STORAGE OPERATORS AND DIRECTED LAMBDA-CALCULUS

René David, Karim Nour

STORAGE OPERATORS
AND
DIRECTED LAMBDA-CALCULUS

René DAVID & Karim NOUR

LAMA - Equipe de Logique

Université de Savoie - Campus Scientifique

73376 Le Bourget du Lac cédex

email [david,nour]@univ-savoie.fr

Abstract

Storage operators have been introduced by J.L. Krivine in [5] ; they are closed l-terms which, for a data type, allow to simulate a "call by value" while using the "call by name" strategy. In this paper, we introduce the directed l-calculus and show that it has the usual properties of the ordinary l-calculus. With this calculus we get an equivalent - and simple - definition of the storage operators that allows to show some of their properties :

- the stability of the set of storage operators under the b-equivalence (theorem 5.1.1) ;
- the undecidability (and its semi-decidability) of the problem "is a closed l-term t a storage operator for a finite set of closed normal l-terms ? " (theorems 5.2.2 and 5.2.3) ;
- the existence of storage operators for every finite set of closed normal l-terms (theorem 5.4.3) ;
- the computation time of the "storage operation" (theorem 5.5.2).

Résumé

Les opérateurs de mise en mémoire ont été introduits par J.L. Krivine dans [5] ; il s'agit de l-termes clos qui, pour un type de données, permettent de simuler "l'appel par nom" dans le cadre de "l'appel par valeur". Dans cet article, nous introduisons le l-calcul dirigé et nous démontrons qu'il garde les propriétés usuelles du l-calcul ordinaire. Avec ce calcul nous obtenons une définition équivalente - et simple - pour les opérateurs de mise en mémoire qui permet de prouver plusieurs de leurs propriétés :

- la stabilité de l'ensemble des opérateurs de mise en mémoire par la b-équivalence (théorème 5.1.1) ;
- l'indécidabilité (et sa semi-décidabilité) du problème "un terme clos t est il un opérateur de mise en mémoire pour un ensemble fini de termes normaux clos ? " (théorèmes 5.2.2 et 5.2.3) ;
- l'existence d'opérateurs de mise en mémoire pour chaque ensemble fini de termes normaux clos (théorème 5.4.3) ;
- une inégalité controlant le temps calcul d'un opérateur de mise en mémoire (théorème

5.5.2).

Summary

§ 0. Introduction

<u>0.1</u> Lambda-calculus as such is not a computational model. A reduction strategy is needed. In this paper, we consider l-calculus with the left reduction (iteration of the head reduction denoted by ∑). This strategy has some advantages : it always terminates when applied to a normalizable l-term, and it seems more economic since we compute a l-term only when we need it. But the major drawback of this strategy is that a function must compute its argument every time it uses it. This is the reason why this strategy is not really used. We would like a solution to this problem.

Let F be a l-term, D a set of closed normal l-terms, and tAD. During the computation, by left reduction, of $(F)h_t$ (where $h_t{:}_b t$), h_t may be computed several times (as many times as F uses it). We would like to transform (F)h_t to (F)t. We also want that this transformation depends only on h_t (and not F). In other words we look for some closed l-term T which satisfies the following propreties :

- For every F, and for every tAD, (T)h_tF∑(F)t ;
- The computation time of (T)h_tF depends only on h_t.

Definition (temporary) :

A closed l-term T is called a **storage operator** for D if and only if for every tAD, and for every $h_t:_b$t, (T)h_tf∑(f)t (where f is a new variable).

It is clear that a storage operator satisfies the required properties. Indeed,

- Since we have (T)h_tf∑(f)t, then the variable f never comes in head position during the reduction, we cmay then replace f by any l-term.
- The computation time (T)h_tF depends only on h_t.

K. Nour has shown (see [9]) that it is not always possible to get a normal form (it is the case for the set of Church integers). We then change the definition.

Definition (temporary) :

A closed l-term T is called **storage operator** for D if and only if for every tAD, there is a closed l-term $t_t:_b$t, such that for every $h_t:_b$t, (T)h_tf∑(f)t_t (where f is a new variable).

J.L. Krivine has shown that, by using Gödel translation from classical to intuitionitic logic, we can find, for every data type, a very simple type for the storage operators. But the l-term t_t obtained may contain variables substituted by l-terms depending on h_t. Since the l-term t_t is by-equivalent to a closed l-term, the left reduction of $t_t[u_1/x_1,…,u_n/x_n]$ is equivalent to the left reduction of t_t, the l-terms $u_1,…,u_n$ will therefore never be evaluated during the reduction. We then modify again the definition.

Definition (final) :

A closed l-term T is called a **storage operator** for D if and only if for every tAD, there is a l-term $t_t:_{by}$t, such that for every $h_t:_b$t, there is a substitution s, such that (T)h_tf∑(f)s(t_t) (where f is a new variable).

In the case where t_t=t, we say that T is a **strong storage operator**, and in the case where t_t is closed, we say that T is a **proper storage operator**. These special operators are studied in [9] and [12].

The previous definition is not well adapted to study these operators. Indeed, it is, a priori, a Pstatement (Vt Et$_t$ Vh$_t$ Es A(T,t,t$_t$,h$_t$,s)). We will show that it is in fact equivalent to Pstatement (t$_t$ can be computed from t, and s from h$_t$).

We now describe the intuitive meaning of the directed lambda calculus.

0.2 Consider the particular case of the set $\underline{N}$ of Church integers.
A closed l-term T is a storage operator for $\underline{N}$ if and only if for every n≥0, there is a l-term $t_n:_{by}\underline{n}$, such that for every $h_n:_b\underline{n}$, there is a substitution s, such that $(T)h_nf∑(f)s(t_n)$.

Let's analyse the head reduction $(T)h_nf∑(f)s(t_n)$, by replacing each l-term which comes from h_n by a new variable.
If $h_n:_b\underline{n}$, then $h_n∑lglx(g)t_{n-1}$, $t_{n-k}∑(g)t_{n-k-1}$ $1\leq k\leq n\text{-}1$, $t_0∑x$, and $t_k:_b(g)^kx$ $0\leq k\leq n\text{-}1$.
Let u_n be a new variable (u_n represents h_n). $(T)u_nf$ is solvable, and its head normal form does not begin by l, therefore it is a variable applied to some arguments. The free variables of $(T)u_n$ f are u_n and f, we then have two possibilities for its head normal form :
(f)d (in this case we stop) or $(u_n)a_1\ldots a_m$.

Assume we obtain $(u_n)a_1\ldots a_m$. The variable u_n represents h_n, and $h_n∑lglx(g)t_{n-1}$, therefore $(h_n)a_1\ldots a_m$ and $((a_1)t_{n-1}[a_1/g,a_2/x])a_3\ldots a_m$ have the same head normal form. The l-term $t_{n-1}[a_1/g,a_2/x]$ comes from h_n. Let $u_{n-1,a1,a2}$ be a new variable ($u_{n-1,a1,a2}$ represents $t_{n-1}[a_1/g,a_2/x]$). The l-term $((a_1)u_{n-1,a1,a2})a_3\ldots a_m$ is solvable, and its head normal form does not begin by l, therefore it is a variable applied to some arguments. The free variables of $((a_1)u_{n-1,a1,a2})a_3\ldots a_m$ are among $u_{n-1,a1,a2}$, u_n, and f, we then have three possibilities for its head normal form :
(f)d (in this case we stop) or $(u_n)b_1\ldots b_r$ or $(u_{n-1,a1,a2})b_1\ldots b_r$.

Assume we obtain $(u_{n-1,a1,a2})b_1\ldots b_r$. The variable $u_{n-1,a1,a2}$ represents $t_{n-1}[a_1/g,a_2/x]$, and $t_{n-1}∑(g)t_{n-2}$, therefore $(t_{n-1}[a_1/g,a_2/x])b_1\ldots b_r$ and $((a_1)t_{n-2}[a_1/g,a_2/x])b_1\ldots b_r$ have the same head normal form. The l-term $t_{n-1}[a_1/g,a_2/x]$ comes from h_n. Let $u_{n-2,a1,a2}$ be a new variable ($u_{n-2,a1,a2}$ represents $t_{n-2}[a_1/g,a_2/x]$). The l-term $((a_1)u_{n-2,a1,a2})b_1\ldots b_r$ is solvable, and its head normal form does not begin by l, therefore it is a variable applied to arguments. The free variables of $((a_1)u_{n-2,a1,a2})b_1\ldots b_r$ are among $u_{n-2,a1,a2}$, $u_{n-1,a1,a2}$, u_n, and f, therefore we have four possibilities for its head normal form :
(f)d (in this case we stop) or $(u_n)c_1\ldots c_s$ or $(u_{n-1,a1,a2})c_1\ldots c_s$ or $(u_{n-2,a1,a2})c_1\ldots c_s$
…and so on…

Assume we obtain $(u_{0,d1,d2})e_1\ldots e_k$ during the construction. The variable $u_{0,d1,d2}$

represents $t_0[d_1/g,d_2/x]$, and t_0∑x, therefore $(t_0[d_1/g,d_2/x])e_1\ldots e_k$ and $(d_2)e_1\ldots e_k$ have the same head normal form ; we then follow the construction with the l-term $(d_2)e_1\ldots e_k$.

The l-term $(T)h_nf$ is solvable, and has $(f)s(t)$ as head normal form, so this construction always stops on $(f)d$. We will prove later by a simple argument that $d{:}_{by}\underline{n}$.

According to the previous construction, the reduction $(T)h_nf$∑$(f)s(t_n)$ can be divided into two parts :

- A reduction that does not depend on $\underline{n}$:

$(T)u_nf$∑$(u_n)a_1\ldots a_m$,

$((a_1)u_{n-1,a1,a2})a_3\ldots a_m$ ∑$(u_{n-1,a1,a2})b_1\ldots b_r$,

$((a_1)u_{n-2,a1,a2})b_1\ldots b_r$∑$(u_{n-2,a1,a2})b_1\ldots b_r$,

…

- A reduction that depends on $\underline{n}$ (and not on h_n) :

the reduction from $(u_n)a_1\ldots a_m$ to $((a_1)u_{n-1,a1,a2})a_3\ldots a_m$,

the reduction from $(u_{n-1,a1,a2})b_1\ldots b_r$ to $((a_1)u_{n-2,a1,a2})c_1\ldots c_s$,

…,

the reduction from $(u_{0,d1,d2})e_1\ldots e_k$ to $(d_2)e_1\ldots e_k$,

…

If we allow some new reduction rules to get the later reductions, (something as : $(u_n)a_1a_2$∑$(a_1)u_{n-1,a1,a2}$; $u_{i+1,a1,a2}$∑$(a_1)u_{i,a1,a2}$ (for i>0) ; $u_{0,a1,a2}$∑a_2)

we obtain an equivalent -and easily expressed - definition for the storage operators for $\underline{N}$:

A closed l-term T is a storage operator for $\underline{N}$ if and only if for every $n\geq0$, $((T)u_nf$∑$(f)d_n$, and $d_n{:}_{by}\underline{n}$.

<u>0.3</u> The **directed l-calculus** is an extension of the ordinary l-calculus built for tracing a normal l-term t during some head reduction. Assume u is some, non normal, l-term having t as a subterm. We wish to trace the places where we really have to know what t is, during the reduction of u. Assume we have for every normal l-term t with free variables $x_1,\ldots,x_n$, and any l-terms $a_1,\ldots,a_n$ a "new" variable $u_{t,a1,\ldots,an}$.

We want the following rules :

if t=lxv, then $(u_{t,a1,\ldots,an})a$∑$u_{v,a1,\ldots,an,a}$ or $u_{t,a1,\ldots,an}$∑$\mathrm{l}x u_{v,a1,\ldots,an,x}$;

if t=(v)w, then $u_{t,a1,\ldots,an}$∑$(u_{v,a1,\ldots,an})u_{w,a1,\ldots,an}$;

if $t=x_i$ $1\leq i\leq n$, then $u_{t,a1,\ldots,an}$∑a_i.

We will prove later the following result (theorem 4-1) :

A closed l-term T is a storage operator for a set of closed normal l-terms D if and only if for every tAD, (T)u_tf∑(f)d_t, and d_t:$_{by}$t.

0.4 By interpreting the variable $u_{t,a_1,\dots,a_n}$ (that will be denoted by [t]<a_1/x_1,…,a_n/x_n> and called a box) by t**[**a_1/x_1,…,a_n/x_n**]** (the l-term t with an explicit substitution), the new reduction rules are those that allow to really do the substitution. This kind of l-calculus (l-calculus with explicit substitution) has been studied by P.L.Curien (see [1] and [4]) ; his ls-calculus contains terms and substitutions and is intended to better control the substitution process created by b-reduction, and then the implementation of the l-calculus. The main difference between the ls-calculus and the directed l-calculus is :

- The first one produces an explicit substitution after each b-reduction ;
- The second only " executes " the substitutions given in advance.

We can therefore consider the directed l-calculus as a restriction (the interdiction of producing explicit substitutions) of ls-calculus ; a well adapted way to the study of the head reduction.

0.5 This paper studies some properties of storage operators. It is organized as follows :

- The section 1 is devoted to preliminaries.
- In section 2, we define the storage operators, and we give the general form of their head normal forms.
- In section 3, we introduce the directed l-calculus, and we prove that it has the main properties of the ordinary l-calculus : the Church-Rosser theorem, the normalisation theorem, the resolution theorem. We focus on the head reduction, and we will prove that the reduction with the boxes represents correctly the reduction of terms where boxes are replaced by b-equivalent l-terms.
- In section 4, we present an equivalent definition for the storage operators.
- In section 5, we give some properties of storage operators :

- If T is a storage operator for a set of closed normal l-terms, and if T:$_b$T', then T' also is a storage operator for this set.
- The problem " Let t be a closed l-term. Is it a storage operator for a set of closed normal l-terms ?" is undecidable. It is semi-decidable in case of a finite set.
- Each finite set of closed normal l-terms has a storage operators.
- the number of b-reductions to go from (T)h_tf to (f)s(t_t) is linear in the number of reductions to normalize h_t.

Note : The presentation made below hides some technical uninteresting difficulties. Since we work with name for the variables, and modulo a-equivalence, there is a problem to define precisely the notion of subterms.

- We suppose, for example, that the l-terms (x)x, (y)y, (z)z,… are subterms of the l-term lx(x)x.
- A l-term may have equal subterms ; we assume that we can distinguish these subterms.

These problems could be solved by indexing subterms with the paths from the root of the l-term and using de Bruijn notation. We will do not do that here.

Acknowledgements. We thank J.L. Krivine, S. Ronchi, and H. Barendregt for helpful discussions.

§ 1. Basic notions of pure l-calculus

1.1. Notations

They are standard (see [2] and [6]).

- We shall denote by L the set of terms of pure l-calculus, also called **l-terms**.
- Let t,u,u_1,…, u_nAL, the application of t to u is denoted by (t)u or simply tu. In the same way we write (t)u_1…u_n or tu_1…u_n instead of (…((t)u_1)…)u_n.
- The b (resp. y, resp. by) -reduction is denoted by $\mathbf{t5_{b}u}$ (resp. $\mathbf{t5_{y}u}$, resp. $\mathbf{t5_{by}u}$).
- One step of b (resp. y) -reduction is denoted by $\mathbf{t5_{b0}u}$ (resp. $\mathbf{t5_{y0}u}$).
- The b (resp. y, resp. by) -equivalence is denoted by $\mathbf{t{:}_{b}u}$ (resp. $\mathbf{t{:}_{y}u}$, resp. $\mathbf{t{:}_{by}u}$).
- The set of free variables of a l-term t is denoted by **Fv(t)**.
- The notation t[a_1/x_1,…,a_n/x_n] represents the result of the simultaneous substitution of l-terms a_1,…,a_n to the free variables x_1,…,x_n of t (after a suitable renaming of the bounded variables of t).

The notation **s(t)** represents the result of the simultaneous substitution **s** to the free variables of t.

- The lenght of a l-term t (number of symbols used to write t) is denoted by **lg(t)**.
- We denote by **ST(t)**, the set of subterms of t.
- If t is b-normalizable, we denote by $\mathbf{t^{b}}$ its b-normal form.
- If t is by-normalizable, we denote by $\mathbf{t^{by}}$ its by-normal form.
- The notation $\mathbf{t∑∑_{0}t'}$ (resp. **t∑∑t'**) means that t' is obtained from t by one step of left reduction (resp. by some left reductions).

Theorem 1.1.1 (normalization theorem). *u is normalizable if and only if u is left normalizable.*

Proof. See [2] and [6]. ■

- If t is a normalizable l-term, then t∑∑t^{b}. We denote by **Tps(t)**, the number of steps used to go from t to t^{b}.
- The notation **t∑$_{0}$t'** (resp. **t∑t'**) means that t' is obtained from t by one step of head reduction (resp. by some head reductions).
- A l-term t is said **solvable** if and only if for every l-term u, there are variables x_1, …,x_k, and a l-terms $u_1,\ldots,u_k,v_1,\ldots,v_l$ k,l≥0, such that $(t[u_1/x_1,\ldots,u_k/x_k])v_1\ldots v_l :_b u$.

Theorem 1.1.2 (resolution theorem). *The following conditions are equivalent :*

1) t is solvable ;

2) the head reduction of t terminates ;

3) t is b-equivalent to a head normal form.

Proof. See [6]. ■

- If t is a solvable l-term, then there is a term t' in head normal form, such that t∑t'. We denote by **tps(t)**, the number of step used to go from t to t'.
- For each l-term, we associate a set of l-terms denoted by **STE(t)**, and called the set of essential subterms of t, by induction :
    - If t is unsolvable, then STE(t)={U}where U is a new symbol ;
    - If t is solvable, and $ly_1\ldots ly_m(y)t_1\ldots t_r$ is its head normal form, then STE(t)={t}".

Theorem 1.1.3. *If t is a normalizable l-term, then Tps(t)=tps(u).*

Proof. Trivial. ■

1.2. Properties of head reduction

Definitions.

- We define an **equivalence relation :** on L by : u:v if and only if there is a t, such that u∑t, and v∑t. In particular, if t is solvable, then u:t if and only if u is solvable, and has the same head normal form of t. If u is in head normal form, then t:u means u is the head normal form of t.
- If t∑t', we denote by **n(t,t')**, the number of steps to go from t to t'.

Theorem 1.2.1. *If t∑t', then for every $u_1,\ldots,u_r$AL :*

*1) There is v*AL*, such that $(t)u_1\ldots u_r$∑v, $(t')u_1\ldots u_r$∑v, and $n((t)u_1\ldots u_r,v)= n((t')u_1\ldots u_r,v)$*

+n(t,t').
2) $t[u_1/x_1,\ldots,u_r/x_r]$∑$t'[u_1/x_1,\ldots,u_r/x_r]$, and $n(t[u_1/x_1,\ldots,u_r/x_r],t'[u_1/x_1,\ldots,u_r/x_r])= n(t,t')$.

Proof. See [7]. ■

Remarks.
- 1) shows that to make the head reduction of $(t)u_1\ldots u_n$, it is equivalent (same result, and same number of steps) to make some steps in the head reduction of t, and then make the head reduction of $(t')u_1\ldots u_n$.
- 2) shows that to make the head reduction of $t[u_1/x_1,\ldots,u_n/x_n]$, it is equivalent (same result, and same number of steps) to make some steps in the head reduction of t, and then make the head reduction of $t'[u_1/x_1,\ldots,u_n/x_n]$.

This will be used everywhere without mention in the following.

Corollary 1.2.2. *Let $t,u_1,\ldots,u_n,v_1,\ldots,v_m$AL. If $(t[u_1/x_1,\ldots,u_n/x_n])v_1\ldots v_m$ is solvable, then t is solvable.*

Proof. Easy. ■

Corollary 1.2.3. *If t:t', then for every $u_1,\ldots,u_r$AL :*
1) $(t)u_1\ldots u_r$:$(t')u_1\ldots u_r$.
2) $t[u_1/x_1,\ldots,u_r/x_r]$:$t'[u_1/x_1,\ldots,u_r/x_r]$.

Proof. See [7]. ■

Corollary 1.2.4. *Let $t:_bu$, and u does not contain the variables $x_1,\ldots,x_n$, then the left reduction of $t[u_1/x_1,\ldots,u_n/x_n]$ is equivalent to the left reduction of t. This reduction is independent of the l-terms $u_1,\ldots,u_n$ which will never be evaluated.*

Proof. See [7]. ■

§ 2. Storage operators

2.1 Definition of storage operators

Definitions.
- A l-term t is said **essential** if and only if it is b-equivalent to a b-normal closed l-term.

- Let T be a closed l-term, and t an essential l-term. We say that T is a **storage operator** (shortened to **o.m.m**. for *opérateur de mise en mémoire*) **for t** if and only if there is $t_t:_{by}t$, such that for every $h_t:_bt$, $(T)h_t\sum lf(f)t_t[h_1/x_1,\ldots,h_n/x_n]$, where $Fv(t_t)=\{x_1,\ldots,x_n,f\}$, and $h_1,\ldots,h_n$ are l-terms which depend on h_t.

- Let T be a closed l-term, D a set of essential l-terms. We say that T is an **o.m.m for D** if and only if it is an o.m.m. for every t in D.

Lemma 2.1.1. *T is an o.m.m. for t if and only if there is a l-term $t_t:_{by}t$, such that for every $h_t:_bt$, $(T)h_tf:(f)t_t[h_1/x_1,\ldots,h_n/x_n]$, where $Fv(t_t)=\{x_1,\ldots,x_n,f\}$, and $h_1,\ldots,h_n$ are l-terms which depend on h_t.*

Proof.

1 Clear.

0 By corollary 1.2.2, $(T)h_t$ is solvable. Let T' be its head normal form.

- If T'=lfw, then w is the head normal form of $(T)h_tf$, therefore $w=(f)t_t[h_1/x_1,\ldots,h_n/x_n]$, therefore $(T)h_t\sum lf(f)t_t[h_1/x_1,\ldots,h_n/x_n]$.

- If $T'=(v)T_1\ldots T_r$; we can choose h_t, such that $fFv(h_t)$, $v\neq f$, therefore the head normal form of $(T)h_tf$ is $(v)T_1\ldots T_rf=(f)t_t[h_1/x_1,\ldots,h_n/x_n]$. A contradiction. ■

Remark. Let F be any l-term, and h_t a l-term b-equivalent to tAD. During the computation of $(F)h_t$, h_t may be computed many times (for example, each time it comes in head position). Insead of computing $(F)h_t$, let us look at the head reduction of $(T)h_tF$. Since it is $(T)h_tf[F/f]$, by theorem 1.2.1, we shall first reduce $(T)h_tf$ to its head normal form, which is $(f)t_t[h_1/x_1,\ldots,h_n/x_n]$, and then compute $(F)t_t[c_1/x_1,\ldots,c_n/x_n, F/f]$ where $c_i=h_i[F/f]$. By corollary 1.2.4, the computation has been decomposed into two parts, the first being independent of F. This first part is essentially a computation of h_t, the result being t_t, which is a kind of normal form of h_t, because it only depends on the b-equivalent class of h_t : the substitutions made in t_t have no computational importance, since t is essential. So, in the computation of $(T)h_tF$, h_t is computed first, and the result is given to F as an argument, T has stored the result, before giving it, as many times as needed, to any function.

2.2 General forms of head normal form of a storage operator

Proposition 2.2.1. *If T is an o.m.m. for t, then T is solvable, and its head normal form T' has one of the following form : $T'=ln(n)T_1\ldots T_r$ $r\geq1$, $T'=lnlf(n)T_1\ldots T_r$ $r\geq1$, or*

T'=lnlf(f)T_1 where $T_1 :_{by} t$.

Corollary 2.2.3. *If t is unsolvable, and T is an o.m.m. for t, then T∑lnlf(f)T_1, and $T_1 :_{by} t$.*

Proof. If T∑ln(n)$T_1 \ldots T_r$ r≥1 or T∑lnlf(n)$T_1 \ldots T_r$ r≥1, then (T)t is unsolvable. Therefore, by proposition 2.2.1, T∑lnlf(f)T_1, and $T_1 :_{by} t$. ■

Proof of proposition 2.2.1. If T is an o.m.m. for t, then there is a l-term $t_t :_{by} t$, such that for every $h_t :_b t$, (T)h_t∑lf(f)$t_t[u_1/y_1,\ldots,u_n/y_n]$, with Fv($t_t$)={$y_1,\ldots,y_n$,f}, and $u_1,\ldots,u_n$ are l-terms wich depend on h_t. Therefore, by corollary 1.2.2, T is solvable. Let T' its head normal form. Since T is closed, T' also is closed, and T'=l$x_1 \ldots$l$x_m(x_i)T_1 \ldots T_r$ r≥1.
By theorem 1.2.1, (T')h_t∑lf(f)$t_t[u_1/y_1,\ldots,u_n/y_n]$, therefore m =1 or 2.
- If m=1, then T'=ln(n)$T_1 \ldots T_r$ r≥1.
- If m=2 :
  - If i=1, then T'=lnlf(n)$T_1 \ldots T_r$ r≥1.
  - If i=2, then T'=lnlf(f)$T_1 \ldots T_r$ r≥1. Therefore lf(f)$T_1[h_t/n] \ldots T_r[h_t/n]$= lf(f)t'$[u_1/y_1,\ldots,u_n/y_n]$, therefore r=1, and $T_1[h_t/n]=t_t[u_1/y_1,\ldots,u_n/y_n]$.

It remains to show that $T_1 :_{by} t$.

Lemma 2.2.4. *Let x,y be two variables of the l-calculus.*
1) If t[(x)y/z]5_{b0}u, then u=v[(x)y/z], and t5_{b0}v.
2) If t is a closed l-term, and t[(x)y/z]5_bt, then t5_bt.

Proof.
1) By induction on t.
- If t is a variable, it is impossible.
- If t=lrw, then u=lra, and w[(x)y/z]5_{b0}a. By induction hypothesis, we have a=b[(x)y/z], and w5_{b0}b. Therefore if we take v=lrb, we get u=v[(x)y/z], and t5_{b0}v.
- If t=(a)b, and u=(c)b where a[(x)y/z]5_{b0}c. By induction hypothesis, we have c=d[(x)y/z], and a5_{b0}d. Therefore if we take v=(d)b, we get u=v[(x)y/z], and t5_{b0}v.
- If t=(a)b, and u=(a)c where b[(x)y/z]5_{b0}c. By induction hypothesis, we have c=d[(x)y/z], and b5_{b0}d. Therefore if we take v=(a)d, we get u=v[(x)y/z], and t5_{b0}v.
- If t=(lra)b, and u=a[(x)y/z][b[(x)y/z]/r]=a[b/r][(x)y/z], then, if we take v=a[b/r], we get u=v[(x)y/z], and t5_{b0}v.

2) By induction on the number of b_0-reductions. We use 1) to prove t=u[(x)y/z], and t5_bu. Since t is closed, then t=u and t5_bt. ■

By lemma 2.2.4, we may assume that t_t does not contain a $(y_i)y_j$ 1≤i,j≤nas subterm.

Lemma 2.2.5. *Let $d,t,t_1,\ldots,t_n$ be l-terms , and $s_1,\ldots,s_n$ substitutions, such that : $Fv(d)=\{x_1,\ldots,x_n\}"\{a_1,\ldots,a_r\}$, $Fv(t)=\{y_1,\ldots,y_m\}"\{b_1,\ldots,b_k\}$, and for all $1\leq i,j\leq m$ $(y_i)y_j$ is not a subterm of t. If for all 1≤i≤nand for every $h_i:_b t_i$, there are $h_1,\ldots,$ $h_{i-1},h_{i+1},\ldots h_n,u_1,\ldots,u_m$, such that $d[s_1(h_1)/x_1,\ldots,s_n(h_n)/x_n]=t[u_1/y_1,\ldots,u_m/y_m]$, then there are $w_1,\ldots,w_m$, such that $d=t[w_1/y_1,\ldots,w_m/y_m]$.*

Proof. By induction on d and t**.**

It is clear that we may assume that any variable $x_1,\ldots,x_n$ (resp. $y_1,\ldots,y_m$) appears at most once in d (resp. t).

- If $d=a_1$, then $a_1=t[u_1/y_1,\ldots,u_m/y_m]$, therefore $t=y_1$, and $u_1=a_1$ or $t=b_1=a_1$.
- If $d=x_1$, then $s_1(h_1)=t[u_1/y_1,\ldots,u_m/y_m]$.
  - If $t=b_1$, then $s_1(h_1)$ is a variable, that is impossible if we take $h_1=(lxt_1)x$.
  - If $t=y_1$, then $d=t[x_1/y_1]$.
  - If t=lxt', then $s_1(h_1)$ begins by l, that is impossible if we take $h_1=(lxt_1)x$.
  - If t=(u)v :
    - If $t=(\ldots(((lxa)b)v_1)\ldots)v_r$, then $s_1(h_1)$ begins with r+1 (, that is impossible if we take $h_1=(\ldots(((lx_1lx_2\ldots lx_{n+2}t_1)x_1)x_2)\ldots)v_{n+2.}$
    - If $t=(\ldots((b_1)v_1)\ldots)v_r$, then that is impossible if we take $h_1=(lxt_1)x$.
    - If $t=(\ldots((y_1)v_1)\ldots)v_r$ and r≥2, then $s_1(h_1)$ begins by at least r (, that is impossible if we take $h_1=(lxt_1)x$. Therefore r=1 and $t=(y_1)v_1$.

      The l-term v_1 can not begin by l. (it suffices to take $h_1=(lxt_1)(lxx)x$)

      The l-term v_1 can not begin by (. (it suffies to take $h_1=(lxt_1)lxx$)

      Therefore v_1 is a variable.

      If $v_1=b_1$, then that is impossible if we take $h_1=(lxt_1)(lxx)x$.

      If $v_1=y_2$, then that is impossible because in this case we have $t=(y_1)y_2$.
- If d=lxu, then :
  - If $t=b_1$, then lg(d)=1, that is impossible.
  - If $t=y_1$, then $d=t[lxu/y_1]$.
  - If t=lxt', then $u[s_1(h_1)/x_1,\ldots,s_n(h_n)/x_n]=t'[u_1/y_1,\ldots,u_m/y_m]$, and we use the induction hypothesis
  - If t=(u)v, then d begins by (, that is impossible.
- If d=(u)v, then :
  - If $t=b_1$, then lg(d)=1, that is impossible.
  - If $t=y_1$, then $d=t[(u)v/y_1]$.
  - If t=lxt', then d begins by l, that is impossible.

- If t=(a)b, then $u[s_1(h_1)/x_1,\ldots,s_n(h_n)/x_n]=a[u_1/y_1,\ldots,u_m/y_m]$, and $v[s_1(h_1)/x_1,\ldots,s_n(h_n)/x_n]=b[u_1/y_1,\ldots,u_m/y_m]$, and we use the induction hypothesis. ■

By lemma 2.2.5, there are $w_1,\ldots,w_m$, such that $T_1=t_t[w_1/x_1,\ldots,w_n/x_n]$, we have $T_1:_{by}t$. ■ (of proposition 2.2.1)

2.3 Examples of storage operators

2.3.1 The projections

For all 0≤i≤n, let $P=lx_1\ldots lx_nx_i$ (the ith projection among n). Let P_n be the set of projections.

Define T=ln(n) lf(f)Plf(f)P... lf(f)P, and T=lnlf(n) (f)PP... (f)P.

Tand Tare two o.m.m. for P_n.

Let $h:_bP1≤i≤n$, then h∑P.

Behaviour of T:

Thf:(h) lf(f)Plf(f)P... lf(f)Pf:(P) lf(f)Plf(f)P... lf(f)Pf:(lf(f)P)f:

(f)P.

It is easy to check that tps(Thf)=Tps(h)+n+2. ■

Behaviour of T:

Thf:(h) (f)PP... (f)Pf:(P) (f)PP... (f)Pf:(f)P.

It is easy to check that tps(Thf)=Tps(h)+n+2. ■

2.3.2 The Church integers

For n≥0, we define the Church integer $\underline{n}$=lflx(f)^{n}x. Let $\underline{N}$ be the set of Church integers. Let $\underline{s}$=lnlflx(f)((n)f)x. It is easy to check that $\underline{s}$ is a l-term for the successor. Define T=ln(n)Gd where G=lxly(x)lz(y)($\underline{s}$)z, and d=lf(f)$\underline{0}$;

T=lnlf(n)F f $\underline{0}$ where F=lxly(x)($\underline{s}$)y.

Tand Tare o.m.m. for $\underline{N}$.

Let $h_n:\underline{n}$, then h_n∑lglx(g)t_{n-1}, t_{n-k}∑(g)t_{n-k-1} 1≤k≤n-1, t_0∑x.

Behaviour of T:

(T)h_nf:(h_n)Gdf:(G)t_{n-1}[G/g,d/x]f:(t_{n-1}[G/g,d/x])lz(f)($\underline{s}$)z.

We define a sequence of l-terms $(t_i)_{1≤i≤n}$:

t_1=lz(f)($\underline{s}$)z, and for all 1≤k≤n-1 let t_{k+1}=lz(t_k)($\underline{s}$)z.

We prove (by induction on k) that for all 1≤k≤nwe have (T)h_nf:(t_{n-k}[G/g,d/x])t_k.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

(T)h_nf:(t_{n-k}[G/g,d/x])t_k:(G)t_{n-k-1}[G/g,d/x]t_k:t_{n-k-1}[G/g,d/x])lz(t_k)($\underline{s}$)z=
(t_{n-k-1}[G/g,d/x])t_{k+1}.

Therefore, in particular, for k=n we have (T)h_nf:(t_0[G/g,d/x])t_n=(d)t_n:(t_n)$\underline{0}$.

We prove (by induction on k) that for all 1≤k≤nwe have t_k:lz(f)($\underline{s}$)^{k}z.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

t_{k+1}=lz(t_k)($\underline{s}$)z:lz(lz(f)($\underline{s}$)^{k}z)($\underline{s}$)z:lz(f)($\underline{s}$)$^{k+1}$z.

Therefore, in particular, for k=n we have t_n:lz(f)($\underline{s}$)^{n}z and (T)h_nf:(lz(f)($\underline{s}$)^{n}z)$\underline{0}$: (f)($\underline{s}$)^n$\underline{0}$.

It is easy to check that tps((T)h_nf)=Tps(h_n)+3n+4. ■

Behaviour of T:

(T)h_nf:(h_n)Ff$\underline{0}$:(F)t_{n-1}[F/g,f/x]$\underline{0}$:(t_{n-1}[F/g,f/x])($\underline{s}$)$\underline{0}$.

We prove (by induction on k) that for all 1≤k≤nwe have
(T)h_nf:(t_{n-k}[F/g,f/x])($\underline{s}$)^k$\underline{0}$.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

(T)h_nf:(t_{n-k}[F/g,f/x])($\underline{s}$)^k$\underline{0}$:(F)t_{n-k-1}[F/g,f/x]($\underline{s}$)^k$\underline{0}$:t_{n-k-1}[F/g,f/x])($\underline{s}$)$^{k+1}$$\underline{0}$.

Therefore, in particular, for k=n we have (T)h_nf:(t_0[F/g,f/x])($\underline{s}$)^n$\underline{0}$=(f)($\underline{s}$)^n$\underline{0}$.

It is easy to check that tps((T)h_nf)=Tps(h_n)+2n+4. ■

2.3.3 The recursive integers

For n≥0, we define the recursive integer by :=lflxx and =lflx(f). Let be the set of recursive integers. Let =lnlflx(f)n. It is easy to check that is a l-term for the successor.

Define T=(Y)H where Y=(lxlf(f)(x)xf)lxlf(f)(x)xf, H=lxly((y)lz(G)(x)z)d, G=lxly(x)lz(y)()z, and d=lf(f) ;

T=ln(n)rtr where t=ldlf(f) , and r=lylz(G)(y)ztz.

Tand Tare o.m.m. for .

Let h_n:$_b$, then :

if n=0, h_n∑lglxx, and if n≠0, h_n∑lglx(g)h_{n-1} where h_{n-1}:$_b$.

Behaviour of T:

We prove (by induction on n) that ((Y)H)h_n:lf(f)()^n.

If n=0, then ((Y)H)h_0:((H)(Y)H)h_0:((h_0)lz(G)((Y)H)z)d:d=lf(f) .

If n≠0, then ((Y)H)h_n:((H)(Y)H)h_n:((h_n)lz(G)((Y)H)z)d:

(lz(G)((Y)H)z)h_{n-1}[lz(G)((Y)H)z/g,d/x]:(G)((Y)H)h_{n-1}[lz(G)((Y)H)z/g,d/x]:

lf(((Y)H)h_{n-1}[lz(G)((Y)H)z/g,d/x])lz(f)( )z.

Since $h_{n-1}:_b$, then h_{n-1}[lz(G)((Y)H)z/g,d/x]$:_b$, and, by induction hypothesis, ((Y)H)h_{n-1}:lf(f)()$^{n-1}$.

Therefore ((Y)H)h_n:lf(lf(f)()$^{n-1}$)lz(f)( )z:lf(f)()^n.

It is easy to check that tps((T))h_nf)=Tps(h_n)+10n+7. ■

Behaviour of T:

We prove (by induction on n) that (h_n)rtr:lf(f)()^n.

If n=0, then (h_0)rtr:(t)r:lf(f).

If n≠0, then (h_n)rtr:(r)h_{n-1}[r/g,t/x]r:(G)(h_{n-1}[r/g,t/x])rtr:

lf((h_{n-1}[r/g,t/x])rtr)lz(f)( )z.

Since $h_{n-1}:_b$, then h_{n-1}[r/g,t/x]$:_b$, and, by induction hypothesis,

h_{n-1}[r/g,t/x]rtr:lf(f)()$^{n-1}$.

Therefore (h_n)rtr:lf(lf(f)()$^{n-1}$)lz(f)()z:lf(f)()^n.

It is easy to check that tps((T))h_nf)=Tps(h_n)+7n+5. ■

2.3.4 The finite lists

Let U be a set of essential l-terms. We define the set of the finite lists of objects of U, L_U={lflx((f)u_1)((f)u_2)…((f)u_n)x where nA**N**, u_iAU}.

Let nil=lxlyy, cons=lxlylfla((f)x)((y)f)a and cons'=lxlylfla((y)f)((f)x)a. It is easy to check that cons and cons' are two l-terms for the concatenation.

Let T_U be an o.m.m. for U.

Define T=ln(n)Hd where H=lxlylz((T_U)x)lu(y)lv(z)((cons)u)v , and d=lf(f)nil ;

T=lnlf(n)K f nil where K=lxlylu((T_U)x)lv(y)(cons')v)u.

Tand Tare o.m.m. for L_U.

Let $h_n:_b$lflx((f)u_1)((f)u_2)…((f)u_n)x, then :

h_n∑lglx(g)v_1t_1, $v_1:_bu_1$, t_i∑(g)$v_{i+1}t_{i+1}$, $v_{i+1}:_bu_{i+1}$ 1≤i≤n-1, t_n∑x.

T_U is an o.m.m. for U, therefore for all 1≤i≤n, there is $t_i:_bu_i$, such that (T_U)v_i[H/g,d/x]∑lf(f)s_i(t_i).

Behaviour of T:

(T)h_nf:(h_n)Hdf:(H)v_1[H/g,d/x]t_1[H/g,d/x]f:

((T_U)v_1[H/g,d/x])lu(t_1[H/g,d/x])lv(f)((cons))u)v:(lf(f)s_1(t_1))lu(t_1[H/g,d/x])lv(f)

((cons))u)v:(t_1[H/g,d/x])lv(f)((cons))s_1(t_1))v.

We define a sequence of l-terms $(d_i)_{1\leq i\leq n}$: $d_1=lv(f)((\underline{cons}))s_1(t_1))v$, and for $1\leq k\leq n-1$ Let $d_{k+1}=lv(d_k)((\underline{cons}))s_{k+1}(t_{k+1}))v$.

We prove (by induction on k) that for all $1\leq k\leq n$we have $(T)h_nf:(t_k[H/g,d/x])d_k$.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

$(T)h_nf:(t_k[H/g,d/x])d_k:(H)v_{k+1}[H/g,d/x]t_{k+1}[H/g,d/x]d_k$

$((T_U)v_{k+1}[H/g,d/x])lu(t_{k+1}[H/g,d/x])lv(d_k)((\underline{cons}))u)v:$

$(lf(f)s_{k+1}(t_{k+1}))lu(t_{k+1}[H/g,d/x])lv(d_k)((\underline{cons}))u)v:$

$(t_{k+1}[H/g,d/x])lv(d_k)((\underline{cons}))s_{k+1}(t_{k+1}))v=(t_{k+1}[H/g,d/x])d_{k+1}$.

Therefore, in particular, for k=n we have $(T)h_nf:(t_n[H/g,d/x])d_n=(d)d_n:(d_n)nil$.

We prove (by induction on k) that for all $1\leq k\leq n$we have

$d_k:lv(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_k(t_k))v$.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

$d_{k+1}=lv(d_k)((\underline{cons}))s_{k+1}(t_{k+1}))v:$

$lz(lv(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_k(t_k))v)(((\underline{cons}))s_{k+1}(t_{k+1}))v:$

$lv(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_k(t_k))v)(((\underline{cons}))s_{k+1}(t_{k+1}))v$.

Therefore, in particular, for k=n we have

$d_n:lv(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_n(t_n))v$

and $(T)h_nf:(lv(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_n(t_n))nil:$

$(f)((\underline{cons})s_1(t_1))((\underline{cons})s_2(t_2))\ldots((\underline{cons})s_n(t_n))nil=(f)s(\{((\underline{cons})t_1)((\underline{cons})t_2)\ldots$

$((\underline{cons})t_n)nil\})$.

It is easy to check that if $tps(T_Uv_i)=Tps(v_i)+D_i$, then $tps((T)h_nf)=Tps(h_n)+6n+4$ +

Behaviour of T:

$(T)h_nf:(h_n)K \quad f \quad nil:(K)v_1[K/g,f/x]t_1[K/g,f/x]nil:((T_U)v_1[K/g,f/x])lv(t_1[K/g,f/x])$

$((\underline{cons}')v)nil:(lf(f)s_1(t_1))lv(t_1[K/g,f/x])((\underline{cons}'))v)nil:$

$(t_1[K/g,f/x])((\underline{cons}'))s_1(t_1))nil$.

We prove (by induction on k) that for all $1\leq k\leq n$we have

$(T)h_nf:(t_k[F/g,f/x])((\underline{cons}')s_k(t_k))((\underline{cons}')s_{k-1}(t_{k-1}))\ldots((\underline{cons}')s_1(t_1))nil$.

For k=1 it is true.

Assume that is true for k, and prove it for k+1.

$(T)h_nf:(t_k[K/g,f/x])((\underline{cons}')s_k(t_k))\ldots((\underline{cons}')s_1(t_1))nil:$

$(K)v_{k+1}[K/g,f/x]t_{k+1}[K/g,f/x]((\underline{cons}')s_k(t_k))\ldots((\underline{cons}')s_1(t_1))nil:$

$((T_U)v_{k+1}[K/g,f/x])lv(t_{k+1}[K/g,f/x])((\underline{cons}'))v)((\underline{cons}')s_k(t_k))\ldots((\underline{cons}')s_1(t_1))nil:$

$(lf(f)s_{k+1}(t_{k+1}))lv(t_{k+1}[K/g,f/x])((\underline{cons}'))v)((\underline{cons}'))v)((\underline{cons}')s_k(t_k))\ldots$

$((\underline{cons}')s_1(t_1))nil:(t_{k+1}[F/g,f/x])((\underline{cons}')s_{k+1}(t_{k+1}))\ldots((\underline{cons}')s_1(t_1))nil$.

Therefore, in particular, for k=n we have

(T)h_nf:(t_n[K/g,f/x])(($\underline{\text{cons}}$')s_n(t_n))…(($\underline{\text{cons}}$')s_1(t_1))nil=
(f)(($\underline{\text{cons}}$')s_n(t_n))…(($\underline{\text{cons}}$')s_1(t_1))nil=(f)s({(($\underline{\text{cons}}$')t_n)…(($\underline{\text{cons}}$')t_1)nil}).
It is easy to check that if tps($T_U v_i$)=Tps(v_i)+D_i, then tps((T)h_nf)=Tps(h_n)+5n+4 +

§ 3. The directed l-calculus

3.1 The l[]-terms

Definitions.

• If L is the set of simple l-terms (L without a-equivalence), having V as set of variables, then the set of terms of **simple directed l-calculus**, denoted by **L[]**, is defined in the following way :

- If xAV, then xAL[] ;
- If xAV, and uAL[], then lxuAL[] ;
- If uAL[], and vAL[], then (u)vAL[] ;
- If tAL is a *b-normal* l-term, such that Fv(t)[{x_1,…,x_n}, and a_1,…,a_nAL[], then [t]<a_1/x_1,…,a_n/x_n>AL[].

A l[]-term of the form [t]<a_1/x_1,…,a_n/x_n> is said a **box directed by t** (we also say that t is the **director** of the box).
This notation represents, intuitively, the l-term t where the free variables x_1,…,x_n will be replaced by a_1,…,a_n.

We extend the definition of the a-equivalence by :
[u]<a_1/x_1,…,a_n/x_n>$:_a$[v]<b_1/y_1,…,b_m/y_m> if and only if there are permutations P_n and P_m, 0≤r≤inf(n,m), and new variables z_1,…,z_r, such that :

- Fv(u)={x,…,x} and Fv(v)={y,…,y},
- u[z_1/x,…,z_r/x]$:_a$v[z_1/y,…,z_r/y].
- a$:_a$b1≤i≤r.

• The set of terms of the **directed l-calculus**, denoted by **L[]**, and also called **l[]-terms**, is defined by L[]=L[]/:a.

• We will note <**a**/**x**> the substitution <a_1/x_1,…,a_n/x_n>. The substitution <a_1/x_1,…,a_n/x_n,b_1/y_1,…,b_m/y_m> is denoted by <**a**/**x**,**b**/**y**>, and the substitution <a_1[u_1/y_1,…,u_m/y_m]/x_1,…,a_n[u_1/y_1,…,u_m/y_m]/x_n> is denoted by <**a**[u_1/y_1,…,u_m/y_m]/**x**>.

• For every u,u_1,…,u_mAL[], we extend the definitions of Fv(u) and u[u_1/y_1,…,u_m/y_m] by :

- Fv([t]<**a**/**x**>)=Fv(**a**)=.
- [t]<**a**/**x**>[u_1/y_1,…,u_m/y_m]=[t]<**a**[u_1/y_1,…,u_m/y_m]/**x**>, after a suitable renaming of the bounded variables of a_1,…,a_n that are free in u_1,…,u_m.

3.2 The b[]-reduction

Definitions.

- A l[]-term of the form (lxu)v is called **b-redex** ; u[v/x] is called its **contractum**.

A l[]-term of the form [t]<**a**/**x**> is called **[]-redex** ; its **contractum** R is defined by induction on t :
- If t=x$_i$ 1≤i≤n,then R=**a**$_i$;
- If t=lxu, then R=ly[u]<**a**,/**x**,y/x> where yFv(**a**) ;
- If t=(u)v, then R=([u]<**a**/**x**>)[v]<**a**/**x**>.

- We define a binary relation **5$_{b0}$** by :

t5$_{b0}$t' if and only if t' is obtained by contracting a b-redex of t.

More precisely :
- If t is a variable, t5$_{b0}$t' is false for all t' ;
- If t=lxu, then t5$_{b0}$t' if and only if t'=lxu', and u5$_{b0}$u' ;
- If t=(v)u, then t5$_{b0}$t' if and only if
  t'=(v)u' with u5$_{b0}$u' or
  t'=(v')u with v5$_{b0}$v' or
  v=lxw, and t'=w[u/x] ;
- If t=[u]<**a**/**x**>, then t5$_{b0}$t' if and only if
a$_i$5$_{b0}$a'$_i$, x$_i$AFv(u) 1≤i≤n,and t'=[u]<a$_1$/x$_1$,…,a$_{i-1}$/x$_{i-1}$,a'$_i$/x$_i$,a$_{i+1}$/x$_{i+1}$,…,a$_n$/x$_n$>.

- We define a binary relation **5$_{[]0}$** by :

t5$_{[]0}$t' if and only if t' is obtained by contracting a []-redex of t.

More precisely :
- If t is a variable, t5$_{[]0}$t' is false for all t' ;
- If t=lxu, then t5$_{[]0}$t' if and only if t'=lxu', and u5$_{[]0}$u' ;
- If t=(v)u, then t5$_{[]0}$t' if and only if
  t'=(v)u' with u5$_{[]0}$u' or
  t'=(v')u with v5$_{[]0}$v' ;
- If t =[u]<**a**/**x**>, then t5$_{[]0}$t' if and only if
  t' is the contractum of t or
  a$_i$5$_{[]0}$a'$_i$, x$_i$AFv(u) 1≤i≤n,and t'=[u]<a$_1$/x$_1$,…,a$_{i-1}$/x$_{i-1}$,a'$_i$/x$_i$,a$_{i+1}$/x$_{i+1}$,…,a$_n$/x$_n$>

- We define a binary relation **5$_{b[]0}$** on L[] by t5$_{b0}$t' or t5$_{[]0}$t'.

Therefore t5$_{b[]0}$t' if anf only if t' is obtained by contracting a b[]-redex of t.

- We define the **b-conversion** (resp. the **[]-conversion,** resp. the **b[]-conversion**) as the reflexive and transitive closure of 5$_{b0}$ (resp.5$_{[]0}$, resp. 5$_{b[]0}$).

We have therefore **$t5_bt'$** (resp. **$t5_{[]}t'$**, resp. **$t5_{b[]}t'$**) if and only if there is a sequence $t_0=t,t_1,\ldots,t_{n-1},t_n=t'$, such that $t_i5_{b0}t_{i+1}$ (resp. $t_i5_{[]0}t_{i+1}$, resp. $t_i5_{b[]0}t_{i+1}$) for $1\leq i\leq n-1$.
It is clear that if $t5_{b[]}t'$, then Fv(t')[Fv(t).

- A l[]-term t is said **b[]-normal**, if it does not contain any redex.

A l[]-term t is said **b[]-normalizable**, if there is a b[]-normal l[]-term t', such that $t5_{b[]}t'$.
A l[]-term t is said **b[]-strongly normalizable**, if there is a no infinite sequence $t_0=t,t_1,\ldots,t_n,\ldots$, such that $t_i5_{b[]0}t_{i+1}$ for $i\geq 0$.

Lemma 3.2.1. *t is b[]-normal if and only if tAL, and t is b-normal.*

Proof. Clear. ■

Lemme 3.2.2. *A []-reduction always terminates.*

Proof. Otherwise, there is an infinite sequence $t_0,t_1,\ldots,t_n,\ldots$, such that $t_i5_{[]0}t_{i+1}$ for $i\geq 0$.
For each l[]-term t, we associate an integer **b(t)** by induction on t :

- If t=x, then b(t)=0 ;
- If t=lxu, then b(t)=b(u) ;
- If t=(u)v, then b(t)=b(u)+b(v) ;
- If t=[u]<**a**/**x**>, then :
    - If $u=x_i$ $1\leq i\leq n$, then $b(t)=b(a_i)+1$;
    - If u=lxv, then b(t)=b([v]<**a**/**x**,y/x>)+1 yFv(**a**) ;
    - If u=(v)w, then b(t)=b([v]<**a**/**x**>)+b([w]<**a**/**x**>)+1.

Lemma 3.2.3.
1) b(t)=0 if and only if tAL.
2) If $b(a_i)=b(a'_i)$ $1\leq i\leq n$, then
$b([u]<\mathbf{a}/\mathbf{x}>)=b([u]<a_1/x_1,\ldots,a_{i-1}/x_{i-1},a'_i/x_i,a_{i+1}/x_{i+1},\ldots,a_n/x_n>)$.
3) If $b(a_i)>b(a'_i)$, and x_iAFv(u) $1\leq i\leq n$, then
$b([u]<\mathbf{a}/\mathbf{x}>)>b([u]<a_1/x_1,\ldots,a_{i-1}/x_{i-1},a'_i/x_i,a_{i+1}/x_{i+1},\ldots,a_n/x_n>)$.

Proof. By induction on t. (resp. u) for 1) (resp 2), 3)). ■

Lemma 3.2.4. *If t $5_{[]0}$t', then b(t)>b(t').*

Proof. By induction on t. The only interesting case is t=[u]<**a**/**x**>. Then :
- If $u=x_i$ $1\leq i\leq n$, then $t'=a_i$, and $b(t)=b(a_i)+1>b(t')$.
- If u=lxv, then t'=[u]<**a**/**x**,y/x> yFv(**a**), therefore, by lemma 3.2.3,

b(t)=b([u]<**a**/**x**,y/x>)+1>b(t').

- If u=(v)w, then t'=([v]<**a**/**x**>)[w]<**a**/**x**>, and b(t)=b([v]<**a**/**x**>)+b([w]<**a**/**x**>)+1>b(t').
- If $a_i5_{[]0}a'_i$, x_iAFv(u) 1≤i≤n, and t'=[u]<$a_1/x_1,\ldots,a_{i-1}/x_{i-1},a'_i/x_i,a_{i+1}/x_{i+1},\ldots,a_n/x_n$>). By induction hypothesis, we have b(a_i)>b(a'_i), therefore, by lemma 3.2.3, b(t)>b(t'). ■

Therefore, by lemma 3.2.4, there is an infinite sequence b(t_0),b(t_1),…,b(t_n),…, such that b(t_i)>b(t_{i+1}) for i≥0. A contradiction. ■ (of lemma 3.2.2)

Definition. For each l[]-term t, we associate a l-term **l(t)** by induction on t :

- If t=x, then l(t)=x ;
- If t=lxu, then l(t)=lxl(u) ;
- If t=(u)v, then l(t)=(l(u))l(v) ;
- If t=[u]<**a**/**x**>, then l(t)=u[l(a_1)/x_1,…,l(a_n)/x_n].

It is clear that for tAL[], Fv(t)=Fv(l(t)).

Theorem 3.2.5. *t is b[]-strongly normalizable if and only if l(t) is strongly normalizable.*

Theorem 3.2.6 (Church-Rosser theorem). *Assume $t_05_{b[]}t_1$, and $t_05_{b[]}t_2$, then there is a t_3, such that $t_15_{b[]}t_3$ and $t_25_{b[]}t_3$.*

Proof of theorem 3.2.5.
1 If l(t) is not strongly normalizable, then there is an infinite sequence t_0=l(t),t_1,…,t_n, …, such that $t_i5_{b0}t_{i+1}$ for all i≥0.

Lemma 3.2.7. *If $t5_{[]}t'$, then l(t)=l(t').*

Proof. By induction on t. ■

Lemma 3.2.8.
*1) [u]<**a**/**x**>$5_{[]}$u[$a_1/x_1,\ldots,a_n/x_n$].*
2) If $u_i5_{[]}v_i$ 1≤i≤n, then u[$u_1/x_1,\ldots,u_n/x_n$]$5_{[]}$u[$v_1/x_1,\ldots,v_n/x_n$].

Proof. By induction on u. ■

Lemma 3.2.9. *If t is a l[]-term, then $t5_{[]}$l(t).*

Proof. By induction on t. The only interesting case is t=[u]<**a**/**x**>.

By lemma 3.2.8, [u]<**a**/**x**>$5_{[]}$u[a_1/x_1,…,a_n/x_n]. By induction hypothesis, we have $a_i5_{[]}$l(a_i) 1≤i≤n, therefore, by lemma 3.2.8, t$5_{[]}$u[l(a_1)/x_1,…,l(a_n)/x_n]=l(t). ■

By lemma 3.2.9, t$5_{[]}$l(t), therefore t is not b[]-strongly normalizable. A contradiction. ■ (of **1** theorem 3.2.5).

0 (theorem 3.2.5) If t is not b[]-strongly normalizable then there is an infinite sequence t_0=t,t_1,…,t_n,…, such that $t_i5_{b0}t_{i+1}$ or $t_i5_{[]0}t_{i+1}$ for i≥0.

Lemma 3.2.10. *l(u[v/x])=l(u)[l(v)/x].*

Proof. By induction on u. ■

Lemma 3.2.11. *If u5_{b0}v, then l(u)5_{b0}l(v).*

Proof. By induction on u. The only non-trivial case is u=(lxt)w : we then have v=t[w/x], therefore, by lemma 3.2.10, l(u)=(lxl(t))l(w)5_{b0} l(t)[l(w)/x]=l(v). ■

Corollary 3.2.12. *If u$5_{b[]}$v, then l(u)5_bl(v).*

Proof. Use lemmas 3.2.7 and 3.2.11. ■

By lemma 3.2.2, and lemma 3.2.11, tthere is an infinite sequence t'_0=l(t),t'_1,…,t'_n,…, such that $t'_i5_{b0}t'_{i+1}$ for all i≥0, therefore l(t) is not strongly normalizable. A contradiction. ■ (of **0** theorem 3.2.5)

Proof of theorem 3.2.6. If $t_05_{b[]}t_1$, and $t_05_{b[]}t_2$, then, by corollary 3.2.12, l(t_0)5_bl(t_1), and l(t_0)5_bl(t_2). Therefore, by the Church-Rosser theorem of l-calculus, there is a t_3, such that l(t_1)5_bt_3, and l(t_2)5_bt_3, therefore, by lemma 3.2.9, $t_15_{b[]}t_3$, and $t_25_{b[]}t_3$. ■

Remarks.

- By the Church-Rosser theorem, the b[]-normal form is unique.
- We define the **b[]-equivalence** (denoted by **$:_{b[]}$**), as the symetric closure of $5_{b[]}$; In other words : t$:_{b[]}$t' if there are t_0=t,t_1,…,t_n=t' with $t_i5_{b[]0}t_{i+1}$ or $t_{i+1}5_{b[]0}t_i$ 0≤i≤n-1. By the Church-Rosser theorem : t$:_{b[]}$t' if and only if there is a l[]-term u, such that t$5_{b[]}$u and t'$5_{b[]}$u, and a l[]-term t is b[]-normalizable if and only if there is a b[]-normal l[]-term u such that t$:_{b[]}$u.

3.3 The b[]-left reduction

Definitions.

- A sequence of symbols of the form (l or [corresponds to a redex. We may then define the **leftmost b-redex** and the **leftmost []-redex** of t. If t' is the l[]-term obtained by contracting this redex, we say that :

t gives t' by **b$_0$-left reduction** (resp. by **[]$_0$-left reduction**, resp. by **b[]$_0$-left reduction**), and write by **t∑∑$_{b0}$t'** (resp. **t∑∑$_{[]0}$t'**, resp. **t∑∑$_{b[]0}$t'**), if it is a b-redex (resp. a []-redex, resp. a b-redex or a []-redex).

- We say that t reduces to t' by **b-left reduction** (resp. **[]-left reduction**, resp. **b[]-left reduction**), and we write **t∑∑$_b$t'** (resp. **t∑∑$_{[]}$t'**, resp. **t∑∑$_{b[]}$t'**) if and only if t' is obtained from t by a sequence of b$_0$-left reductions (resp. of []$_0$-left reductions, resp. of b[]$_0$-left reductions).

- A l[]-term t is said **b[]-left normalizable** if and only if there is a b[]-normal l[]-term t', such that t∑∑$_{b[]}$t'.

Theorem 3.3.1. *u is b[]-left normalizable if and only if l(u) is left normalizable.*

Theorem 3.3.2 (normalization theorem). *u is b[]-normalizable if and only if u is b[]-left normalizable.*

Proof of theorem 3.3.1.

1 Use lemmas 3.2.7 and 3.3.3.

Lemma 3.3.3.

1) If R is the leftmost b-redex of u, then l(R) is the leftmost redex of l(u).
2) If u∑∑$_{b0}$v, then l(u)∑∑$_0$l(v).

Proof.

1) Clear.

2) By induction on u. The only non-trivial case is u=(lxt)w : then we have v=t[w/x], then, by lemma 3.2.10, l(u)=(lxl(t))l(w)∑∑$_0$l(t)[l(w)/x]= l(t[w/x]). ■

0 If not, there is an infinite sequence of l[]-terms u$_0$=u,u$_1$,…,u$_n$,…, such that u$_i$∑∑$_{b0}$u$_{i+1}$ or u$_i$∑∑$_{[]0}$u$_{i+1}$ for i≥0. Therefore, by lemmas 3.2.2, 3.2.7, and 3.3.3, there is an infinite sequence of l[]-terms v$_0$=l(u),v$_1$,…,v$_n$,…, such that v$_i$∑∑$_0$v$_{i+1}$ for i≥0, therefore l(u) is not left normalizable. A contradiction. ■

Proof of theorem 3.3.2.

0 Clear.

1 If u is b[]-normalizable, then l(u) is normalizable (same proof as theorem 3.3.1 **1**). By the normalization theorem of l-calculus, l(u) is left normalizable, therefore, by theorem 3.3.1, u is b[]-left normalizable. ■

3.4 The b[]-head reduction

Proposition 3.4.1. *Every l[]-term t can be - uniquely - written as $lx_1\ldots lx_n(R)t_1\ldots t_m$ $n,m\geq 0$, R being a variable or a redex*

Proof. By induction on t. ■

Definitions.

- Let t be a l[]-term, then, by proposition 3.4.1, $t=lx_1\ldots lx_n(R)t_1\ldots t_m$.

If R is a variable, we say that t is a **b[]-head normal form**.

If R is a redex, we say that R is the **head redex** of t.

If t' is the l[]-term obtained from t by contracting its head redex, we say that :

t gives t' by **b_0-head reduction** (resp. by **$[]_0$-head reduction**, resp. by **$b[]_0$-head reduction**), and we write **$t\sum_{b0}t'$** (resp. **$t\sum_{[]0}t'$**, resp. **$t\sum_{b[]0}t'$**), if the head redex is a b-redex (resp. a []-redex, resp. a b-redex or a []-redex).

- We say that t reduces to t' by **b-head reduction** (resp. **[]-head reduction**, resp. **b[]-head reduction**), and we write **$t\sum_{b}t'$** (resp. **$t\sum_{[]}t'$**, resp. **$t\sum_{b[]}t'$**) if and only if t' is obtained from t by a sequence of b_0-head reduction (resp. $[]_0$-head reduction, resp. $b[]_0$-head reduction).

A b[]-head reduction is, in particular, a b[]-left reduction.

- If $t\sum_{b[]}t'$, we denote by **n(t,t')**, the number of steps to go from t to t'.

- A l[]-term t is said **b[]-solvable** if and only if for every l[]-term u, there are variables $x_1,\ldots,x_k$, and l[]-terms $u_1,\ldots,u_k,v_1,\ldots,v_l$ $k,l\geq 0$, such that $(t[u_1/x_1,\ldots,u_k/x_k])v_1\ldots v_l :_{b[]}u$.

Theorem 3.4.2. *If $t\sum_{b[]}t'$, then for every $u_1,\ldots,u_r A L[]$:*

1) There is $v A L$, such that $(t)u_1\ldots u_r\sum_{b[]}v$, $(t')u_1\ldots u_r\sum_{b[]}v$, and $n((t)u_1\ldots u_r,v)= n((t')u_1\ldots u_r,v)+n(t,t')$.

2) $t[u_1/x_1,\ldots,u_r/x_r]\sum_{b[]}t'[u_1/x_1,\ldots,u_r/x_r]$, and $n(t[u_1/x_1,\ldots,u_r/x_r],t'[u_1/x_1,\ldots,u_r/x_r])= n(t,t')$.

Remarks.

- 1) shows that to make the b[]-head reduction of $(t)u_1\ldots u_n$, it is equivalent (same result, and same number of steps) to make some steps in the b[]-head reduction of t, and then make the b[]-head reduction of $(t')u_1\ldots u_n$.
- 2) shows that to make the b[]-head reduction of $t[u_1/x_1,\ldots,u_n/x_n]$, it is equivalent (same result, and same number of steps) to make some steps in the b[]-head reduction of t, and then make the b[]-head reduction of $t'[u_1/x_1,\ldots,u_n/x_n]$.

Corollary 3.4.3. *Let $t,u_1,\ldots,u_n,v_1,\ldots,v_m$AL[]. If the b[]-head reduction of $(t[u_1/x_1,\ldots,u_n/x_n])v_1\ldots v_m$ terminates, then the b[]-head reduction of t terminates.*

Proof. Use theorem 3.4.2. ■

Theorem 3.4.4 (resolution theorem). *The following conditions are equivalent :*
1) t is b[]-solvable ;
2) the b[]-head reduction of t terminates ;
3) t is b[]-equivalent to a b[]-head normal form.

Proof of theorem 3.4.2. It is enough to do the proof for one step of reduction.
1) By induction on r ; it is enough to do the proof for r=1. Then $t=lx_1\ldots lx_n(R)t_1\ldots t_m$, and $t'=lx_1\ldots lx_n(R')t_1\ldots t_m$ where R' is the contractum of R.
If n=0, then $(t)u=(R)t_1\ldots t_m u$, and $(t')u=(R')t_1\ldots t_m u$, therefore $(t)u\sum_{b[]}v$, where v=(t')u.
If n≥1, then one step of b[]-head reduction of (t)u gives $lx_2\ldots lx_n(\underline{R})\underline{t}_1\ldots\underline{t}_m$ (where $\underline{w}=w[u/x_1]$ for every wAL[]). One step of b[]-head reduction of (t')u gives $lx_2\ldots lx_n(\underline{R'})\underline{t}_1\ldots\underline{t}_m$.

Lemma 3.4.5. *If R a redex, R' its contractum, and $u_1,\ldots,u_m$AL[], then $R[u_1/y_1,\ldots,u_m/y_m]$ is a redex, and $R'[u_1/y_1,\ldots,u_m/y_m]$ is its contractum.*

Proof. If R is a b-redex, then R=(lxu)v, and R'=u[v/x].
$R[u_1/y_1,\ldots,u_m/y_m]=(lxu[u_1/y_1,\ldots,u_m/y_m])v[u_1/y_1,\ldots,u_m/y_m]$ is a b-redex, and its contractum is $u[u_1/y_1,\ldots,u_m/y_m][v[u_1/y_1,\ldots,u_m/y_m]/x]=R'[u_1/y_1,\ldots,u_m/y_m]$.

If R is a []-redex, then $R=[t]<\mathbf{a}/\mathbf{x}>$:
- If $t=x_i$ $1\leq i\leq n$, then $R'=a_i$. $R[u_1/y_1,\ldots,u_m/y_m]=[t]<\mathbf{a}[u_1/y_1,\ldots,u_m/y_m]/\mathbf{x}>$ is a []-redex, and its contractum is $a_i[u_1/y_1,\ldots,u_m/y_m]=R'[u_1/y_1,\ldots,u_m/y_m]$.
- If t=lxu, then $R'=ly[u]<\mathbf{a}/\mathbf{x},y/x>$ where yFv($\mathbf{a}$).
$R[u_1/y_1,\ldots,u_m/y_m]=[lxu]<\mathbf{a}[u_1/y_1,\ldots,u_m/y_m]/\mathbf{x}>$ is a []-redex, and its contractum is $ly[u]<\mathbf{a}/\mathbf{x},y/x>[u_1/y_1,\ldots,u_m/y_m]=R'[u_1/y_1,\ldots,u_m/y_m]$ where yFv($\mathbf{a}$)"
- If t=(u)v, then $R'=([u]<\mathbf{a}/\mathbf{x}>)[v]<\mathbf{a}/\mathbf{x}>$.

$R[u_1/y_1,\ldots,u_m/y_m]=([u]<\mathbf{a}/\mathbf{x}>)[v]<\mathbf{a}[u_1/y_1,\ldots,u_m/y_m]/\mathbf{x}>$ is a []-redex, and its contractum is $([u]<\mathbf{a}/\mathbf{x}>)[v]<\mathbf{a}/\mathbf{x}>[u_1/y_1,\ldots,u_m/y_m]=R'[u_1/y_1,\ldots,u_m/y_m]$. ■

By lemma 3.4.5, $(t)u\sum_{b[]}v$, and $(t')u\sum_{b[]}v$ where $v=lx_2\ldots lx_n(\underline{R'})\underline{t}_1\ldots\underline{t}_m$.
2) Same proof as 1). ■

Proof of theorem 3.4.4.
1)12) If t is b[]-solvable, then there are variables $x_1,\ldots,x_k$, and terms $u_1,\ldots,u_k,v_1,\ldots,v_l$ $(k,l\geq0)$, such that $(t[u_1/x_1,\ldots,u_k/x_k])v_1\ldots v_l:_{b[]}lxx$, therefore, by the Church-Rosser theorem, $(t[u_1/x_1,\ldots,u_k/x_k])v_1\ldots v_l\sum_{b[]}lxx$, therefore, by corollary 3.4.3, the b[]-head reduction of t terminates.
2)13) Clear.
3)11) Assume $t:_{b[]}lx_1\ldots lx_n(y\ )t_1\ldots t_m$, and let u be a l[]-term :
- If $y=x_i$ $1\leq i\leq n$, then $(((t)x_1\ldots x_{i-1})ly_1\ldots ly_m u)x_{i+1}\ldots x_n:_{b[]}u$ where $y_j Fv(u)$ $1\leq j\leq m$.
- If $y\neq x_i$ $1\leq i\leq n$, then $(t[ly_1\ldots ly_m u/y])x_1\ldots x_n:_{b[]}u$ where $y_j Fv(u)$ $1\leq j\leq m$.

Therefore t is b[]-solvable. ■

Lemma 3.4.6. *If $u\sum_{b0}v$, then $l(u)\sum_0 l(v)$.*

Proof. Same proof as lemma 3.4.5. ■

Theoreme 3.4.7. *u is b[]-solvable if and only if l(u) is solvable.*

Proof.
1 Use lemmas 3.2.7 and 3.4.6.
0 Otherwise there is an infinite sequence of l[]-terms $u_0=u,u_1,\ldots,u_n,\ldots$, such that $u_i\sum_{b0}u_{i+1}$ or $u_i\sum_{[]0}u_{i+1}$ for $i\geq0$. Therefore, by lemmas 3.2.7, 3.4.6, and 3.2.2, there is an infinite sequence of l[]-terms $v_0=l(u),v_1,\ldots,v_n,\ldots$, such that $v_i\sum_0 v_{i+1}$ for $i\geq0$, therefore l(u) is unsolvable. A contradiction. ■

§ 4. An equivalent definition for storage operators

Theorem 4.1. *Let t be a closed b-normal l-term, and T a closed l-term. T is an o.m.m. for t if and only if there is a l-term $t_t:_{by}t$, such that*
$T[t]f\sum_{b[]}(f)t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$.

To prove this theorem we need some definitions

Definition. Let t be a b-normal l-term, and u a l[]-term. We say that u is **directed by t** if and only if the directors of boxes of u are subterms of t.

More precisely u is directed by t if and only if :

- If u=x, then u is directed by t ;
- If u=lxv, then u is directed by t if and only if v is directed by t ;
- If u=(v)w, then u is directed by t if and only if v and w are directed by t ;
- If u=[v]<**a/x**>, then u is directed by t if and only if v is a subterm de t, and for all 1≤i≤na$_i$ is directed by t.

Lemma 4.2.

1) If u and v are directed by t, then u[v/x] is directed by t.

2) If u is directed by t, and u5$_{b[]}$v, then v is directed by t.

Proof. By induction on u. ■

Definition. Let t be a b-normal l-term. A **t-special application** h is a function from ST(t) to L which satisfies the following properties :

- h(x)∑x ;
- h(lxu)∑lxh(u) ;
- h((u)v)∑(h(u))h(v).

Lemma 4.3. *If h is a t-special application, then, for every uAST(t), h(u):$_b$u.*

Proof. by induction on u. ■

Lemma 4.4. *Let t be a b-normal l-term, and uAST(t). For every h$_u$:$_b$u, there is a t-special application h, such that h(u)=h$_u$.*

Proof. Let vAST(t) ; we define h(v) as follows :

- If vAST(u), h(v) is defined by induction on li(v)=lg(u)-lg(v), and we check that h(v):$_b$v.
  - If li(v)=0, then v=u. Take h(v)=h$_u$, we have h(v):$_b$v.
  - If li(v)≥1, then v is a proper subterm of u :
    - If there is an x, such that lxvAST(u) then by induction hypothesis, we have h(lxv):$_b$lxv, therefore h(lxv)∑lxh$_v$ where h$_v$:$_b$v. Take h(v)=h$_v$, we have h(v):$_b$v.
    - If there is wAST(t), such that (v)wAST(t) then by induction hypothesis, we have h((v)w):$_b$(v)w. Since t is b-normal, then h((v)w)∑(h$_v$)h$_w$ where h$_v$:$_b$v and h$_w$:$_b$w.

Take $h(v)=h_v$, we have $h(v):_b v$.

- If there is wAST(t), such that (w)vAST(t) then by induction hypothesis, we have $h((w)v):_b(w)v$. Since t is b-normal, then $h((w)v)∑(h_w)h_v$ where $h_v:_b v$, and $h_w:_b w$. Take $h(v)=h_v$, we have $h(v):_b v$.

- If uAST(v)\{v}, take h(v) the l-term v where u is replaced by h(u), we have $h(v):_b v$.

- Otherwise, we put h(v)=v.

By construction, h is a t-special application. ■

Definition. Let t be a b-normal l-term, and h a t-special application. The **t-special substitution** S_h is the function from the set of l[]-terms directed by t into L defined by induction :

- If u=x, then $S_h(u)=x$;
- If u=lxv, then $S_h(u)=lyS_h(v[y/x])$ where yFv(h(t)) ;
- If u=(v)w, then $S_h(u)=(S_h(v))S_h(w)$;
- If u=[v]<**a**/**x**>, then $S_h(u)=h(v)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]$.

A t-special substitution is the function S_h associated to a b-normal l-term t, and some t-special application h.

It is easy to see that if u does not contain boxes, then $S_h(u)=u$.

Lemma 4.5. *If $y_1,\ldots,y_m$Fv(h(t)), then*
$S_h(u[v_1/y_1,\ldots,v_m/y_m]) =S_h(u)[S_h(v_1)/y_1,\ldots,S_h(v_m)/y_m]$.

Proof. By induction on u. ■

Lemma 4.6. *$S_h(u):_b l(u)$.*

Proof. By induction on u. ■

Lemma 4.7. *If $u∑_{b[]} v$, then $S_h(u):S_h(v)$.*

Proof. It is enough to do the proof for one step of reduction.

Let $u=lx_1\ldots lx_n(R)u_1\ldots u_m$, and $v=lx_1\ldots lx_n(R')u_1\ldots u_m$ where R' is the contractum of redex R :

If R=(lxa)b, then R'=a[b/x].

$S_h((lxa)b)=(lyS_h(a[y/x]))S_h(b)∑S_h(a[y/x])[S_h(b)/y]=S_h(a)[y/x][S_h(b)/y]$
$=S_h(a)[S_h(b)/x]$, therefore, by lemma 4.5, $S_h((lxa)b)∑S_h(a[b/x])$.

If R=[u]<**a**/**x**> :

- If $u=x_i$ $1\leq i\leq n$,then $R'=a_i$, and $S_h(R)=S_h(R')$.

- If u=lxv, then R'=ly[v]<**a**/**x**,y/x> where yFv(**a**).
$S_h(R)=h(u[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]$∑$lxh(v)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]=$
$lzh(v)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n,z/x]$ where zFv(h(t))"Fv(**a**), therefore
$S_h(R)$∑$S_h(R')$.

- If u=(c)d, then R'=([c]<**a**/**x**>)[d]<**a**/**x**>
$S_h(R)=h(u)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]$∑$(h(c))h(d)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]=$
$(h(c)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n])h(d)[S_h(a_1)/x_1,\ldots,S_h(a_n)/x_n]$,
therefore $S_h(R)$∑$S_h(R')$. ■

Corollary 4.8. *u is b[]-solvable if and only if $S_h(u)$ is solvable.*

Proof.
1 Use lemma 4.7.
0 $S_h(u)$ is solvable, therefore, by lemma 4.6, l(u) is solvable, therefore, by theorem 3.4.7, u is b[]-solvable. ■

Definition. We say that a l[]-term t is **good** if and only if there is a l-term u, such that $t=u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, and for all $1\leq i\leq m$ if $\mathbf{a_i}=a_{1,i},\ldots,a_{n_i,i}$, then $a_{j,i}$ is good $1\leq j\leq n$.

It is clear that we have :

- x is good ;
- If lxt is good, then t is good ;
- (u)v is good if and only if u and v are good ;
- [w]<**a**/**x**> is good if and only if a_i is good $1\leq i\leq n$.

Example. The l[]-term $[x_1]<x/x_1>$ is good, but the l[]-term $lx[x_1]<x/x_1>$ is not. Indeed, the variable x becomes bounded, and so we can not find a l-term u, such that $lx[x_1]<x/x_1>=$ u[[t]<**a**/**x**>/y].

Lemma 4.9. *If $t,v_1,\ldots,v_r$ are good, then $t[v_1/y_1,\ldots,v_r/y_r]$ is good.*

Proof. By induction on t. ■

Definitions.
- A **[]'-redex** is a l[]-term of the form $([ly_1\ldots ly_m(y)u_1\ldots u_r]<\mathbf{a}/\mathbf{x}>)v_1\ldots v_m$. Its contractum R is defined by : $R=(b)[u_1]<\mathbf{a}/\mathbf{x},\mathbf{v}/\mathbf{y}>\ldots[u_r]<\mathbf{a}/\mathbf{x},\mathbf{v}/\mathbf{y}>$ where $b=v_i$ if $y=y_i$ $1\leq i\leq m$, and $b=a_i$ if $y=x_i$ $1\leq i\leq n$.

It is easy to see that if R is a []'-redex, and R' its contractum, then R$∑_{b[]}$R'.

Let $t=lx_1\ldots lx_n(R)t_1\ldots t_m$ where R is a []'-redex. If t' is the l[]-term obtained from t by contracting the []'-redex R, we say that t gives t' by a **[]'$_0$-head reduction**, and we write **$t\sum_{[]'0}t'$**.

We say that t reduces to t' by **[]'-head reduction**, and we write **$t\sum_{[]'}t'$** if and only if t' is obtained from t by a sequence of []'$_0$-head reductions.

- If t' is the l[]-term obtained from t by contracting its head redex (b-redex or []'-redex), we say that t gives t' by **b[]'$_0$-head reduction**, and we write **$t\sum_{b[]'0}t'$**.

We say that t reduces to t' by **b[]'-head reduction**, and we write **$t\sum_{b[]'}t'$** if and only if t' is obtained from t by a sequence of b[]'$_0$-head reductions.

- A head reduction $t\sum_b t'$ is said **complete** if and only if for every l[]-term u, if $t'\sum_b u$, then t'=u.

Lemma 4.10.
1) If f is a variable, and $t\sum_{b[]}(f)u_1,\ldots u_r$, then there is a sequence $t_0=t,t_1,\ldots,t_n=(f)u_1,\ldots u_r$, such that $t_i\sum_b t_{i+1}$ is complete or $t_i\sum_{[]'}t_{i+1}$ $0\leq i\leq n\text{-}1$.
2) If moreover t is directed by u, then every director of t_i $0\leq i\leq n$is an element of STE(u).

Proof.
1) If $t\sum_{b[]}(f)u_1,\ldots u_r$, then there is a sequence $t=(v_0)w_0,(v_1)w_1,\ldots,(v_m)w_m=(f)u_1,\ldots u_r$, such that $(v_i)w_i\sum_{b0}(v_{i+1})w_{i+1}$ or $(v_i)w_i\sum_{[]0}(v_{i+1})w_{i+1}$ $0\leq i\leq m\text{-}1$. If $(v_i)w_i\sum_{[]0}(v_{i+1})w_{i+1}$ $0\leq i\leq m\text{-}1$, then $(v_i)w_i=([ly_1\ldots ly_p(y)d_1\ldots d_q]<\mathbf{a}/\mathbf{x}>)b_1\ldots b_p\ c_1\ldots c_s$. Therefore there is j>i, such that $(v_i)w_i\sum_{[]'0}(v_j)w_j$, therefore there is a sequence $t=(v'_0)w'_0,(v'_1)w'_1,\ldots,(v'_k)w'_k=(f)u_1,\ldots u_r$, such that $(v'_i)w'_i\sum_b(v'_{i+1})w'_{i+1}$ or $(v'_i)w'_i\ \sum_{[]'}(v'_{i+1})w'_{i+1}$ $0\leq i\leq k\text{-}1$. Gathering consecutive b-reductions, it is clear that we can suppose that the b-reductions $(v'_i)w'_i\sum_b(v'_{i+1})w'_{i+1}$ are complete.

2) Easy. ■

Lemma 4.11. *Let t be a good l[]-term.*
1) If $t\sum_b t'$ then, t' is good.
2) If $t\sum_{[]'}(a)b$, then (a)b is good.
3) If $t\sum_{b[]}(f)u_1\ldots u_r$, then $u_1,\ldots,u_r$ are good.

Proof. If t is good, then there is a l-term u, such that $t=u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, and for all $1\leq i\leq m$ if $\mathbf{a_i}=a_{1,i},\ldots,a_{ni,i}$, then $a_{j,i}$ is good $1\leq j\leq$ņ.

1) It is enough to do the proof for one step of reduction. If $t\sum_{b0}t'$, then $t'=u'[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$ where $u\sum_0 u'$, therefore t' is good.

2) It is enough do the proof for one step of reduction. For every l[]-term u, denote by u''

the l[]-term $u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$.

It is clear that we may suppose that $y_i Fv(\mathbf{a_j})$ $1\leq i,j\leq m$.

If $t\sum_{[]'0}(a)b$, then $u=(y_i)v_1\ldots v_q w_1\ldots w_s$ $1\leq i\leq m$, $t_i=lf_1\ldots lf_q(y)u_1\ldots u_r$, and $(a)b=\{(c)z_1\ldots z_r w_1\ldots w_s\}[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m,[u_1]<\mathbf{a_i}/\mathbf{x_i},\mathbf{v''}/\mathbf{f}>/y_1,\ldots,$ $[u_r]<\mathbf{a_i}/\mathbf{x_i},\mathbf{v''}/\mathbf{f}>/z_r]$ where $c=v_i$ if $y=f_i$ $1\leq i\leq q$, and $c=a_{j,i}$ if $y=x_{j,i}$ $1\leq i\leq m$, $1\leq j\leq n_i$, and $z_1,\ldots,z_r$ are a new variables, therefore, by lemma 4.9, (a)b is good.

3) Use lemma 4.10, and 1) and 2). ■

Proof of theorem 4.1.

1 If T is an o.m.m. for t, then there is $d_t:_{by}t$, such that for every $h_t:_b t$, there is a substitution s, such that $Th_t f\sum(f)s(d_t)$. $l(T[t]f)=Ttf$ is solvable, therefore, by theorem 3.4.7, $T[t]f$ is b[]-solvable, and $T[t]f\sum_{b[]}(f)t'$. By lemma 4.4, let h be a t-special application, such that $h(t)=h_t$. Then $S_h(t')=s(d_t)$. $T[t]f$ is a good l[]-term, therefore, by lemma 4.11, t' is good, therefore $t'=t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$ where t_t is a l-term. Therefore $S_h(t')=t_t[h_1/y_1,\ldots,hm/y_m]$ where $h_i=h(t_i)[S_h(a_{i,1})/x_{i,1},\ldots,S_h(a_{i,mi})/x_{i,mi}]$ $1\leq i\leq m$, therefore, by lemmas 2.2.4, 2-6, and 4.3, $t_t=s'(d_t)$, therefore $t_t:_{by}t$.

0 Assume that $T[t]f\sum_{b[]}(f)t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, and $t_t:_{by}t$. Let $h_t:_b t$. By lemma 4.4, let h be a t-special application, such that $h(t)=h_t$. By lemma 4.7, we have $S_h(T[t]f)=Th_t f\ \sum(f)t_t[h_1/y_1,\ldots,hm/y_m]$. Therefore T is an o.m.m. for t. ■

Examples.

- Tis an o.m.m. for P_n. Indeed,

$T[P_)]f\sum_{b0}([P])lf(f)P\ldots lf(f)Pf\sum_{[]'0}(lf(f)P)f\sum_{b0}(f)P1\leq i\leq n$.

- Tis an o.m.m. for $\underline{N}$. Indeed,

$T[\underline{n}]f\sum_{b0}([\underline{n}])Ff\underline{0}\sum_{[]'0}((F)[(x_1)^{n-1}x_2]<F/x_1,f/x_2>)\underline{0}\sum_b([(x_1)^{n-1}x_2]<F/x_1,f/x_2>)(\underline{s})\underline{0}$ $\sum_{[]'0}((F)[(x_1)^{n-2}x_2]<F/x_1,f/x_2>)(\underline{s})\underline{0}\sum_b([(x_1)^{n-2}x_2]<F/x_1,f/x_2>)(\underline{s})^2\underline{0}\sum_{[]'0}\ldots\sum_b$ $([x_2]<F/x_1,f/x_2>)(\underline{s})^n\underline{0}\sum_{[]'0}(f)(\underline{s})^n\underline{0}$.

§ 5. Properties of storage operators

5.1 Storage operators and b-equivalence

Theorem 5.1.1. *Let t be a closed b-normal l-term, T and T' be closed l-terms. If T is an o.m.m. for t, and $T':_bT$, then T' also is an o.m.m. for t.*

Proof. On the set of good l[]-terms, we define an **equivalence relation g** by :
If $t=u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$ where u is a l-term, then tgt' if and only if $t'=u'[[t_1]<\mathbf{a'_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a'_m}/\mathbf{x_m}>/y_m]$ where $u:_bu'$, and for all $1\leq i\leq m$ if $\mathbf{a_i}=a_{1,i},\dots,a_{n_i,i}$, then $\mathbf{a'_i}=a'_{1,i},\dots,a'_{n_i,i}$, and $a_{j,i}ga'_{j,i}$ $1\leq j\leq n_i$.
It is clear that if $lx_1\dots lx_n(f)u_1\dots u_mgt$ (f is a variable), then $t=lx_1\dots lx_n(f)u'_1\dots u'_m$ where $u_igu'_i$ $0\leq i\leq m$.
Lemma 5.1.2. *If tgt', and $v_igv'_i$ $1\leq i\leq r$,then $t[v_1/y_1,\dots,v_r/y_r]gt'[v'_1/y_1,\dots,v'_r/y_r]$.*

Proof. By induction on t. ■

Lemma 5.1.3. *Let t be a good l[]-term.*
1) If $t\sum_bt'$ is complete, and tgT, then for someT' : t'gT', and $T\sum_bT'$ is complete.
2) If $t\sum_{[]'0}(c)d$, and tgT, then tor someT' with the same b[]-head normal form as T : (c)dgT'.
3) If $t\sum_{b[]}(f)u_1\dots u_r$, and tgT, then tor someT' : $(f)u_1\dots u_rgT'$, and $T\sum_{b[]}T'$.

Proof. If t is good, then there is a l-term u, such that
$t=u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$ where $\mathbf{a_i}=a_{1,i},\dots,a_{n_i,i}$ $1\leq i\leq n$.
1) If $t\sum_bt'$ is complete, then $t'=u'[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$ where u' is the head normal form of u. If tgT, then $T=U[[t_1]<\mathbf{a'_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a'_m}/\mathbf{x_m}>/y_m]$ where $u:_bU$, $\mathbf{a'_i}=a'_{1,i},\dots,a'_{n_i,i}$, and $a_{j,i}ga'_{j,i}$ $1\leq j\leq n_i$. Let U' be the head normal form of U.
Let $T'=U'[[t_1]<\mathbf{a'_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a'_m}/\mathbf{x_m}>/y_m]$. It is clear that we have t'gT', and $T\sum_bT'$ is complete.
2) For every l[]-term u, we denote by u" the l[]-term $u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\dots, [t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$.
It is clear that mat suppose that $y_iFv(\mathbf{a_j})$ $1\leq i,j\leq m$.
If $t\sum_{[]'0}(c)d$, then $u=(y_i)v_1\dots v_qw_1\dots w_s$ $1\leq i\leq m$, $t_i=lf_1\dots lf_q(y)u_1\dots u_r$, and
$(c)d=\{(b)z_1\dots z_rw_1\dots w_s\}[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m,[u_1]<\mathbf{a_i}/\mathbf{x_i},\mathbf{v''}/\mathbf{f}>/z_1,\dots,[u_r]<\mathbf{a_i}/\mathbf{x_i},\mathbf{v''}/\mathbf{f}>/z_r]$ where $b=v_i$ if $y=f_i$ $1\leq i\leq q$, and $b=a_{j,i}$ if $y=x_{j,i}$ $1\leq i\leq m$, and $1\leq j\leq n_i$, and $z_1,\dots,z_r$ are a new variables.
If tgT, then $T=U[[t_1]<\mathbf{a'_1}/\mathbf{x_1}>/y_1,\dots,[t_m]<\mathbf{a'_m}/\mathbf{x_m}>/y_m]$ where $u:_bU$, $\mathbf{a'_i}=a'_{1,i},\dots,a'_{n_i,i}$ and $a_{j,i}ga'_{j,i}$ $1\leq j\leq n_i$. Since $u:_bU$, then $U\sum(y_i)c_1\dots c_qd_1\dots d_s$ where $v_i:_bc_i$ $1\leq i\leq q$, and $w_j:_bd_j$ $1\leq j\leq s$.
For every l[]-term u, we denote by u'" the l[]-term $[[t_1]<\mathbf{a'_1}/\mathbf{x_1}>/y_1,\dots, [t_m]<\mathbf{a'_m}/\mathbf{x_m}>/$

y_m].

Let T'={(b)z_1…$z_r$$d_1$…$d_s$}[[$t_1$]<$\mathbf{a'_1/x_1}$>/$y_1$,…,[$t_m$]<$\mathbf{a'_m/x_m}$>/$y_m$,[$u_1$]<$\mathbf{a'_i/x_i,c'''/f}$> /$z_1$,…, [$u_r$]<$\mathbf{a'_i/x_i,c'''/f}$>/$z_r$] where b'=$c_i$ if y=f_i 1≤i≤q, and b'=$a'_{j,i}$ if y=$x_{j,i}$ 1≤i≤m, and 1≤j≤n_i.

It is clear that T and T' have the same b[]-head normal form, and, by lemma 5.1.2, t'gT'.

3) Use 1), 2), and lemma 4.10. ■

If T is an o.m.m. for t, then, by theorem 4.1,
T[t]f ∑$_{b[]}$ (f)t_t[[t_1]<$\mathbf{a_1/x_1}$>/y_1,…,[t_m]<$\mathbf{a_m/x_m}$>/y_m], and t_t:$_{by}$t. If T':$_b$T, then T'[t]fgT[t]f, therefore, by 3) of lemma 5.1.3,
T'[t]f ∑$_{b[]}$ (f)t'_t[[t_1]<$\mathbf{a'_1/x_1}$>/y_1,…,[t_m]<$\mathbf{a'_m/x_m}$>/y_m], and t'_t:$_b$$t_t$:$_{by}$t. Therefore, by theorem 4.1, T' is an o.m.m. for t. ■ (of theorem 5.1.1)

5.2 Decidability

Theorem 5.2.1. *If X is a non trivial set of closed l-terms stable by b-equivalence, then X is not recursive.*

Proof. See [2], [5], and [14]. ■

Theorem 5.2.2. *The set of o.m.m. for a set of closed b-normal l-terms is not recursive.*

Proof. Use theorems 5.1.1 and 5.2.1. ■

Theorem 5.2.3. *The set of o.m.m. for a finite set of closed b-normal l-terms is recursively enumerable.*

Proof. Use theorem 4.1. ■

5.3 Storage operators and y-equivalence

Theorem 5.3.1. *Let t be a closed b-normal l-term, and T be closed l-term.*
If T is an o.m.m. for t, and t5yt', then T also is an o.m.m. for t'.

Remark. The theorem 5.3.1 is no more true if we replace t5$_y$t' by t:$_y$t'. Indeed, if we take t=lxx, t'=lxlylz((x)y)z, and T=ln(n)lf(f)lxx, then :

- $t'5_yt$, therefore $t:_yt'$.
- For every l-term u such that $u:_bt$, (T)u∑lf(f)lxx, therefore T is an o.m.m. for t.
- (T)t'f∑lz((f)lxx)z, therefore T is not an o.m.m. for t'.

Proof of theorem 5.3.1. On the set L[], we define the **binary relation c** as the least relation satisfying :

- tct ;
- If tct', then lxtclxt';
- If ucu', and vcv', then (u)vc(u')v';
- If $t5yx_i$, and $a_ica'_i$ $1\leq i\leq n$, then [t]<**a**/**x**>ca'_i ;
- If t5yt', and $a_ica'_i$ $1\leq i\leq n$, then [t]<**a**/**x**>c[t']<**a'**/**x**>.

It is clear that :

- If $lx_1\ldots lx_n(u_0)u_1\ldots u_m$ct, then $t=lx_1\ldots lx_n(u'_0)u'_1\ldots u'_m$ where $u_icu'_i$ $0\leq i\leq m$.

- Let t be a good l[]-term, therefore there is a l-term u, such that $t=u[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, and for all $1\leq i\leq m$ if $\mathbf{a_i}=a_{1,i},\ldots,a_{ni,i}$, then $a_{j,i}$ is good $1\leq j\leq n_i$. If tct', then it is easy to check that $t'=u[c_1/y_1,\ldots,c_n/y_m]$ where $c_i=[t'_i]<\mathbf{a'_i}/\mathbf{x_i}>$ with $t_i5_yt'_i$, $a_{j,i}ca'_{j,i}$ $1\leq i\leq m$, $1\leq j\leq n_i$, or $c_i=a'_{j,i}$ $1\leq j\leq n$ with $t_i5_yx_{j,i}$ and $a_{j,i}ca'_{j,i}$ $1\leq i\leq m$, $1\leq j\leq n_i$.

Lemma 5.3.2. *If ucu', and $v_icv'_i$ $1\leq i\leq n$, then :*
$u[v_1/x_1,\ldots,v_n/x_n]c\ u'[v'_1/x_1,\ldots,v'_n/x_n]$.

Proof. By induction on u. ■

Lemma 5.3.3. *If $u=lx_1\ldots lx_n(y)u_1\ldots u_m5_yv$, then $v=lx_1\ldots lx_{n-r}(y)u'_1\ldots u'_{m-r}$ where $u_j5_yu'_j$ $1\leq j\leq m-r$, $u_{m-s}5_yx_{n-s}$ $0\leq s\leq r-1$, and $x_{n-s}\neq y$ does not appear in $u_1,\ldots u_{m-r}$.*

Proof. By induction on the number n of y_0-reductions to go from u to v.

n=0 : clear.

If $n\geq 1$, then $u5_yw5_{y0}v$, therefore $w=lx_1\ldots lx_{n-r}(y)u'_1\ldots u'_{m-r}$ where $u_j5_yu'_j$ $1\leq j\leq m-r$, $u_{m-s}5_yx_{n-s}$ $0\leq s\leq r-1$, and $x_{n-s}\neq y$ does not appear in the $u_1,\ldots u_{m-r}$. Since $w5_{y0}v$, then $v=lx_1\ldots lx_{n-r}(y)u'_1\ldots u''_k\ldots u'_{m-r}$ where $u'_k5_yu''_k$, or $u'_{m-r}=x_{n-r}$, $x_{n-r}\neq y$ does not appear in the $u_1,\ldots,u_{m-r-1}$, and $v=lx_1\ldots lx_{n-r-1}(y)u'_1\ldots u'_{m-r-1}$ as required. ■

Lemma 5.3.4.

1) If $t\sum_{b0}t'$, and tcT, then for some t' : t'cT', and $T\sum_{b0}T'$.

2) If $t\sum_{[]'0}t'$, and tcT, then for some t' : t'cT', and $T\sum_{[]'0}T'$.

3) If $t\sum_{b[]'}t'$, and tcT, then for some t' : t'cT', and $T\sum_{b[]'}T'$.

Proof.

1) If $t\sum_{b0}t'$, then $t=lx_1\ldots lx_n(lxu)vt_1\ldots t_m$, and $t'=lx_1\ldots lx_n(u[v/x])t_1\ldots t_m$. If tcT, then $T=lx_1\ldots lx_n(lxu')v't'_1\ldots t'_m$ where ucu', vcv', and $t_ict'_i$ $1\leq i\leq m$.

Let $T'=lx_1\ldots lx_n(u'[v'/x])t'_1\ldots t'_m$. It is clear that $T\sum_{b0}T$ ', and, by lemma 5.3.2, t'cT'.

2) If $t\sum_{[]'0}t'$, then $t=ly_1\ldots ly_m([lz_1\ldots lz_k(y)u_1\ldots u_r]<\mathbf{a}/\mathbf{x}>v_1\ldots v_kw_1\ldots w_s$, and $t'=ly_1\ldots ly_m(b)[u_1]<\mathbf{a}/\mathbf{x},\mathbf{v}/\mathbf{z}>\ldots[u_r]<\mathbf{a}/\mathbf{x},\mathbf{v}/\mathbf{z}>w_1\ldots w_s$ where $b=v_i$ if $y=y_i$ $1\leq i\leq m$, and $b=a_i$ if $y=x_j$ $1\leq j\leq n$. Assume tcT.

- If $lz_1\ldots lz_k(y)u_1\ldots u_r5_yy$, then k=r, $u_i5_yz_i$ $1\leq i\leq m$, and $z_i\neq y=x_j$ $1\leq j\leq n$, then $T=ly_1\ldots ly_m(a'_j)v'_1\ldots v'_kw'_1\ldots w'_s$ where $a_jca'_j$, $v_icv'_i$ $1\leq i\leq k$, and $w_icw'_i$ $1\leq i\leq s$. Let T'=T. It is clear that t'cT', and $T\sum_{[]'0}T'$.

- If $lz_1\ldots lz_k(y)u_1\ldots u_r5_ylz_1\ldots lz_{k-l}(y)u'_1\ldots u'_{r-l}$ where $u_j5_yu'_j$ $1\leq j\leq r-l$, $u_{r-s}5_yz_{k-s}$ $0\leq s\leq l-1$, and $z_{k-s}\neq y$ does not appear in the $u_1,\ldots u_{r-l}$, then $T=ly_1\ldots ly_m([lz_1\ldots lz_{k-l}(y)u'_1\ldots u'_{r-l}]<\mathbf{a'}/\mathbf{x}>v'_1\ldots v'_kw'_1\ldots w'_s$ where $v_icv'_i$ $1\leq i\leq k$, and $w_icw'_i$ $1\leq i\leq s$. Let $T'=ly_1\ldots ly_m(b_i)[u_1]<\mathbf{a'}/\mathbf{x},v'_1/z_1,\ldots,v'_{k-l}/z_{k-l}>\ \ldots[u_r]<\mathbf{a'}/\mathbf{x},v'_1/z_1,\ldots,v'_{k-l}/z_{k-l}>\ v'_{m-l+1}\ldots v'_k\ w'_1\ldots w'_s$ where $b=v'_i$ if $y=y_i$ $1\leq i\leq m$, and $b=a_i$ if $y=x_j$ $1\leq j\leq n$. It is clear that t'cT', and $T\sum_{[]'0}T'$.

3) Use 1) and 2). ■

If T is an o.m.m. for t, then, by theorem 4.1, there is a l-term $t_t:_{by}t$, such that $T[t]f\sum_{b[]}(f)t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$.

If $t5_yt'$, then T[t]fcT[t']f, therefore, by lemma 5.3.4, $T[t']f\sum_{b[]}(f)t'$, and $t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]ct'$. Therefore there is a l-term t''_t, such that $t''_t:_bt_t:_{by}t$, and $t'=t''_t[[u_1]<\mathbf{b_1}/\mathbf{z_1}>/y_1,\ldots,[u_m]<\mathbf{b_r}/\mathbf{z_r}>/y_r]$. Therefore, by theorem 4.1, T is an o.m.m. for t'. ■ (of theorem 5.3.1)

5.4. Storage operators for a set of b-normal l-terms

Theorem 5.4.1. *Let $u_1,\ldots,u_n,v_1,\ldots,v_m$ be closed l-terms. Assume $u_{iby}u_j$ for i<j, there is a closed l-term T, such that $(T)u_i:_bv_i$ $1\leq i\leq n$.*

Proof. See [3]. ■

Theorem 5.4.2. *Every finite set of b-normal l-terms having all distinct by-normal forms has an o.m.m..*

Proof. Let D={t_1,…,t_n} be such a set. By theorem 5.4.1, there is a closed l-term T', such that T't_i:$_b$P1≤i≤n, therefore for every h_i:$_b$$t_i$, T'$h_i$:$_b$P, therefore T'$h_i$∑P. Let T=ln((T')n)lf(f)$t_1$…lf(f)$t_n$. It is easy to check that T is an o.m.m. for D. ■

Theorem 5.4.3. *Every finite set of b-normal l-terms has an o.m.m..*

Remarks.

- The theorem 5.4.3 is no more true if we remove the hypothesis "the l-terms of D are b-normal". If we take t_1=lxx, and t_2=(lx(x)x)lx(x)x, then D={t_1,t_2} have no o.m.m.. Indeed, if T is an o.m.m. for t_2, then, by corollary 2.2.3, T =lnlf(f)u_2 where u_2:$_{by}$$t_2$, therefore T is not an o.m.m. for t_1.

- The theorem 5.4.3 is no more true if we remove the hypothesis "D is finite". If we take D the set of all Pi≥1, then D have no o.m.m.. Indeed, if T is an o.m.m. for D, let T' its head normal form. By proposition 2.2.1, T'=lx_1…lx_e(x_i)t_1…t_n where e=1 or 2, and, by theorem 5.1.1, T' also is an o.m.m. for D. It is easy to prove that T' is not an o.m.m. for the l-term P.

Proof of theorem 5.4.3. Let D={t_1,…,t_n} be a finite set of b-normal l-terms. Gathering the l-terms having the same by-normal form, we can write D=where D_i={t,…,t}1≤i≤m, for all 1≤i≤m, and 1≤j,j'≤m_i, t^{by}=t^{by}, and for all 1≤i,i'≤m, t^{by}≠t^{by}.

Lemma 5.4.4. *Let t,t' be b-normal l-terms. If t:$_y$t', then there is a b-normal l-term u, such that u5$_y$t, and u5$_y$t'.*

Proof. By induction on t and t'. If t:$_y$t', then there is a b-normal l-term v, such that t5$_y$v, and t'5$_y$v. If v=lx_1…lx_n(y)v_1…v_m, then, by lemma 5.3.3, t=lx_1…lx_nly_1…ly_k(y)v'$_1$…v'$_m$$u_1$…$u_k$, and t'=l$x_1$…l$x_nly_1$…l$y_r$(y)v"$_1$…v"$_m$ w_1…w_r where v'$_i$5$_y$$v_i$, v"$_i5_y$$v_i$ 1≤i≤m, u_j5$_y$$y_j$ 1≤j≤k, y_j≠y does not appear in v_1,…v_m, w_j5$_y$$y_j$ 1≤j≤r, and y_j≠ydoes not appear in v_1,…v_m. Assume that k≤r. By induction hypothesis, let a_i be a b-normal l-term, such that a_i5$_y$v'$_i$, and a_i5$_y$v"$_i$ 1≤i≤m, and b_j be a b-normal l-term, such that b_j5$_y$$u_j$, and b_j5$_y$$w_i$ 1≤j≤k. Let u=lx_1…lx_nly_1…ly_r(y)a_1…$a_m$$b_1$…$b_k$$w_{k+1}$…$w_r$. It is clear that u is a b-normal l-term, and that u5$_y$t, and u5$_y$t'. ■

An **y-bound** for a set $B=\{u_1,\ldots,u_m\}$ is a b-normal l-term u, such that $u5_y u_i$ $1\leq i\leq m$.

Corollary 5.4.5. *Every finite set B of b-normal l- terms having all the same y-normal form has an y-bound.*

Proof. By induction on the number of l-terms of B using lemma 5.4.4. ■

By corollary 5.4.4, let u_i be a y-bound for D_i $1\leq i\leq m$. By theorem 5.4.2, the set $\{u_1, \ldots,u_m\}$ has an o.m.m., therefore, by theorem 5.3.1, D has an o.m.m.. ■ (of theorem 5.4.3)

5.5 Computation time of a storage operator

Lemma 5.5.1. *Let $(t_i)_{1\leq i\leq n}$and $(t'_i)_{1\leq i\leq n}$be sequences of l-terms, such that :*
1) For all $1\leq i\leq n$,$t_i\sum t'_i$.
2) For all $1\leq i\leq n$-1, $t_i=(u_i)v_{i,1}\ldots v_{i,ri}$, $t'_i=(u'_i)v_{i,1}\ldots v_{i,ri}$, and $u'_i\sum u_{i+1}$.
3) $t'_n=(f)v_1\ldots v_r$ where f is a variable.
Then $t_1\sum t'_n$, and $tps(t_1)=n(t_1,t'_n)=+$.

Proof. By induction on n.
n=1: trivial
for n>2 : Let $n_i=n(t_i,t'_i)$ and $m_i=n(u'_i,u_{i+1})$. By induction hypothesis, we have $t_2\sum t'_n$, and $n(t_2,t'_n)=+$.
$u'_1\sum u_2$, therefore, by theorem 1.2.1, for some w, $(u'_1)v_{1,1}\ldots v_{1,r1}\sum w$, $(u_2)v_{1,1}\ldots v_{1,r1}\sum w$, and $n((u'_1)v_{1,1}\ldots v_{1,r1},w)=n((u_2)v_{1,1}\ldots v_{1,r1},w)+n(u'_1,u_2)=$ $n((u_2)v_{1,1}\ldots v_{1,r1},w)+m_1$.
Therefore $t'_1\sum t'_n$, and $n(t'_1,t'_n)=n(t'_1,w)+n(w,t'_n)$. Therefore $t_1\sum t'_n$, and
$tps(t_1)=n(t_1,t'_n)=n(t_1,t'_1)+n(t'_1,w)+n(w,t'_n)=n_1+m_1++=+$. ■

Theorem 5.5.2. *Let t be a closed b-normal l-term, and T a closed l-term.*
If T is an o.m.m. for t, there are constants $A_{T,t}$ and $B_{T,t}$, such that for every $h_t{:}_b t$, $tps(Th_tf)\leq A_{T,t}Tps(h_t)+B_{T,t}$.

Proof.
If $t\sum_{b[]}t'$, denote by b(t,t'), the number of b_0-reductions used in this reduction.
For every vAL[], we define D(v) by induction on v :
- If [u]<**a/x**> is the head redex of v, then D(v)=u ;

- If not, D(v)=o where o is a constant.

Let h be a t-special application. For every uAST(t)"{o}, we define the integer $n_h(u)$ by :

- $n_h(x)=n_h(o)=0$;
- $n_h(lxu)=n(h(lxu),lxh(u))$;
- $n_h((u)v)=n(h((u)v),(h(u))h(v))$.

If T is an o.m.m. for t, then
$T[t]f\sum_{b[]}(f)t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, and $t_t:_{by}t$. There is a sequence of l[]-terms $t_0=T[t]f,t_1,\ldots,t_n=(f)t_t[[t_1]<\mathbf{a_1}/\mathbf{x_1}>/y_1,\ldots,[t_m]<\mathbf{a_m}/\mathbf{x_m}>/y_m]$, such that $t_{i-1}\sum_{b0}t_i$ or $t_{i-1}\sum_{[]0}t_i$ $1\leq i\leq n$.

Let $A_{T,t}$=Max{number of boxes directed by u and appearing in head position of t_i $0\leq i\leq n$,uAST(t)}, and $B_{T,t}=b(t_0,t_n)$.

Let $h_t:_bt$. By lemma 4.4, let h be a t-special application, such that $h(t)=h_t$.

By the proof of lemma 4.7, and by lemma 5.5.1, we have
$tps(Th_tf)=b(t_0,t_n)+$.

By theorem 1-3, $Tps(h_t)=n_h(u)$, and then $tps(Th_tf)\leq A_{T,t}Tps(h_t)+B_{T,t}$. ■

Remark. By the proof of theorem 5.5.2, we have $tps(Th_tf)=A_{T,t}Tps(h_t)+B_{T,t}$ if and only if, for all uAST(t), $A_{T,t}$=the number of boxes directed by u and appearing in head position of t_i $0\leq i\leq n$.

References

[1] **M. Abadi, L. Cardelli, P.L. Curien, and J.L. Levy.**
Explicit Substitutions. Technical report 1176, INRIA, 1990.
To appear in Journal of Functional Programming.

[2] **H. Barendregt.**
The lambda calculus : Its Syntax and Semantics.
North Holland, 1984.

[3] **C. Böhm, M. Dezani Ciancaglini, A. Peretti & S. Ronchi Della Rocca.**

A discriminator algorithm inside l-b-calculus.
Theorical Computer Science 8, (271-291), 1979.

[4] **P.L. Curien.**
The lr-calculi : an abstract framework for closures.
Technical report, LIENS - Ecole Normale Supérieure, 1988.

[5] **J.L. Krivine.**
Lambda calcul, évaluation paresseuse et mise en mémoire.
Theorical Informatics and Applications (R.A.I.R.O.), Vol. 25,1, p. 67-84, 1991.

[6] **J.L. Krivine.**
Lambda calcul, types et modèle.
Masson, Paris, 1990.

[7] **J.L. Krivine.**
Opérateurs de mise en mémoire et traduction de Gödel.
Arch. Math. Logic 30. (241-267), 1990.

[8] **K. Nour**.
Opérateurs de mise en mémoire en lambda-calcul pur et typé.
Thèse de doctorat, Université de Savoie, 1993.

[9] **K. Nour**.
Strong storage operators and data types.
Preprint of LAMA, 1993.

[10] **K. Nour**.
Opérateurs de mise en mémoire et typesV-positifs.
Submitted to Theorical Informatics and Applications, 1993.

[11] **K. Nour**.
Storage operators and V-positive types in system TTR.
Preprint of LAMA, 1993.

[12] **K. Nour**.
Opérateurs propres de mise en mémoire.
CRAS Paris, t. 317, Série I, p. 1.-.6, 1993.

[13] **K. Nour**.

Preuve syntaxique d'un théorème de J.L. Krivine sur les opérateurs de mise en mémoire.

CRAS Paris, t. 318, Série I, p. 201-204, 1994.

[14] **D.S. Scott**.

A systeme of functional abstraction.

Unpublished, 1963.

❈ ❈ ❈ ❈ ❈