Separating Words with Small Grammars

James Currie*
Department of Mathematics
University of Winnipeg
Winnipeg, Manitoba, Canada R3B 2E9
currie@io.uwinnipeg.ca

Holger Petersen
Institut für Informatik der Universität Stuttgart
Breitwiesenstraße 20–22
D-70565 Stuttgart, Germany
petersen@informatik.uni-stuttgart.de

John Michael Robson
LaBRI
Université Bordeaux I
351, cours de la Libération
33405 TALENCE Cedex France
robson@labri.u-bordeaux.fr

Jeffrey Shallit*
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
shallit@graceland.uwaterloo.ca

February 16, 1999

Abstract

We study the following problem: given two words w and x, with $|w|, |x| \leq n$, what is the size of the smallest context-free grammar G which generates exactly one of $\{w, x\}$? If $|w| \neq |x|$, then we prove there exists a G separating w from x of size $O(\log \log n)$, and this bound is best possible. If $|w| = |x|$, then we get an upper bound on the size of G of $O(\log n)$, and a lower bound of $\Omega(\frac{\log n}{\log \log n})$.

*Research supported in part by a grant from NSERC.

1 Introduction and Definitions

Consider two deterministic finite automata (DFA's) $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$. A well-known theorem [7, 11] states that if M_1 and M_2 accept different languages, then there is a "short" string accepted by one but not the other. More precisely, if $L(M_1) \neq L(M_2)$, then there exists $w \in \Sigma^*$ such that

$$w \in (L(M_1) - L(M_2)) \ \cup \ (L(M_2) - L(M_1))$$

and

$$|w| \leq \mathrm{card}(Q_1) + \mathrm{card}(Q_2) - 2. \tag{1}$$

Here $\mathrm{card}(Q)$ denotes the cardinality of the set Q. The bound (1) is best possible, even over a unary alphabet.

The "inverse" problem — where we are given two distinct *words* and want to find a "small" DFA separating them — appears to be much more difficult. More formally, let Σ be a finite alphabet, and let $w, x \in \Sigma^*$ be distinct words of length $\leq n$. Goralčík and Koubek [4] were apparently the first to study the problem of finding a small DFA M that *separates* w from x, i.e., such that

$$\mathrm{card}(L(M) \cap \{w, x\}) = 1.$$

They observed that w and x can be separated with $O(\log n)$ states if $|w| \neq |x|$, and sketched a proof that $o(n)$ states suffice if $|w| = |x|$. Robson [8] improved the latter upper bound to $O(n^{2/5}(\log n)^{3/5})$. Also see [9].

In this note we study this last problem, where "deterministic finite automaton" is replaced with "context-free grammar" (CFG). We need a notion of the size of a context-free grammar. If $G = (V, \Sigma, P, S)$ then we define the *description size* of G, $\mathrm{ds}(G)$, as follows:

$$\mathrm{ds}(G) = 1 + \mathrm{card}(V) + \mathrm{card}(\Sigma) + \sum_{(A,\beta) \in P} (|\beta| + 3).$$

(Compare similar measures of Gruska [5].) For a variable $A \in V$, we define

$$L(A) = \{w \in \Sigma^* \ : \ A \Longrightarrow^* w\},$$

and

$$\mathrm{ds}(A) = \sum_{(A,\beta) \in P} (|\beta| + 3).$$

2 The case of unequal lengths

In this section we find an upper bound on the size of a separating grammar in the case that the two words are of different length.

Theorem 1 *Suppose $w, x \in \Sigma^*$ with $|w|, |x| \leq n$ and $|w| \neq |x|$. Let $k = \mathrm{card}(\Sigma)$. Then there exists a CFG $G_{w,x}$ that separates w from x, with description size $\mathrm{ds}(G_{w,x}) = O(k + \log\log n)$.*

Proof. By the prime number theorem, if $|w|, |x| \leq n$ and $|w| \neq |x|$, then there exists a prime $p = O(\log n)$ such that $|w| \not\equiv |x| \pmod{p}$. More precisely, for $n \geq 2$, such a prime exists with $p \leq 4.4 \log n$ (see [11, Example 6]). Let $i = |w| \bmod p$. Then $w \in (\Sigma^p)^*\Sigma^i$ and $x \notin (\Sigma^p)^*\Sigma^i$. Thus it suffices to show how to generate $(\Sigma^p)^*\Sigma^i$ with a grammar of description size $O(k + \log p)$.

For a non-negative integer r, define

$$s(r) = \begin{cases} 0, & \text{if } r = 0; \\ r - 1, & \text{if } r \text{ odd}; \\ r/2, & \text{if } r > 0 \text{ is even}, \end{cases}$$

and set $T_r = \{r, s(r), s^2(r), \ldots\}$. Note that T_r is a finite set; in fact, it is easy to see that $\mathrm{card}(T_r) \leq 2 + 2\log_2 r$ for $r \geq 1$.

Now define $G_{w,x} = (V, \Sigma, P, S)$ as follows. Let $T = T_p \cup T_i$, and let

$$V = \{A_r \ : \ r \in T\} \cup \{S, B, C\}.$$

Let P be the following set of productions:

$$\begin{aligned} S &\to CA_i, \\ C &\to \epsilon \mid A_pC, \\ B &\to c, && \forall\ c \in \Sigma; \\ A_0 &\to \epsilon, \\ A_{2j} &\to A_jA_j, && \text{if } 2j \in T - \{0\}; \\ A_{2j+1} &\to BA_{2j}, && \text{if } 2j + 1 \in T. \end{aligned}$$

Then it is easy to prove by induction that $L(A_j) = \Sigma^j$ for all $j \in T$. Hence $L(C) = (\Sigma^p)^*$, and $L(S) = (\Sigma^p)^*\Sigma^i$. The total description size of $G_{w,x}$ is $O(k + \log p) = O(k + \log\log n)$. ■

3 The case of equal lengths

In this section we find an upper bound on the size of a separating grammar in the case that the two words are of equal length.

Theorem 2 *Suppose $w, x \in \Sigma^*$ with $|w|, |x| \leq n$ and $|w| = |x|$. Let $k = \mathrm{card}(\Sigma) \geq 2$. Then there exists a CFG $G_{w,x}$ that separates w from x with description size $\mathrm{ds}(G_{w,x}) = O(k + \log n)$.*

Proof. If $w \neq x$ and $|w| = |x|$ then there must exist a position j, $1 \leq j \leq n$, in which $a = w_j \neq x_j = b$ for $a, b \in \Sigma$. Then $w \in \Sigma^{j-1} a \Sigma^{n-j}$ and $x \notin \Sigma^{j-1} a \Sigma^{n-j}$.

Now define $G_{w,x} = (V, \Sigma, P, S)$ as follows. Let $T' = T_{j-1} \cup T_{n-j}$, where T is defined as in the proof of Theorem 1, and let

$$V = \{A_r \ : \ r \in T'\} \cup \{S, B, C\}.$$

Let P be the following set of productions:

$$\begin{aligned}
S &\rightarrow A_{j-1}\ a\ A_{n-j} \\
B &\rightarrow c, && \forall\ c \in \Sigma; \\
A_0 &\rightarrow \epsilon, \\
A_{2i} &\rightarrow A_i A_i, && \text{if } 2i \in T' - \{0\}; \\
A_{2i+1} &\rightarrow B A_{2i}, && \text{if } 2i + 1 \in T'.
\end{aligned}$$

It is easy to see that $L(S) = \Sigma^{j-1} a \Sigma^{n-j}$, so $w \in L(S)$ and $x \notin L(S)$. We observe that $\mathrm{ds}(G_{w,x}) = O(k + \log n)$. ■

4 Lower bounds

It is natural to wonder how close our results are to being optimal. In this section we obtain some lower bounds.

Our technique is based on the following lemma. Let S be a finite set. We call a finite collection $\mathcal{U} = \{U_1, U_2, \ldots, U_j\}$ of subsets of S a *separating collection* if for all $x, y \in S$ with $x \neq y$, there exists a set $C_{xy} \in \mathcal{U}$ such that

$$\mathrm{card}(C_{xy} \cap \{x, y\}) = 1.$$

Lemma 3 *Suppose S is a finite set of cardinality $m \geq 1$. If $\mathcal{U}$ is a separating collection for S, then* $\mathrm{card}(\mathcal{U}) \geq \lceil \log_2 m \rceil$. *Furthermore, this bound is best possible.*

Proof. For each $x \in S$ consider the set of indices of members of $\mathcal{U}$ to which x belongs, that is,

$$V_x = \{i : x \in U_i\}.$$

Then we claim that all the sets V_x are distinct; for if not we would have $V_x = V_y$ for some $y \neq x$, and then every set in $\mathcal{U}$ containing x would also contain y. Hence $\mathcal{U}$ is not a separating collection. It follows that $2^{\mathrm{card}(\mathcal{U})} \geq m$, and hence $\mathrm{card}(\mathcal{U}) \geq \log_2 m$. Since the cardinality of a set must be an integer, we obtain $\mathrm{card}(\mathcal{U}) \geq \lceil \log_2 m \rceil$.

We now show the bound is best possible. Without loss of generality, we may assume $S = \{0, 1, 2, \ldots, m-1\}$. Then define

$$U_i = \{r \in S \ : \ \text{the } i\text{'th least significant bit of the binary expansion of } r \text{ is } 1\}$$

for $0 \leq i < \lceil \log_2 m \rceil$, and set $\mathcal{U} = \{U_i \ : \ 0 \leq i < \lceil \log_2 m \rceil\}$. It works. ■

Remark. We do not know who first proved Lemma 3, but it must be very well known. It can be found (stated in the language of perfect hash families) in [1, Thm. 1.2, Cor. 1.3]. We thank C. J. Colbourn for bringing this reference to our attention.

We now apply Lemma 3 to get lower bounds corresponding to Theorems 1 and 2.

Theorem 4 *Let $k = \mathrm{card}(\Sigma)$ be fixed. For all $n \geq 1$ there exists a pair of distinct words $w, x \in \Sigma^n$ requiring a context-free grammar of description size $\Omega(\frac{\log n}{\log\log n})$ to separate them.*

Proof. Without loss of generality we may assume that $\Sigma = \{1, 2, \ldots, k\}$. Let $G = (V, \Sigma, P, S)$ be a CFG separating w from x. Without loss of generality we may assume that $V = \{A_1, A_2, \ldots, A_r\}$ and $A_1 = S$.

We may encode G as a string s over the alphabet $V \cup \Sigma \cup \{\#\}$ as follows: if

$$P = \{B_1 \to \beta_1, B_2 \to \beta_2, \ldots, B_t \to \beta_t\}$$

then

$$s = B_1\#\beta_1\#B_2\#\beta_2\#\cdots\#B_t\#\beta_t\#.$$

Then

$$|s| = \sum_{1 \leq i \leq t} (|\beta_i| + 3) \leq \mathrm{ds}(G),$$

and each position in s can take on at most $|\Sigma| + |V| + 1 \leq \mathrm{ds}(G)$ different values. Suppose $\mathrm{ds}(G) \leq d$. Then there are at most d^d different strings encoding such a grammar, and hence at most d^d different grammars with description size $\leq d$. Hence there are at most d^d different languages generated by CFG's with description size $\leq d$. Let $\mathcal{U}$ be the collection of all these languages.

Now apply Lemma 3. There are k^n distinct words of length n. If $\mathcal{U}$ is a separating collection for the set of words of length $\leq n$, then we have

$$d^d \geq \mathrm{card}(\mathcal{U}) \geq \log_2 k^n.$$

It follows that

$$d \log d \geq \log n + \log\log_2 k,$$

and so, for fixed k, we have $d = \Omega(\frac{\log n}{\log\log n})$. ■

Theorem 5 *Let $k = \mathrm{card}(\Sigma)$ be fixed. For all $n \geq 1$ there exists a pair of words w, x with $|w| \neq |x|$ and $|w|, |x| \leq n$ requiring a context-free grammar with description size $\Omega(\frac{\log\log n}{\log\log\log n})$ to separate them.*

Proof. Suppose G is a context-free grammar separating w and x, where $|w| \neq |x|$. Without loss of generality we may assume $k = 1$.

There are $n + 1$ strings in 0^* of length $\leq n$, so by applying Lemma 3 as in the proof of the previous theorem, we find that $d^d \geq \log_2(n + 1)$, and so $d = \Omega(\frac{\log\log n}{\log\log\log n})$. ■

We finish this section by observing that the lower bound technique in Lemma 3 cannot improve the lower bounds in Theorems 4 and 5 without a significant new idea. This is because, as we show below in Theorem 6, there are $d^{\Omega(d)}$ distinct languages generable by context-free grammars with description size $\leq d$ — even over a unary alphabet.

Theorem 6 *Any subset $S \subseteq \{\epsilon, 0, 0^2, \ldots, 0^{n \cdot 2^n - 1}\}$ can be generated using a grammar with description size $O(2^n)$. Hence $2^{n \cdot 2^n} = (2^n)^{2^n}$ different languages can be generated by grammars with description size $O(2^n)$.*

Proof. Let $A_j = \{0^i \ : \ jn \leq i < (j+1)n\}$ for $0 \leq j < 2^n$, and for $0 \leq j < 2^n$ define $S_j = S \cap A_j$ for $0 \leq j < 2^n$. We observe that $S_j \subseteq A_0 0^{jn}$ for $0 \leq j < 2^n$. Let W_j be the subset of A_0 such that $S_j = W_j 0^{jn}$. Then we have

$$S = \bigcup_{0 \leq j < 2^n} W_j 0^{jn}.$$

Now we describe a "small" grammar $G_S = (V, \{0\}, P, R)$ generating S. The idea is to generate all nonempty subsets of A_0 and all strings 0^{jn}, $0 \leq j < 2^n$, with $O(2^n)$ description size.

First, we create productions

$$B_j \to \begin{cases} \epsilon, & \text{if } j = 0; \\ 0\, B_{j-1}, & \text{if } j > 0. \end{cases}$$

for $0 \leq j \leq n$. Note that $L(B_j) = \{0^j\}$, and $\sum_{0 \leq j \leq n} \mathrm{ds}(B_j) = 5n + 3$.

Second, for all $2^n - 1$ nonempty subsets $T \subseteq A_0$, we create productions as follows:

$$C_T \to \begin{cases} B_j, & \text{if } \mathrm{card}(T) = 1 \text{ and } T = \{0^j\}; \\ C_{T - \{j\}} \mid B_j, & \text{if } \mathrm{card}(T) > 1 \text{ and } j = \max_{t \in T} t. \end{cases}$$

Note that $L(C_T) = T$, and

$$\sum_{\substack{T \subseteq A_0 \\ T \neq \emptyset}} \mathrm{ds}(C_T) = 4n + 8(2^n - n - 1).$$

Third, for $0 \leq j < 2^n$, we create productions as follows:

$$X_j \to \begin{cases} \epsilon, & \text{if } j = 0; \\ B_n, & \text{if } j = 1; \\ B_n X_{j-1}, & \text{if } 1 < j < 2^n; \end{cases}$$

Note that $L(X_j) = 0^{jn}$ and

$$\sum_{0 \le j < 2^n} \mathrm{ds}(X_j) = 3 + 4 + 5(2^n - 2).$$

Finally, we create productions

$$R \to C_{W_j} X_j$$

for all j such that $0 \le j < 2^n$ and W_j is nonempty. Then $\mathrm{ds}(R) \le 5 \cdot 2^n$.

Now let

$$V = \{R\} \cup \{X_j : 0 \le j < 2^n\} \cup \{C_T : T \subseteq A_0\} \cup \{B_j : 0 \le j \le n\}.$$

It is now easy to see that $G_S = (V, \{0\}, P, R)$ has description size $O(2^n)$ and generates the set S. ■

5 An improved lower bound for unequal lengths

In this section we improve the lower bound of Theorem 5. The method is similar to that used by Goralčík and Koubek [4] in the case of finite automata.

We assume that the grammars in the section are in "binary normal form"; that is, that every production is of the form $A \to BC$ or $A \to B$ for $B, C \in V$, or $A \to a$ for $a \in \Sigma$, or $A \to \epsilon$. We can do this by (1) replacing all occurrences of terminals a in the right-hand sides of productions of length ≥ 2 by a new variable X_a, and then adding the production $X_a \to a$; and (2) replacing all productions of the form $A \to X_1 X_2 \cdots X_t$ for $t > 2$ with $t - 1$ productions of the form $A \to X_1 B_1$, $B_1 \to X_2 B_2$, ..., $B_{t-3} \to X_{t-2} B_{t-2}$, $B_{t-2} \to X_{t-1} X_t$. This change converts every grammar to one in binary normal form with only a linear increase in description size.[1]

We begin with some terminology. Fix a grammar $G = (V, T, P, S)$. A word a^i is called a *pumping word* if there exist a variable A and integers $i_1, i_2 \ge 0$ such that $i = i_1 + i_2$ and $A \stackrel{*}{\underset{G}{\Longrightarrow}} a^{i_1} A a^{i_2}$. A *cutting operation* or *cut* can be performed on a parse tree T if there exist two occurrences of the same variable A at nodes N_1 and N_2, with N_2 a descendant of N_1. The cut is performed by replacing the subtree rooted at N_1 by that rooted at N_2, obtaining a new tree T'. If the yield of T is w, and the yield of T' is w', then we say w' is *obtained by cutting* from w. If $w = a^j$ and $w' = a^k$, then a^{j-k} is a pumping word. A *pasting operation* is the result of undoing a cut. We say w is *obtained by pasting* from w'.

We say a cut is *basic* if there is no other possible cut, possibly involving other variables, within the subtree rooted at N_1. A pumping word is called *basic* if it is cut out by some basic cut. Note that if a cut is possible, then a basic cut is also possible.

[1] Note this may not be the case for Chomsky normal form, since Chomsky normal form also demands no unit productions. The standard algorithm for removing unit productions can potentially increase the size of the grammar by a quadratic factor.

For the rest of this section, unless otherwise stated, we assume $G = (V, T, P, S)$ is in binary normal form, with $s := \text{card}(V)$ and $m := 2^s$.

Lemma 7 *If a parse tree T has yield x with $|x| \geq m$, then it is possible to perform a cut in it.*

Proof. The proof is the same as the familiar proof of the pumping lemma [2]. Since $|x| \geq 2^s$, we have $|x| > 2^{s-1}$. Since every node of the parse tree has outdegree ≤ 2, there is a path from the root of T to a leaf of length $\geq s + 1$. Such a path has $\geq s + 2$ nodes, all of which but the last are labeled with variables. Hence some variable occurs twice on this path. ■

Lemma 8 *Every basic pumping word is of length $< 2m$.*

Proof. Let a^i be a word with $i \geq 2m$, and consider an arbitrary cut operation which cuts out a^i. Consider the subtree T_1 rooted at node N_1 of the cut. The yield of T_1 is of length $\geq 2m$. Since G is in binary normal form, the node N_1 has at most two children. Thus N_1 has a child which is the root of a subtree T_2 with yield of length $\geq m$. By Lemma 7, it is possible to perform a cut in T_2. Hence the cut at N_1 is not basic. Since we considered an arbitrary cut, a^i is not cut out by any basic cut and hence a^i is not a basic pumping word. ■

Lemma 9 *Given a derivation $S \stackrel{*}{\Longrightarrow} w$, there is a word w' with $|w'| < 2ms$, obtained from w by a sequence of zero or more basic cuts, such that w' has a derivation in G using all the variables used in the derivation of w.*

Proof. Let T be the parse tree corresponding to the given derivation $S \stackrel{*}{\Longrightarrow} w$. Consider reducing T to a tree T'' with yield w'' by a sequence of basic cuts $c_1, c_2, \ldots, c_j$, such that no further cuts are possible in T''. By Lemma 7 we have $|w''| < m$. Suppose that the basic cuts which removed the last occurrence of some variable are $c_{n_1}, \ldots, c_{n_k}$, in that order. Then we can perform pasting operations to T', reversing the effects of cuts $c_{n_k}, \ldots, c_{n_1}$, and obtaining a parse tree T' with yield w'. Then all the other pasting operations corresponding to the remaining c_i can be performed in any order to obtain a tree with yield w. Since w has been obtained from w' by pasting basic pumping words, w' can be obtained from w by basic cuts. Then, since $k \leq s - 1$, we have

$$|w'| < |w''| + 2m(s-1) < m + 2m(s-1) < 2ms.$$

■

We now fix one such sequence of basic cuts, and define $r(w)$ to be the word w' obtained from w by this sequence of basic cuts.

Theorem 10 *Let $G = (V, T, P, S)$ be a context-free grammar in binary normal form with $s = \mathrm{card}(V)$. Define $m = 2^s$, $m' = \mathrm{lcm}(1, 2, \ldots, 2m)$, and $M = 4m^2m'$. Then G does not separate a^{2M} from a^{3M}.*

Proof. We show $a^{2M} \in L(G)$ iff $a^{3M} \in L(G)$.

$\Longrightarrow$: Suppose $w = a^{2M} \in L(G)$, and consider a parse tree T for a^{2M}. Since $|w| = 2M \geq m$, we can perform a pasting operation on T, which increases the length of the resulting yield by i, where a^i is a basic pumping word. By Lemma 8 we have $i < 2m$, and so $i \mid m'$. But $m' \mid M$, so we can perform this pasting operation M/i times to get a derivation of a^{3M}.

$\Longleftarrow$: Consider the set $B = \{a^{b_1}, a^{b_2}, \ldots, a^{b_j}\}$ of all nonempty basic pumping words, where $b_1 \leq b_2 \leq \cdots \leq b_j$. Define $g := \gcd_{x \in B} |x|$. Now if t is an integer linear combination of the b_i, then $g \mid t$. Furthermore, by a well-known result [3], if $u \geq (b_1/g - 1)(b_j/g - 1)$, then gu is a non-negative integer linear combination of the b_i. Since $b_i < 2m$ for $1 \leq i \leq j$, it follows that if $i \geq 4m^2$, then $a^i \in B^*$ iff $g \mid i$.

Now suppose $w = a^{3M} \in L(G)$. Without loss of generality, we may assume that every variable in V is used in the derivation of w; for if not, we could replace V by V', where V' consists only of those variables appearing in the derivation of w. Consider the word $a^x = r(a^{3M})$, where r is the function defined after Lemma 9.

Since a^x was obtained from a^{3M} by a sequence of basic cuts, we must have $a^{3M-x} \in B^*$. By Lemma 9, we have $x < 2ms < M$. Hence $3M - x > 2M - x > M \geq 4m^2$. Since $g \mid M$, it follows that $a^{2M-x} \in B^*$. By Lemma 9, the parse tree for a^x uses all variables of V, so we can perform any sequence of paste operations. Thus we can perform pasting operations that add $2M - x$ a's to a^x, obtaining a^{2M}, and so $a^{2M} \in L(G)$. ■

Corollary 11 *Let $k = \mathrm{card}(\Sigma)$ be fixed. For all $n \geq 1$ there exist words w, x with $|w| \neq |x|$ and $|w|, |x| \leq n$ requiring a context-free grammar with description size $\Omega(\log \log n)$ to separate them.*

Proof. Let G be a context-free grammar with description size d. By previous remarks we may convert G to an equivalent grammar G' in binary normal form with description size αd for some constant α. Then by Theorem 10, G' (and hence G) fails to separate a^{2M} from a^{3M}. But $m = 2^s \leq 2^{\alpha d}$. Also, $m' = \mathrm{lcm}(1, 2, \ldots, 2m) \leq e^{2.08m}$ by a well-known estimate [10]. It follows that $3M \leq 12 \cdot 2^{2\alpha d} \cdot e^{2.08 \cdot 2^{\alpha d}}$, and so $d = \Omega(\log \log n)$, where $n = 3M$. ■

6 Separating grammars with words

Finally, we note that there is no bound similar to that in (1) for CFG's.

Theorem 12 *There is no computable function $f : \mathbb{N} \to \mathbb{N}$ with the following property: if G_1, G_2 are context-free grammars with $L(G_1) \neq L(G_2)$ and $\mathrm{ds}(G_1), \mathrm{ds}(G_2) \leq n$, then there exists w with $|w| \leq f(n)$ such that $w \in (L(G_1) - L(G_2)) \cup (L(G_2) - L(G_1))$.*

Proof. Assume that such a function exists. Then a Turing machine could test if $L(G_1) = L(G_2)$ by computing $b = f(\max(\mathrm{ds}(G_1), \mathrm{ds}(G_2)))$ and testing membership of w in $L(G_1)$ and $L(G_2)$ for all $|w| \leq b$. The membership test can be done, for example, using the well-known Cocke-Younger-Kasami algorithm [6, pp. 139–141]. If such a w is found with

$$w \in (L(G_1) - L(G_2)) \ \cup \ (L(G_2) - L(G_1)),$$

then reject; otherwise accept. This would give a decision procedure for testing equality of context-free languages, which is well-known to be recursively unsolvable [2, Thm. 6.3b].

7 Acknowledgments.

We would like to thank David Swart and the referees for their careful reading of this paper.

References

[1] M. Atici, S. S. Magliveras, D. R. Stinson, and W.-D. Wei. Some recursive constructions for perfect hash families. *J. Combinatorial Designs* **4** (1996), 353–363.

[2] Y. Bar-Hillel, M. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. *Z. Phonetik. Sprachwiss. Kommunikationsforsch.* **14** (1961), 143–172.

[3] A. Brauer. On a problem of partitions. *Amer. J. Math.* **64** (1942), 299–312.

[4] P. Goralčík and V. Koubek. On discerning words by automata. In L. Kott, editor, *Proc. 13th Int'l Conf. on Automata, Languages, and Programming (ICALP)*, Vol. 226 of *Lecture Notes in Computer Science*, pp. 116–122. Springer-Verlag, 1986.

[5] J. Gruska. On the size of context-free grammars. *Kybernetika* **8** (1972), 213–218.

[6] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[7] E. F. Moore. Gedanken-experiments on sequential machines. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, Vol. 34 of *Annals of Mathematics Studies*, pp. 129–153. Princeton University Press, Princeton, 1956.

[8] J. M. Robson. Separating strings with small automata. *Inform. Process. Lett.* **30** (1989), 209–214.

[9] J. M. Robson. Separating words with machines and groups. *RAIRO Inform. Théor. App.* **30** (1996), 81–86.

[10] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Ill. J. Math.* **6** (1962), 64–94.

[11] J. Shallit and Y. Breitbart. Automaticity I: Properties of a measure of descriptional complexity. *J. Comput. System Sci.* **53** (1996), 10–25.