3.

UNIVERSITA' DI ROMA

ISTITUTO DI AUTOMATICA

C.BATINI A.PETTOROSSI

SOME PROPERTIES OF SUBBASES IN WEAK COMBINATORY LOGIC

R.75-04

MARZO 1975

RAPPORTI DELL'ISTITUTO DI AUTOMATICA DELL'UNIVERSITÀ DI ROMA E DEL CENTRO
DI STUDIO DEI SISTEMI DI CONTROLLO E CALCOLO AUTOMATICI DEL C. N. R.

C. Batini - A. Pettorossi

SOME PROPERTIES OF SUBBASES IN WEAK COMBINATORY LOGIC

Ing. Carlo Batini, Centro di Studio dei Sistemi di Controllo
e Calcolo Automatici del C.N.R., Via Eudossiana, 18
00184 Roma
Ing. Alberto Pettorossi, Istituto di Automatica dell'Università di Roma, Via Eudossiana, 18 - 00184 Roma

This work was partially supported by "Consiglio Nazionale delle Ricerche"

ABSTRACT

In this paper weak combinatory logic as an algorithmic language is considered and various notions of structural and computational complexity are introduced. Particular attention is devoted to the definitional power of a system of combinators, that is to the concept of "subbase". Some results concerning the relations between specific subbases and their generative power are presented.

1. INTRODUCTION

The main purpose of this work is to introduce the concepts of computational complexity and subrecursiveness in combinatory logic [1]. Since we are interested in the algorithmic and computational properties of combinatory logic, we think that a fruitful way of approaching these propierties is to consider how limiting the power of the calculus (that is limiting the allowed definitions of "programs" or the allowed amount of "resource" used by programs) entails limitations on the ability of manipulating "data".

Studies on computational complexity have given well established and meaningful results for several abstract machines and languages such as Turing machines, LOOP programs, rewriting systems [2]. Less considered seems to have been the definition of suitable concepts of computational complexity in combinatory logic and λ-calculus. An interesting step in this direction was made by H.R. Strong [3] who defined a measure of depth of computation in a programming language based on Wagner's URS and showed that for each partial recursive function there is an index with uniformly bounded measure of computation.

In order to carry on our investigation on complexity properties of combinatory logic in §2 and §3 we shall examine properties of structural and computational complexity of combinators in weak calculus [4] and in §4 we examine the ability of combinators in generating pure applicative combinations[1]. For reasons that will be made clear later we restricted ourselves to considering only proper combinators [1] or, in general, combinators for which it is possible to define suitable input-output relations.

2. APPROACHES TO COMPUTATIONAL COMPLEXITY IN COMBINATORY LOGIC

The two basic notions of complexity in the literature are structural and computational complexity.

We now introduce the formal definitions of some possible measures of these notions. Structural complexity is inherent to a combinator as a static well formed object in a specific base.

DEFINITION 2.1. - *Lenght* (SL) of a combinator χ is recursively defined by the following rules:

i) if χ is a basic combinator, $SL(\chi) = 1$;

ii) if $\chi = (\chi_1\chi_2)$, $SL(\chi) = SL(\chi_1) + SL(\chi_2)$.

DEFINITION 2.2. - *Depth of the parenthesis structure* (SD) of a combinatory χ is defined by:

i) if χ is a basic combinator, $SD(\chi) = 0$;

ii) if $\chi = (\chi_1\chi_2)$, $SD(\chi) = 1 + \max\{SD(\chi_1), SD(\chi_2)\}$.

A simple relation between the two measures of structural complexity is the following:

Fact 2.1. - $\lceil \log_2 SL(w) \rceil \leq SD(w) \leq SL(w)-1$ for any combinator w.

Computational complexity measures are related to the reduction of combinators to normal form in the weak calculus.

The measures we may define depend on the computation rule choosen in the reduction process. Among the rules that guarantee to reach the normal form we will choose the standard (leftmost outermost) rule.

DEFINITION 2.3. - *Number of steps of computation* (CT) of a combinator χ is defined by:

i) $CT(\chi) = t$ if χ reaches a normal form in t steps;

ii) $CT(\chi)$ = undefined otherwise.

DEFINITION 2.4. - *Size of computation* (CS) of a combinator χ is defined by:

$CS(\chi) = \ell$ if $\ell = \max_i\{SL(\chi_i)\}$ where χ_i is a formula achieved during the reduction process.

DEFINITION 2.5. - *Depth of computation* (CD) of a combinator χ is defined by:

$CD(\chi) = d$ if $d = \max_i\{SD(\chi_i)\}$ where χ_i is a formula achieved during the reduction process.

Analogously to what is made [5,6,7] for acceptable Gödel numbering of partial recursive functions we will now give the following definitions for measures of complexity in the weak calculus of combinators:

(1) $|\ |$ is a structural complexity measure if:

i) $|\ |$ is a recursive mapping from the set of combinators to the integers;

ii) $\forall n$ the number of combinators w such that $|w| = n$, is finite.

(2) C is a computational complexity measure for a combinator w if C is a partial recursive mapping from the set of combinators to the integers and:

i) if a combinator w has normal form then $C(w)$ is defined;

ii) $C(w)$ is defined implies that "w has normal form" is decidable;

iii) $C(w) = n$ is decidable.

It is not difficult to verify that, if the cardinality of the base is finite, SL and SD are structural complexity measures, while CT, CS and CD are computational complexity measures (*).

(*) The first property of computational complexity measures is satisfied by the norm introduced in [8] in the definition of NURS.

As we have already remarked, in general the way the properties of complexity measures are studied is to consider how limitations on the measures result in limitations on the power of computational systems. On the other hand, for the above listed measures this does not seem to be the way of achieving interesting general results. It is certainly possible to simulate tape and time bounded Turing machines, restricted rewriting systems, primitive recursive computations, etc. in combinatory logic, and to define in this way the corresponding classes of combinators, but this approach does not give classifications of combinators well matched to the computational peculiarities of combinatory logic. These peculiarities are essentially:

i) the ability of "packing" data in such a way that "unpacking" is impossible from outside [9] and that data can be accessed in other than a sequential way;

ii) the "rightward" mechanism of operating of the calculus that makes it more similar to a tag-machine or to a non-erasing Turing machine than to other classical computing systems;

iii) the impossibility in a non typed calculus of an "a priori" distinction between programs and data and, inside a program, between primitives and constructs.

The last point is particularly interesting because the variability of the argument (or at least of the size of the argument) is at the base of the classification of the complexity of programs and functions (typically characterized by asyntotic behaviour).

3. COMPLEXITY MEASURES IN SUBBASES

Taking into account the type of properties of combinatory logic listed at the end of §2, we think that a promising way of studying the computational complexity of combinators is to use the concept of subbase and to analyze the computational power of various subbases. This somehow corresponds to the limitation of definitions in the formalism of recursive functions, which allows the generation of interesting subsets of partial recursive functions, such as the class of primitive recursive functions, the class of elementary functions and the classes of Grzegorczyk [10].

DEFINITION 3.1. - A *subbase* is a non-empty (possibly infinite) class of combinators $\mathcal{B} = \{\Phi_1,\ldots,\Phi_n\}$.

In general we will refer to finite subbases of independent combinators.

DEFINITION 3.2. - The *applicative closure* of the subbase $\mathcal{B}$ (denoted $\mathcal{B}^+$) is the class of all finite (applicative) combinations of Φ_i's. We wish, in the future, to refer to the subclass of $\mathcal{B}^+$ whose elements are in normal form and proper. We will indicate this class by $\mathcal{B}^+_{np}$.

We now summarize a few examples of basic results holding for particular subbases.

THEOREM 3.1. *For any combinator* w *in the subbase* $\{B,C,K\}^+$ *the structural and computational complexity satisfy the following limitations:*

i) $CT(w) \leq SL(w) - 1$

ii) $CS(w) = SL(w)$

iii) $CD(w) \leq SL(w) - 1$

iv) $CD(w) \leq \left\lfloor 2^{SD(w)-1} + \frac{SD(w)-1}{2} \right\rfloor$

PROOF. (i), (ii) - From a theorem of Curry [1] no combinator exists in {B,C,K} with duplicative effect. So, at every reduction step the length of the formula decreases at least by 1.

(iii). In a formula, whose length is n, appear n-1 applications.

(iv). Since SD(w) is fixed, the maximum number of basic combinators in w is $2^{SD(w)}$.

In every contraction step the basic combinators may increase the depth of a formula by at most 1. Therefore, if n is the number of basic combinators used to achieve the maximum depth, we have

$CD(w) = SD(w) + n$, where n is the maximum

integer such that: $SD(w)+n \leq 2^{SD(w)}-n-1$. (The last inequality is the (iii) written for the deepest formula achieved). Hence point (iv) follows.

Q.E.D.

THEOREM 3.2. - *In the subbase* {B} *the expotential growth of the depth is achievable. In fact:*

(i) $\forall w \varepsilon \{B\}^+$ such that $SD(w) \leq 2$ then $CD(w) = SD(w)$;

(ii) $\forall n > 2 \; \exists \, w_n \varepsilon \{B\}^+$ such that:

$$SD(w_n) = n$$

$$CD(w_n) = n + \sum_{i=0}^{n-1} b_i \quad \text{where } b_o = b_1 = 0 \text{ and } b_{i+2} = b_{i+1} + b_i + 1$$

PROOF. The point (i) is immediate.
For the point (ii) it is not difficult to see that the class of combinators w_n, whose structure is recursively defined as follows:

$$w_n = d_{n-1} w_{n-1} \qquad \text{where: } w_o = d_o$$

$$d_n = B\, d_{n-2} d_{n-1}$$

$$d_o = B$$

$$d_1 = B\,B\,,$$

has the property that all its B's with 3 arguments, in the reduction process, increase by 1 level the initial depth of the w_n.
We also have $w_n = d_{n-1}(d_{n-2}(\ldots(d_o d_o)\ldots))$, and $n = SD(w_n)$.

Let b_i be the number of B's with 3 arguments in d_i. We can see $b_o = b_1 = 0$ and $b_i = b_{i-1} + b_{i-2} + 1$.

Therefore: $CD(w_n) = n + \sum_{i=0}^{n-1} b_i$ Q.E.D.

REMARK 3.1. In the subbase {B,C} the limit of theorem 3.2 can be improved.

For example, if $q = B^2(CBB)B\, d_4 d_5 w_6$, whose $SD(q) = 7$,

we have $CD(q) = SD(q) + \sum_{i=0}^{SD(q)-1} b_i + 1$.

As far as combinators without normal form are concerned, we may define SL(w)(n) to be the length and the depth of the combinator w at the n-th reduction step and we may show some properties of these functions of n for the subbase {W} and {B,W}. In particular, we will show that, in the subbase {W}, SL(w)(n) is somehow linear, while an exponential growth is possible in the subbase {B,W}.

We will first introduce the following:

DEFINITION 3.3. The *number* $n_r(w)$ *of right-applied objects of* $w\varepsilon\mathcal{B}^+$, is defined by: $n_r(w)=k+1$, where $w=(\ldots((b\chi_1)\chi_2)\ldots\chi_k)$, $b\varepsilon\mathcal{B}$ and $\chi_i\varepsilon\mathcal{B}^+$, $1\leq i\leq k$.

REMARK 3.2. The decomposition of w in right-applied objects is unique.

DEFINITION 3.4. The *subwords* of a combinator $w\varepsilon\mathcal{B}^+$ are recursively defined as follows:

(i) if w=(ab) where $a,b\varepsilon\mathcal{B}$ then a and b are subwords of w;

(ii) if w=(AB) where $A,B\varepsilon\mathcal{B}^+$ then A,B and the subwords of A and B are subwords of w.

It is now necessary to prove the following lemma.

LEMMA 3.1. Let w be a combinator of $\{W\}^+$. Let $\overline{w}$ be the leftmost subword of w such that $n_r(\overline{w})=3$. We will call $\overline{w}$ the *leading subword of* w.

(i) If $\overline{w}$ exists:

- w has not normal form;
- $\forall n$ SL(w)(n) = SL($\overline{w}$)(n)+c, where c is a constant.

(ii) If $\bar{w}$ does not exist, w is in normal form.

PROOF. Point (ii) is immediate. Point (i) is proved by the following facts:

(i) In any reduction step of a word $w\varepsilon\{W\}^+$, we have: $n_r(w)(n)\leq n_r(w)(n+1)$.(*)

(ii) If $w=w_1w_2$ and no W in w_1 has 2 or more arguments, then in the reduction process the W's in w_1 do not change the number of their arguments.

(iii) If $w=w_1w_2$ where $n_r(w_1)\geq 3$, then the reduction process which follows does not take into account w_2.

Q.E.D.

For the base {W} we have the following result:

THEOREM 3.3. For any $w\varepsilon\{W\}^+$ without normal form $\exists\bar{n}$ finite such that, if $\Delta(n)=SL(w)(n+1)-SL(w)(n)$, we have:

(i) $\forall n<\bar{n} : \Delta(n)\geq 0$ depending on n;

(ii) $\forall n\geq\bar{n} : \Delta(n)=\bar{\Delta}\geq 0$.

PROOF. Given a $w\varepsilon\{W\}^+$ without normal form by lemma 3.1. $\bar{w}$ exists (defined as in lemma 3.1) such that $SL(w)(n)=SL(\bar{w})(n)+c$.

We can easily verify that the following procedure gives the value of $\bar{\Delta}$. Consider $\bar{w}=Ww_1w_2$:

a) if $w_1=w_2=\underbrace{W(\ldots(W(WW))\ldots)}_{i}$ then $\bar{\Delta}=i$;

(*) We denote by w(n) the combinator derived by w after n reductions. Obviously w(0)=w.

b) if $n_r(w_1)\geq 1$ and $w_1\neq w_2$ then $\bar{\Delta}$ is determined by reducing the leftmost W in $\bar{w}$ and considering only the leading subword of the achieved formula.

Finally, in order to show that $\bar{n}$ is finite, we have to prove the following assertion: given $\bar{w}=Ww_1w_2$ the case (b) does not occur an infinite number of times.

Infact the structure of w_1 may be only one of the following:

1) $w_1 = \bar{\bar{w}}$

2) $w_1 = Ww_k$

3) $w_1 = W$, where $\bar{\bar{w}}_1 w_k \varepsilon \{W\}^+$ and $n_r(\bar{\bar{w}})\geq 3$.

In case 1) the assertion is proved applying the principle of induction on combinator's lenght $(SL(\bar{\bar{w}})<SL(\bar{w}))$.

In case 2) we obtain $\bar{w}\geq w_1w_2\geq w_kw_2w_2w_2$ and we may apply the induction as in case 1), since $SL(w_k)=SL(w_1)-1$.

In case 3) we have $\bar{w}\geq w_1w_2w_2\geq w_2w_2w_2$, that is the following subcases:

3.1) $w_2=\bar{\bar{\bar{w}}}$

3.2) $w_2=Ww_j$

3.3) $w_2=W$, where $\bar{\bar{\bar{w}}},w_j\, \varepsilon \{w\}^+$ and $n_r(\bar{\bar{\bar{w}}})\geq 3$

Cases 3.1) and 3.2) are as cases 1) and 2).

In case 3.3) $\bar{w}=WWW$ and $\bar{\Delta} = 0$.

Q.E.D.

REMARK 3.3. This proof allows to determine the value of $\overline{n}$ as the number of occurrences of the case b).

REMARK 3.4. We can construct a combinator $w \varepsilon \{W\}^+$ satisfying a given infinite succession $s=\{\Delta(0),\Delta(1),\dots\}$, such that

$$\overline{n} \geq 0 : \forall n \geq \overline{n},\ \Delta(n) = \overline{\Delta} \geq 0.$$

Let us indicate:

(i) the succession s without its first k elements by $s_{(k)}$ (*);

(ii) the word in $\{W\}^+$ satisfying $s_{(k)}$ by $w_{(k)}$;

(iii) the word W(W(...(WW)...)) with m W's by $w^{(m)}$.

The $w \varepsilon \{W\}^+$ satisfying s is recursively defined as follows:

$w_{(\overline{n})} = WWw^{(\overline{\Delta}+1)}$;

$w_{(k-1)} = Ww_{(k)}\overline{w}$, where $\overline{w} \varepsilon \{W\}^+$ and $SL(\overline{w}) = \Delta(k-1)+1$.

For the subbase {B,W} we have:

THEOREM 3.4. There exists w in $\{W,B\}^+$, such that SL(w)(n) grows exponentially with n.

(*) Obviously $s \equiv s_{(0)}$.

PROOF. In a constant number of reduction steps (6) $W(W_{(2)}B_{(1)})(W(W_{(2)}B_{(1)}))p$, where $p\varepsilon\{W,B\}^+$, reaches $W(W_{(2)}B_{(1)})(WW_{(2)}B_{(1)}))p'$ where $p'\varepsilon\{W,B\}^+$ and $SL(p')=2\cdot SL(p)$.

Q.E.D.

4. SUBBASES AND DEFINITIONAL COMPLEXITY

A particularly interesting type of results concerning subbases are related to their generative power:

DEFINITION 4.1. Let $\mathcal{B}$ be a subbase; let V be an infinite ordered set of variables $\{x_1,x_2,\ldots,x_n,\ldots\}$; let V^+ be the set of all finite applicative combinations of variables.

We say that $L(\mathcal{B})$ is the *language generated by* $\mathcal{B}$ if $L(\mathcal{B})$ is the smallest subset of V^+ satisfying the property that, given any $w\varepsilon\mathcal{B}^+_{np}$, then if n is the order of w, there is $X\varepsilon L(\mathcal{B})$ such that

$$wx_1,\ldots,x_n \text{ reduces to } X^{(*)} .$$

We prove first the following lemma.

LEMMA 4.1. If ξ is a proper combinator whose order is 2, then $\{\xi\}^+_{np}$ is the set: $\{\xi,\xi\xi,\xi(\xi\xi),\ldots,\xi(\xi(\ldots(\xi\xi)\ldots)),\ldots\}$.

(*) Notice that we consider for example $a_1a_2a_3$ and $((a_1a_2)a_3)$ to be the same word.

PROOF. If we suppose in $\{\xi\}^{+}_{np}$ a ψ exists with $n>2$ right-applied objects, then ψ is not in normal formal.

Q.E.D.

The following fact can easily be verified:

FACT 4.1.

(i) $L(\{K\}) = \{x_n | n \geq 1\}$;

(ii) $L(\{W\}) = \{x_1x_2x_2\} \cup \{x_1^k | k \geq 3\}$ (*)

(iii) $L(\{B,I\})$ = the set of *complete ordered applications* that is the language generated by the production $S \rightarrow (SS)$ and by substitution from left to right of all occurrences of S in a sentential form by the variables $x_1, x_2, \dots, x_n, \dots$ in this order;

(iv) $L(\{B\}) = L(\{B,I\}) - \{\bar{X}x_n |$ if $n=1$ $\bar{X}$ is the empty word, if $n>1$ $\bar{X}$ is a word of the language of complete ordered applications with the variables $x_1, \dots, x_{n-1}\}$;

(v) $L(\{C\}) = \{x_1x_3x_2\} \cup \{x_2x_3x_1\}$.

PROOF of Fact 4.1. (i). From Lemma 4.1. we know the structure of the elements of $\{K\}^{+}_{np}$. For every $n \geq 1$ there exists only one combinator $\xi_n \varepsilon \{K\}^{+}_{np}$ such that $SL(\xi_n)=n$, and $\xi_n = K\xi_{n-1}$. For $\xi_1 = K$ the corresponding X is x_1.

If the fact is valid $\forall n' < n$ then

$$\xi_n x_1 \dots x_{n+1} = K\xi_{n-1} x_1 \dots x_{n+1} \geq \xi_{n-1} x_2 \dots x_{n+1} \geq x_n$$

(*) Notice that we consider x_i^k and $\underbrace{x_i x_i \dots x_i}_{k}$ to be the same word.

where the last contraction is guaranteed by the induction hypothesis.

Q.E.D.

PROOF of Fact 4.1. (ii). Like proof of Fact 4.1. (i).

Q.E.D.

PROOF of Fact 4.1. (iii) and Fact. 4.1. (iv). The language $L(\{B,I\})$ is contained in the language of complete ordered applications [1].

Viceversa the language of complete ordered applications is contained in $L(\{B,I\})$, because, if X is a word of the language of complete ordered applications, and:

- $X = x_1x_2\ldots x_n$, where $n \geq 1$, then $B^{n-1}I$ corresponds to it;
- $X = \overline{X}\, x_j\ldots x_n$ and $\overline{w} \varepsilon$ ({B}) corresponds to $\overline{X}$, then $B^nI\overline{w}$ corresponds to it.

The last case is the one in which $X = x_1X_1X_2\ldots X_k$ where at least X_k is a combination of 2 or more variables x_i, and possibly k=1. In this case we will now prove inductively that a combinator of $\{B\}^+$ corresponds to X. One parenthesis can be removed from X eliminating that one surrounding X_k by $B_{(k-1)}$. Let us assume we succeded, in the expansion procedure, to remove p parentheses, obtaining a combination of the form:

$\xi x_1Y_1Y_2\ldots Y_\ell$ where at least one Y_j is a combination of 2 or more variables, and ξ is a combination of B's.

Now we may remove one more parenthesis as the one surrounding Y_j, by the deferred combinator $B_{(j)}$.

Q.E.D.

PROOF of Fact 4.1. (v). We prove first that: $\forall\xi\varepsilon\{C\}^{+}_{np}$ the order of ξ is ≤ 3.

In every reduction step of $\xi x_1 x_2 \ldots x_n$ where $\xi\varepsilon\{C\}^{+}_{np}$, there are at least 3 arguments between the first basic combinator in ξ and x_4.

Infact there will always be x_1,x_2 and x_3, because:

- on the first reduction step there are x_1,x_2 and x_3;
- if there are x_1,$x_2$and x_3 on the i-th step, then they are also on the (i+1)-th step, because the order of C is 3, C has not compositive effect and no x_i, where $1\leq i\leq 3$, may be on the left of the leftmost C in a formula (ξ is proper).

Therefore in order to prove the fact 4.1.(v), we can consider only the combinations of x_1,x_2 and x_3. For $x_1x_2x_3$,$x_2x_1x_3$,$x_3x_1x_2$ and $x_3x_2x_1$ after the first expansion step the last variable, that cannot be reused, is not x_3, as it should be.

Instead for $x_1x_3x_2$ and $x_2x_3x_1$ the last variable is x_3, and they are actually computed by C and CC, respectively.

Q.E.D.

For specific subbases the completeness (meta)-algorithms (such as those given in [11] for the base {S,K} and in [1] for the base {B,C,W,K}) become more interesting. In fact while in a complete base those metaalgorithms cannot always give the shortest combinator corresponding(*) to a given combination of variables, this may be accomplished in the case of a subbase which is complete only with respect to a subset of combinations.

(*) In the sense of [1] pag. 160.

This is the case of the base {B}. For the proof of the theorem we need some definitions.

DEFINITION 4.2. Given a combination of B's and variables we define *free parenthesis* associated to the combination a couple of parenthesis such that:

1) includes a combination of B's and variables or a combination of variables;
2) can be eliminated in an expansion step by only one B on the head of the combination.

For instance, in the combination

$$\mathrm{Ba}\underset{1}{(}\mathrm{b}\underset{2}{(}\mathrm{cd}))\underset{3}{(}\mathrm{ef})$$

the couples (1) and (3) are free, the couple (2) is not free. The couple (1) for instance can be eliminated with the expansion

$$\mathrm{Ba(b(cd))(ef)} \leq \mathrm{B(Ba)b(cd)(ef)}$$

DEFINITION 4.3. Given an expansion with n free parenthesis we say that a free parenthesis 1 is the *last* one if it is on the righthand of all the other free parenthesis.

Now, given a combination X of the variables $x_1,\ldots,x_n$ we want to find the shortest $w \in B^+$ such that

$$wx_1,\ldots,x_n \geq X$$

Proceeding in the expansion steps from the object X to the object $Wx_1,\ldots,x_n$ we may follow various strategies according to the order in which we remove the parentheses.

DEFINITION 4.4. We call 0-*strategy* the strategy that always removes the last free parenthesis.

DEFINITION 4.5. We call i-*strategy* a strategy that i times does not remove the last free parenthesis.

We may now state the following theorem:

THEOREM 4.1. The 0-strategy is an optimal strategy.

PROOF. By induction we prove firstly that the 0-strategy is not worse of every 1-strategy. Secondly, we prove that if the theorem is true for every i-strategy, it is true for every (i+1)-strategy. We have to notice that the couple of parenthesis we have to remove can be of 3 different types according to the structure of the two objects w and w in the application.

We define the parenthesis of type:

i) VV if w_1 and w_2 are combinations of variables;
ii) BVV if w_1 is a combination of B's and variables or only of B's and w_2 is a combination of only variables.
iii) BBV if w_1 is a combination of B's and w_2 is a combination of B's and variables.

Firstly we prove the initial step of the induction for the 3 cases.

INITIAL STEP

1) <u>Case VV</u>

We compare the two strategies from the point in which they diverge.

The structure of the combination at this point has the following form

$$X_1X_2x_k\ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})\ldots(t_kt_{k+1})x_t\ldots x_z \qquad (1)$$

where X_1 is a combination of only B's and X_2 of B's and the first k-1 variables. We construct now the histories of the two computations. We establish in the future not to write the terminal part of the combination composed of only variables without compositions among these variables, that is the longest terminal string of the form

$$x_n)x_{n+1})\ldots)x_z.$$

0-STRATEGY

$$X_1X_2x_k\ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})\ldots(t_kt_{k+1}) \geq \qquad (1)$$

$$B(X_1X_2x_k\ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})\ldots)t_kt_{k+1} \geq \qquad (2)$$

$$\xi_{kk+1}[B(X_1X_2\ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})\ldots)] \geq \qquad (3)$$

where in (3) ξ_{kk+1} is the combination that eliminates all the parentheses from w_kw_{k+1} with the 0-strategy, leaving only one parenthesis surrounding the object that in (2) preceeds w_k.

$$B\xi_{kk+1}B(X_1 \ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})\ldots) \geq \qquad (4)$$

$$\xi_{\ell ii+1}(B\xi_{kk+1}B)(X_1 \ldots x_\ell(t_1t_2)\ldots(t_it_{i+1})) \geq \qquad (5)$$

where in (5) $\xi_{\ell ii+1}(B\xi_{kk+1}B)$ is the object that applied to $(X_1 - x_\ell(t_1t_2)\ldots(t_it_{i+1}))$ and the succeding variables, gets the object (4).

We have to notice that in $\xi_{\ell ii+1}(B\xi_{kk+1}B)$ the parenthesis following the last B are in general $n>1$. Since we are interested in computing lengths of combinators, we give no weight to this imperfect notation.

$$B[\xi_{\ell ii+1}(B\xi_{kk+1}B)](X_1 \ldots x_\ell(t_1t_2)\ldots)(t_it_{i+1}) \geq \qquad (6)$$

$$B(B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots))t_it_{i+1} \geq \qquad (7)$$

$$\xi_{i+1}[B(B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots))t_i] \geq \qquad (8)$$

$$B\xi_{i+1}[B(B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots))]t_i \geq \qquad (9)$$

$$\xi_i\{B\xi_{i+1}[B(B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots))]\} \geq \qquad (10)$$

$$B\xi_i(B\xi_{i+1})\xi B(B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots))\} \geq \qquad (11)$$

$$B(B\xi_i(B\xi_{i+1}))B\{B(\xi_{\ell ii+1}(B\xi_{kk+1}B))(X_1 \ldots x_\ell(t_1t_2)\ldots)\} \geq \qquad (12)$$

$$B[B(B\xi_i(B\xi_{i+1}))B]\{B(\xi_{\ell ii+1}(B\xi_{kk+1}B))\}(X_1 \ldots x_\ell(t_1t_2)\ldots) \geq \qquad (13)$$

$$\xi_{END}\{B[B(B\xi_{i}(B\xi_{i+1}))B]\{B(\xi_{\ell ii+1}(B\xi_{kk+1}B))\}\}X_{1}X_{2}' \qquad (14)$$

where X_2' is the part of X_2 composed of only B's and ξ_{END} is the combinator that eliminates all the remaining compositions between the variables.

We come now to the generic 1-strategy. Suppose that the 1-strategy eliminates initially the parenthesis $(t_i t_{i+1})$ and after eliminates always the last parenthesis.

$$X_1 \ldots x_\ell (t_1 t_2) \ldots (t_i t_{i+1}) \ldots (t_k t_{k+1}) \geq \qquad (1)$$

$$B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i t_{i+1} \ldots (t_k t_{k+1}) \geq \qquad (2)$$

$$B[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i t_{i+1} \ldots] t_k t_{k+1} \geq \qquad (3)$$

$$\xi_{kk+1}\{B[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i t_{i+1} \ldots]\} \geq \qquad (4)$$

where ξ_{kk+1} is the same defined in step (3) of the 0-strategy.

$$B\xi_{kk+1}B[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i t_{i+1} \ldots] \geq \qquad (5)$$

$$\xi_{\ell ii+1}(B\xi_{kk+1}B)[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i t_{i+1}] \geq \qquad (6)$$

where for the object $\xi_{\ell ii+1}$ the same considerations made at step (5) of the 0-strategy are valid.

$$B(\xi_{\ell ii+1}(B\xi_{kk+1}B))[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i] t_{i+1} \geq \qquad (7)$$

$$\xi_{i+1}\{B(\xi_{\ell ii+1}(B\xi_{kk+1}B))[B(X_1 \ldots x_\ell (t_1 t_2) \ldots) t_i]\} \geq \qquad (8)$$

$$B\xi_{i+1}[B(B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))(X_1...x_\ell(t_1t_2)...))]t_i \geq \quad (9)$$

$$\xi_i\{B\xi_{i+1}[B(B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))(X_1..x_\ell(t_1t_2)..))]\} \geq \quad (10)$$

$$B\xi_i(B\xi_{i+1})\{B(B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))(X_1..x_\ell(t_1t_2)...))\} \geq \quad (11)$$

$$B(B\xi_i(B\xi_{i+1}))B\{B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))(X_1..x_\ell(t_1t_2)..)\} \geq \quad (12)$$

$$B[B(B\xi_i(B\xi_{i+1}))B]\{B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))\}(X_1..x_\ell(t_1t_2)..) \geq \quad (13)$$

$$\xi_{END}\{B[B(B\xi_i(B\xi_{i+1}))B]\{B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))\}\}\ X_1X_2^1$$

where X_2^1 is the part of X_2 composed of only B's and ξ_{END} is the combinator that eliminates all the remaining compositions between the variables. We come now to the generic 1-strategy. Suppose that the 1-strategy eliminates initially the parenthesis (t_it_{i+1}) and after eliminates always the last parenthesis.

$$X_1..x_\ell(t_1t_2)..(t_it_{i+1})..(t_\kappa t_{\kappa+1}) \geq \quad (1)$$

$$B(X_1..x_\ell(t_1t_2)..)t_it_{i+1}...(t_\kappa t_{\kappa+1}) \geq \quad (2)$$

$$B[B(X_1..x_\ell(t_1t_2)..)t_it_{i+1}...]t_\kappa t_{\kappa+1} \geq \quad (3)$$

$$\xi_{\kappa\kappa+1}\{B[B(X_1..x_\ell(t_1t_2)...)t_it_{i+1}..]\} \geq$$

where $\xi_{\kappa\kappa+1}$ is the same defined in step (3) of the 0-strategy.

$$B\xi_{KK+1}B[B(X_1..x_\ell(t_1t_2)..)t_it_{i+1}..] \geq \tag{5}$$

$$\xi_{\ell ii+1}(B\xi_{KK+1}B)\ [B(X_1..x_\ell(t_1t_2)..)t_it_{i+1}] \geq \tag{6}$$

where for the object $\xi_{\ell ii+1}$ the same considerations made at step (5) of the O-strategy are valid.

$$B(\xi_{\ell ii+1}(B\xi_{KK+1}B))[B(X_1..x_\ell(t_1t_2)..)t_i]t_{i+1} \geq \tag{7}$$

$$\xi_{i+1}\{B(\xi_{\ell ii+1}(B\xi_{KK+1}B))\ [B(X_1..x_\ell(t_1t_2)..)t_i]\} \geq \tag{8}$$

$$B\xi_{i+1}\{B[\xi_{\ell ii+1}(B\xi_{KK+1}B)]\}[B(X_1..x_\ell(t_1t_2)..)t_i] \geq \tag{9}$$

$$B\{B\xi_{i+1}[B(\xi_{\ell ii+1}(B\xi_{KK+1}B))]\}[B(X_1..x_\ell(t_1t_2)..)]t_i \geq \tag{10}$$

$$\xi_i\{B[B\xi_{i+1}[B(\xi_{\ell ii+1}(B\xi_{KK+1}B))]][B(X_1..x_\ell(t_1t_2)..)]\} \geq \tag{11}$$

$$B\xi_i\{B[B\xi_{i+1}(B[\xi_{\ell ii+1}(B\xi_{KK+1}B)])]\}\ [B(X_1..x_\ell(t_1t_2)..)] \geq \tag{12}$$

$$B[B\xi_i\{B[B\xi_{i+1}(B[\xi_{\ell ii+1}(B\xi_{kk+1}B)])]\}]B(X_1..x_\ell(t_1t_2)..) \geq \tag{13}$$

$$\xi_{END}\{B[B\xi_i\{B[B\xi_{i+1}(B[\xi_{\ell ii+1}(B\xi_{KK+1}B)])]\}]B\}X_1X_2^1 \tag{14}$$

The structure of (t_it_{i+1}) influences the lenght of combinators corresponding to X according to the two above strategies.

Let's define a combination t *COMPLEX* if it is the combination of at least two objects. Let's define *SIMPLE* if it is a variable x_j.

Let's consider the four possible cases.

1) t_i simple, t_{i+1} simple.

The steps (8),(9),(10),(11),(12) of the O-strategy are eliminated.

The steps (8),(9),(11),(12) of the 1-strategy are eliminated.

The new final combinators are:

$$\xi_{END}\{BB[B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))]\} \qquad \text{for the O-strategy;}$$

$$\xi_{END}\{B[B\{B[\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B)]\}]B\} \qquad \text{for the I-strategy.}$$

The O-strategy is shorter.

2) t_i complex, t_{i+1} complex.

The final combinators are the ones appearing in step (14) of the two strategies: they are equally long.

3) t_i complex, t_{i+1} simple.

The two final combinators are:

$$\xi_{END}\{B[B\xi_i B][B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))]\}X_1X_2^1 \qquad \text{for the O-strategy;}$$

$$\xi_{END}\{B[B\xi_i\{B[B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B))]\}]B\}X_1X_2^1 \qquad \text{for the 1-strategy.}$$

The O-strategy is shorter.

4) t_i simple, t_{i+1} complex.

The two final combinators are:

$$\xi_{END}\{B(B\xi_i B)(B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B)))\}X_1X_2^1 \qquad \text{for the O-strategy;}$$

$$\xi_{END}\{B\xi_i[B(B(\xi_{\ell i i+1}(B\xi_{\kappa\kappa+1}B)))B]\}X_1X_2^1 \qquad \text{for the 1-strategy.}$$

The two strategies are equally long.

The initial step for the case VV is so proved.

2) <u>Case BVV</u>

The initial structure of the object has the form

$$X_1(X_2X_3)t(t_\kappa t_{\kappa+1})x_t..x_z$$

where X_1 is a combination of B's, X_2 of B's and variables, X_3 of variables, t is a string of variables and combinations of variables, the parenthesis surrounding $t_\kappa t_{\kappa+1}$ is the last one.

O-STRATEGY

$$X_1(X_2X_3)t(t_\kappa t_{\kappa+1}) \geq \tag{1}$$

$$B(X_1(X_2X_3)t)t_\kappa t_{\kappa+1} \geq \tag{2}$$

$$\xi_{\kappa+1}[B(X_1(X_2X_3)t)t_\kappa] \geq \tag{3}$$

$$B\xi_{\kappa+1}[B(X_1(X_2X_3)t)]t_\kappa \geq \tag{4}$$

$$\xi_\kappa\{B\xi_{\kappa+1}[B(X_1(X_2X_3)t)]\} \geq \tag{5}$$

$$B\xi_\kappa(B\xi_{\kappa+1})[B(X_1(X_2X_3)t)] \geq \tag{6}$$

$$B(B\xi_\kappa(B\xi_{\kappa+1}))B(X_1(X_2X_3)t) \geq \tag{7}$$

$$B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)(X_1(X_2X_3))t \geq \tag{8}$$

$$\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)(X_1(X_2X_3))] \geq \tag{9}$$

$$B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B](X_1(X_2X_3)) \geq \tag{10}$$

$$B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}X_1(X_2X_3) \geq \tag{11}$$

$$B[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}X_1]X_2X_3 \geq \tag{12}$$

$$\xi_3\{B[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B]\}X_1]X_2\} \geq \tag{13}$$

$$B\xi_3\{B[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B]\}X_1]\}X_2 \geq \tag{14}$$

$$\xi_{END}[B\xi_3\{B[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B]\}X_1]\}]X_2^1 \tag{15}$$

while removing before the inner parenthesis we have

$$X_1(X_2X_3)t(t_\kappa t_{\kappa+1}) \geq \tag{1}$$

$$BX_1X_2X_3t(t_\kappa t_{\kappa+1}) \geq \tag{2}$$

$$B(BX_1X_2X_3t)t_\kappa t_{\kappa+1} \geq \tag{3}$$

$$\xi_{\kappa+1}(B(BX_1X_2X_3t)t_\kappa) \geq \tag{4}$$

$$B\xi_{\kappa+1}(B(BX_1X_2X_3t))t_\kappa \geq \tag{5}$$

$$\xi_\kappa[B\xi_{\kappa+1}(B(BX_1X_2X_3t))] \geq \tag{6}$$

$$B\xi_\kappa(B\xi_{\kappa+1})(B(BX_1X_2X_3t)) \geq \tag{7}$$

$$B(B\xi_\kappa(B\xi_{\kappa+1}))B(BX_1X_2X_3t) \geq \tag{8}$$

$$B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)(BX_1X_2X_3)t \geq \tag{9}$$

$$\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)(BX_1X_2X_3)] \geq \tag{10}$$

$$B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)](BX_1X_2X_3) \geq \tag{11}$$

$$B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}(BX_1X_2)X_3 \geq \tag{12}$$

$$\xi_3[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}(BX_1X_2)] \geq \tag{13}$$

$$B\xi_3[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}](BX_1X_2) \geq \tag{14}$$

$$B\{B\xi_3[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}]\}(BX_1)X_2 \geq \tag{15}$$

$$\xi_{END}[B\{B\xi_3[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]\}]\}(BX_1)]X_2^1 \tag{16}$$

Conducting an analisys for cases as in precedence, the initial step of the induction is satisfied.

3) <u>Case BBV</u>

O-STRATEGY

The first 12 steps are equal as in the case of BVV parenthesis

$$\xi_{END}\{B[B\{B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B]\}X_1]X_2\} \tag{13}$$

that is the final step.

1 - STRATEGY

The first 12 steps are equal as in the case of BVV parenthesis

$$\xi_{END}\{B[B\xi_t[B(B(B\xi_\kappa(B\xi_{\kappa+1}))B)]](BX_1X_2)\} \tag{13}$$

In this case too, for all possible subcases the first step of the induction is proved.

GENERIC STEP OF THE INDUCTION

We have to prove that, if the theorem is true for all the i-strategies, than it is true for all the (i+1)-strategies. Let's consider the expression preceeding the last step in which a (i+1)-strategy choiches a parenthesis that is not the last one.
Let's compare the (i+1)-strategy from this step to the final

one and the i-strategy that at this step and always in the future choiches the last free parenthesis. Only the three preceeding cases can verify and so no (i+1)-strategy can be shorter than a i-strategy.

Q.E.D.

As a consequence of theorem 4.1, we may notice that, if $wx_1\ldots x_n \geq X$ and $w'x_1\ldots x_n \geq X'$, where $X,X'\varepsilon L(\{B\})$ and $w,w'\varepsilon \{B\}^+$:

(i) if X' has a lower number of parentheses to be eliminated [(*)] than X, then $SL(w')<SL(w)$;

(ii) if X' is obtained from X by moving on the left one couple of parentheses of X to be eliminated, then $SL(w')<SL(w)$.

We can also establish the following:

<u>Theorem</u> 4.2. - For any X in $L(\{B\})$ such that $SL(X)=n$ [(**)] we have that if $w\varepsilon\{B\}^+$ corresponds to X, then $SL(w)=0(n)$.

Proof. The structure of X such that $X\varepsilon L(\{B\})$ and $SL(X)=n$, in which there is the minimum number (1) of parentheses to be eliminated, is of the form:

$$\bar{X}_n = x_1\ldots x_{n-2}(x_{n-1}x_n).$$

On the other hand, the structure of X such that $X\varepsilon L(\{B\})$ and $SL(X)=n$, in which there is the maximum number of parentheses to be eliminated and these are in the rightmost position, is of the form:

(*) We suppose all parentheses to be eliminated are explicited.

(**) The definitions of structural complexity obviously can be extended to pure combinations.

$$\bar{\bar{X}}_n = x_1(x_2(\dots(x_{n-1}x_n)\dots)).$$

It can be easily verified that: if $\bar{w}_n x_1 \dots x_n \geq \bar{X}_n$, then $\bar{w}_{n+1} = B\bar{w}_n$;

if $\bar{\bar{w}}_n x_1 \dots x_n \geq \bar{\bar{X}}_n$, then $\bar{\bar{w}}_{n+1} = B(B\bar{\bar{w}}_n)B$;

$\bar{w}_3 = \bar{\bar{w}}_3 = B.$

Q.E.D.

REFERENCES

[1] H.B. CURRY, R. FEYS: *Combinatory logic*, vol. 1, North Holland, 1974.

[2] G. AUSIELLO: *Computational complexity - Main results and a commentary*, Séminaires IRIA, 1972.

[3] H.R. STRONG: *Depth-bounded computation*, JCSS, Vol. 4, n. 1, 1970.

[4] H.B. CURRY, J.R. HINDLEY, J.P. SELDIN: *Combinatory logic*, vol. 2, North Holland, 1972.

[5] M. BLUM: *On the size of machines*, Information and Control 2, 1967.

[6] M. BLUM: *A machine independent theory of the complexity of recursive functions*, J.A.C.M. 14, April 1967.

[7] G. AUSIELLO: *Abstract computational complexity and cycling computations*, J.C.S.S. Vol. 5, 1971.

[8] H. BARENDREGT: *Normed uniformly reflexive structure*, (in these proceedings).

[9] C. BÖHM, W. GROSS: *Introduction to the CUCH*, Automata Theory (ed. Caianiello) N.Y., 1966.

[10] A. GRZEGORCZYK: *Some classes of recursive functions*, Rozprawy Matematyczne, 1953.

[11] P. ROSENBLOOM: *The elements of mathematical logic*, Dover, 1950.

Finito di stampare nel Luglio 1975
dall'Artigiana Multistampa Snc
Via Mecenate 20 - Roma