Discerning Two Words by a Minimum Size Automaton

Wiedermann, Jiří
2016

Institute of Computer Science
The Czech Academy of Sciences

Discerning Two Words by a Minimum Size Automaton

Jiří Wiedermann

Technical report No. V-1230

June 2016

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 053 520, fax: +420 286 585 789,
e-mail:wieder@cs.cas.cz

Institute of Computer Science
The Czech Academy of Sciences

Discerning Two Words by a Minimum Size Automaton[1]

Jiří Wiedermann

Technical report No. V-1230

June 2016

Abstract:

In 1986 Goralčík and Koubek proved that there is a DFA of sublinear size distinguishing between two words of length $\leq n$ (by accepting one and rejecting the other). In 1989 Robson designed a DFA of size $O(n^{2/5}(\log n)^{3/5})$ representing the best known upper bound for this problem until today. Improving this bound has been formulated as an open problem in automata theory by several authors. In this paper we definitely resolve this problem by showing a DFA of size $O(\log n)$ which is asymptotically optimal. We also characterize the class of regular languages recognized by the underlying automata.

Keywords:
Finite automaton, discerning two words, complexity

[1]This research was partially supported by the institutional fund RVO 67985807.

1 Introduction

The task of discerning two given words calls for the design of a finite automaton that is as small as possible, accepting one of the words and rejecting the other. Note the substantial difference between this task and the classical task of (formal) language recognition. In general, in the latter case, the task is to recognize a subset of an infinite set of (finite) words and to reject any word from the complement of this subset. In our case, we are given just one concrete word that is to be accepted and one concrete word that is to be rejected. We assume that except of these two words the underlying automaton is never confronted with any other word. Hence, it can be tailored, specialized to just these two words and in its discerning task it can concentrate to inspection of those parts of the words in which they differ. This solution leads to substantial savings in the number of states of the underlying automaton when compared with the straightforward design of an automaton accepting either of the two words and rejecting anything else.

The origin of this task goes back to nineteen eighties when Christian Choffrut has formulated the problem of two words separation and asked for the size of the smallest automaton solving this problem. The first answer has been given thirty years ago by Goralčík and Koubek [3] who have proved that there is an automaton of sublinear size (w.r.t. to the length of either word) for this problem. The next progress occurred in 1989 when J. M. Robson [7] has devised an automaton of size $O(n^{2/5}(\log n)^{3/5})$ representing the best known upper bound for this problem till the present time. In 2011, Demain et all in an overview paper [2] published a lower bound of form $\Omega(\log n)$. In this paper a number of special instances of the input words have been identified for which this lower bound is achievable. However, the general case for arbitrary words remained unsolved and in fact, it has been mentioned as an open problem in automata theory by several authors (cf. [7], [8], [2], [5]). J. Shallit in his recent talk at BCTS 2014 conference [11] offered 100 GBP for any nontrivial progress on this problem.

It seems that the motivation for considering this problem comes from the work of H. Johnson [4] on data compression. In the presentation accompanying the talk [2] the problem of telling two strings apart has been presented as *"the simplest computational problem you could ask to solve"*. As an additional motivation, this problem can be viewed as an inverse to a classical problem from the early days of automata theory: given two finite state automata accepting different languages, what length of word suffices to distinguish them (cf. [1])? (This problem has been routinely solved by using the classical procedures for automata minimization.)

In our opinion, the optimal solution of the word separation problem — as presented in this paper — brings an important insight. Namely, specializing the task of an automaton to distinguish between exactly two inputs leads to a dramatic decrease of its size as compared to the "obvious", first-hand solution considering only one given input to be accepted.

The structure of the paper is as follows. In Section 2 we introduce the basic notions and definitions. Especially, we introduce the notion of discerning sets and discerning automata. The next Section 3 presents the main result of this paper. First, we prove an important lemma showing that instead of studying discerning automata it is enough to investigate the properties of the discerning sets. Then we show a lower bound $\Omega(\log n)$ on the size of discerning automata and the construction of a discerning automata of that size. In Section 4 we characterize the regular language recognized by a discerning automaton. Section 5 contains the concluding remarks.

2 Preliminaries

We start by presenting a simple lemma (which has already been mentioned in [3]) that can be used for separation of words of different lengths and also for discerning the words that differ in one or two symbols. We present a complete proof of its claim in order to illustrate how to find parameter m of an automaton discerning two words of different lengths — cf. Theorem 3.4.

Lemma 2.1 *Let $0 < a < b$ be two integers. Then there exists $m \in O(\log(b-a))$ such that $a \not\equiv b \pmod{m}$.*

Proof: Let $p_1 < p_2 < \ldots < p_k$ be the first k primes, let $p_k\#$, the primorial of p_k, be the product of the first k primes. Let k be minimal such that $p_k\# > b - a$.

We prove that there exists $m \in \{p_1, p_2, \ldots, p_k\}$ such that $a \not\equiv b \pmod{m}$. By contradiction, assume that for all $m \in \{p_1, p_2, \ldots, p_k\}$, $a \equiv b \pmod{m}$. Then for all $i, 1 \leq i \leq k, p_i | b - a$, and hence $p_k\# | b - a$. But this is impossible since $p_k\# > b - a$.

As far as the size of m is concerned, we know that $2 \leq m \leq p_k$ and $p_{k-1}\# \leq b - a < p_k\#$. It is known (cf. [9], formula 316) that for any $i : p_i \geq 41$, $\log p_i\# > p_i(1 - 1/\ln p_i) > p_i/2$. Hence $\log(b-a) \geq \log p_{k-1}\# \geq p_{k-1}/2$. Now, from the Bertrand's postulate (cf. [6]) stating that for any integer $j > 1$ there is a prime number p such that $j < p < 2j$ we get that $p_{k-1} < p_k < 2p_{k-1}$. Therefore $m \leq p_k < 2p_{k-1} \leq \log(b-a)$ and therefore $m \in O(\log(b-a))$.

□

Thus, given two strings, u and v, with $|u| \neq |v|$ and $|u| < |v|$, there exits m such that $|u| \not\equiv |v| \pmod{m}$. Therefore, an automaton cycling through m states can distinguish u from v as follows. If the automaton halts (by reaching the end of a string) in the $(|u| \mod m)$-th state it accepts its input ("knowing" that it was u). However, if it halts in the $(|v| \mod m)$-th state, it rejects its input ("knowing" that it was v).

It follows that discerning two words of equal length is the harder case that we will investigate in the sequel.

First of all, consider two words of length n differing in exactly one symbol. This means that the words differ in the number of ones (or zeros, for that matter). In such a case Lemma 1 can be used again, this time discerning between the number of ones in both words rather than between the lengths of the words (as it was the case in the previous case).

Similarly, when the words differ in exactly two positions, an automaton exists discerning between these two positions using Lemma 1 again.

For the remaining cases we will consider specific, so-called *discerning automata* (also known as permutation machines in [8]). These automata have a simple structure and work as follows. A discerning automaton starts in state $(0, 0)$ and accepts in any state $(q, 1)$. Variable q counts, modulo m, the symbols in the input string read thus far while p records the parity of the number of 1's encountered at positions congruent to $i \pmod{m}$. Therefore, the machine accepts precisely those words with an odd number of occurrences of 1's in positions with index congruent to $i \pmod{m}$.

Definition 2.2 *A* discerning automaton $A_{i,m}$, *with* $0 \leq i < m$, *is a finite automaton* $A_{i,m} = (Q, \Sigma, \delta, q_0, F)$, *where the elements in the previous five-tuple are defined as follows:*

- Q *is a finite set of* $2m$ *states:* $Q = \{(q, p) | 0 \leq q < m, p \in \{0, 1\}\}$;
- $\Sigma = \{0, 1\}$ *is a finite set of input symbols called the* alphabet*;*
- $\delta : \Sigma \times Q \to Q$ *is the* transition function *defined as follows:*
  $\delta(0, (q, p)) = ((q + 1) \mod m, p)$, *and*
  $\delta(1, (q, p)) = ((q + 1) \mod m, \textbf{if } q = i \textbf{ then } 1 - p \textbf{ else } p)$.
- q_0 *is the* start state, *defined as* $q_0 = (0, 0) \in Q$, *and*
- F *is the set of* accepting states, $F = \{(q, 1) | (q, 1) \in Q\}$.

Obviously, any discerning automaton $A_{i,m}$ has $O(m)$ states, i.e., is of size $O(m)$.

The next notion we will need is that of the discerning sets. Consider two words, $u, v \in \{0, 1\}^n$. By u_i we denote the i-th symbol of atring u, and similarly for string v.

Definition 2.3 *Let* $u, v \in \{0, 1\}^n$ *be two different words. The set* $D(u, v) = \{i | 1 \leq i \leq n, u_i \neq v_i\}$ *is called the* discerning set *for words* u *and* v.

Definition 2.4 *We say that an automaton* discerns *two given strings* u *and* v, *respectively, if and only if it accepts one and rejects the other. That is, it either accepts* u *and rejects* v, *or accepts* v *and rejects* u.

3 The main result

The key to the efficient solution of the discerning problem is given in the next theorem which reveals the relation between discerning sets and discerning automata. It shows that instead of studying discerning automata one can investigate the properties of the discerning sets.

Theorem 3.1 *Let $u, v \in \{0,1\}^n$ be two different words, let $D(u,v)$ be their discerning set. Then there exist m and i, with $0 \leq i < m \leq n$ such that the following two statements are equivalent:*

(i) the residue class $R_{D(u,v)}(i,m) = \{x | x \in D(u,v) \wedge x \equiv i \pmod m\}$ has an odd number of elements;

(ii) the discerning automaton $A_{i,m}$ discerns u and v.

Proof: In order to get insight into the claim of the theorem consider the arrangement of elements of set $\{1, 2, \ldots, n\}$ as in the Table 1.

Residua of $j \pmod m$, **for** $j = 1, 2 \ldots, n, n = im + r$						
0	**1**	**2**	...	**r**	...	**m-1**
-	1	2	...	r	...	m-1
m	m+1	m+2	...	m+r	...	2m-1
2m	2m+1	...				3m-1
	...					
(i-1)m	...					im-1
im	im+1	im+2	...	im+r	-	-

Table 3.1: The diagram of the residue classes

In this diagram, the indices of the symbols of u and v, respectively, will occupy those positions that correspond to positions of 1's in the respective words. Let us color the entries representing word u by green color and the entries representing v by blue color. It may happen that the same entry in the diagram belongs to both u and v an thus it will be colored both green and blue. The resulting color of such entry will be yellow color.

Now, it is obvious that the green and blue entries in the resulting diagram correspond exactly to the discerning set $D(u,v)$. The entries corresponding to individual residue classes (mod m) find themselves in the columns of this diagram. The theorem claims that there is m and i such that in the column $R_{D(u,v)}(i,m)$ there is an odd number of blue and green elements (plus possibly some number of yellow elements). Therefore, to discern the two words it is enough for the automaton to determine the parity of the set of green and yellow (or blue and yellow) elements in the respective residue class. (Consideration of the yellow elements will not change the disparity between the green and blue elements since the same number of yellow elements is considered in both cases.)

Now to the proof itself.

Assume (i) holds. Then $R_{D(u,v)}(i,m)$ is the set of all positions of those 1's in both u and v that are congruent to $i \pmod m$, with $u_i \neq v_i$. Since this set contains an odd number of elements, one of the strings must contain an odd number of such positions (colored green, say) while the other one contains an even number of such positions (colored blue). However, note that in addition to the 1's whose counterpart in the other string are 0's $A_{i,m}$ also counts (mod m) the 1's in both strings whose counterparts are again 1's (such entries are yellow in the previous diagram). Fortunately, the number of yellow elements is the same for both strings and therefore their participation in counting cannot change the disparity between green and blue elements. Therefore, $A_{i,m}$ discerns the two strings. Note that which of the two strings gets accepted and which gets rejected by $A_{i,m}$ depends only on the parity of the elements of both u and v in $R_{D(u,v)}(i,m)$.

If (ii) holds, then if $A_{i,m}$ discerns u and v, then there must be a disparity between the number of 1's in both u and v at positions congruent to $i \pmod m$. Consider the sets of positions of all 1's congruent to $i \pmod m$ in both strings. The parity of these sets must differ, since one of them was

accepted by $A_{i,m}$ while the other was rejected. After deleting the same positions (i.e., yellow elements) from both sets we get exactly the set $D(u,v)$ (of green and blue elements) whose cardinality must be odd, since the same number of positions (of yellow elements) has been deleted from both original sets which were of different parity. Therefore $R_{D(u,v)}(i,m)$ has an odd number of elements.

□

In the previous theorem we showed that instead of investigating discerning automata pertinent to two given words of equal length one can investigate the parity of residue classes of the corresponding discerning set. If there is a residue class of odd cardinality, then there is a discerning automaton discerning any two strings with the given discerning set. This result alone suffices for the proof of a lower bound on the size of discerning automata:

Theorem 3.2 *In the worst case, a discerning automaton discerning any two words of length n must be of size at least $\Omega(\log n)$.*

Proof: For any $0 \leq i < j \leq m \leq n$ consider the sequence $\alpha(m)$ of length $N(m)$ of automata $A_{i,j}$ sorted lexicographically w.r.t. parameters i, j. We are looking for a lower bound on $N(m)$ such that sequence $\alpha(m)$ will contain discerning automata separating any two strings $u, v \in \{0,1\}^n$. When thinking about the elements of $\alpha(m)$ it is quite tempting to consider automata as defined in Definition 2.2 that accept any string in $\{0,1\}^n$ rather than those that discern any sets of pairs of strings in $(\{0,1\}^n)^2$ according to some criteria. Note that in the former case we would get a lower bound on the size of discerning automata accepting any particular string of length n whereas in the latter case we get a bound for automata discerning any two strings of length n (without distinguishing which of the two strings gets accepted) which is what we are after.

Having the latter idea in mind, let us define $R_{i,j}$ as the class of all strings of length n whose discerning sets contain residue class $i \pmod j$ of odd cardinality: $R_{i,j} = \{(u,v) | u, v \in \{0,1\}^n and R_{D(u,v)}(i,j)$ *has an odd number of elements*$\}$ for all $0 \leq i < j \leq n$.

We show two important facts about the sets $R_{i,j}$'s: (i) any pair of distinct strings of length n will find itself in some $R_{i,j}$, and (ii), no two $R_{i,j}$'s are equal.

To see (i), consider the union $\cup_{0 \leq i < j \leq n} R_{i,j}$. We show that this union contains all strings — it is equal to $\{\{0,1\}^n\}^2$. The reason is that for any u, v, and any i, any non-empty set $R_{D(u,v)}(i,n)$ is of odd cardinality, containing exactly one element, and therefore any pair (u,v) will be a member of some $R_{i,n}$. Obviously, some pairs will find themselves also in other $R_{i,j}$'s.

As far as the case (ii) is concerned, we show that for any $0 \leq i < j \leq n$ and $0 \leq p < q \leq n$, $(i,j) \neq (p,q)$ the classes $R_{i,j}$ and $R_{p,q}$ are distinct: $R_{i,j} \neq R_{p,q}$. To that end we construct (u,v) such that $(u,v) \in R_{i,j}$ but $(u,v) \notin R_{p,q}$.

Let u be a string of length n such that $u_r = 1$, where r is the first index congruent to $i \pmod j$ and $r \neq p \mod q$; all other symbols in v are zeros. Let u be the string consisting of n zeros. Then $R_{D(u,v)}(i,j) = \{r\}$ and $R_{D(u,v)}(p,q) = \emptyset$. Therefore $(u,v) \in R_{i,j}$ and at the same time, $(u,v) \notin R_{p,q}$.

Now, to each $R_{i,j}$ assign a Boolean vector $W(i,j)$ of length $N(m)$ whose k-th entry is set to 1 if and only if the k-th automaton in sequence $\alpha(m)$ discerns some $(u,v) \in R_{i,j}$. Then, w.r.t. the properties of $R_{i,j}$'s for any pair of indices $(i,j) \neq (p,q)$ we have $W(i,j) \neq W(p,q)$.

The number of different Boolean vectors $W(i,j)$ of length $N(m)$ is upper-bounded by $2^{N(m)}$. This number cannot be smaller than the number of $R_{i,j}$'s, for $0 \leq i < j \leq n$. The latter number is of order $\Omega(n^2)$ and therefore we have $2^{N(m)} = \Omega(n^2)$, or $N(m) = \Omega(\log n)$. Consequently, the biggest discerning automaton must have at least $\Omega(\log n)$ states. □

Now we will establish an upper bound on the size of discerning automata. To that end, in what follows we will consider arbitrary discerning sets $S = D(u,v)$ for arbitrary strings u, v of equal length n.

It is clear that in the case of discerning sets of odd cardinality there must always be a residue class of odd cardinality:

Theorem 3.3 *Let $S \subseteq \{1, 2, \ldots, n\}$ be the set of odd cardinality, with $n \geq |S| \geq 3$. Then for $m = 2$ the cardinality of either $R_S(0,m)$ or $R_S(1,m)$ is odd.*

Proof: This is a simple consequence of the fact that a set with an odd number of elements can be split into two disjoint sets (of odd and even numbers, respectively) one of which must contain an odd number of elements.

□

Hence we are left with one so-far unsolved discerning problem — namely the case of discerning sets of even cardinality.

The following theorem handles this case for even word lengths:

Theorem 3.4 *Let $S \subseteq \{1, 2, \ldots, n\}$ be the set with $n \geq |S| \geq 2$, n even. Then there exist $m \in O(\log n)$ and number $i \in \{0, 1, \ldots, m-1\}$ such that the cardinality of $R_S(i, m)$ is odd.*

Proof: By contradiction, assume that under the assumption of the theorem it holds for all $m \in O(\log n)$ and for all $i \in \{0, 1, \ldots, m-1\}$, $|R_S(i, m)|$ is even.

We will distinguish two cases: $S = \{1, 2, \ldots, n\}$ and $S \subset \{1, 2, \ldots, n\}$, respectively.

In the former case, choose any m which does not divide n and for this m consider the diagram of the residue classes of set S (cf. Table 1). Clearly, in this diagram residue classes with both even and odd number of elements exist — a contradiction with the assumption that for all $i \in \{0, 1, \ldots, m-1\}$, $|R_S(i, m)|$ is even.

In the latter case, when $S \subset \{1, 2, \ldots, n\}$, consider the set S' which is the complement of S : $S' = \{1, 2, \ldots, n\} - S$. Since n and $|S|$ are both even, also the cardinality of S' is even, and therefore, according to the contradictory assumption, for all $m \in O(\log n)$ and for all $i \in \{0, 1, \ldots, m-1\}$, $|R_{S'}(i, m)|$ is even. Now observe that for any i, $R_S(i, m) \cup R_{S'}(i, m) = R_{\{1,2,\ldots,n\}}(i, m)$ and the residue classes $R_{\{1,2,\ldots,n\}}(i, m)$ have an even number of elements for all m and i. But from the former case we know that there is an m such that not all residue classes $R_{\{1,2,\ldots,n\}}(i, m)$ have the same parity — again, a contradiction.

Thus, under the assumption of our theorem it holds that there exist $m \in O(\log n)$ and number $i \in \{0, 1, \ldots, m-1\}$ such that the cardinality of $R_S(i, m)$ is odd. In fact, in our case we can assume that there exist two residue classes with an odd number of elements since in set S with an even number of elements there cannot exist exactly one residue class modulo m with an odd number of elements.

In order to pin down the size of m, let us assume that for some m there are two residue classes with an odd number of elements, $R_S(i, m)$ and $R_S(j, m)$, respectively. Then for any $x \in R_S(i, m)$ and $y \in R_S(j, m)$ it holds $x \not\equiv y \pmod{m}$.

Let $p_1 < p_2 < \ldots < p_k$ be the first k primes such that $p_{k-1}\# \leq n < p_k\#$. We prove that there exists $m \in \{2, 3, 5, \ldots, p_k\}$ such that for any $x \in R_S(i, m)$ and $y \in R_S(j, m)$ it holds $x \not\equiv y \pmod{m}$. By contradiction, assume that for all $m \in \{2, 3, 5, \ldots, p_k\}$ there exist x and y such that $x \equiv y \pmod{m}$. Then for all m, $m|(x - y)$ and therefore, for all m, $p_k\#|(x - y)$. But the latter is impossible since $p_k\# > n > |(x - y)|$. The size of m, i.e., $m \in O(\log n)$, is then determined similarly as in Lemma 1.

□

Note that the upper bound for m from the latter theorem matches the lower bound from Theorem 3.2. The best previously known upper bound concerning the size of discerning automata of the type considered here was $O(\sqrt{n})$ [7].

For discerning sets of even cardinality that are subsets of set $\{1, 2, \ldots, n\}$ of even cardinality the previous theorem guarantees the existence of a discerning automaton of size $O(\log n)$. If n is odd then the discerning automaton for strings u and v, respectively, of the same length n, can be constructed as follows.

If $u_1 \neq v_1$ then the discerning automaton can decide by merely inspecting the first symbol of the input. However, if $u_1 = v_1$ then the discerning automaton simply skips the first symbol of the input and proceeds as on the input of length $(n - 1)$, i.e., as on an input of even length.

Using the claims of Lemma 1, Theorem 3.3 and 3.4, from Theorem 3.1 and remarks concerning short words and words of odd length we get the following final result:

Theorem 3.5 *Let $u, v \in \{0, 1\}^*$ be two different words. Then there exists a finite automaton of size $O(\log n)$, with $n = \max\{|u|, |v|\}$ that discerns u and v.*

The latter result is optimal since a lower bound of order $\Omega(\log n)$ on the size of any DFA discerning two strings is known [2].

4 The power of discerning automata

It appears that, in general, a discerning automaton constructed for two different words $u, v \in \{0,1\}^n$ is able to set apart more than these two words. Namely, from the proof of Theorem 3.1 one can see that the activity of such an automaton only depends on the properties of the discerning set $D(u,v)$. The following claim, characterizing the set of discerning words separated by automaton $A_{i,m}$ is in fact a consequence of Theorem 3.1 and has been used in the proof of Theorem 3.2:

Corollary 4.1 *Any discerning automaton $A_{i,m}$ will discern all pairs $(u,v) \in \{\{0,1\}^n\}^2$ for which $R_{D(u,v)}(i,m)$ has an odd number of elements.*

Clearly, since any discerning automaton is a finite automaton it recognizes a regular language. In the next theorem we characterize this language by means of regular expressions.

Theorem 4.2 *Let $m \geq 2$ and $0 \leq i < m$, let $A_{i,m}$ be a discerning automaton. Then the language recognized by $A_{i,m}$ is described by the regular expression of form $a\{b^*cb^*c\}^*b^*cb^*d$, with*

- $a = \{0|1\}^{i-1}$;
- $b = 0\{0|1\}^{m-1}$;
- $c = 1\{0|1\}^{m-1}$;
- $d = \varepsilon|\{0|1\}|\{0,1\}^2|\ldots|\{0,1\}^{m-1}$.

Proof: From the informal description of the discerning automaton (to be found immediately before its definition) one can see that starting with the position i the automaton checks its input at equidistant points (or "check-points") whose positions are congruent to $i \pmod m$ and accepts if and only if an odd number of ones can be found at these check-points.

We construct a regular expression describing strings accepted by $A_{i,m}$. In order to get an idea about the construction we first solve a simpler problem: constructing a regular expression describing all strings over $\{0,1\}^*$ containing an odd number of ones. The respective expression looks, e.g., as follows: $\{0^*10^*1\}^*0^*10^*$. That is, all the eligible strings start with a sequence of zeros, then there is a part containing zeros in between the two ones. All this can be iterated giving rise to an even number of ones in that part. This is followed by zeros, and finally there is a one followed by zeros. The latter symbol one ensures an odd number of ones in the string.

To return to our case, the regular expression we are after has a similar structure as the previous expression, but more restrictions have to be considered. First, there is a starting sequence consisting of an arbitrary string of length $i-1$. Second, at the check-points an odd number of ones must be located, possibly interspersed with an arbitrary number of zeros at the remaining check-points. Third, the substrings between the checkpoints can be arbitrary strings of length $m-2$.

One can verify that the final expression can be constructed from four regular subexpressions a, b, c, and d, and will have the form $a\{b^*cb^*c\}^*b^*cb^*d$. The respective regular expressions a, b, c, and d, are defined as follows:

- $a = \{0|1\}^{i-1}$; this is a "displacement" — the prefix of a string of length $i-1$ where the check-points start;
- $b = 0\{0|1\}^{m-1}$; a substring starting with zero at a check-point, followed by an arbitrary string of length $m-1$.
- $c = 1\{0|1\}^{m-1}$; a substring starting with one at a check-point, followed by an arbitrary string of length $m-1$.
- $d = \varepsilon|\{0|1\}|\{0,1\}^2|\ldots|\{0,1\}^{m-1}$; an arbitrary closing string of length at most $m-1$.

Note that the two occurrences of c in the substring $e = \{b^*cb^*cb^*\}$ result in an even number of ones at the check-points in the iterated string e^* and the single occurrence of c afterwards, at a check-point, results in an odd number of ones at check-points in the entire string.

□

Denoting as $\mathcal{L}_{i,m}$ the language recognized by automaton $A_{i,m}$, we see that, in fact, the automaton discerns any pair of words (u, v), with $u \in \mathcal{L}_{i,m}$ and $v \notin \mathcal{L}_{i,m}$.

5 Conclusions

In their recent post [5], R.J. Lipton and K.W. Regan speculated on the length of proofs of open problems. Among the problems they termed as *"the problems ripe for short solutions"* they have also mentioned the problem of word separation investigated in this paper.

The previous efforts to solve this problem (cf. [3], [7] and [8]) have come with cumbersome solutions described on a quite a number of pages, using nonintuitive, hard to understand approaches resulting into automata of size $O(n^\varepsilon)$, for some $0 < \varepsilon < 1$. This is to be compared with our approach bringing an exponential improvement of the size of the underlying automata, leading thus to an optimal solution. In order to achieve such result only elementary properties of natural numbers have been used. The main "breakthrough", if we can term it like this, bringing the necessary and, as it appeared later, sufficient insight into the problem, was the discovery of the relation between the parity of the residue classes of discerning sets and discerning automata, as described in Theorem 3.1. This has eventually lead to the (constructive) proof of existence of discerning automata of minimal size.

Bibliography

[1] Currie, H. Petersen, J. M. Robson, and J. Shallit: Separating words with small grammars. J. Automata, Languages, and Combinatorics, 4:101-110, 1999.

[2] Demaine, E., Eisenstat, S., Shallit, J, and Wilson, D.: Remarks on Separating Words. In: Proceedings of the 13th International Workshop on Descriptional Complexity of Formal Systems (DCFS 2011), Lecture Notes in Computer Science, volume 6808, Giessen, Germany, July 25–27, 2011, p. 147–157, 2011.

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

[4] Johnson, J. H.: Rational equivalence relations. Theoretical Computer Science, 47: 39-60, 1986

[5] Lipton, R. J., Regan, K. W.: Open Problems That Might Be Easy. In: Gödel Lost Letter and P=NP, September 3, 2015, https://rjlipton.wordpress.com/

[6] Ramanujan, S.: A Proof of Bertrand's Postulate. J. Indian Math. Soc. 11, 181-182, 1919

[7] Robson, J. M.: Separating strings with small automata. Inform. Process. Lett., 30:209-14, 1989.

[8] Robson, J. M.: Separating words with machines and groups. RAIRO Inform. Theor. App., 30:81-86, 1996.

[9] Rosser, J. B., and Schoenfeld, L.: Approximate formulas for some functions of prime numbers, Illinois J. Math. 6 (1962), 64-94.

[10] Shallit, J.: A Second Course in Formal Languages and Automata Theory. Cambridge University Press, 2009.

[11] Shallit, L.: Open problems in automata theory: an idiosyncratic view, LMS Keynote Address in Discrete Mathematics, BCTCS 2014, April 10 2014, Loughborough, England