RR/LSI-96-60-R

Linear Lower Bounds and Simulations in Frege Systems with Substitutions

M. Bonet
N. Galesi

Report LSI–96–60–R

Linear lower bounds and simulations in Frege systems with substitutions

M. Bonet [*] N. Galesi [†]
Universitat Politecnica de Catalunya
Dep. de Llenguatges i Sistemes Informatics
Pau Gargallo 5, 08028 Barcelona, Spain
e-mail: {bonet, galesi}@goliat.upc.es

October 29, 1996

Abstract

Our work concerns Frege systems ($\mathcal{F}$), substitution Frege systems ($s\mathcal{F}$), renaming Frege systems ($r\mathcal{F}$) , $\top/\bot$-Frege systems ($\top/\bot$-$\mathcal{F}$) and extended Frege systems ($e\mathcal{F}$). Urquhart shows that tautologies associated to a binary strings require $\Omega(n/\log n)$ lines to be proved in $s\mathcal{F}$. Here we prove, by giving a $s\mathcal{F}$ proof of $O(n/\log n)$ lines, that his lower bound is optimal and we show that in the tree-like case $\Omega(n)$ lines are required for a proof of the same tautologies.
We also show the following two simulation results: (1) tree-like $s\mathcal{F}$ p-simulates non tree-like $s\mathcal{F}$; (2) Tree-like $\mathcal{F}$ linearly simulates tree-like $r\mathcal{F}$ and tree-like $\top/\bot$-$\mathcal{F}$.

1 Introduction

A *Frege system* $\mathcal{F}$ is an inference system for propositional logic based on (1) a language of well-formed formulas obtained from a numerable set of propositional variables and any finite propositionally complete set of connectives; (2) a finite set of axiom schemes; and (3) the rule of *Modus Ponens* ($\frac{A \quad A \rightarrow B}{B}$).

A *proof* P of the formula A in a Frege system is a sequence $A_1, \ldots, A_n$ of formulas such that A_n is A end every A_i is either an *instance* of an axiom scheme or it is obtained by the application of the Modus Ponens from the premises A_j and A_k with $j, k < i$. We will call A *theorem* and we write $\vdash A$. A proof P is said to be *tree-like* if every A_i is used only once as premise of a rule in P. Any Frege system must be *sound* and *complete*. We write $A \models B$ if every truth assignement satisfying A also satisfies B, and $A \vdash B$ if adding the formula A among the axioms we can give a proof of B. A Frege proof system is *implicationally complete*

[*]Research partly supported by NSF grant number CCR-9403447.
[†]Supported by EC grant under the TMR project.

if whenewer $A \models B$, then $A \vdash B$, and *implicationally sound* if whenewer $A \vdash B$, then $A \models B$.

The main notions of complexity of proofs are: (1) the *number of lines* of a proof $P = A_1, \ldots, A_n$ equals to n; (2) the *number of symbols* or *size* $|P|$ defined as $\sum_{i=1}^{n} |A_i|$ where $|A_i|$ is the number of symbols in A_i.
A Frege system $\mathcal{F}_1$ *p-simulates* another Frege system $\mathcal{F}_2$ if whenever a formula A has a proof P in $\mathcal{F}_2$ of size m, there is a proof P' of A in $\mathcal{F}_1$ of size $p(m)$, where p is a polynomial.

Frege systems can be extended with some extra rules.
An *extended Frege system* $e\mathcal{F}$ is a Frege system augmented with the *extension rule*. This rule allows to infer, for any formula B the formula $p \leftrightarrow B$ under the restrictions that p does not occur in B, in previous lines and in the last line of the proof.

A *substitution* σ is a mapping from a finite set of propositional variables Dom(σ) to a set of well-formed formulas Rng(σ). It will be called *renaming* whenever σ is a bijection and Rng(σ) is a set of propositional variables and $\top/\bot$-*substitution* whenever Rng(σ) $= \{\top, \bot\}$. By $A\sigma$ we denote the result of *simulataneously* substitute in A any propositional variable p_i in A with $\sigma(p_i)$.

A *substitution Frege system* is a Frege system augmented with the substitution rule $\frac{A}{A\sigma}$ that from a formula A allows to infer the formula $A\sigma$ for any σ.
It will be called *Renaming Frege system* ($r\mathcal{F}$) or $\top/\bot$-*Frege system* ($\top/\bot$-$\mathcal{F}$) whenever the substitutions are respectively renaming and $\top/\bot$.

Some properties can be lost by adding extra rules. For example $s\mathcal{F}$, $r\mathcal{F}$ and $e\mathcal{F}$ are implicationally complete but not implicationally sound. The following Theorem holds for any two implicationally complete systems differing at most in the set of connectives used.

Theorem 1.1 ([CR]) *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two systems differing at most in the set of connectives used. There exists a constant c such that if P is a proof of A in $\mathcal{F}_2$ of n lines and m symbols, there is a proof of A in $\mathcal{F}_1$ of $\leq cn$ lines and $\leq cm$ symbols.*

Since these linear simulations do not affect our work, from now on we won't worry about the set of axiom schemes we use, and we fix the set of the connectives as $\{\wedge, \rightarrow, \vee, \neg\}$. Also, we add to our language the constant $\top$ and $\bot$ whose intended meaning is TRUE and FALSE and the extra axiom $\top$.
Let σ and λ be two substitutions, by $\sigma\lambda$ we denote the new substitutions θ such that Dom(θ) = Dom(λ), Rng(θ) = Rng(σ) and if $\lambda(p_i) = A_i$, then $\theta(p_i) = A_i\sigma$. Two formulas A and B are isomorphic if there is a renaming σ such that $A = B\sigma$. Let $\Sigma = \{A_1, \ldots, A_k\}$ be a set of formulas and Var(Σ) $= \bigcup_{i=1}^{k}$ Var(A_i). A substitution σ unifies Σ if $A_1\sigma = \ldots = A_k\sigma$ and in this case Σ is said to be *unifiable*. A substitution σ is a *most general unifier* (mgu) for Σ if and only if it unifies Σ and for any other unifier θ there is a substitution λ with Rng(λ) = Var(Rng(σ)) such that $\theta = \sigma\lambda$. This means that any two mgu of Σ are isomorphic

The following Theorem is due to Robinson (see for example [G] cap.8 for its proof in the general case of tree over a ranked alphabet).

Theorem 1.2 *There is a deterministic algorithm $\mathcal{A}$, that always halts, such that for any set Σ of formulas, if Σ is unifiable, then $\mathcal{A}$ outputs its most general unifier.*

Let A and $B \rightarrow C$ be two formulas. The rule of *condensed detachment* with premises A and $B \rightarrow C$ and conclusions D, denoted by $\mathbf{CD}(A, B \rightarrow C)$ is defined as follows:

1. change A into an isomorphic formulas A' such that $\mathrm{Var}(A') \cap \mathrm{Var}(B \rightarrow C) = \emptyset$;
2. if $\mathcal{A}(\{A', B \rightarrow C\}) = \sigma$ then change, by a renaming, σ to σ^* in such a way that $[\mathrm{Var}(B\sigma^*) - \mathrm{Var}(B)] \cap \mathrm{Var}(C) = \emptyset$ and define $D = C\sigma^*$. If $\{A', B \rightarrow C\}$ is not unifiable then $\mathbf{CD}(A, B \rightarrow C)$ is undefined.

A *Condensed Detachment Frege system $CD(\mathcal{F})$* is a Frege system whose only rule is the condensed detachment and where we use a finite number of axioms instead of axiom schemes. We suppose that the axioms used in $CD(\mathcal{F})$ are indexed by an order and that in the proofs the axioms are introduced in increasing order in the first lines. It is easy to see that any proof in $CD(\mathcal{F})$ can be trasformed into such an equivalent one.

The following are the *Urquhart's tautologies.* Let x a binary string, then

$$\top_x = \begin{cases} \top & \text{if } x = \epsilon \\ (\top \rightarrow \top_y) & \text{if } x = 1y \\ (\bot \vee \top_y) & \text{if } x = 0y \end{cases}$$

The paper is divided as follows: in Section 2 we show that Urquahrt's tautologies require $\Omega(n)$ lines in tree-like $s\mathcal{F}$. In Section 3 we show that there is a proof of $O(n/\log n)$ lines in $s\mathcal{F}$ of the same tautologies, so proving optimality of [Ur] lower bound. In Section 4 we prove the following result: (1) if $e\mathcal{F}$ simulates in $O(n^{O(1)})$ lines a $s\mathcal{F}$ proof of n lines, then tree-like $s\mathcal{F}$ simulates in $O(n^{O(1)} \log^2 n)$ lines a $s\mathcal{F}$ proof of n lines. This is obtained by improving to $O(n \log n)$ the number of lines of the [CR] $s\mathcal{F}$ simulation of $e\mathcal{F}$ and by extend to $e\mathcal{F}$ the tree-like $\mathcal{F}$ simulation of $\mathcal{F}$ in $O(n \log n)$ lines of [BoBu]. (2) Tree-like $\mathcal{F}$ linearly simulates tree-like $r\mathcal{F}$ and tree-like $\top/\bot$-$\mathcal{F}$.

2 Lower bound for tree-like $s\mathcal{F}$ proof

In this section we will work under the hypothesis that $s\mathcal{F}$ has the same axioms (instead of axiom schemes) of $CD(\mathcal{F})$. We will show that Urquhart's tautologies require a tree-like $s\mathcal{F}$ proof of $\Omega(n)$ lines. The result is essentially due to the fact that in tree-like case we can improve to $O(n)$ the size of a *succint representation* for a $CD(\mathcal{F})$ proof P of n lines instead of the $O(n \log n)$ bound of [Ur] in the non tree-like case. This will be possible since, when we code a rule, we do not need to save the numbers of lines of the premises, opposite to what happen with the succint representation of [Ur] or with the *scheme of a proof* of [P] and [Or]. The proof proceeds as follows: given a tree-like proof P of A in $CD(\mathcal{F})$, first we show

how to encode it in more succint way using the concept of *proof-graph*. Then we prove how to recover a tree-like proof P' of A with the same number of lines of P. Finally we observe that the relation between $s\mathcal{F}$ and $CD(\mathcal{F})$ expressed by Theorem 2.2 of [Ur] holds aslo in the tree-like case. A lower bound theorem analogous to that of [Ur] can therefore be proved.
The method to compress the succint representation cannot be extended to the case of non tree-like proofs. This is also proved in Section 3 where we give an $O(\frac{n}{\log n})$ upper bound for the number of lines that Urquhart's tautologies require in non tree-like $s\mathcal{F}$.
The combinatorial argument to show the lower bound is a direct application of the Incompressiblity Methods arising in the theory of Kolmogorov Complexity [LV]. The argument is essentially the same of [Ur], but it allows to think in a more general way how to prove lower bounds using the technique of encoding proofs.

2.1 Proof-graphs

We represent a proof P in $CD(\mathcal{F})$ as a graph G_P is such a way it is possible recover uniquely (in the sense of [HM] pag 93) P from G_P. We will work with labelled directed acyclic graphs in which there are: (1) *leaf-nodes* with in-degree 0, out-degree ≥ 1 and labelled with a natural number; (2) *internal nodes* with in-degree 2 and out-degree ≥ 1, labelled with one natural number;(3) *root node*: an internal node with out-degree 0; (4) *arcs* labelled either 0 or 1.
Given a proof P in $CD(\mathcal{F})$ the *proof-graph* G_P associated to P is built as follows:

1. Create one leaf-node for each of the axiom used in the proof;
2. for each line i in the proof
   - if it corresponds to an axiom, then put i as label of the leaf node corresponding to that axiom;
   - if $i = CD(j, k)$ with $j, k < i$, then
     - create a new internal node and label it with i;
     - create an arc from node labelled with j and label the arc 1;
     - create an arc from node labelled with k and label the arc 0;

Clearly if P is tree-like then G_P is a tree. The following two properties are very easy to note:

1. since the proofs in $CD(\mathcal{F})$ have all axioms in the first lines, then the values of the labels of the nodes at depth i in G_P will be strictly greater than the values of the labels of the nodes at depth $i-1$;
2. If P is a proof in $CD(\mathcal{F})$ of A, then, following Theorem 3.1 of [Ur], from G_P we can reconstruct P uniquely (this is since in G_P we have the same informations of his succint representation).

Suppose to have a proof-graph without labels in the nodes (a *skeleton*) and let G_{P_1} and G_{P_2} be obtained by ℓ_1 and ℓ_2, two enumerations of the nodes preserving previous property 1. If P_1 and P_2 are the proofs recovered from G_{P_1} and G_{P_2}, then:

Lemma 2.1 *P_1 and P_2 have the same number of lines and are proofs of isomorphic formulas.*

Proof. The first part follows from the fact that the number of node is the same in the two graph and it corresponds to the number of lines in the proof.
Moreover the structure of the proof is preserved by the skeleton since (1) the number of CD rules is the same in the two proofs and (2) dependencies between CD rules and between axioms and CD rules are preserved. The second part now follows since mgu is unique and since the root node of a proof-graph corresponds to the last line in the proof. □

2.2 Recovering proofs from their succint representations

We are going to define a more succint representation. Given a tree-like proof P in $CD(\mathcal{F})$ it will be a description of the skeleton of the proof-graph G_P associated to P. Let n be the number of lines in P and let d be the number of axioms in the system. The *succint representation* $\langle P \rangle$ is a list L built over the alphabet $\{[,],0,1\}$ obtained, starting from the root node, in the following way:

- if the node is a leaf-node, then L is the binary notation for index number of the axiom associated to the node;
- if the node is an internal node, then $L = [L_1 L_0]$ where L_i is the list associated to the node linked by the arc labelled i.

Observe that the number of '[', of ']' in $\langle P \rangle$ is the number of lines in P in which we apply CD rule while the number of times we put in $\langle P \rangle$ the index number of an axiom scheme in $\langle P \rangle$ is the number of times we use that axiom in any CD rule. So the length (number of symbols) of $\langle P \rangle$ is $|\langle P \rangle| \leq n(2 + \log d)$.

Theorem 2.1 *Let $\langle P \rangle$ be the succint representations of the tree-like $CD(\mathcal{F})$ proof P of the formula A. We can recover from $\langle P \rangle$ a proof P' of A that has the same number of lines of P.*

Proof. First we recover from $\langle P \rangle$ the skeleton of P, then we give an enumeration of the node preserving property 1. The Theorem then follows by Lemma 2.1. □

2.3 Lower bound

We will give an extremely brief introduction to Kolmogorov Complexity and outline the idea of our proof.
The main concept of Kolmogorov Complexity is to measure the hardness of a discrete object in terms of its quantity of informations i.e. the number of bits required to *describe* it. In order to let these concepts to be applied generally, what is required is: (1) an universal description method, (2) a mechanism to produce an object from its description, (3) indipendence between the method of description and the notion of information content of an object. Given a method f, that can be considered as a partial function over non-negative integers, the *complexity* of an object x is defined to be $\min\{|p| \mid f(p) = x\}$, where $|p|$ is the length of

the binary representation of the number p. In terms of Computer Science we can think to f as a computer, to p as a program, so that the complexity of x is the minimal length of a program for f to compute x. Since it can be shown that different choices of the method f only affects by a constant amount the complexity of x we can fix a method f as a Universal Turing Machine M and think to p as binary codification of a program that let M compute x. Moreover, since discrete objects can be codified as strings, we set the class of objects to be the class of binary strings.
The incompressibility method is based on the simple fact that there are strings whose shorter description is close to their length (obviously the worst description of a string is the string itself). These strings are called *incompressible* or *random*. For example is easy to show that there is at least a binary string of size n that cannot be described by a program (another binary string) of size less than n. Indeed, there are 2^n binary string of size n but only $\sum_{i=0}^{n} 2^i = 2^n - 1$ possible descriptions of size $n-1$ or less. In general, if we define a string x to be c-incompressible if its complexity is $\geq n - c$, then there are at least $2^n - 2^{n-c} + 1$ c-incompressible strings of size n (see [LV] pag. 96) and this means that among the strings of size n almost all (precisely $(1 - 2^{-c})$) are random. This simple fact can now be used to show combinatorial properties.

For example, in our case we want to show that for all x, $|x| = n$, all proof of $\top_x$ in $CD(\mathcal{F})$ require $\Omega(n)$ number of lines. The proof proceds as follows:

1. Assume by the contrary that for all $|x| = n$ there exist a proof P' proving $\top_x$ in less than n lines;
2. fix a c-incompressible string x of size n;
3. obtain the absurd, proving that from the alleged proof P' of less than n lines it can be given a description of x stricly less than n.

First we show how to improve Theorem 2.2 of [Ur] for the tree-like case, also we reformulate his lower bound proof in the framework of Kolmogorov Complexity.

Theorem 2.2 *Let P be a tree-like proof in $s\mathcal{F}$, then there is a tree-like $CD(\mathcal{F})$ proof P' such that every step in P is a substitution instance of a step in P and $|P| \geq |P'|$.*

Proof. The proof follows [Ur]. We only prove that the tree-like property is preserved. Let A be a formula in a line of P not istance of any axiom. A step of the form $\frac{A}{A\sigma}$ in P is compressed in only one line in P' into a formula B such that $B\sigma_1 = A$ for some σ_1. Since P is tree-like, then A can be used only to produce $A\sigma$. This means that in P' the formula B will be used only once depending on where $A\sigma$ is used in P. □

Theorem 2.3 *For almost all strings x of size n, any tree-like proof of $\top_x$ in $s\mathcal{F}$ must have $\Omega(n)$ lines.*

Proof. Suppose, by the contrary, that for all x exists a tree-like proof P in $s\mathcal{F}$ of $\leq \frac{n}{c'}$ lines with $c' > (2 + \log d)\log 4$. Then by Theorem 2.2 there is a proof P' in $CD(\mathcal{F})$ of a formula A of which $\top_x$ is a substitution instance. As $\top_x$ is not a substitution istance of any shorter tautology, then the sequence of connectives is the same in in A and $\top_x$. Fix a c-incompressible string x of size n. Note that, once $CD(\mathcal{F})$ is fixed, a description of x can be obtained from $\langle P' \rangle$ and from a program $\mathcal{P}$ - indipendent from x - that on input $\langle P' \rangle$ (1) reconstructs the proof, (2) takes the last formula and (3) put it into a binary string form. By Theorem 2.2 we have that $|P'| \leq \frac{n}{c'}$ and so $|\langle P' \rangle\ | \leq \frac{n(2+\log d)\log 4}{c'}$ since we can identify string of length n over an alphabet of k symbols with binary strings of length $n \log k$. Since $\mathcal{P}$ is indipendent from x its size can be considered constant, so $|\mathcal{P}| = O(1)$. It follows that the complexity of x is $\leq \frac{n(2+\log d)\log 4}{c'} + O(1)$ and since $c' > (2+\log d)\log 4$, we have that this is $< n - c$ for n sufficiently large. Since there are $2^n - 2^{n-c} + 1$ strings of length n that are c-incompressible, then for almost all tautologies $\top_x$ a tree-like $s\mathcal{F}$ proof of $\Omega(n)$ is required. $\square$

3 Upper bound for Urquhart's tautologies

Urquhart shows in [Ur] that tautologies $\top_x$ associated to a binary string of size n require $\Omega(\frac{n}{\log n})$ lines $s\mathcal{F}$ proofs proving by a counting argument and without costructing them explicitly, that each proof of $\top_x$ must be greater than $\lfloor \frac{n}{(4 \log n)} \rfloor$. Here we show how to obtain an $O(n/\log n)$ upper bound for the number of lines needed to obtain a $s\mathcal{F}$ proof of $\top_x$. We will show that this is $\leq 2c\frac{n}{\log n}$ for some constant c depending on the choosen system. Consider the following formulas associate to a binary string x:

$$\top^p_x = \begin{cases} p & \text{if } x = \epsilon \\ (\top \to \top^p_y) & \text{if } x = 1y \\ (\bot \vee \top^p_y) & \text{if } x = 0y \end{cases}$$

Lemma 3.1 *All tautologies $p \to \top^p_x$ for any $x \in \{0,1\}^n$ with $n > 0$ can be obtained in a $s\mathcal{F}$ proof of $O(2^n)$ lines.*

Proof. First observe that $p \to \top^p_1 = p \to (\top \to p)$ and $p \to \top^p_0 = p \to (\bot \vee p)$ can be obtained in a constant number of steps. Now supose to have in the proof all the tautologies $p \to$ for $|y| = n-1$. To obtain $p \to \top^p_x$ we use all $p \to \top^p_y$, $p \to \top^p_1$ and $p \to \top^p_0$ in the following way: if $x = y1$ (respectively $x = y0$) then $p \to \top^p_x$ is obtained in the following two steps substitute $(\top \to p)$ to p in $p \to \top^p_y$; (2) cut the formula $p \to (\top \to p)$ (respectively $p \to (\bot \vee$ with the formula obtained at the previous step. Each one on the $\top^p_x$'s can be obtained in a constant number of lines from one of the $\top^p_y$'s and one between the first tautologies. So the total numbers of lines to put in the proof all the $\top^p_x$ is $c\sum_{i=1}^n 2^i = O(2^n)$. $\square$

Theorem 3.1 *Given $x \in \{0,1\}^n$ there is a $s\mathcal{F}$ proof of $\top_x$ in $O(\frac{n}{\log n})$ lines.*

Proof. The proof is divided in two parts. In the first, by previous Lemma, we obtain in $O(\frac{n}{\log n})$ lines a proof of all tautologies $p \to \top^p_y$ for all $|y| = \log(\frac{n}{\log n})$. To obtain $p \to \top$ suppose x divided in $\frac{n}{\log(\frac{n}{\log n})}$ substrings $x_1, \ldots, x_k$ (with $k = \frac{n}{\log(\frac{n}{\log n})}$) of size $\log(\frac{n}{\log n}$. In the first part we have already proved the tautologies $p \to \top^p_{x_i}$ for all i. We put them together to form $p \to \top^p_x$ starting from the innermost in the following way: Consider the sequence $p \rightarrow \mathsf{T}^p_{x_1}, \ldots, p \rightarrow \mathsf{T}^p_{x_k}$; obtain the formula $p \rightarrow \overline{\mathsf{T}}^p_{x_{k-1}}$ by the following two steps: (1) substitute $\mathsf{T}^p_{x_k}$ to p in $p \rightarrow \mathsf{T}^p_{x_{k-1}}$; (2) cut this formula with $p \rightarrow \mathsf{T}^p_{x_k}$. Now consider the sequence $p \rightarrow \mathsf{T}^p_{x_1}, \ldots, p \rightarrow \overline{\mathsf{T}}^p_{x_{k-1}}$ and iterate the two previous steps. After $k = \frac{n}{\log(\frac{n}{\log n})}$ iterations we obtain T^p_x and in a constant number of steps T_x. Since each $p \rightarrow \mathsf{T}^p_{x_i}$ is obtained in a constant number of steps from $p \rightarrow \mathsf{T}^p_{x_i}$ and $p \rightarrow \overline{\mathsf{T}}^p_{x_{i-1}}$, the total number of steps of the second part is $O(\frac{n}{\log(\frac{n}{\log n})})$. Observe that for $0 < c < 1$, $(1-c)\log n \geq \log\log n$ for almost every n. So, choosing $c = 0.5$ we have $\frac{n}{\log(\frac{n}{\log n})} \leq (2n/\log n)$. So putting together the two parts of the proof we obtain an $O(n/\log n)$ lines $s\mathcal{F}$ proof of T_x.□

4 Simulations for Frege systems with substitutions

Result of [Ur] and Theorems 2.3 and 3.1 seem to suggest that could be a result of the following type:
If P is a proof of n lines of the formula A in $s\mathcal{F}$, then there is a tree-like $s\mathcal{F}$ proof of A in $O(n \log n)$ lines,
that would imply that our lower bound is optimal. This is also supported by the fact that it holds for $\mathcal{F}$ [BoBu] and for $e\mathcal{F}$ (see 4.1.1). But the technique used in [BoBu], that substantially is applied also for $e\mathcal{F}$, does not work for $s\mathcal{F}$, since it seems difficult to infer the formula $A_1 \wedge \ldots \wedge A_j\sigma \wedge \ldots \wedge A_n$ from $A_1 \wedge \ldots \wedge A_j \wedge \ldots \wedge A_n$ in a tree-like way and without working on the definition of A_j (in fact in this case we loose the dependency from only the number of lines of the proof we want to simulate). Moreover, note that there is no known result about polynomial simulation of $s\mathcal{F}$ systems by tree-like $s\mathcal{F}$ systems. Here we solve this problem showing that if $e\mathcal{F}$ simulation of a n lines $s\mathcal{F}$ proof, showed in [KP], increments the number of lines to $O(n^{O(1)})$, then an $s\mathcal{F}$ proof of n lines can be simulated by a tree-like $s\mathcal{F}$ proof with $O(n^{O(1)} \log^2 n)$.
Buss in [Bu1] shows that $r\mathcal{F}$ and $\top/\bot$-$\mathcal{F}$ p-simulate $s\mathcal{F}$. We show that if these simulations can be extended to the tree-like case, then $\mathcal{F}$ p-simulates $s\mathcal{F}$.

4.1 Tree-like $s\mathcal{F}$ simulation of $s\mathcal{F}$

We divide the task to obtain a tree-like $s\mathcal{F}$ simulation of $s\mathcal{F}$ in the following steps:

1. $e\mathcal{F}$ simulates $s\mathcal{F}$;

2. tree-like $e\mathcal{F}$ simulates $e\mathcal{F}$;

3. tree-like $s\mathcal{F}$ simulates tree-like $e\mathcal{F}$

The first is shown im [KP] without saying anything about the degree of the polynomial involved. For the second simulation we show that a $e\mathcal{F}$ proof of n lines can be simulated by a tree-like $e\mathcal{F}$ proof of $O(n \log n)$ lines using a technique developed in [BoBu]. For the third simulation we show that a tree-like $e\mathcal{F}$ proof of n lines can be simulated by a tree-like $s\mathcal{F}$ proof of $O(n \log n)$, improving the $O(n^2)$ result of [CR].
Consider the following formulas introduced in [Bo1]:

Definition 4.1 *Let $A_1, \dots, A_n$ be formulas with n a power of 2. The Balanced conjunction $\bigwedge_{i=1}^{n} A_i$ of $A_1, \dots, A_n$ is defined inductively by*

- *if $n = 1$, then $\bigwedge_{i=1}^{n} A_i$ is A_1;*
- *otherwise, $(\bigwedge_{i=1}^{n/2} A_i) \wedge (\bigwedge_{i=1}^{n/2} A_{n/2+i})$.*

Definition 4.2 *Let $A_1, \dots, A_n$ be formulas with $n = 2^m + s$ with $0 < s \leq 2^m$. The Psuedobalanced conjunction $\bigwedge_{i=1}^{n} A_i$ of $A_1, \dots, A_n$ is defined inductively by*

- *if $n = 1$,then $\bigwedge_{i=1}^{n} A_i$ is A_1;*
- *otherwise, $(\bigwedge_{i=1}^{2^m} A_i) \wedge (\bigwedge_{i=1}^{s} A_{(2^s+i)})$ where the first conjunct is balanced and the second pseudobalenced.*

The following Lemma is showed in [BoBu]:

Lemma 4.1 ([BoBu]) *The formula $(\bigwedge_{i=1}^{k-1} A_i) \wedge A_k \rightarrow (\bigwedge_{i=1}^{k} A_i)$, where the conjunction are associated in a pseudobalenced way, has a tree-like $\mathcal{F}$ proof of $O(\log k)$ lines.*

4.1.1 Tree-like $e\mathcal{F}$ simulation of $e\mathcal{F}$

This proof follows in a very easy way from the analogous thoerem for $\mathcal{F}$ showed in [BoBu]. We only sketch it.

Theorem 4.1 *If P is a $e\mathcal{F}$ proof of the formula A in n lines, then there is a tree-like $e\mathcal{F}$ proof of A in $O(n \log n)$ lines.*

Proof. Let $A_1, \dots A_n$ be the proof in $e\mathcal{F}$. Let $B_i = \bigwedge_{j=i}^{n} A_j$ for $i = 1, \dots, n$ and $B_0 = \top$. Depending on how A_{i+1} is inferred in P it can be show that there is atree-like $e\mathcal{F}$ proof of $B_i \rightarrow B_{i+1}$ in $O(\log i)$ lines. Suppose that A_{i+1} is inferred by the extension rule and it is $p_k \leftrightarrow B_k$. Then obtain $B_i \rightarrow B_i$ in a costant number of steps, introduce $p_k \leftrightarrow B_k$ (this is correct since in $B_0, \dots, B_i$ never occurs p_k) and in a costant number of steps obtain $B_i \rightarrow B_i \wedge p_k \leftrightarrow B_k$ which is $B_i \rightarrow B_{i+1}$. The proof then follows as in [BoBu] □.

4.1.2 tree-like $s\mathcal{F}$ simulation of tree-like $e\mathcal{F}$

Cook and Reckhow show in [CR] that $s\mathcal{F}$ p-simulates $e\mathcal{F}$. Here we show that this simulation is preseved also in the tree-like case and improving from $O(n^2)$ to $O(n \log n)$ the number of lines used. Observe that there is a $\mathcal{F}$ tree-like proof of $(\bigwedge_{i=1}^{k} A_i) \rightarrow A_l$ for any $l = 1, \dots k$ in $O(\log k)$ lines.

Theorem 4.2 *Given a tree-like $e\mathcal{F}$ proof P of the formula A in n lines, there is a tree-like $s\mathcal{F}$ proof of A in $O(n \log n)$ lines.*

Proof. Let P be the $e\mathcal{F}$ tree-like proof $A_1, \dots, A_n$, and suppose that k between the A_i's are formulas of the form $p_j \leftrightarrow B_j$ introduced by the extension rule. Consider the formula B defined by $\bigwedge_{j=1}^{k} (p_j \leftrightarrow B_i)$ following the order of introduction of the extension rules. First we give a tree-like $\mathcal{F}$ proof of $B \rightarrow A$. This is obtained by showing that for all $i = 1, \dots, n$ there is a proof of $B \rightarrow A_i$. By cases on how A_i is inferred in P:

case 1: A_i is an axiom, then there is a tree-like $\mathcal{F}$ proof of $B \rightarrow A_i$ in a costant number of lines;

case 2: A_i is obtained by *Modus Ponens* from A_j and A_k with $j, k < i$. We can therefore assume that there are a tree-like $\mathcal{F}$ proofs of $B \rightarrow A_j$ and $B \rightarrow (A_j \rightarrow A_i)$. A tree-like proof of $B \rightarrow A_i$ can be easy obtained in a costant number of lines.

case 3: A_i is $p_l \leftrightarrow B_l$ for some l. By previous observation we can obtain a proof of $B \rightarrow A_i$ in $O(\log k)$ lines.

This first part requires $O(n \log k)$ lines. Now A is obtained by the following steps:

1. obtain $\bigwedge_{j=1}^{k-1}(p_j \leftrightarrow B_i) \wedge (p_k \leftrightarrow B_k) \rightarrow \bigwedge_{j=1}^{k}(p_j \leftrightarrow B_i)$ in $O(\log k)$ lines from Lemma 4.1;
2. substitute in the formula previously obtained, B_k to p_k. This will not affect any other formula since p_k does not ocurr in previous lines in P;
3. obtain $B_k \leftrightarrow B_k$ in a constant number of lines;
4. substitute B_k to p_k in $B \rightarrow A$
5. obtain $\bigwedge_{j=1}^{k-1}(p_j \leftrightarrow B_i) \rightarrow A$ in a costant number of lines from the previous steps;
6. reiterate this process by substituting the p_i's in the reverse order respect to their introduction in P.

This second part requires $O(k \log k)$ lines. Since we can only bound above k with n, the total number of lines is $O(n \log n)$. □

4.2 Tree-like $\mathcal{F}$ simulations of tree-like $r\mathcal{F}$ and tree-like $\top/\bot$-$\mathcal{F}$

First we show that tree-like $\mathcal{F}$ simulates tree-like $s\mathcal{F}$ with a simulation polynomial in terms of lines but not of symbols. Then we discuss differences between $s\mathcal{F}$, $r\mathcal{F}$ and $\top/\bot$-$\mathcal{F}$.
The technique used is that of pushing substitution lines up above Modus Ponens lines, up to obtain isomoprhic istances of the axioms. This is also stated in Lemma 1.11 of [HM]. Let P be a tree-like $s\mathcal{F}$ proof. At each formula $A\sigma$ obtained by substitutions, we associate its *degree* d_σ as the depth of the formula $A\sigma$ in the tree associate to the proof P, and define the degree d_P of the proof as the maximal d_σ. Note that a *degree*-0 proof is a tree-like Frege proof.

Lemma 4.2 *Given a degree d tree-like $s\mathcal{F}$ proof P ending in a substitutions, of the formula A, there is a tree-like $s\mathcal{F}$ proof P' of the formula A with degree strictly less than d.*

Proof. Let P be a tree like proof $s\mathcal{F}$ and let $A\sigma$ be the last substitution in the proof. There are three cases.

case 1: A is an istance of an axiom scheme. Then the trasformation from P in P' is :

$$\frac{A}{A\sigma} \quad \text{in} \quad A\sigma$$

The proof P' is valid since $A\sigma$ is an istance of an axiom and the degree is reduced to 0.

case 2: A is obtained by Modus Ponens from B and $B \to A$. Then the trasformation is:

$$\frac{\dfrac{B \quad B \to A}{A}}{A\sigma} \quad \text{to} \quad \frac{\dfrac{B}{B\sigma} \quad \dfrac{B \to A}{B\sigma \to A\sigma}}{A\sigma}$$

P' is a valid $d-1$ degree proof.

case 3: A is obtained by substitutions, then the trasformations is:

$$\frac{\dfrac{B}{B\lambda = A}}{A\sigma} \quad \text{in} \quad \frac{B}{B\theta}$$

where θ is the substitution $\lambda\sigma$ obatined by compositions. Also in this case P' remains a valid $d-1$ degree proof.

Theorem 4.3 *Given a tree-like $s\mathcal{F}$ proof P in $s\mathcal{F}$, of a formula A of n lines of which k are obtained by the rule of substitutions, there is a tree like Frege proof P' of the same formula with $n-k$ lines.*

Proof. Given P we construct, by induction on d_P a proof P' of degree 0. At each inductive step we copy the proof tree if no substitution rule is used; when a substitution is used we apply Lemma 4.2. At the base case d_P is 0 and so we have done. Observe that in P' we eliminate all the lines obtained by substitution and maintain all the other lines, so the number of lines of P' is $n-k$. □

Let m be the size of P. It is easily seen that if we apply the previous theorem to a $s\mathcal{F}$ system then the formulas introduced by the axioms in P' can have $O(m^{k+1})$ number of symbols and so we cannot conclude that $\mathcal{F}$ system p-simulate tree-like $s\mathcal{F}$ systems. But it is easy to see that in the case of tree-like $r\mathcal{F}$ or $\top/\bot$-$\mathcal{F}$ systems the number of symbols in the formulas introduced by axioms in P' is the same as the original proof.

Theorem 4.4 *If P is a tree-like $r\mathcal{F}$ or $\top/\bot$-$\mathcal{F}$ proof of n lines, of which k introduced by the rule of substitutions, and m symbols of the formula A, then there is a tree-like $\mathcal{F}$ proof of A in $n-k$ lines and m symbols.*

Note that the known $r\mathcal{F}$ simulation of $s\mathcal{F}$ does not preserve the tree-like property (see [Bu1] Lemma 17).

References

[Bol] M. Bonet. *Number of symbols in Frege proofs with and without the deduction rule.* In Arithmetic, proof theory and computational complexity, Oxford University Press Eds. P. Clote and J. Krajìček -1992.

[BoBu] M. Bonet, S. Buss. *The duduction rule and linear and near-linear proof simulations.* Journal of Symbol Logic, **58** (1993) pg 688-709.

[Bu1] S. Buss. *Some remarks on length of propositional proofs.* Archive for Mathemtical Logic. *To appear.*

[Bu2] S. Buss, *Lecture on proof theory.* TR SOCS-96.1 School of Computer Science - Mc Gill University 1996.

[CR] S. Cook, R. Reckhow. *The relative efficiency of propositional proof systems.* Journal of Symbolic Logic, **44** (1979) pg 57-64.

[G] J. H. Gallier. Logic for Computer Science - Foundations of Automatic Theorem Proving. J. Wiley & Sons. 1987

[HM] J. R. Hindley, D. Meredith. *Principal type schemes and condensed detachment.* Journal of Symbolic Logic, **55** (1990) pg 90-105.

[Kr] J. Krajìček. *On the number of Steps in proof.* Annals of Pure and Applied Logic **41** (1989) pg 153-178.

[KP] J. Krajìček, P. Pudlàk. *Propositional proof systems,the consistency of first order theories and the complexity of computation.* Journal of Symbolic Logic. **54**(1989) pg 1063-1079.

[LV] M.Li. P. Vitanyi. An Introduction to Kolmogorov Complexity and its Application. Springer Verlag 1993.

[Or] V.P. Orevkov. *Recostruction of a proof from its scheme.* Soviet Mathematics Doklady **35**(1987) pg 326-329.

[P] R. Parikh *Some results on the length of proofs.* Trans. A.M.S. **177**(1973) pg 29-36.

[Ur] A. Urquhart. *The number of lines in Frege proof with substitution.* Archive for Mathematical Logic. *To appear.*

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

Research Reports – 1996

LSI-96-1-R "(Pure) Logic out of Probability", Ton Sales.

LSI-96-2-R "Automatic Generation of Multiresolution Boundary Representations", C. Andújar, D. Ayala, P. Brunet, R. Joan-Arinyo, and J. Solé.

LSI-96-3-R "A Frame-Dependent Oracle for Linear Hierarchical Radiosity: A Step towards Frame-to-Frame Coherent Radiosity", Ignacio Martin, Dani Tost, and Xavier Pueyo.

LSI-96-4-R "Skip-Trees, an Alternative Data Structure to Skip-Lists in a Concurrent Approach", Xavier Messeguer.

LSI-96-5-R "Change of Belief in SKL Model Frames (Automatization Based on Analytic Tableaux)", Matías Alvarado and Gustavo Núñez.

LSI-96-6-R "Compressibility of Infinite Binary Sequences", José L. Balcázar, Ricard Gavaldà, and Montserrat Hermo.

LSI-96-7-R "A Proposal for Word Sense Disambiguation using Conceptual Distance", Eneko Agirre and German Rigau.

LSI-96-8-R "Word Sense Disambiguation Using Conceptual Density", Eneko Agirre and German Rigau.

LSI-96-9-R "Towards Learning a Constraint Grammar from Annotated Corpora Using Decision Trees", Lluis Màrquez and Horacio Rodríguez.

LSI-96-10-R "POS Tagging Using Relaxation Labelling", Lluís Padró.

LSI-96-11-R "Hybrid Techniques for Training HMM Part-of-Speech Taggers", Ted Briscoe, Greg Grefenstette, Lluís Padró, and Iskander Serail.

LSI-96-12-R "Using Bidirectional Chart Parsing for Corpus Analysis", A. Ageno and H. Rodríguez.

LSI-96-13-R "Limited Logical Belief Analysis", Antonio Moreno.

LSI-96-14-R "Logic as General Rationality: A Survey", Ton Sales.

LSI-96-15-R "A Syntactic Characterization of Bounded-Rank Decision Trees in Terms of Decision Lists", Nicola Galesi.

LSI-96-16-R "Algebraic Transformation of Unary Partial Algebras I: Double-Pushout Approach", P. Burmeister, F. Rosselló, J. Torrens, and G. Valiente.

LSI-96-17-R "Rewriting in Categories of Spans", Miquel Monserrat, Francesc Rosselló, Joan Torrens, and Gabriel Valiente.

LSI-96-18-R "On the Depth of Randomly Generated Circuits", Tatsuie Tsukiji and Fatos Xhafa.

LSI-96-19-R "Learning Causal Networks from Data", Ramon Sangüesa i Solé.

LSI-96-20-R "Boundary Generation from Voxel-based Volume Representations", R. Joan-Arinyo and J. Solé.

LSI-96-21-R "Exact Learning of Subclasses of CDNF Formulas with Membership Queries", Carlos Domingo.

LSI-96-22-R "Modeling the Thermal Behavior of Biosphere 2 in a Non-Controlled Environment Using Bond Graphs", Angela Nebot, François E. Cellier, and Francisco Mugica.

LSI-96-23-R "Obtaining Synchronization-Free Code with Maximum Parallelism", Ricard Gavaldá, Eduard Ayguadé, and Jordi Torres.

LSI-96-24-R "Memoisation of Categorial Proof Nets: Parallelism in Categorial Processing", Glyn Morrill.

LSI-96-25-R "Decision Trees Have Approximate Fingerprints", Víctor Lavín and Vijay Raghavan.

LSI-96-26-R "Visible Semantics: An Algebraic Semantics for Automatic Verification of Algorithms", Vicent-Ramon Palasí Lallana.

LSI-96-27-R "Massively Parallel and Distributed Dictionaries on AVL and Brother Trees", Joaquim Gabarró and Xavier Messeguer.

LSI-96-28-R "A Maple package for semidefinite programming", Fatos Xhafa and Gonzalo Navarro.

LSI-96-29-R "Bounding the expected length of longest common subsequences and forests", Ricardo A. Baeza-Yates, Ricard Gavaldà, and Gonzalo Navarro.

LSI-96-30-R "Parallel Computation: Models and Complexity Issues", Raymond Greenlaw and H. James Hoover.

LSI-96-31-R "ParaDict, a Data Parallel Library for Dictionaries (Extended Abstract)", Joaquim Gabarró and Jordi Petit i Silvestre.

LSI-96-32-R "Neural Networks as Pattern Recognition Systems", Lourdes Calderón.

LSI-96-33-R "Semàntica externa: una variant interessant de la semàntica de comportament" (written in Catalan), Vicent-Ramon Palasí Lallana.

LSI-96-34-R "Automatic verification of programs: algorithm ALICE", V.R. Palasí Lallana.

LSI-96-35-R "Multiresolution Approximation of Polyhedral Solids", D. Ayala, P. Brunet, R. Joan-Arinyo, I. Navazo.

LSI-96-36-R "Algebraic Transformation of Unary Partial Algebras II: Single-Pushout Approach", P. Burmeister, M. Montserrat, F. Rosselló, and G. Valiente.

LSI–96–37–R "Probabilistic Conditional Independence: A Similarity-Based Measure and its Application to Causal Network Learning", Ramon Sangüesa Solé, Joan Cabós Fabregat, and Ulises Cortés García.

LSI–96–38–R "Analysing the Process of Enforcing Integrity Constraints", Enric Mayol and Ernest Teniente.

LSI–96–39–R "Reducció de l'equivalència inicial visible a teoremes inductius" (written in Catalan), Vicent-Ramon Palasí Lallana.

LSI–96–40–R "A Compendium of Problems Complete for Symmetric Logarithmic Space", Carme Àlvarez and Raymond Greenlaw.

LSI–96–41–R "Semàntica algebraica del llenguatge AL: l'algorisme α" (written in Catalan), V.R. Palasí Lallana.

LSI–96–42–R "Partial Occam's Razor and its Applications", Carlos Domingo, Tatsuie Tsujiki, and Osamu Watanabe.

LSI–96–43–R "Transparent Distributed Problem Resolution in the MAKILA Multi-Agent System", Karmelo Urzelai.

LSI–96–44–R "The Intensional Events Method for Consistent View Updating", Dolors Costal, Ernest Teniente, and Toni Urpí.

LSI–96–45–R "Extending Eiffel as a Full Life-cycle Language", Alonso J. Peralta and Joan Serras.

LSI–96–46–R "Analysis of Methods for Generating Octree Models of Objects from Their Silhouettes", Marta Franquesa and Pere Brunet.

LSI–96–47–R "Learning nearly monotone k-term DNF", Jorge Castro, David Guijarro, and Víctor Lavín.

LSI–96–48–R "Learning Monotone Term Decision Lists", David Guijarro, Víctor Lavín, and Vijay Raghavan.

LSI–96–49–R "Coding Complexity: The Computational Complexity of Succinct Descriptions", José L. Balcázar, Ricard Gavaldà, and Osamu Watanabe.

LSI–96–50–R "Algorithms for Learning Finite Automata from Queries: A Unified View", José L. Balcázar, Josep Díaz, Ricard Gavaldà, and Osamu Watanabe.

LSI–96–51–R "Mètodes de validació d'esquemes de bases de dades deductives" (written in Catalan), Carles Farré.

LSI–96–52–R "The Parallel Complexity of Positive Linear Programming", Luca Trevisan and Fatos Xhafa.

LSI–96–53–R "Polynomial-Time Algorithms for Some Self-Duality Problems", Carlos Domingo.

LSI–96–54–R "Patterns in Random Binary Search Trees", Philippe Flajolet, Xavier Gourdon, and Conrado Martínez.