uniquely written as $\overrightarrow{\lambda}\ (R\ \overrightarrow{r})$ where R is either a variable or $\perp$ or $(\lambda x\ u\ v)$ or $c(a, \sigma)$.*

Proof. By induction on the term.

Definition 4. *1. Let $t = \overrightarrow{\lambda}\ (R\ r_1\ ...\ r_q)$ be in Λ' and f be a Böhm function. One step of f-reduction of t is defined as follows :*
   - *If $R = x$ then t is in f-head normal form and t has no f-reduct.*
   - *If $R = \perp$*
     - *If $t = \perp$ then t is in f-head normal form and t has no f-reduct.*
     - *otherwise, the f-reduct of t is $\perp$.*
   - *If $R = (\lambda x\ u\ v)$ then the f-reduct of t is $\overrightarrow{\lambda}\ (\sigma(u)\ r_1\ ...\ r_q)$ where $\sigma(x) = v$.*
   - *If $R = c(a, \sigma)$*
     - *If f(a) = ($\{x_1, ..., x_k\}$, x, p) then the f-reduct of t is $\overrightarrow{\lambda}\ \lambda x_{j+1} ... \lambda x_k\ (\sigma'(x)\ c([a :: 1], \sigma')\ ...\ c([a :: p], \sigma')\ r_{j+1} ... r_q)$ where j =Min(k, q), $\sigma' = \tau \circ \sigma$ and τ is defined by $\tau(x_i) = r_i$ for 1 $\leq i \leq j$.*
     - *If $f(a) = \perp$, then the f-reduct of t is $\perp$.*
     - *If f(a) is not defined the f-reduct of t is not defined.*

2. *$t \rightarrow_f t'$ (resp $t \twoheadrightarrow_f t'$) means that t' is the f-reduct of t (resp t' is obtained from t by some, possibly zero, steps of f-reductions).*

Comments and conventions

- An example of f-reduction is given after the definition 10.
- If t is in Λ the f-reduction is the ordinary head reduction (f is never used and thus can be anything).
- If t is in Λ' and f "represents" the term u (see the definition 8) the f-reduction "corresponds" to the (ordinary) head reduction of t' where

- t' is the term t where the constants $c(a, \sigma)$ have been replaced by the subterm of the Böhm tree of u at the address a in the environment σ.
- "corresponds" means that the reduction is the same except that the part of the computation of t' that "comes from" the computation of the node at the address a in the Böhm tree of u has been forgotten and is given by the "oracle" f.

– I allow $f(a)$ to be undefined in the definition of the f-reduction of t because I made no restrictions in the definition of Λ'. However the typical situation where the f-reduction is used is the following. Let $t = (u\ \overrightarrow{r})$ be in Λ, f "represents" u and $t' = (c(nil, Id)\ \overrightarrow{r})$. In this case the f-reduction will clearly always be defined.
– Similarly, if t "comes from" a λ term, since I only do head reductions the composition $\sigma' = \tau \circ \sigma$ (in the case $R = c(a, \sigma)$) in fact is a concatenation of substitutions (cf the definition 12 and the lemma 8) but I must allow also composition when I know nothing on t.
– When t is in Λ, I will not write the symbol f. For example I will write $t \twoheadrightarrow t'$ instead of $t \twoheadrightarrow_f t'$ and similarly for all the definitions in this section. For example $hnf(t)$ instead of $hnf(f, t)$ in the next definition.
– The letters $a, b, c, ...$ are reserved for elements of A, the letters $f, g, ...$ for Böhm functions and the letters $r, s, t, ...$ for terms in Λ'. This will avoid possible confusions.

Definition 5. *$hnf(f, t)$ (the f-head normal form of t) is defined by*

1. – *If some step of the f-reduction of t is undefined, then $hnf(f, t)$ is not defined.*
   – *If $t \twoheadrightarrow_f t'$ for some term t' in f-head normal form and $t' \neq \bot$, then $hnf(f, t) = t'$. In this case t is said to be f-solvable.*
   – *If the f-reduction of t does not terminate or if $t \twoheadrightarrow_f \bot$, then $hnf(f, t) = \bot$. In this case t is said to be f-unsolvable.*

Definition 6. *Let a be an address, t be in Λ' and f be a Böhm function*

1. *a is f-accessible in t is defined by*
   – *nil is f-accessible in t*
   – *$[i :: l]$ is f-accessible in t iff hnf(f, t) = $\overrightarrow{\lambda}(x\ t_1 ... t_n), 1 \leq i \leq n$ and l is f-accessible in t_i*
2. *Let a be f-accessible in t. $hnf(f, t, a)$ is defined by*
   – *$hnf(f, t, nil) = hnf(f, t)$.*
   – *$hnf(f, t, [i :: l]) = hnf(f, t_i, l)$ where $hnf(f, t) = \overrightarrow{\lambda}(x\ t_1 ... t_n)$*
3. *Let a be f-accessible in t. $adr(f, t, a)$ is defined by*
   – *$adr(f, t, nil) = t$.*
   – *$adr(f, t, [i :: l]) = adr(f, t_i, l)$ where $hnf(f, t) = \overrightarrow{\lambda}(x\ t_1 ... t_n)$*

Comments In the following t is assumed to be in Λ.

- a is accessible in t iff the Böhm tree of t (denoted by $BT(t)$) has a node at the address a.
- $hnf(t,a)$ is the λ term we get at the address a when the computation of the node at this address in $BT(t)$ is *terminated*.
- $adr(t,a)$ is the λ term we get at the *beginning* of the computation of the node at this address in $BT(t)$.

Definition 7. *Let a be in A, t be in Λ' and f be a Böhm function.*

1. *a is an f-branch in t iff*
   - *$\forall i < lg(a)$ $a \restriction i$ is f-accessible in t.*
   - *if a is finite, then $hnf(f,t,a) = \perp$*
2. *Assume a is an f- branch in t and k be in N. $Res(f,t,a,k)$ and $Br(f,t,a,k)$ are defined by*
   - *$Res(f,t,a,0) = t$ and $Br(f,t,a,0) = a$*
   - *If $Res(f,t,a,k)$ is not an f-head normal form then $Res(f,t,a,k+1) =$ the f- reduct of $Res(f,t,a,k)$ and $Br(f,t,a,k+1) = Br(f,t,a,k)$*
   - *If $Res(f,t,a,k) = \overrightarrow{\lambda}(x\ t_1 ... t_n)$ and $a = [i :: l]$ then $Res(f,t,a,k+1) = t_i$ and $Br(f,t,a,k+1) = l$*
   - *Otherwise $Res(f,t,a,k)$ and $Br(f,t,a,k)$ are undefined.*
3. *$t \twoheadrightarrow_{f,a} t'$ means that t' = Res(f, t, a, k) for some k.*

Comments and examples In the following t is assumed to be in Λ.

1. $Res(t,a,k)$ is the term we get after k many steps in the computation of the branch a of $BT(t)$.
2. If $t' = Res(t,a,k)$ then $a' = Br(t,a,k)$ is the branch of t' that has to be computed to finish the computation of the branch a of t. Thus, if $t \twoheadrightarrow_a t'$ and $t' \twoheadrightarrow_{a'} t"$ then $t \twoheadrightarrow_a t"$.
3. Let t be in Λ. If t is unsolvable, then nil is the only accessible address (and the only branch) in t.
4. Let $t = (I\ \ \lambda x\ (x\ (\delta\ \delta)))$. Then $hnf(t,nil) = \lambda x\ (x\ (\delta\ \delta))$, $adr(t,[1]) = (\delta\ \delta)$ and $hnf(t,[1]) = \perp$. The only branch of t is $[1]$.
5. $hnf(Y,nil) = \lambda f\ \ (f\ (B\ B\ f))$. $hnf(Y,[1,1,...,1]) = (f\ (B\ B\ f))$. The only branch of Y is $1^\infty = [1,\ 1,\ ...]$.
6. Let $w = \lambda xyz\ (z\ (y\ (x\ x\ y))\ z)$ and $t = (w\ w)$.
   - $hnf(t,nil) = \lambda yz\ (z\ (y\ (w\ w\ y))\ z)$,
   - $hnf(t,[1]) = (y\ (w\ w\ y))$,
   - $hnf(t,[2]) = z$,
   - $hnf(t,[1,1]) = \lambda z_1(\ z_1\ (y\ (w\ w\ y))\ z_1)$
   - a is accessible in t iff $a = [1,1,...,1]$ or $a = [1,1,...,1,2]$. The only branch of t is 1^∞.

Definition 8. *Let u be in Λ' and g be a Böhm function.*

1. *$\psi(g,u)$ is the Böhm function f defined as follows*

- *f(a) is defined iff a is g-accessible in u.*
- *f(a) = $(\{x_1, \ldots, x_k\}, x, p)$ iff $hnf(g,u,a) = \lambda x_1 \ldots \lambda x_k\ (x\ t_1 \ldots t_p)$ for some terms $t_1, \ldots, t_p$*
- *f(a) = $\perp$ iff $hnf(g,u,a) = \perp$.*

2. *Let t be in Λ'. $t[g,u]$ is the term obtained by replacing in t the occurrences of $c(a,\sigma)$ by $\sigma(adr(g,u,a))$ for every a and σ.*

Comment and example

- Most of the time the previous definition will be used with u in Λ and thus $t[g,u]$ also is in Λ and g is useless. In this case the function ψ describes the nodes of $BT(u)$. Remember (cf. the conventions after the definition 4) that, in this case, we "forget" the argument g i.e. we write $\psi(u)$ and $t[u]$. However the more general definition is necessary to prove that (see the proposition 2) "to be useful" is a transitive notion.
- Let $f = \psi(Y)$. Since $Y \twoheadrightarrow \lambda x\ (x\ (x$ ... we have $f(nil) = (\{x\}, x, 1)$ and $f([1,1,1...,1]) = (\emptyset, x, 1)$

Definition 9. *Let t be in Λ'.*

1. *The notion of subterm of t is defined as usual, with the following additional rule. u is a (strict) subterm of $c(a,\sigma)$ if u is a subterm of $\sigma(x)$ for some x.*
2. *Let f be a Böhm function, b be f-accessible in t and $t \twoheadrightarrow_{f,b} t'$.*
   - *A subterm u' of t' is a residue of a subterm u of t if it is a "copy by β-reduction" of u where, possibly, the free variables have been substituted. u' is a descendent of u if it is a residue of u and the free variables have not been substituted.*
   - *The subterm $u' = c(a',\sigma')$ of t' is an immediate successor of the subterm $u = c(a,\sigma)$ of t if*
     $t \twoheadrightarrow_{f,b} t_1 = \overrightarrow{\lambda}(c(a,\tau)\ \overrightarrow{r}) \rightarrow_f t_2 = \overrightarrow{\lambda}(\tau'(x)\ c([a :: 1],\tau')\ \ldots\ c([a :: p],\tau')\ \overrightarrow{r'}) \twoheadrightarrow_{f,b} t'$
     u' is a residue of some element of the sequence $c([a :: 1],\tau')\ \ldots\ c([a :: p],\tau')$ in t_2
     the occurrence of $c(a,\tau)$ in t_1 is a residue of u.
3. *The successor relation (between terms as $c(a,\sigma)$) is the transitive closure of the immediate successor relation.*

Remark A more "formal" definition of these notions (that are intuitively very clear) is rather tedious. For more details see [2]. It is clear that the notion of descendent given above is exactly the one in [2]. In particular, if $t = (d\ \overrightarrow{u}) \twoheadrightarrow_a (d'\ \overrightarrow{u'})$ and d' is a residue of some element of the sequence $\overrightarrow{u}$ then it is also a descendent of this element.

Definition 10. *Let t, u be in Λ and assume that $t = D(\sigma(u))$ for some context D and some substitution σ. Let $t' = D(c(nil,\sigma))$ and $f = \psi(u)$. Let a be a branch in t.*

1. *Let b be an address accessible in u. b is (t, a) useful if, for some k, $\overrightarrow{v}$ and σ, $Res(f, t', a, k) = \overrightarrow{\lambda}\ (c(b, \sigma)\ \overrightarrow{v})$.*
2. *Let b be a branch in u. b is (t, a) useful if there is a sequence $< k_i, \sigma_i, \overrightarrow{v_i} >_{i<lg(b)}$ such that, for every i, $Res(f, t', a, k_i) = \overrightarrow{\lambda}\ (c(b \restriction i, \sigma_i)\ \overrightarrow{v_i})$; moreover the occurrence of $c(b \restriction i+1, \sigma_{i+1})$ in $Res(f, t', a, k_{i+1})$ is an immediate successor of the occurrence of $c(b \restriction i, \sigma_i)$ in $Res(f, t', a, k_i)$.*

Remarks and examples

- A context is a λ term (not a λ' term !) with some holes. As usual, in a substitution in a context some variables may be captured.
- It will be shown (see the proposition 1) that, with the notations of the previous definition, a is an f-branch in t' and thus the definition makes sense.
- Most often, either σ is the identity (i.e. u is a subterm of t) or D is an applicative context (i.e. $t = (\sigma(u)\ \overrightarrow{r})$) but it is not always the case (see the proposition 6) and I thus need this general definition. In fact both cases are essentially the same since it is not difficult to prove the following fact.
  Let $t = D(u)$ for some context D and a be a branch in t. Assume that the address nil in u is (t, a) useful, then $t \twoheadrightarrow_a \overrightarrow{\lambda}(\sigma(u)\ \overrightarrow{r})$ for some σ which is the identity except on the free variables of u that are captured by the context D.
- Let $t = (Y\ I)$. t is unsolvable and thus nil is a branch in t. 1^∞ is a branch in Y. It is easy to check that 1^∞ is (t, nil) useful.
- Note that a term t may have many subterms each of them has a branch that is (t, a) useful. For example, let $t = (Y_1\ F)\ (Y_2\ F)$ where $Y_1 = Y_2 = Y$ and $F = \lambda f \lambda g\ (g\ f)$. The following reduction shows that the branch 1^∞ in Y_1 (and similarly for Y_2) is (t, nil) useful.
  Let $f = \psi(Y)$ and $t' = (c(nil, Id)\ F\ (Y\ F))$. Remember that $f(nil) = (\{x\}, x, 1)$ and $f([1, 1, ..., 1]) = (\emptyset, x, 1)$. The f-reduction of t' is given by (where $\sigma(x) = F$) : $t' \to (F\ c([1], \sigma)\ (Y\ F)) \twoheadrightarrow (Y\ F\ c([1], \sigma)) \twoheadrightarrow$
  $(F\ (Y\ F)\ c([1], \sigma)) \twoheadrightarrow (c([1], \sigma)\ (Y\ F)) \to (F\ c([1, 1], \sigma)\ (Y\ F)) \twoheadrightarrow$ ...
- Also note that, for an infinite branch b, being (t, a) useful is stronger that simply asking that for every $i, b \restriction i$ is (t, a) useful. Let $t = (Y_1\ H\ Y_2\ 0)$ where $Y_1 = Y_2 = Y$, $H = \lambda f n p\ (u\ n\ p\ (f\ n\ (s\ p)))$, $u = \lambda n p a\ (n\ F\ (p\ F\ \lambda x\ a))$, $F = \lambda x y\ (y\ x)$, $0 = \lambda x y\ y$ and $s = \lambda n f x\ (f\ (n\ f\ x))$. For every k, the address 1^k is (t, nil) useful both in Y_2 and Y_1. The branch 1^∞ of Y_1 is (t, nil) useful but the branch 1^∞ of Y_2 is not. The reason is the following : u is a term (given by Maurey) such that $(u\ n\ p\ a) \to a$ for every Church integers $n \geq p$. Since Y may be seen as an "infinite" Church integer, $(u\ Y\ k\ a) \to a$ for every k and this computation "uses" the address $1^k$of Y. It follows that, letting $G = (Y_1\ H)$, $t = (G\ Y_2\ 0) \to (G\ Y_2\ 1) \to (G\ Y_2\ 2) \to ...$. It is easy to see that, in this computation, the node at the address 1^{k+1} of Y_1 that is used for the reduction $(G\ Y_2\ k) \to (G\ Y_2\ k+1)$ satisfies the descendent condition whereas, since the occurrence of Y_2 in $(G\ Y_2\ k+1)$ is a "new" one, the node at the address 1^{k+1} of Y_2 that is used in this reduction does not satisfy the condition.

Definition 11. *Let t be in Λ, a be a branch of t and (d_n) a sequence of λ terms. (d_n) is a decoration for (t, a) if there is a strictly increasing sequence (k_n) of integers and a sequence $(\overrightarrow{r}_n)$ such that for every $n \geq 0$*

1. *$Res(t, a, k_n) = \overrightarrow{\lambda}(d_n\ \overrightarrow{r_n})$*
2. *d_{n+1} is the descendent of an element of $\overrightarrow{r_n}$*
3. *d_n is solvable.*

Theorem 1. *Let t be in Λ and a be a branch in t. Then (t, a) has a decoration.*

Corollary 1. *Every unsolvable λ term has a decoration in the sense of [2].*

4 Proof of the theorem

4.1 Some lemmas on the f-reduction and usefulness

In this section I prove essentially two things : The notion of computation and the notion of usefulness are "transitive". Moreover in both cases the notion of descendence is preserved by this transitivity.

The first one (mainly the lemma 7) means that a computation (by left reduction) can be "partitioned" in the following way : Let u be a subterm of t. Get t' by replacing in t the subterm u by its Böhm tree. The computation of a branch a of t is the same as the computation of the branch a of t' where, when a node of $BT(u)$ appears in head position, the computation of this node is "inserted". There is a (non essential) technical difficulty showed in the following example : Assume $u \twoheadrightarrow \lambda x\ u_1 \twoheadrightarrow \lambda x\ (x\ v)$ then $(u\ r) \twoheadrightarrow (\lambda x\ u_1\ r) \rightarrow u_1[x := r] \twoheadrightarrow (r\ v[x := r])$ and the order is not exactly the same as $(u\ r) \twoheadrightarrow (\lambda x\ u_1\ r) \twoheadrightarrow (\lambda x\ (x\ v)\ r) \twoheadrightarrow (r\ v[x := r])$. This is why we have to use big steps of head reduction.

The second one is given by the proposition 2.

Lemma 2. *Let t, t' be in Λ', f be a Böhm function and a be f-accessible in t. Assume $t \twoheadrightarrow_{f,a} t'$. Then, for some $a' \leq a$, $t \twoheadrightarrow_{f,a} adr(f, t, a') \twoheadrightarrow_f t'$.*

Proof. Immediate from the definition.

Lemma 3. *Let v, v' be in Λ' and f be a Böhm function. Assume that $v \twoheadrightarrow_f v'$.*

1. *Let σ be a substitution. Then $\sigma(v) \twoheadrightarrow_f \sigma(v')$.*
2. *Let $\overrightarrow{r}$ be a sequence of terms and assume v' does not begin with λ. Then $(v\ \overrightarrow{r}) \twoheadrightarrow_f (v'\ \overrightarrow{r})$*
   Moreover in both cases the length of the f-reduction remains the same.

Proof. Note that the more general case, where v' begins with λ, is treated in the lemma 5. The proof is by induction on the length of the reduction and case analysis. Use the fact that $\sigma(u[x := v]) = \sigma(u)[x := \sigma(v)]$.

Lemma 4. *Let t be in Λ' and f be a Böhm function such that t is f-unsolvable.*

1. *Let σ be a substitution. Then $\sigma(t)$ is f-unsolvable.*
2. *Let $\overrightarrow{r}$ be a sequence of terms. Then $(t\ \overrightarrow{r})$ is f-unsolvable. Moreover $(t\ \overrightarrow{r})$ has no reduct of the form $\overrightarrow{\lambda}(r_i\ \overrightarrow{v})$ where r_i is a descendent of an element of $\overrightarrow{r}$.*

Proof. 1. This follows immediately from the lemma 3.

2. If t does not reduce to a term beginning with λ this follows immediately from the lemma 3. Otherwise let $\overrightarrow{r} = (r_1\ ...\ r_n)$ and t' be the least step where λ appears. Then (by the lemma 3) $(t\overrightarrow{r}\ ) \twoheadrightarrow_f (t'\ \overrightarrow{r}) = (\lambda x\ t_1\ \overrightarrow{r}) \twoheadrightarrow_f (\sigma(t_1)\ r_2\ ...\ r_n)$ where $\sigma(x) = r_1$. The result follows by the lemma 3 and by repeating, if necessary, the same argument.

Corollary 2. *Let t be in Λ, a be a branch of t, (d_n) be a sequence of λ terms, (k_n) be a strictly increasing sequence of integers and $(\overrightarrow{r}_n)$ be a sequence of finite sequences of λ terms. Assume that for every $n \geq 0$*

1. *$Res(t, a, k_n) = \overrightarrow{\lambda}(d_n\ \overrightarrow{r_n})$*
2. *d_{n+1} is the descendent of an element of $\overrightarrow{r_n}$*

Then (d_n) is a decoration for (t, a).

Proof. The fact that d_n is solvable follows immediately from the lemma 4.

Lemma 5. *Let v, r_1, ..., r_p be in Λ', σ be a substitution and f be a Böhm function. Assume that $v \twoheadrightarrow_f \lambda x_1 ... \lambda x_k\ (u\ \overrightarrow{t})$. Then $(\sigma(v)\ r_1\ ...\ r_p) \twoheadrightarrow_f \lambda x_{j+1} ... \lambda x_k\ (\sigma'(u)\ \overrightarrow{\sigma'(t)}\ r_{j+1}\ ...\ r_p)$ where $j = Min(k, p)$, $\sigma' = \tau \circ \sigma$ and τ is given by $\tau(x_i) = r_i$ for $1 \leq i \leq j$.*

Proof. By induction on k. The case $k = 0$ is given by the lemma 3. Assume $k \geq 1$. Look at the least step in the reduction $v \twoheadrightarrow_f v'$ where v' begins with λ, say $v' = \lambda x_1\ v_1$. Then, we have the following sequence of f-reductions : $(\sigma(v)\ r_1\ ...\ r_p) \twoheadrightarrow_f (\lambda x_1\ \sigma(v_1)\ r_1\ ...\ r_p) \rightarrow_f (\sigma_1(v_1)\ r_2\ ...\ r_p) \twoheadrightarrow_f \lambda x_{j+1}\ ...\ \lambda x_k\ (\sigma'(u)\ \overrightarrow{\sigma'(t)}\ r_{j+1}\ ...\ r_p)$ where $\sigma_1 = \tau \circ \sigma$ and τ is given by : $\tau(x_1) = r_1$. The first $\twoheadrightarrow_f$ is given by the lemma 3 and the last $\twoheadrightarrow_f$ is given by the induction hypothesis.

Lemma 6. *Let t, u be in Λ', g be a Böhm function and $f = \psi(g, u)$. Assume $t = \overrightarrow{\lambda}\ (R\ r_1\ ...\ r_p)$ and t' is the f-reduct of t. Then*

1. *if $R = x$, then t[g, u] is in g-head normal form.*
2. *if $R = (\lambda x\ v\ w)$ or $\perp$, then the g-reduct of t[g, u] is t'[g, u].*
3. *if $R = c(a, \sigma)$*
   - *If $f(a) = \perp$, then t[g, u] is not g-solvable.*
   - *If $f(a) = (\{x_1, ..., x_k\}, x, q)$ then t[g, u] $\twoheadrightarrow_g$ t'[g, u].*

Proof. (1) and (2) are clear. (3.1) follows from the lemma 4 and (3.2) follows from the lemma 5.

Lemma 7. *Let t, u be in Λ', g be a Böhm function, $f = \psi(g,u)$ and a be f-accessible in t. Assume $t \twoheadrightarrow_{f,a} t' = \overrightarrow{\lambda}(R\ \overrightarrow{s})$ and R = either x or $(\lambda x\ v\ w)$ or $c(b,\sigma)$ and f(b) $\neq \perp$. Then, t[g, u] $\twoheadrightarrow_{g,a}$ t'[g, u]. Moreover, let d' be a subterm of t' that is a residue (resp a descendent) of a subterm d of t. Then d'[g, u] is a residue (resp a descendent) of the corresponding subterm d[g, u].*

Proof. By induction on the length of the reduction of t. For $a = nil$ this follows from the lemma 6. If $a = [i :: b]$, then $t \twoheadrightarrow_f \overrightarrow{\lambda}(x\ t_1\ ...\ t_n)$. By the lemma 6, $t[g,u] \twoheadrightarrow_g \overrightarrow{\lambda}(x\ t_1[g,u]\ ...\ t_n[g,u]) \twoheadrightarrow_{g,a} t_i[g,u]$ and the result follows easily by induction on the length of a.

Proposition 1. *Let t, u be in Λ', g be a Böhm function and $f = \psi(g,u)$. Let a be in A. Then a is an f-branch in t iff a is a g-branch in t[g, u].*

Proof. It follows immediately from the lemma 6 that t has an f-head normal form iff $t[g,u]$ has a g-head normal form. Moreover if $hnf(f,t,nil) = \lambda x_1 ... \lambda x_k\ (x\ t_1 ...\ t_p)$ then $hnf(g,t[g,u],nil) = \lambda x_1 ... \lambda x_k\ (x\ t_1[g,u]\ ...\ t_p[g,u])$. The result follows easily.

Definition 12. *Let σ, σ' be substitutions. $\tau = \sigma \oplus \sigma'$ if for every variable x*

- *if $\sigma(x) \neq x$ then $\tau(x) = \sigma(x)$ and $\sigma'(x) = x$*
- *if $\sigma'(x) \neq x$ then $\tau(x) = \sigma'(x)$ and $\sigma(x) = x$*
- *otherwise $\tau(x) = x$*

Definition 13. *Let u be in Λ. Define, for a accessible in u, FV(u, a) by :*

- *FV(u, nil) = $\emptyset$*
- *FV(u, [a :: i]) = Fv(u, a) $\cup \{x_1 ... x_k\}$ where hnf(u, a) = $\lambda x_1 ... x_k\ (x\ \overrightarrow{r})$*

Lemma 8. *1. Let $t = (\sigma(u)\ \overrightarrow{r})$ be in Λ, $t' = (c(nil, \sigma)\ \overrightarrow{r})$, b be accessible in t, $f = \psi(u)$, $t' \twoheadrightarrow_{f,b} t"$ and $c(a, \tau)$ be a subterm of t". Then $\tau = \sigma \oplus \sigma'$ for some σ' whose domain is included in FV(u,a). Moreover, for every variable y in the domain of τ, for every a' > a and every x in FV(u, a') - FV(u, a), x is not free in $\tau(y)$.*

2. *Similarly for $t = D(\sigma(u))$ with $\tau = \sigma \oplus \sigma" \oplus \sigma'$ where the domain of $\sigma"$ is included in the set of variables captured by the context D.*
3. *Moreover if $c(a', \tau')$ is a descendent of $c(a, \tau)$ then $\tau' = \tau \oplus \mu$ for some μ whose domain is included in FV(u, a') - FV(u, a)*

Proof. This comes immediately from the fact that we are doing head reduction (and of course the renaming rule to avoid capture). More precisely, this is proved by induction on the length of the reduction $t' \twoheadrightarrow_{f,b} t"$ by a simple case analysis.

Lemma 9. *Let $t = (\sigma(u)\ \overrightarrow{r})$ be in Λ, b be a branch in t and $f = \psi(u)$. Let t' = $(c(nil, \sigma)\ \overrightarrow{r})$.*

1. *Assume* $t' \twoheadrightarrow_{f,b} \overrightarrow{\lambda}\ (c(a,\tau)\ \overrightarrow{s})$ *and* $u \twoheadrightarrow_a adr(u,a) \twoheadrightarrow \lambda x_1 ... \lambda x_k\ (d\ \overrightarrow{v}) \twoheadrightarrow \lambda x_1 ... \lambda x_k\ ...\ \lambda x_{k+k'}\ (d'\ \overrightarrow{v'})$ *and* d' *is the descendent of an element of* $\overrightarrow{v}$. *Then* $t \twoheadrightarrow_b \overrightarrow{\lambda}(\mu(d)\ \mu(\overrightarrow{v})\ \overrightarrow{w}) \twoheadrightarrow_b \overrightarrow{\lambda'}\ (\mu'(d')\ \mu'(\overrightarrow{v'})\ \overrightarrow{w'})$ *and* $\mu'(d') = \mu(d')$ *is a descendent of the corresponding element of* $\mu(\overrightarrow{v})$.
2. *Similarly assume that :*
   - $t' \twoheadrightarrow_{f,b} \overrightarrow{\lambda}\ (c(a,\tau)\ \overrightarrow{s}) \twoheadrightarrow_{f,b} \overrightarrow{\lambda'}\ (c(a',\tau')\ \overrightarrow{s'})$ *for some* $a < a'$ *and* $c(a',\tau')$ *is a successor of* $c(a,\tau)$.
   - $u \twoheadrightarrow_{a'} adr(u,a) \twoheadrightarrow \overrightarrow{\lambda}(d\ \overrightarrow{v}) \twoheadrightarrow_{a'} adr(u,a') \twoheadrightarrow \overrightarrow{\lambda}(d'\ \overrightarrow{v'})$ *and* d' *is the descendent of an element of* $\overrightarrow{v}$.

Then $t \twoheadrightarrow_b \overrightarrow{\lambda}(\mu(d)\ \mu(\overrightarrow{v})\ \overrightarrow{w}) \twoheadrightarrow_b \overrightarrow{\lambda}(\mu'(d')\ \mu'(\overrightarrow{v'})\ \overrightarrow{w'})$ *and* $\mu'(d') = \mu(d')$ *is a descendent of the corresponding element of* $\mu(\overrightarrow{v})$.

Proof. 1. By the lemma 8, $\tau = \sigma \oplus \sigma_1$. By the lemma 7, $t \twoheadrightarrow_b \overrightarrow{\lambda}(\tau(adr(u,a))\ \overrightarrow{s[u]})$ and, by the lemma 5, $\overrightarrow{\lambda}(\tau(adr(u,a))\ \overrightarrow{s[u]}) \twoheadrightarrow \overrightarrow{\lambda}(\mu(d)\ \mu(\overrightarrow{v})\ \overrightarrow{w}) \twoheadrightarrow \overrightarrow{\lambda'}(\mu'(d')\ \mu'(\overrightarrow{v'})\ \overrightarrow{w'})$ where $\mu = \sigma' \circ \tau$ (resp $\mu' = \sigma" \circ \tau$) and the domain of σ' (resp $\sigma"$) is included in $\{x_1\ ...\ x_k\}$ (resp $\{x_1\ ...\ x_{k+k'}\}$). By the lemma 8, $\mu = \tau \oplus \sigma'$ and $\mu' = \tau \oplus \sigma"$. Since d' is the descendent of an element of $\overrightarrow{v}$ the variables $x_{k+1}\ ...\ x_{k+k'}$ do not appear in d' and $\mu(d') = \mu'(d')$.

2. Similarly $t \twoheadrightarrow_b \overrightarrow{\lambda}(\mu(d)\ \mu(\overrightarrow{v})\ \overrightarrow{w}) \twoheadrightarrow_b \overrightarrow{\lambda}(\mu'(d')\ \mu'(\overrightarrow{v'})\ \overrightarrow{w'})$ where $\mu = \tau \oplus \sigma'$, $\mu' = \mu \oplus \sigma"$and the domain of $\sigma"$ is included in $FV(u,a') - FV(u,a)$. Since d' is the descendent of an element of $\overrightarrow{v}$, d' has no free variables in $FV(u,a') - FV(u,a)$ and thus $\mu'(d') = \mu(d')$.

Proposition 2. *Let t, u, v be in* Λ, *a (resp b, c) be a branch in t (resp in u, v). Assume that b is (t, a) useful and c is (u, b) useful. Then c is (t, a) useful.*

Proof. Let $t = D(\sigma(u))$, $u = E(\tau(v))$. Let $t' = D(c(nil,\sigma)), u' = E(c(nil,\tau))$. Let $F = D(\sigma(E))$. Then $t = F(\sigma \circ \tau(v))$. Let $t"$ $= F(c(nil, \sigma \circ \tau))$. I only prove $t" \twoheadrightarrow_{g,a} \overrightarrow{\lambda}(c(c \restriction j, \tau_j)\ \overrightarrow{r_j})$ for every $j < lg(c)$, where $g = \psi(v)$. I should prove a bit more, namely that the corresponding $c(c \restriction j, \tau_j)$ are in the immediate successor relation (see the definition 10). This is rather tedious to write but this follows immediately from the proof.

Let $f = \psi(u)$ and $d = c \restriction j$. Since c is (u,b) useful, $u' \twoheadrightarrow_{g,b} \overrightarrow{\lambda}(c(d,\tau')\ \overrightarrow{r'})$. Thus, by the lemma 2, $u' \twoheadrightarrow_{g,b} adr(g,u',b') \twoheadrightarrow_g \overrightarrow{\lambda}(c(d,\tau')\ \overrightarrow{r'})$ for some $b' \leq b$. Since b is (t,a) useful, $t' \twoheadrightarrow_{f,a} \overrightarrow{\lambda}(c(b',\sigma')\ \overrightarrow{s'})$. Clearly $t" = t'[g,u']$. Thus, by the lemmas 7 and 5 , $t" \twoheadrightarrow_{g,b} \overrightarrow{\lambda}(\sigma'(adr(u',b')\ \overrightarrow{s'}) \twoheadrightarrow_{g,b} \overrightarrow{\lambda}(c(d,\tau")\ \overrightarrow{r'})$.

Proposition 3. *Let* $t = (\sigma(u)\ \overrightarrow{r'})$ *be in* Λ *and b be a branch in t. Let a be a branch in u that is (t, b) useful. Assume that Res(u, a, k) =* $\overrightarrow{\lambda}(u_1\ \overrightarrow{v_1})$. *Then,*

- *For some j and some* τ, *Res(t, b, j) =* $\overrightarrow{\lambda}(\tau(u_1)\ \tau(\overrightarrow{v_1})\ \overrightarrow{w})$.
- *Let c be a branch in* u_1 *that is (Res(u, a, k), Br(u, a, k)) useful. Then c is (Res(t, b, j), Br(t, b, j)) useful.*

Proof. By the lemma 2, $u \twoheadrightarrow_a adr(u,a_1) \twoheadrightarrow \overrightarrow{\lambda}(u_1\ \overrightarrow{v_1}) = u'$. Let $t' = (c(nil,\sigma),\ \overrightarrow{r})$ and $f = \psi(u)$. Since a is (t,b) useful $t' \twoheadrightarrow_{f,b} \overrightarrow{\lambda}(c(a_1,\sigma_1)\ \overrightarrow{s})$. Thus $t \twoheadrightarrow_b \overrightarrow{\lambda}(\sigma_1(adr(u,a_1))\ \overrightarrow{s}) \twoheadrightarrow_b \overrightarrow{\lambda}(\tau(u_1)\ \tau(\overrightarrow{v_1})\ \overrightarrow{w}) = Res(t,b,j) = t$". Let $a' = Br(u,a,k)$ and $b" = Br(t,b,j)$. Since a is (t,b) useful, it is clear that a' is $(t",b")$ useful and since c is (u',a') useful, by the proposition 2, c is $(t",b")$ useful.

4.2 The key results

The propositions 5 and 6 give the key points mentioned in the section 2. Intuitively the proposition 6 gives the next step of the decoration and the proposition 7 is the technical result that allows to iterate the construction.

Proposition 4. *Let u be in Λ. Assume that u is unsolvable and (d_k) is a decoration for (u, nil).*

1. *Let σ be a substitution. Then $(\sigma(d_k))$ is a decoration for $(\sigma(u), nil)$.*
2. *Let $t = (u\ \overrightarrow{r})$. Then there is a sequence (σ_k) of substitutions such that $(\sigma_k(d_k))$ is a decoration for (t, nil).*

Proof. The first case is trivial since, by the lemma 3, if $u \twoheadrightarrow u'$ then $\sigma(u) \twoheadrightarrow \sigma(u')$. For the second case let p be the length of $\overrightarrow{r}$. If $p = 0$, this is trivial. Assume $p \geq 1$. If, for every $k, Res(u, nil, k)$ does not begin with λ the result follows from the lemma 3. Otherwise, let k be the least integer such that $Res(u, nil, k) = \lambda x\ u'$. Since (d_k) is a decoration for (u, nil), let (k_n) be the sequence such that $Res(u, nil, k_n) = \overrightarrow{\lambda}(d_n\ \overrightarrow{v_n})$.

Assume first that $k_0 > k$. Then (by the lemma 3) $(u\overrightarrow{r}) \twoheadrightarrow (\lambda x\ u'\ \overrightarrow{r}) \twoheadrightarrow (\sigma(u')\ r_2\ ..\ r_p)$ where $\sigma(x) = r_1$. Repeating the same argument with $(\sigma(u')\ r_2\ r_p)$ yields the result.

Assume that $k_0 \leq k$. Let n_0 be the largest integer such that $k_{n_0} \leq k$. Then (by the lemma 3) for $n \leq n_0$ $Res(t, nil, k_n) = (d_n\ \overrightarrow{v}_n\ \overrightarrow{r})$. $Res(t, nil, k_{n_0}) \twoheadrightarrow (\lambda x\ u'\ \overrightarrow{r}) \twoheadrightarrow (\sigma(u')\ r_2\ ...\ r_p)$ where $\sigma(x) = r_1$. Since $(d_n)_{n>n_0}$ is a decoration for $(u', nil), (\sigma(d_n))_{n>n_0}$ is a decoration for $(\sigma(u'), nil)$. Since d_{n_0+1} is a descendent of an element of v_{n_0}, x is not free in d_{n_0+1}. Repeating the same argument with $((\sigma(u')\ r_2\ ...\ r_p)\ , nil)$ yields the result.

Proposition 5. *Let t, u be in Λ and b (resp a) be a branch in t (resp u). Assume a is (t, b) useful and let (d_k) be a decoration for (u, a). Then there is a sequence (σ_k) of substitutions such that $(\sigma_k(d_k))$ is a decoration for (t, b).*

Proof. - If a is infinite, the sequence (σ_k) is easily constructed by using the lemma 9.

- If a is finite the sequence (σ_k) is easily constructed by using the lemma 9 for the finite part of the branch and the proposition 4 for its last node.

Proposition 6. *Let $t = (u\ r_1\ ...\ r_n)$ be a λ term and a be a branch in t. Assume there is no branch neither in u nor in any r_i that is (t, a) useful. Then there is $< i, k, u_1, \nu >$ such that, letting $t' = Res(t, a, k)$ and $a' = Br(t, a, k)$:*

- $t' = \overrightarrow{\lambda}\ (\nu(u_1)\ \overrightarrow{v})$ *for some* $\overrightarrow{v}$,
- $u_1 = (r_i\ s_1\ ...\ s_m)$ *and* $\nu(r_i) = r_i$ *is a descendent of its occurrence in* t.
- *For* $1 \leq j \leq m$, s_j *has no branch that is* (t', a') *useful*
- u_1 *has a branch that is* (t', a') *useful.*

Comments The intuition of the proof is the following : Since there is no useful branch in u the set of useful nodes in $BT(u)$ is (by König's lemma) finite. Assume, for example, that $t = (\lambda x \lambda y\ (x\ s_1\ s_2)\ r_1\ r_2)$. Then $t \twoheadrightarrow (r_1\ s'_1\ s'_2)$. If there is no useful branch neither in s'_1 nor in s'_2 we are done. Otherwise there is such a useful branch in, say, s'_1. Thus $t \twoheadrightarrow \overrightarrow{\lambda}(s'_1\ \overrightarrow{w})$ for some $\overrightarrow{w}$. By the lemmas of the section 4.1 it is mainly enough to prove the result for s'_1. But $t' = (\lambda x \lambda y\ s_1\ r_1\ r_2) \twoheadrightarrow s'_1$ and the cardinality of the set of useful nodes of t' is smaller than the one of t. We get the result by repeating the previous argument.

Before giving the proof I give an example of the difficult case (the case 2.b in the proof). This is the example 4.3.6 in [1]. Let $w = \lambda xyz\ (y\ (x\ (z\ x)))$, $R = \lambda z\ (z\ I\ w)$ and $t = (w\ R\ I\ w)$. t is unsolvable. w, R, I are normal and so they do not have a branch that is (t, nil) useful. $t \twoheadrightarrow (I\ (R\ (w\ R))) \twoheadrightarrow (R\ (w\ R))$. We cannot choose the step $(I\ (R\ (w\ R)))$ and the argument I as the first element of the decoration for t since the unsolvability is already created (and "used") in $(R\ (w\ R))$. We will choose the next step $(R\ (w\ R))$ and the argument R because, at this step, the unsolvability is not yet created since R and $(w\ R)$ are solvable. Thus, here, the solution is : $k = 4, u_1 = (R\ (w\ R))$, $i = 1, \nu = Id$ and $\overrightarrow{v}$ is empty.

Proof. Let $E = \{b\ /\ b$ is an address accessible in u, that is (t, a) useful$\}$. Note that for b in $E, hnf(u, b) \neq \perp$ because otherwise b would be a branch in u that is (t, a) useful.

I define a procedure to construct the desired $< i, k, u_1, \nu >$ and a branch in u. This procedure halts (and I thus get the result) because otherwise this means we always are in the case (1) below and this procedure has constructed an infinite branch in u that is (t, a) useful and this is a contradiction. Note that I cannot use the fact that E is finite (and prove the result by induction on the cardinality of E). Intuitively this is actually the argument used but we cannot formalize it in this way. If E is infinite, by König's lemma, there is an infinite branch b such that for every i, $b \restriction i \in E$ but (see the example after the definition 10) this does not imply that b is (t, a) useful.

nil clearly is in E. Let $hnf(u, nil) = \lambda x_1\ ...\ x_k\ (x\ w_1\ ...\ w_p)$, $j_0 = Min(k, n)$ and σ is given by $\sigma(x_j) = r_j$ for $j \leq j_0$. It is clear that $j_0 \geq 1$ because otherwise t reduces to $\overrightarrow{\lambda}(x\ \overrightarrow{w}\ \overrightarrow{r})$ and then u or some r_i would have a branch that is (t, a) useful.

1) Assume first that $x \notin \{x_1\ ...\ x_k\}$. Then $t \twoheadrightarrow \lambda x_{j_0+1}\ ...\ x_k\ (x\ \ \sigma(w_1)\ ...\ \sigma(w_p)\ r_{j_0+1}\ ...\ r_n)$ and thus $a \neq nil$. Let $a = [i :: l]$. If $i > p$, there is a branch in r_i that is (t, a) useful and this contradicts the hypothesis. Thus $i \leq p$. Let $u' = \lambda x_1\ ...\ x_{j_0}\ w_i$. Then $t \twoheadrightarrow_a \sigma(w_i)$ and $(u'\ \ r_1\ ...\ r_n) \twoheadrightarrow \sigma(w_i)$. The first node of the branch constructed by the procedure is i. Repeat the procedure (to get the other nodes) with $(u'\ \ r_1\ ...\ r_n)$.

2) Assume that $x = x_i$. Then $t \twoheadrightarrow \lambda x_{j_0+1} .. x_k\ (r_i\ \ \sigma(w_1) .. \sigma(w_p)\ r_{j_0+1} .. r_n)$.

a) Assume first that for $1 \leq q \leq p$, $\sigma(w_q)$ has no branch that is (t, a) useful. Then $< i, j_0, u_1, Id >$ where $u_1 = (r_i\ \ \sigma(w_1) \ldots \sigma(w_p)\ r_{j_0} \ldots r_n)$ clearly satisfies the conclusion of the proposition.

b) Assume that, for some $1 \leq q \leq p$, $\sigma(w_q)$ has a branch that is (t, a) useful.

Claim

There is b in E and $j \leq j_0$ such that $hnf(u, b) = \overrightarrow{\lambda}(x_j\ s_1 \ldots s_l)$ and $\sigma(hnf(u, b))$ has a branch that is (t, a) useful but no $\sigma(s_m)$ has such a branch.
Proof

Note that $adr(u, [q]) = w_q$. By the hypothesis, $[q]$ is in E. Let $hnf(u, [q]) = \overrightarrow{\lambda}(y\ s_1 \ldots s_l)$. If $y = x_j$ and no $\sigma(s_m)$ has a branch that is (t, a) useful, $b = [q]$ satisfies the conclusion of the claim. Otherwise some $\sigma(s_m)$ has a branch that is (t, a) useful. (*Proof :* If $y = x_j$ this is clear. If $y \notin \{x_1 \ldots x_k\}$, $\sigma(hnf(u, [q])) = \overrightarrow{\lambda}(y\ \sigma(s_1) \ldots \sigma(s_l))$ and this is again clear since a branch in $\sigma(hnf(u, [q]))$ is a branch in some $\sigma(s_m)$). We may repeat the argument with $b = [q :: m]$. If the claim fails we get in this way an infinite branch in u that is (t, a) useful. (Q.E.D. of the claim)

Let (b, j) be given by the claim. Let $t' = (c(nil, Id)\ r_1 \ldots r_n)$ and $f = \psi(u)$. $t' \twoheadrightarrow_{f,a} \overrightarrow{\lambda}(c(b, \tau)\ \overrightarrow{w})$ for some $\tau = \sigma \oplus \sigma'$ and thus $t \twoheadrightarrow_a \overrightarrow{\lambda}(\tau(adr(u, b))\ \overrightarrow{w})$. By the lemmas 5 and 8, there is a substitution τ' such that $\overrightarrow{\lambda}(\tau(adr(u, b))\ \overrightarrow{w}) \twoheadrightarrow \overrightarrow{\lambda}(\mu(x_j)\ \mu(\overrightarrow{s})\ \overrightarrow{v}) = Res(t, a, k)$ where $\mu = \tau \oplus \tau' = \sigma \oplus \sigma' \oplus \tau'$. Then, $< j, k, u_1,\ \sigma' \oplus \tau' >$ satisfies the conclusion of the proposition, where $u_1 = (r_j\ \overrightarrow{\sigma(s)}) = \sigma((x_j\ s_1 \ldots\ \ s_l))$.

Proposition 7. *Let $(d_n)_{n\geq 0}$ (resp. $(\overrightarrow{u_n})_{n\geq 0}$, $(\overrightarrow{v_n})_{n\geq 1}$, resp. $(a_n)_{n\geq 0}$, resp. $(\sigma_n)_{n\geq 1}$) be a sequence of λ terms (resp. be sequences of finite sequences of λ terms, resp. be a sequence of elements of A, resp. be a sequence of substitution). Assume that for every $n \geq 0$*

- *$t_n = (d_n\ \overrightarrow{u_n})$ and a_n is a branch in t_n.*
- *For some k_n, $Res(t_n, a_n, k_n) = \overrightarrow{\lambda_n}(\sigma_{n+1}(t_{n+1})\ \ \overrightarrow{v}_{n+1})$ and a_{n+1} is $(Res(t_n, a_n, k_n),\ Br(t_n, a_n, k_n))$ useful.*
- *d_{n+1} is the descendent of an element of the sequence $\overrightarrow{u_n}$*
- *$\sigma_{n+1}(d_{n+1}) = d_{n+1}$.*

  Then, there is an increasing sequence (τ_n) of substitutions such that the sequence $(\tau_n(d_n))$ is a decoration for $(t_0,\ a_0)$.

Proof. I construct (by induction on n) a sequence $< j_n, r_n, b_n, \tau_n >$ such that : $r_0 = t_0, j_0 = 0, \tau_0 = Id, b_0 = a_0$ and, for $n \geq 1$, $r_n = Res(r_0, b_0, j_n) = \overrightarrow{\lambda}(\tau_n(t_n)\ \overrightarrow{w_n}), b_n = Br(r_0, b_0, j_n), \tau_n(d_n) = \tau_{n-1}(d_n)$ and a_n is (r_n, b_n) useful. It is clear that the sequence (τ_n) satisfies the conclusion.

$t_n \twoheadrightarrow_{a_n} \overrightarrow{\lambda_n}(\sigma_{n+1}(t_{n+1})\ \overrightarrow{v_{n+1}})$. Since a_n is (r_n, b_n) useful and by the proposition 3, $r_n \twoheadrightarrow_{b_n} r'_n = \overrightarrow{\lambda}(\overrightarrow{\lambda}_n\ (\tau_n(\sigma_{n+1}(t_{n+1}))\ \tau_n(\overrightarrow{v_{n+1}}))\ \ \overrightarrow{w_n})$ for some τ_n and $\overrightarrow{w_n}$.

Clearly $r'_n \twoheadrightarrow \overrightarrow{\lambda'}\ (\tau_{n+1}(t_{n+1})\ \overrightarrow{w_{n+1}}) = Res(r_0, a_0, j_{n+1})$ for some $\overrightarrow{w_{n+1}}$ where $\tau_{n+1} = \tau_n \circ \sigma_{n+1} \oplus \mu_n$ and the domain of μ_n is included in the variables in $\overrightarrow{\lambda}_n$. Since d_{n+1} is the descendent of an element of $\overrightarrow{u_n}$, d_{n+1} is not affected by μ_n . Since, by the hypothesis, $\sigma_{n+1}(d_{n+1}) = d_{n+1}$, we have $\tau_{n+1}(d_{n+1}) = \tau_n(d_{n+1})$. Finally, again by the proposition 3, a_{n+1} is $(r_{n+1},\ b_{n+1})$ useful.

4.3 End of the proof of the theorem

Let t be a λ term and a be branch in t. The existence of a decoration is proved by induction on the complexity of t.

- If $t = \lambda x\ u$ or $t = (x\overrightarrow{r})$ the result follows immediately from the induction hypothesis.
- If $t = (u\ r_1 ...\ r_n)$ and there is, either in u or in some r_i, a branch that is (t, a) useful. For example, say b is such a branch in u. By the induction hypothesis there is a decoration of (u, b) and by the proposition 5 there is a decoration for (t, a).
- Otherwise $t = (u\ r_1 ...\ r_n)$ and there is no branch neither in u nor in any r_i that is (t, a) useful. Let $a_0 = a, d_0 = u$, $\overrightarrow{u_0} = r_1\ ...\ r_n, t_0 = (d_0\ \overrightarrow{u_0})$ and $\overrightarrow{v_0}$ be the empty sequence. By the proposition 6 there is $< i, k_0, t_1, \sigma >$ such that, letting $t' = Res(t_0, a_0, k_0)$ and $a' = Br(t_0, a_0, k_0)$:
  - $t' = \overrightarrow{\lambda}\ (\sigma(t_1)\quad \overrightarrow{v_1})$, $t_1 = (r_i\ s_1\ ...\ s_m), \sigma(r_i) = r_i$ for some terms $s_1\ ...\ s_m\ \overrightarrow{v_1}$ and some substitution σ.
  - For $1 \leq j \leq m$, s_j has no branch that is (t', a') useful
  - t_1 has a branch a_1 that is (t', a') useful.

Let $d_1 = r_i$ and $\overrightarrow{u_1} = s_1\ ...\ s_m$. No s_j has a branch that is (t_1, a_1) useful since, otherwise, by the proposition 2 such a branch would be (t', a') useful. We may again use the proposition 6 with t_1 and the branch a_1. By repeating the same argument we get sequences satisfying the hypothesis of the proposition 7 and thus a decoration for t.

References

1. R. Kerth *"Isomorphisme et équivalence équationnelle entre modèles du λ calcul"* Ph.D. thesis Université Paris 7, 1995.
2. R. Kerth *"The interpretation of Unsolvable λ Terms in Models of Untyped λ Calculus"*. To appear in the JSL
3. R. Kerth *"On the Construction of Stable Models of Untyped λ Calculus"*. To appear in TCS
4. R. David & K. Nour *"Storage operators and directed λ calculus"*. JSL 60, n°4, 1054-1086, 1995.