J. M. ROBSON

Separating words with machines and groups

Informatique théorique et applications, tome 30, nº 1 (1996), p. 81-86.

<http://www.numdam.org/item?id=ITA_1996__30_1_81_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 30, n° 1, 1996, pp. 81 à 86)

SEPARATING WORDS WITH MACHINES AND GROUPS (*)

by J. M. Robson [1]

Communicated by Christian Choffrut

Abstract. – *In the study of small automata which separate pairs of strings of a given length, those automata whose syntactic monoids are groups, appear to be of central importance. We show an upper bound on the number of states required for such a restricted automaton to separate two words of length N, not too much larger than the corresponding bound in the general case.*

Résumé. – *Les automates, dont le monoïde syntaxique est un groupe, sont apparemment d'une importance centrale dans l'étude des petits automates capables de distinguer deux mots de la même longueur. On montre un majorant sur le nombre d'états d'un tel automate séparant deux mots de longueur N, qui n'est pas par trop supérieur au majorant connu dans le cas d'un automate général.*

1. INTRODUCTION AND HISTORY

This paper continues the study [1], [2], [3] of the minimal finite state machine separating two arbitrary words of length N (that is accepting one but not the other), in particular the question of how many states are required in the worst case. The cases where the two words have different lengths or where the alphabet has a cardinality different from 2 were solved or reduced to the case of equal length words over a binary alphabet in [1] and so we do not study them here. We therefore take the alphabet to be $\{0, 1\}$ and we restrict our attention to deterministic machines. [3] showed an upper bound of $O(N^{2/5} log^{3/5} N)$ on the number of states required. The results of [1] imply a lower bound of $\Omega(log N)$.

In this paper we study the related problem where the transition associated with the input of a given symbol is restricted to be a permutation of the states of the machine, that is we study separation of words by groups without the extra interest added by the fact that a word over a group is generally

(*) Received July 1, 1995.

[1] LaBRI, Université Bordeaux-I, Talence Cedex, France, E-mail: robson@labri.u-bordeaux.fr

taken to allow inverses. It may be noted that the results in [1] hold equally for this restricted case. We will show an upper bound of $O(\sqrt{N})$ for the number of states required.

For the rest of the paper a "machine" means a finite automaton with alphabet $\{0,\ 1\}$ whose transition functions are both permutations of the states.

2. SEPARATION BY SMALL PERMUTATION GROUPS

2.1. Introduction

In this section we will prove the new result that two words of length N can be separated by a very simple machine with $O(\sqrt{N})$ states. The description of the set of machines is very simple. The fact that one machine of the set must separate two given words is less clear.

2.2. The Machines

We define the permutation machine $M_{x,m}$ $(x < m)$ with $2m$ states $(c,\ p)$ $(0 \le c < m,\ 0 \le p \le 1)$ and transition functions for input "0" $(c,\ p)- > ((c+1) \mod m,\ p)$ and for input "1" $(c,\ p)- > ((c+1) \mod m,\ \textbf{if } c = x \textbf{ then } 1-p \textbf{ else } p)$. This machine starts in state $(0,\ 0)$ and accepts in any state $(c,\ 1)$. Clearly c counts the number of input characters read (modulo m) and p records the parity of the number of "1" characters encountered at positions congruent to x modulo m. Thus the machine accepts precisely those words with an odd number of occurrences of "1" in positions with index congruent to x modulo m.

2.3. Statement of the Main Theorem

THEOREM: *Two words w and w' of the same length N are separated by a machine $M_{x,m}$ such that (1) $m = O(\sqrt{N})$, (2) m is square free and (3) either m and x have no common factor or $m = 2$ and $x = 0$.*

Conditions (2) and (3) are not essential but simplify the proof. Dealing with all $M_{x,m}$ subject to condition (1) would merely reduce the constant factor in the O notation. Conversely we could simplify the argument even further by restricting the machines to those with m prime, but that would give a slightly weaker result $(O(\sqrt{NlogN}))$.

The proof of this theorem will take up the remainder of section 2.

2.4. Simple Properties of the Machines

We define l as the smallest integer such that the number of machines $M_{x,m}$ satisfying conditions (2) and (3) of the statement of the theorem and having $m \leq l$ is at least N. The machine separating w and w' will be one of these machines with $m \leq l$. We write S for the set of all these machines and number them arbitrarily $S_1, ..., S_{|S|}$. We define the *result vector* of a word u as the vector $res(u)$ of length $|S|$ having its element i equal to "1" if S_i accepts u and "0" otherwise. We further define an equivalence relationship on words by saying $u \equiv_S v$ iff $res(u) = res(v)$.

LEMMA 1: *If u and v are two words with $|u| = |v|$, then $res(u \not\equiv v) = res(u) \not\equiv res(v)$ (where $\not\equiv$ is the componentwise exclusive or operation).*

Proof: Consider the element i in each of the three vectors $res(u)$, $res(v)$ and $res(u \not\equiv v)$ corresponding to a given machine $M_{x,m}$. These elements are the exclusive or of the characters of the three words (u, v and $u \not\equiv v$) which lie in positions with index congruent to x modulo m. By the associativity and commutativity of the exclusive or operator, it follows that element i of $res(u \not\equiv v)$ is given by the exclusive or of those of $res(u)$ and $res(v)$.

2.5. The Number of Equivalence Classes

LEMMA 2: *The equivalence relation $\equiv_S$ has exactly $2^{|S|}$ equivalence classes.*

Proof: We will show how to construct a word of length $l!$ for each of the $2^{|S|}$ possible result vectors. By lemma 1 it suffices to do this for each result vector which has exactly one "1" element. So consider the result vector with a single "1" element in the position corresponding to the machine $M_{x,m}$. Machine $M_{0,2}$ is a special case: its word is the one with a single "1" at position $l!$. In all other cases, since m is square free, we can write it as a product of distinct primes $m = \prod_{i=1}^{k} \pi_i$. Since m and x have no common factor, we know that $x \mod \pi_i$ is nonzero for each π_i. Write x_i for this value. Consider the word u with 2^k "1" characters, whose positions are obtained by choosing either x_i or zero for the value modulo π_i for each i $(1 \leq i \leq k)$ and zero for the value modulo every other prime less than or equal to l. (By the Chinese remainder theorem we can always choose such positions.)

We claim that $res(u)$ has the required form of a single "1" in the position corresponding to $M_{x,m}$. Certainly it does have a "1" in this position since it has one and only one "1" character in a position congruent to x modulo m.

To see that $res(u)$ has no "1" in any other position, consider an arbitrary other position corresponding to $M_{x',m'}$. There are three possibilities: (1) $m = m'$ but $x \neq x'$: every "1" character in u except the one at position x modulo m was chosen so as to have a common factor with m and so cannot affect any $M_{x',m'}$ in S ($x' \neq x$); (2) there is some prime factor π_i of m which is not a factor of m': any positions of "1"s in u congruent to x' modulo m' can be divided into two equal parts, those congruent to x_i modulo π_i and those congruent to zero; thus their number is even so that $M_{x',m'}$ does not accept u; (3) m' has some prime factor π which is not a factor of m (and m' is not equal to 2 since that case would have been included in case (2)): every position of a "1" in u has a common factor (π) with m' and so does not affect acceptance by any $M_{x',m'}$ in S.

Thus, as required, u is accepted by $M_{x,m}$ and by no other machine in S. This completes the proof of lemma 2.

2.6. Canonical Members of Equivalence Classes

Now we define w_0 to be the shortest word equivalent (under $\equiv_S$) to the empty word and not consisting entirely of "0" characters. Note that w_0 must end with a "1" character.

LEMMA 3: *Any word of the form* $0^* w_0$ *is equivalent to the empty word.*

Proof: We will show that, if v is any word equivalent to the empty word, $0v$ is also equivalent. Recall that being equivalent to the empty word means not being accepted by any of the machines in S. Consider an arbitrary such machine $M_{x,m}$. This machine will accept $0v$ iff $0v$ has an odd number of "1" characters at positions congruent to x modulo m, that is if v has an odd number of "1" characters at positions congruent to $x - 1$ modulo m. This is already ruled out by the fact that v is not accepted by $M_{x-1,m}$ except in the case that $x - 1$ has a common factor with m. We will show by induction on the number of common factors of y and m that v has an even number of occurrences of "1" at positions congruent to y modulo m. The basis of the induction is the case where this number of common factors is zero and is established by the fact that $M_{y,m}$ does not accept v. Suppose the number of common factors of y and m is $c + 1$: by reordering the prime factors $\pi_1 ... \pi_k$ of m we can write y in modular representation as $(0, 0, ...0, y_{c+2}, ..., y_k)$ with all the explicitly written y_i nonzero. The positions congruent to y modulo m can be described in terms of their modular representation as all those which agree with y on every position except possibly $c + 1$ minus those which definitely differ from y on position $c+1$ but agree with it on the other positions. That is they are those congruent to y' modulo m' minus all those congruent to some y'' modulo m where (a) $m' = m/\pi_{c+1}$, (b) y' has modular representation $(0, 0, ..., 0, y_{c+2}, ..., y_k)$ with respect to the prime factors of m' and (c) the values y'' are those whose modular representation with respect to m is obtained by replacing the 0 in position $c+1$ by some nonzero value less than π_{c+1}. But each of these sets of positions (the set for y' and each of the $\pi_{c+1} - 1$ for the various y'') is one corresponding to some machine $M_{x,m}$ where x has only c prime factors in common with m. Hence, by the inductive hypothesis, each of these sets of positions has an even number of "1" characters and therefore so too does the difference. The case where m was itself a prime does not follow this pattern exactly since the first set of positions is now the set of all positions and does not correspond to a machine $M_{x,m}$; in that case the fact that this set has an even number of "1" occurrences comes from the fact that neither $M_{0,2}$ nor $M_{1,2}$ accepts v.

LEMMA 4: *Every equivalence class of $\equiv_S$ contains a word with length $|w_0| - 1$.*

Proof: Take an arbitrary representative u of an equivalence class C. If $|u| < |w_0|$, then u can be lengthened to an equivalent word of the correct length by appending an appropriate number of "0" characters. Otherwise, if u ends with a "0", deleting this "0" gives a shorter equivalent word. If u ends with a "1", take the exclusive or of u with a word of the same length obtained by prefixing "0" characters to w_0. This will give a word equivalent to u (by lemmas 1 and 3) and of the same length but ending "0", to which we can apply the same process of deleting the final "0". Iterating this process will eventually give us a word equivalent to u and of length exactly $|w_0| - 1$.

LEMMA 5: *Every equivalence class of $\equiv_S$ contains exactly one word of length $|w_0| - 1$.*

Proof: We know each class contains at least one such word. If it contained two, their exclusive or would be equivalent to the empty word, in contradiction to the definition of w_0.

Now we can conclude from lemmas 2 and 5 that $|w_0| = 1 + |S|$ since there is a bijection between words of length $|w_0| - 1$ and equivalence classes and we know that there are $2^{|S|}$ equivalence classes. We can also conclude that any two words w and w' of the same length less than $|w_0|$ are separated by a machine in S since otherwise we could produce two equivalent words of length $|w_0| - 1$ by appending the same suffix to w and w'.

2.7. The Required Value of l

LEMMA 6: $|S| = \Omega(l^2)$.

Proof: S contains every machine $M_{x,m}$ for $0 < x < m \leq l$ except those for which m has a squared prime π^2 as a factor or both x and m have the same prime π as a factor. There are $\Omega(l^2)$ pairs x, m satisfying $0 < x < m \leq l$ of which a fraction $\leq \pi_i^{-2}$ have π_i as a squared factor of m and similarly a fraction $\leq \pi_i^{-2}$ have π_i as a common factor of m and x. Since $\sum_{\pi_i \text{ prime}} 2\pi_i^{-2} < 1$, this must leave $\Omega(l^2)$ pairs which do define machines in S.

Putting together the conclusion of 2.6 and lemma 6, we see that to separate w from w', it is sufficient to take l large enough to make $\Omega(l^2)$ greater than N. Hence $l = O(\sqrt{N})$ suffices.

3. FURTHER WORK

Since the machines $M_{x,m}$ have been shown to be surprisingly effective at economical separation of words, it is natural to ask whether some generalisation of their definition might lower the bound to $O(N^{1/3})$ or even beyond. An obvious attempt would be to define $M_{x,y,m}$ which accepts those words which have an odd number of "1"s in positions which are congruent to y modulo m and are preceded by an odd number of "1"s in positions congruent to x modulo m. These machines are certainly permutation machines with $4m$ states; whether two words of length N must be separated by one of them with $m = O(N^{1/3})$ is a completely open question.

REFERENCES

1. P. GORALCIK and V. KOUBEK, On Discerning words by automata, *13th International Colloquium on Automata Languages and Programming*, Springer (LNCS 226), 1986, pp. 116-122.
2. J. H. JOHNSON, Rational Equivalence Relations, *13th International Colloquium on Automata Languages and Programming*, Springer (LNCS 226), 1986, pp. 167-176.
3. J. M. ROBSON, Separating Strings with Small Automata, *Information Processing Letters*, 1989, *30*, pp. 209-214.