View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Elsevier - Publisher Connector

Theoretical Computer Science 85 (1991) 135–153
Elsevier

Multihead one-way finite automata*

Mirosław Kutyłowski
Institute of Computer Science, University of Wrocław, Przesmyckiego 20, 51-151 Wrocław, Poland

Communicated by A. Meyer
Received May 1988
Revised April 1989

Abstract

Kutyłowski, M., Multihead one-way finite automata, Theoretical Computer Science 85 (1991) 135–153.

We consider one-way non-sensing multihead finite automata. Let

$$P_m = \{1^{a_1} * 1^{a_2} * \cdots * 1^{a_m} \# 1^{a_m} * 1^{a_{m-1}} * \cdots * 1^{a_1}\colon a_1, \ldots, a_m \in \mathbb{N}\}.$$

We show that no k-head automaton can recognize the language P_m if $m > \frac{1}{2}k^3$. It partially confirms the conjecture of Rosenberg. It shows that the languages P_m and the languages L_m, where

$$L_m = \{w_1 * w_2 * \cdots * w_m \# w_m * w_{m-1} * \cdots * w_1\colon w_1, \ldots, w_m \in \{0, 1\}^*\},$$

are of similar complexity for one-way multihead finite automata. We present a technique which can be used in some cases to estimate computational complexity of languages with respect to multihead automata.

1. Introduction

Multihead finite automata have been introduced in the early sixties by Piatkowski [11]. Obviously, multihead one-way finite automata (1-MFA) can recognize much more than regular languages and many authors have tried to characterize this class of languages. This problem received no satisfactory answer. For instance, the well-known Pattern Matching Problem (see [4]) is still open. It concerns the language

$$L_{sm} = \{w \# ywz\colon w, y, z \in \{0, 1\}^*\}.$$

* This research received support from the Institute of Informatics, University of Warsaw under program RP. I.09. The final version of the paper was prepared under sponsorship of the Alexander von Humboldt-Stiftung during author's stay at Technische Hochschule Darmstadt, Germany.

0304-3975/91/$03.50 © 1991—Elsevier Science Publishers B.V.

It is unknown whether L_{sm} can be recognized by some 1-MFA. There are some partial answers [9, 2], but it seems that we are still far from solving the problem. From the very beginning the problem of hierarchy with respect to the number of heads was considered for multihead automata. After many efforts [12–15, 3, 7] it was finally proved by Yao and Rivest [16] that "$k+1$ heads are better than k heads". Their method was used afterwards by Hromkovič [5, 6] to show that the class of languages recognized by k-head 1-MFA is not closed under many simple operations.

It turns out that the language $\{w \# w^R\colon w \in \{0, 1\}^*\}$ (where w^R stands for word w written in the reverse order) cannot be recognized by any 1-MFA. However, various simpler versions of this language have been used to show several results about multihead automata. Consider e.g. the language

$$L_m = \{w_1 * w_2 * \cdots * w_m \# w_m * w_{m-1} * \cdots * w_1\colon \forall j \leqslant m \; w_j \in \{0, 1\}^*\}.$$

Intuitively, this language is difficult for 1-MFA for the following reason. If automaton M accepting language L_m works on input $w_1 * w_2 * \cdots * w_m \# v_m * v_{m-1} * \cdots * v_1$, then it has to check that for each j the blocks w_j and v_j are equal. A straightforward method to verify this is to put two heads at the beginnings of blocks w_j and v_j and then to move these heads simultaneously through words w_j and v_j checking if the corresponding symbols are equal. Note that each pair of heads can be used only once. Indeed, after checking w_j and v_j one head has only blocks $v_{j-1}, v_{j-2}, \ldots, v_1$ left to read, while the other one has only blocks $w_{j+1}, w_{j+2}, \ldots, w_m$ on its right side before the symbol $\#$. This old idea was used by Yao and Rivest [16] to show that k-head one-way automata can recognize languages L_m only for $m \leqslant \binom{k}{2}$. Their hierarchy theorem was just a simple corollary of this fact. To carry out the proof they had to use the fact that for a fixed n there are about 2^n words of length n in L_m, even if we fix the length of blocks w_j and v_j. In this paper we try to answer if this argument must really be used. We consider relatively simple languages P_m, where

$$P_m = \{1^{a_1} * 1^{a_2} * \cdots * 1^{a_m} \# 1^{a_m} * \cdots * 1^{a_1}\colon a_1, \ldots, a_m \in \mathbb{N}\}.$$

Each block in P_m is a block of 1's, so only its length must be checked against the length of the corresponding block. The proof used for languages L_m does not work for languages P_m for the reason that there are relatively few words in P_m of a given length. The above languages were considered already by Rosenberg [13] in his attempt to prove the hierarchy theorem for 1-MFA.

In this paper we consider only non-sensing automata (a sensing multihead automaton can determine when coincidence of its heads occurs, non-sensing automata cannot detect whether their heads stay at the same place). We consider the following problem.

Problem 1.1. *Given $k \in \mathbb{N}$. Find the maximal number m such that language P_m can be recognized by some k-head (non-sensing) 1-MFA.*

We prove in Theorem 2.2 that for $m > k^3/2$ there is no k-head 1-MFA recognizing language P_m. On the other hand (see [16]), k-head automata are capable of recognizing languages P_m for $m \leqslant (k^2 - k)/2$. We also prove (Theorem 3.1) that for each k there is a k-head automaton which accepts a language $P' \subseteq P_m$, where $m \approx k^3/24$. The language P' contains language $P_{m,c}$ which is a fairly large subset of P_m, namely,

$$P_{m,c} = \{1^{a_1} * 1^{a_2} * \cdots * 1^{a_m} * 1^{a_m} * \cdots * 1^{a_1} \colon \forall k, l \leqslant m \ \ a_k \leqslant c \cdot a_l\}.$$

Anyway, the first-mentioned result shows that for non-sensing automata complexities of P_m and L_m are similar. The aim of this paper is not merely to give an answer to Problem 1.1, which is of rather technical importance. We have in mind a more general problem.

Problem 1.2. *Let L be a language of the form*

$$L = \{1^{f_1(n)} * 1^{f_2(n)} * \cdots * 1^{f_m(n)} \colon n \in \mathbb{N}\},$$

where $f_1, \ldots, f_m$ are some functions over $\mathbb{N}$. Find a minimal number k such that language L can be recognized by a k-head (non-sensing) 1-MFA or show that L cannot be recognized by such a device.

We do not know a complete answer to Problem 1.2. However, the methods used to solve Problem 1.1 can be easily adopted to provide a technical framework allowing to give answers to many subcases of Problem 1.2.

At this moment we have to mention the results obtained by Chrobak [1]. He considered very simple languages, namely,

$$C_n = \{1^x 2^{ix} \colon i, x \in \mathbb{N},\ 1 \leqslant i \leqslant n\},$$

containing only two blocks of different symbols. He proved that for each k there is an n such that C_n can be recognized by some k-head 1-MFA, but C_{n+1} requires already $k+1$ heads. His proof was based on some geometric interpretation of the behaviour of 1-MFA. We follow this idea of such an interpretation. Also by applying some methods of this paper it is possible to determine how many heads are necessary to recognize language C_n [8].

2. The upper bound

In this section we present our main result about recognizing languages P_m. From now on, by automata we mean multihead deterministic non-sensing one-way finite automata (1-MFA). For the sake of completeness we recall their definition taken from [7].

Definition 2.1. A one-way k-head deterministic non-sensing automaton is a device $M = \langle k, K, \Sigma, \delta, q_0, \$, F \rangle$, where $k \geqslant 1$ is the number of heads, K, Σ and F are finite sets of states, input symbols and accepting states ($F \subseteq K$), respectively, q_0 is the initial state, \$ (not in Σ) is the right endmarker for the inputs, and δ is a mapping from $K \times (\Sigma \cup \{\$\})^k$ into $K \times \{0, 1\}^k$. An input to M is a string $a_1 a_2 \ldots a_n$ of symbols in Σ delimited on the right end by the symbol \$. We can think of $a_1 a_2 \ldots a_n\$$ as written on the tape (with each symbol occupying one tape square) and the heads moving left to right on the tape. One execution step of M is described as follows. Let M be in state q with heads $H_1, \ldots, H_k$ scanning symbols $b_1, \ldots, b_k$ (in $\Sigma \cup \{\$\}$). Suppose $\delta(q, b_1, \ldots, b_k) = (p, d_1, \ldots, d_k)$. Then for each $j \leqslant k$, automaton M moves head H_j exactly d_j squares to the right and enters state p (we assume that no head scanning \$ can move further to the right). We say that a word $a_1 a_2 \ldots a_n$ is accepted or recognized by M if when M started on $a_1 a_2 \ldots a_n\$$ in state q_0 with all heads placed at a_1, after some number of steps it reaches an accepting state. Language L is recognized by automaton M if for every word $\boldsymbol{x}$, $\boldsymbol{x} \in L$ if and only if, $\boldsymbol{x}$ is accepted by M.

Theorem 2.2. *Suppose $m > k^3/2$. Then language P_m cannot be recognized by any k-head (non-sensing) 1-MFA.*

The rest of this section is devoted to a proof of this theorem. Assume that M is a k-head automaton recognizing P_m. Consider input words of the form $\boldsymbol{x} = 1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}}$. The subwords 1^{a_1}, 1^{a_2}, $\ldots$, $1^{a_{2m}}$ are called blocks of $\boldsymbol{x}$, so $\boldsymbol{x}$ is built from $2m$ such blocks. The lth block ($l \leqslant 2m$) of $\boldsymbol{x}$ is denoted by B_l. Also the right endmarker \$ forms one additional block. Let the reading heads of M be called H_1, $H_2, \ldots, H_k$.

Consider the computation on $\boldsymbol{x}$ performed by M. It can be divided into several stages with each stage terminating when some head crosses a boundary between two adjacent blocks of $\boldsymbol{x}$. There are k heads, so after at most $2mk$ stages each computation must terminate. Without loss of generality we may assume that M can reach a final state only when it enters a new stage of computation.

Now we shall construct C_M, a computation tree of M with so-called distance functions and case conditions giving a full description of each possible computation performed by M. C_M is a finite tree of height at most $2mk$. For a node a of C_M let $\mathrm{ht}(a)$ denote the height of a, i.e. $\mathrm{ht}(a) = l$ if a is the lth node on the path leading from the root of C_M to node a. Each node of C_M corresponds to the beginning of some stage of computation for some class of inputs. The root of C_M corresponds to the beginning of the first stage (which is the same for all inputs). Its successors correspond to the second stages and so on: a node a of C_M corresponds to the stage $\mathrm{ht}(a)$. With each node a of C_M we associate

(i) a state of M denoted by $\mathrm{st}(a)$,
(ii) distance functions $f_{a,n,l}$ for $n \leqslant k$, $l \leqslant 2m+1$,
(iii) case conditions.

Each case condition is an expression of the form

$$h(x_1, x_2, \ldots, x_{2m}) = 0 \quad \text{or} \quad h(x_1, x_2, \ldots, x_{2m}) > 0,$$

where the variables $x_1, x_2, \ldots, x_{2m}$ stand for the lengths of blocks B_1, B_2, $\ldots$, B_{2m}. We shall set $f_{a,n,l}(x_1, x_2, \ldots, x_{2m})$ to be the distance between head H_n and the end of the lth block of $1^{x_1} * \cdots * 1^{x_{2m}}$ at the beginning of stage $\mathrm{ht}(a)$, if H_n lies in this block. If H_n does not lie in this block, then $f_{a,n,l}$ is undefined.

Let $p(a)$ denote the path connecting node a with the root of C_M. Define X_a to be the set of all words x, $x = 1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}}$, such that for each case condition $g(x_1, \ldots, x_{2m})$ associated with a node lying on path $p(a)$, $g(a_1, a_2, \ldots, a_{2m})$ holds. The key point of the construction of C_M are the following properties.

Claim 2.3. (i) *Suppose a is a node of C_M. Then for every $x \in X_a$, $x = 1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}}$, at the beginning of the stage corresponding to node a, automaton M is in state* st(a) *and for each $n \leqslant k$ the distance functions $f_{a,n,l}(a_1, a_2, \ldots, a_{2m})$ describe correctly the position of head H_n.*

(ii) *Each distance function is a "simple" expression (for a definition see below) and each case condition is of the form*

$$\xi(x_1, \ldots, x_{2m}) = 0 \quad \textit{or} \quad \xi(x_1, \ldots, x_{2m}) < c,$$

where ξ is a simple expression.

Before the definition recall that $r(d, c)$ is the remainder on dividing d by c.

Definition 2.4. (i) Remainder functions are defined inductively as follows. $R(x_1, \ldots, x_{2m})$ is a remainder function if

$$R(x_1, \ldots, x_{2m}) = c \cdot r\left(\sum_{j \leqslant n} \xi_j, e\right),$$

where $c \in \mathbb{Z}$, $n, e \in \mathbb{N}$ and each ξ_j is a remainder function, an integer or an integer multiplied by some x_i, $i \leqslant 2m$.

(ii) We say that $\xi(x_1, \ldots, x_{2m})$ is a simple expression if it takes the form

$$\xi(x_1, \ldots, x_{2m}) = \left(\sum_{i \leqslant n} \xi_i(x_1, \ldots, x_{2m})\right) \Big/ d,$$

where $n, d \in \mathbb{N}$ and for each $i \leqslant n$ expression ξ_i takes one of the possible forms: $c \cdot x_j$, c or $R(x_1, \ldots, x_{2m})$, where $c \in \mathbb{Z}$, $j \leqslant 2m$ and R is a remainder function.

To define C_M we construct inductively a finite sequence of trees $T_0 \subseteq T_1 \subseteq T_2 \subseteq \cdots$ The last tree in this sequence is C_M. Each T_i is an initial subtree of C_M, i.e. if $a \in C_M$ and $a \in T_i$, then $p(a) \subseteq T_i$. Therefore, to prove Claim 2.3 for C_M it suffices to show it for each T_i, i.e. with C_M replaced by T_i.

T_0 is a tree consisting only of r, the root of C_M. Clearly, st(r) is the initial state of M, the set of the case conditions associated with r is empty, the distance functions are defined as follows.

$$f_{r,n,1}(x_1, \ldots, x_{2m}) = x_1 \quad \text{for } n \leqslant k,$$
$$f_{r,n,j}(x_1, \ldots, x_{2m}) \quad \text{is undefined for } j > 1.$$

Now we assume that we have constructed T_i ($i \geq 1$). If for every leaf a of T_i the state st(a) is final, then the construction of C_M is finished and $C_M = T_i$. Otherwise, we single out a leaf a such that st(a) is not a final state. Tree T_{i+1} will be constructed by adding some number of successors to node a. Throughout the construction we shall consider exclusively input words from X_a. Let $n = \text{ht}(a)$. If $x \in X_a$, $x = 1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}}$, then M enters stage n of computation over x in state st(a) with the head positions determined by values $f_{a,t,j}(a_1, a_2, \ldots, a_{2m})$ for $t \leq k$, $j \leq 2m$. We do not know which head first reaches a boundary between blocks at the end of stage n. There are many possibilities, each of them giving rise to a different group of successors of a. So assume that S is a set of heads and for each j, head H_j crosses a boundary between blocks at the end of stage n iff $H_j \in S$. We shall describe all successors of node a corresponding to this situation. Let $H_t \in S$; head H_t crosses the boundary between blocks B_l and B_{l+1} at the end of stage n. Hence, during this stage H_t makes $f_{a,t,l} = f_{a,t,l}(a_1, \ldots, a_{2m})$ moves. Note that since each block contains only 1's (except the first symbol and the block \$), after some initial p_0 steps the behaviour of M in stage n becomes periodic. Of course, $f_{a,t,l}$ can be too small to reach the first period cycle. For each $p \leq p_0$ there is a successor of a in T_{i+1} describing the situation when stage n terminates after exactly p machine steps. So assume that machine M makes p steps ($p \leq p_0$) and stage n terminates. We describe the corresponding node a' of T_{i+1} added to T_i. For each head H_s of M let e_s be the number of moves made by H_s during those p machine steps in stage n. Let e'_s be the number of moves made by H_s during $p-1$ such steps. The distance functions of a' are defined as follows.

For $s \notin S$

$$\begin{aligned} &f_{a',s,j} = f_{a,s,j} - e_s && \text{if } f_{a,s,j} \text{ is defined,} \\ &f_{a',s,j} && \text{is undefined otherwise.} \end{aligned}$$

For $s \in S$ function $f_{a',s,j+1}$ is defined iff $f_{a,s,j}$ is defined and $f_{a',s,j+1} = x_{j+1}$.
The case conditions associated with a' are defined as follows:

$$\begin{aligned} &f_{a,s,j} - e_s > 0 && && \text{for } s \notin S, \\ &f_{a,s,j} - e_s = 0 && \text{and} \quad f_{a,s,j} - e'_s > 0 && \text{for } s \in S. \end{aligned}$$

In the above conditions j is chosen such that $f_{a,s,j}$ is defined. Clearly, node a' satisfies Claim 2.3(i). For Claim 2.3(ii) note that a sum of a simple expression and a constant is a simple expression.

Now we have to consider the case when $f_{a,t,l}$ is big enough to reach the periodic behaviour during stage n. We have to determine the number of moves made by each head. Consider head H_s. Before M reaches the first moment of the periodic part of the execution, H_t makes some c_t moves and H_s makes some c_s moves (c_t, c_s are constants). So there are still $f_{a,t,l} - c_t$ cells left for H_t in block B_l. During each period cycle, H_t makes u_t moves while H_s makes u_s moves, for some constants u_t, u_s. So, to the end of the last fully executed cycle H_t makes

$$(f_{a,t,l} - c_t) - r(f_{a,t,l} - c_t, u_t)$$

moves. It corresponds to $[(f_{a,t,l}-c_t)/u_t]$ full cycles (where $[p]$ stands for the integer part of p, the maximal $n\in\mathbb{Z}$ such that $n\leqslant p$). At the same time head H_s makes

$$[(f_{a,t,l}-c_t)/u_t]\cdot u_s$$

moves. During the last, not fully executed cycle each head makes some constant number of moves depending only on $r(f_{a,t,l}-c_t, u_t)$. Machine M needs some p_1 steps to execute the cycle. For each $p<p_1$ there is a node in T_{i+1} corresponding to the situation when the last cycle terminates after p initial steps. We fix $p<p_1$ and we describe the corresponding node a'' of T_{i+1}. Let v_s be the number of moves made by H_s during p initial machine steps in the cycle, and let v_s' be the number of moves made during $p-1$ such steps. Finally, we can say that during stage n head H_s makes

$$c_s+[(f_{a,t,l}-c_t)/u_t]\cdot u_s+v_s$$

moves. New distance functions associated with a'' are defined as follows. For heads $H_s\in S$ it is simple since these heads enter new blocks. Let $s\notin S$. Then the new distance function $f_{a'',s,j}$ is undefined if $f_{a,s,j}$ is undefined and

$$f_{a'',s,j}=f_{a,s,j}-c_s-[(f_{a,t,l}-c_t)/u_t]\cdot u_s-v_s$$

otherwise. The case conditions associated with a'' are the following.

For each $s\notin S$ and j such that $f_{a,s,j}$ is defined

$$f_{a,s,j}-c_s-[(f_{a,t,l}-c_t)/u_t]\cdot u_s-v_s=0,$$

$$f_{a,s,j}-c_s-[(f_{a,t,l}-c_t)/u_t]\cdot u_s-v_s'>0.$$

For $s\in S$ and j such that $f_{a'',s,j}$ is defined

$$f_{a'',s,j}>0.$$

It follows from the construction that Claim 2.3(i) holds for T_{i+1}. So it remains to prove that each of the above-defined case condition and distance function is of the desired form. What we need is the following lemma.

Lemma 2.5. (i) *A sum of simple expressions is a simple expression.*

(ii) *If ξ is a simple expression and $c\in\mathbb{Z}$, then $c\cdot\xi$ is also a simple expression.*

(iii) *If ξ is a simple expression, $c\in\mathbb{Z}$, $d\in\mathbb{N}$, then $[(\xi-c)/d]$ is a simple expression.*

Proof. (i) and (ii) are obvious. For (iii) consider $\xi=(\sum\xi_j)/e$. Then

$$\begin{aligned}[(\xi-c)/d]&=\frac{\xi-c-r(\xi-c,d)}{d}\\&=\frac{(\sum\xi_j)/e-c-r((\sum\xi_j)/e-c,d)}{d}\\&=\frac{\sum\xi_j-ce-e\cdot r((\sum\xi_j-ce)/e,d)}{de}.\end{aligned}\tag{2.1}$$

Now we have to prove that $e\cdot r((\sum\xi_j-ce)/e,d)$ is of the appropriate form. For $\alpha,\beta,\sigma\in\mathbb{N}$, $\alpha/\beta\in\mathbb{N}$ we have $r(\alpha/\beta,\sigma)=r(\alpha,\beta\sigma)/\beta$. So

$$e\cdot r((\sum\xi_j-ce)/e,d)=e\cdot r(\sum\xi_j-ce,ed)/e=r(\sum\xi_j-ce,ed).$$

The last expression is a remainder function, so by (2.1), $[(\xi-c)/d]$ is a simple expression. □

By Lemma 2.5, each constructed distance function and case condition has the desired form and Claim 2.3(ii) holds for the tree T_{i+1}.

We have just described the construction of tree C_M. Since each node of C_M has finitely many successors and the height of C_M is not greater than $2mk$, C_M is a finite tree.

While looking at the work of M it is troublesome to consider all remainder functions which occur within the case conditions and the distance functions. We shall find a way to elude this difficulty. Let $\mathscr{R}$ be the set of all these remainder functions. Take $g\in\mathbb{N}$ such that if $R\in\mathscr{R}$, say $R(x_1,\ldots,x_{2m})=c\cdot r(\sum\xi_j,e)$, then e divides g.

Definition 2.6. Let $X'=\{1^{a_1}*1^{a_2}*\cdots*1^{a_{2m}}\colon \forall i\ \ g\mid a_i\}$ and $P'=P_m\cap X'$.

Essentially, in the rest of this section we shall use only inputs from X'. The reason for that will become clear when we formulate the following lemma.

Lemma 2.7. *For each $R\in\mathscr{R}$ there is a constant c_R such that for every $x\in X'$, $x=1^{a_1}*1^{a_2}*\cdots*1^{a_{2m}}$, we have $R(a_1,\ldots,a_m)=c_R$.*

Proof. *By induction on complexity of R*: Suppose $R(x_1,\ldots,x_{2m})=c\cdot r(\sum\xi_j,e)$. If $\xi_j(x_1,\ldots,x_{2m})=d\cdot x_j$, then $\xi_j(a_1,\ldots,a_{2m})=d\cdot a_j$. But $e\mid g$ and $g\mid a_j$, so $e\mid d\cdot a_j$. If ξ_j is a remainder function, then by the induction hypothesis $\xi_j(a_1,\ldots,a_{2m})$ has a constant value not depending on $a_1,\ldots,a_{2m}$. Hence, each $\xi_j(a_1,\ldots,a_{2m})$ is either a constant not depending on $a_1,\ldots,a_{2m}$ or a number divisible by e. So the lemma holds for R. □

Definition 2.8. If ϕ is a distance function

$$\phi(x_1,\ldots,x_{2m})=\left(\sum_i\xi_i(x_1,\ldots,x_{2m})\right)\Big/d, \tag{2.2}$$

then ϕ', the reduced distance function, is obtained from ϕ by replacing in (2.2) each remainder function ξ_i by the constant c_{ξ_i} given by Lemma 2.7. Similarly, we define the reduced case conditions.

Lemma 2.9. *If $x\in X'$, $x=1^{a_1}*1^{a_2}*\cdots*1^{a_{2m}}$, then for each distance function ϕ*

$$\phi(a_1,a_2,\ldots,a_{2m})=\phi'(a_1,a_2,\ldots,a_{2m})$$

and for each case condition ψ *we have*

$$\psi(a_1, a_2, \dots, a_{2m}) \Leftrightarrow \psi'(a_1, a_2, \dots, a_{2m}).$$

Proof. It follows immediately from the definition. □

The most important fact about the reduced distance functions and the reduced case conditions is that they involve only linear functions. It considerably simplifies the situation.

Assume that l is a node of C_M. Then let $X_l' = X_l \cap X'$. Let Ψ_l be the set of all reduced case conditions associated with nodes lying on path $p(l)$. Clearly,

$$X_l' = \{1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}} \in X' \colon \forall \phi' \in \Psi_l \;\; \phi'(a_1, a_2, \dots, a_{2m})\}.$$

Before we proceed we recall terminology of geometry which we shall use. If $\phi(x_1, \dots, x_n) = \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n$ (where $\alpha_1, \dots, \alpha_n \in \mathbb{Q}$) and $c \in \mathbb{Q}$, then the set $\{\boldsymbol{x} \in \mathbb{Q}^n \colon \phi(\boldsymbol{x}) = c\}$ is called a *hyperplane* in $\mathbb{Q}^n$. The sets $\{\boldsymbol{x} \in \mathbb{Q}^n \colon \phi(\boldsymbol{x}) > c\}$ and $\{\boldsymbol{x} \in \mathbb{Q}^n \colon \phi(\boldsymbol{x}) \geqslant c\}$ are called *halfspaces* with the edge $(\boldsymbol{x} \in \mathbb{Q}^n \colon \phi(\boldsymbol{x}) = c\}$. An intersection of a finite number of halfspaces is a *polyhedron*. Hence, for each polyhedron U there are $\phi_1, \phi_2, \dots, \phi_j$, linear combinations of $x_1, \dots, x_n$, and $c_1, \dots, c_n \in \mathbb{Q}$ such that $U = \{\boldsymbol{x} \in \mathbb{Q}^n \colon \forall i \leqslant j \;\; \phi_i(\boldsymbol{x}) \, \$_i \, c_i\}$, where each $\$_i$ is $<$ or $\leqslant$. U is called a *layer* if there is $\phi(x_1, \dots, x_n)$, a linear combination of $x_1, \dots, x_n$, and $c_1, c_2 \in \mathbb{Q}$ such that $U = \{\boldsymbol{x} \in \mathbb{Q}^n \colon c_1 \leqslant \phi(\boldsymbol{x}) \leqslant c_2\}$. So a layer is simply a set of points which lie between two parallel hyperplanes. A *polygon* is a polyhedron of dimension 2. A *face* of a polyhedron U is either the empty set or a polyhedron obtained by replacing some of the inequalities that define U with equations. A proper face of U is a face not equal to the empty set or U. A maximal proper face of U is called a *facet* of U. For $S \subseteq \mathbb{Q}^n$ the affine hull of S, aff(S), is the set of all $z \in \mathbb{Q}^n$ which can be expressed as $z = \sum_{x \in S'} \lambda_x \cdot \boldsymbol{x}$ satisfying $\sum_{x \in S'} \lambda_x = 1$ for some finite $S' \subseteq S$. S is an affine subspace of $\mathbb{Q}^n$ if aff$(S) = S$. We define the *dimension* $\dim S$ of $S \subseteq \mathbb{Q}^n$ to be the dimension of aff(S), its affine hull.

Each word $1^{a_1} * 1^{a_2} * \cdots * 1^{a_{2m}}$, corresponds to the string $(a_1, a_2, \dots, a_{2m}) \in \mathbb{Q}^{2m}$. For several reasons it will be more convenient to consider elements of $\mathbb{Q}^{2m}$ rather than words. This enables us to use simple geometric techniques. For that reason we shall identify word $1^{a_1} * \cdots * 1^{a_{2m}}$ with the point $(a_1, \dots, a_{2m}) \in \mathbb{Q}^{2m}$. Also the sets X', X_l', P' shall be treated as subsets of $\mathbb{Q}^{2m}$.

Let $V = \{\boldsymbol{x} \in \mathbb{Q}^{2m} \colon \forall i \;\; x_i = x_{2m-i+1}\}$. Clearly, $P' = V \cap X'$ and V is an affine subspace of $\mathbb{Q}^{2m}$ of dimension m. Recall that $X_l' = \{\boldsymbol{x} \in X' \colon \forall \phi \in \Psi_l \;\; \phi(\boldsymbol{x})\}$. Define

$$U_l = \{\boldsymbol{x} \in \mathbb{Q}^{2m} \colon \forall \phi \in \Psi_l \;\; \phi(\boldsymbol{x})\}.$$

Then, obviously, $X_l' = U_l \cap X'$. Recall that each $\phi \in \Psi_l$ defines in $\mathbb{Q}^{2m}$ a hyperplane or a halfspace depending on whether ϕ is an equation or an inequality. Hence, U_l is a polyhedron in $\mathbb{Q}^{2m}$. Automaton M recognizes language P_m, so

$$X' = \bigcup \{X_l' \colon l \text{ is a leaf of } C_M\},$$

$$P' = \bigcup \{X_l' \colon l \text{ is an accepting leaf of } C_M\}.$$

Polyhedrons U_l might not cover $\mathbb{Q}^{2m}$ but by the first of the above equalities they cover X'. Now we shall seek l, an accepting leaf of C_M, such that U_l is large enough for our purposes. Before that we must prove some auxiliary facts of geometry.

Lemma 2.10. *Suppose* $g \in \mathbb{N}$. *Take a finite set of layers in* $\mathbb{Q}^n$, *say* $F_1, F_2, \ldots, F_t$. *Then there is a point* $x \in \mathbb{Q}^n \setminus \bigcup_i F_i$ *such that* $\forall i \leqslant n \; g \mid x_i$.

Proof. *By simple induction on n*: For $n=1$ the lemma is obviously true, so assume that $n>1$. Consider hyperplanes W_i: $W_i = \{x \in \mathbb{Q}^n\colon x_1 = g \cdot i\}$. If the edges of F_j are not parallel to hyperplane $\{x \in \mathbb{Q}^n\colon x_1 = 0\}$, then $F_j \cap W_i$ is a layer in W_i for each i. If the edges of F_j are parallel to this hyperplane, then for almost all i sets F_j and W_i are disjoint. So there is $i \in \mathbb{N}$ such that the sets $W_i \cap F_1, W_i \cap F_2, \ldots, W_i \cap F_t$ are layers in W_i. Hyperplane W_i is isomorphic to $\mathbb{Q}^{n-1}$, so by the induction hypothesis there is $x \in W_i \setminus \bigcup (W_i \cap F_i) = W_i \setminus \bigcup F_i$ such that $g \mid x_i$ for $i = 2, 3, \ldots, n$. But $x \in W_i$ so $g \mid x_1$. □

Let $\mathbb{Q}^n_+ = \{x \in \mathbb{Q}^n\colon \forall i \leqslant n \; x_i \geqslant 0\}$. Since $X' \subseteq \mathbb{Q}^{2m}_+$, we shall virtually stay in $\mathbb{Q}^{2m}_+$.

Lemma 2.11. *Lemma* 2.10 *holds also if we replace* $\mathbb{Q}^n$ *by* $\mathbb{Q}^n_+$.

Proof. $\mathbb{Q}^n$ is a union of finitely many subsets isomorphic to $\mathbb{Q}^n_+$. So if we could cover $\mathbb{Q}^n_+$ by finitely many layers we could do the same with $\mathbb{Q}^n$. □

Definition 2.12. Let L be a polyhedron, say $L = \bigcap_{i \in J} \{x\colon \phi_i(x) \,\$_i\, c_i\}$ where each $\$_i$ is $\leqslant$ or $<$. If $z \in L$ then by $\mathscr{L}(L, z)$ we mean the polyhedron (Fig. 1)

$$\mathscr{L}(L, z) = \bigcap_{i \in J} \{x\colon \phi_i(x) \geqslant \phi_i(z)\}.$$

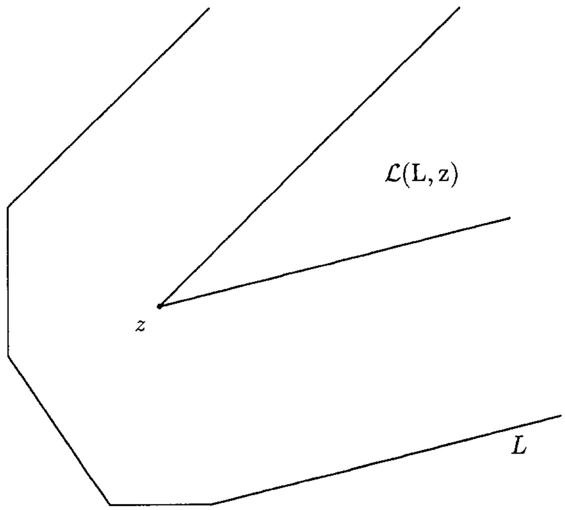

Fig. 1.

Lemma 2.13. *If L is a polyhedron, $z \in L$, then*
(i) $\mathscr{L}(L,z) \subseteq L$.
(ii) *If $\mathscr{L}(L,z)$ is not the point z itself, then*

$$\mathscr{L}(L,z) = \bigcup \{p\colon p \text{ is a halfline with the end } z,\ p \subseteq L\}. \tag{2.3}$$

(iii) *If z, $s \in L$ then $\mathscr{L}(L,s)$ is equal to $\mathscr{L}(L,z)$ translated by vector $\overrightarrow{zs}$.*
(iv) *If $\mathscr{L}(L,z) \subseteq V$ and $\dim \mathscr{L}(L,z) = m$, then $\mathscr{L}(L,z) \cap X' \neq \emptyset$.*

Proof. Let L be defined as in Definition 2.12. Part (i) is obvious. For part (ii) we show first the inclusion $\subseteq$. Suppose $y \in \mathscr{L}(L,z)$, $y \neq z$. Take the halfline p with the end z such that $y \in p$. Consider function ϕ_i on p. It is linear. We have $\phi_i(y) \geqslant \phi_i(z)$, so ϕ_i is not decreasing along p. So $\phi_i(\boldsymbol{u}) \geqslant \phi_i(z)$ for every $\boldsymbol{u} \in p$. It holds for every i so $p \subseteq \mathscr{L}(L,z) \subseteq L$. Hence, p witnesses that y is an element of the union on the right side of (2.3). For the inclusion $\supseteq$ assume that p is a halfline with the end z, $p \subseteq L$. Consider function ϕ_i on p. It is linear and has values not smaller than c_i. So function ϕ_i cannot decrease along p. Hence, its values are not smaller than $\phi_i(z)$. So we get $p \subseteq \mathscr{L}(L,z)$.

For (iii) we show first that $\mathscr{L}(L,z) + \overrightarrow{zs} \subseteq \mathscr{L}(L,s)$. Take $\boldsymbol{u} \in \mathscr{L}(L,z)$. Then $\phi_i(\boldsymbol{u}) \geqslant \phi_i(z)$ for each i. Note that $\phi_i(\boldsymbol{u}+\overrightarrow{zs}) - \phi_i(\boldsymbol{u}) = \phi_i(s) - \phi_i(z)$, because ϕ_i is a linear function. So

$$(\phi_i(\boldsymbol{u}+\overrightarrow{zs}) - \phi_i(\boldsymbol{u})) + \phi_i(\boldsymbol{u}) \geqslant (\phi_i(s) - \phi_i(z)) + \phi_i(z),$$

$$\phi_i(\boldsymbol{u}+\overrightarrow{zs}) \geqslant \phi_i(s).$$

So $(\boldsymbol{u}+\overrightarrow{zs}) \in \mathscr{L}(L,s)$. Hence, $\mathscr{L}(L,z) + \overrightarrow{zs} \subseteq \mathscr{L}(L,s)$. In the same way we get $\mathscr{L}(L,s) + \overrightarrow{sz} \subseteq \mathscr{L}(L,z)$. So $\mathscr{L}(L,s) + \overrightarrow{sz} + \overrightarrow{zs} \subseteq \mathscr{L}(L,z) + \overrightarrow{zs}$. Then $\mathscr{L}(L,s) \subseteq \mathscr{L}(L,z) + \overrightarrow{zs}$.

For (iv) note that, since $\dim(\mathscr{L}(L,z)) = m$ and $\mathscr{L}(L,z)$ is a polyhedron in V, there is a point s which is an interior point of $\mathscr{L}(L,z)$ in the sense of topology of V. Then $\phi_i(s) > \phi_i(z)$ for each i (otherwise, s would lie on one of the facets of $\mathscr{L}(L,z)$). By (ii), there is a halfline p beginning with z and containing point s. Function ϕ_i grows to infinity on p. Since ϕ_i's are linear functions, we can find a number $\eta \in \mathbb{Q}$, $\eta > 0$, such that for each pair of points $\boldsymbol{x}$, y, $|\phi_i(\boldsymbol{x}) - \phi_i(y)| \leqslant \eta \cdot d(\boldsymbol{x}, y)$, where $d(\boldsymbol{x}, y)$ denotes the distance between $\boldsymbol{x}$ and y. Therefore, we can find a point $\boldsymbol{u} \in p$ such that a ball K in V with center at $\boldsymbol{u}$ and radius $\frac{1}{2}g\sqrt{m}$ is a subset of $\mathscr{L}(L,z)$. Indeed, it suffices to take $\boldsymbol{u}$ such that $\phi_i(\boldsymbol{u}) > \phi_i(z) + \frac{1}{2}\eta \cdot g\sqrt{m}$ for each i. If $\boldsymbol{x} \in K$, then

$$\begin{aligned}\phi_i(\boldsymbol{x}) &= \phi_i(\boldsymbol{u}) + (\phi_i(\boldsymbol{x}) - \phi_i(\boldsymbol{u})) \geqslant \phi_i(\boldsymbol{u}) - \eta \cdot d(\boldsymbol{x}, \boldsymbol{u}) \\ &\geqslant \phi_i(z) + \tfrac{1}{2}\eta \cdot g\sqrt{m} - \tfrac{1}{2}\eta \cdot g\sqrt{m} = \phi_i(z).\end{aligned}$$

So $\boldsymbol{x} \in \mathscr{L}(L,z)$. Each ball in V of radius $\frac{1}{2}g\sqrt{m}$ contains a point from X'. So the lemma follows. $\square$

Now consider $l_0, l_1, \ldots, l_t$, all accepting leaves of C_M. We may assume that each Ψ_{l_i}, the set of reduced case conditions leading to l_i, contains the inequalities $x_j \geqslant 0$ for $j \leqslant 2m$. Consider now polyhedrons W_i, where $W_i = V \cap U_{l_i}$. Of course, $V \cap \mathbb{Q}_+^{2m}$ is isomorphic to $\mathbb{Q}_+^m$ and $V\cap\mathbb{Q}_+^{2m}\supseteq\bigcup_i W_i\supseteq V\cap\mathbb{Q}_+^{2m}\cap X'$. For each W_i fix a point $w_i\in W_i$.

Claim 2.14. *For some i polyhedron $\mathscr{L}(W_i, w_i)$ has dimension m.*

Proof. Assume the converse. So each $\mathscr{L}(W_i, w_i)$ is a subset of some layer. On the other hand, by the definition of $\mathscr{L}(W_i, w_i)$, the set $W_i\setminus\mathscr{L}(W_i, w_i)$ is included in a finite union of layers. Consequently, we can cover $V\cap\mathbb{Q}_+^{2m}\cap X'$ by a finite union of layers in $V\cap\mathbb{Q}_+^{2m}$. This contradicts Lemma 2.11. □

By the above claim we may assume that polyhedron $\mathscr{L}(W_0, w_0)$ has dimension m. By Lemma 2.13 there is $x_0\in W_0\cap X'$. Then again by Lemma 2.13, $\mathscr{L}(W_0, x_0)$ has dimension m. Leaf l_0 is the leaf l we have been looking for, for which U_l is "large enough".

Polyhedron U_{l_0} is defined by the reduced case conditions from Ψ_{l_0}. We split Ψ_{l_0} into two subsets Φ_1 and Φ_2. For $\phi\in\Psi_{l_0}$ we put ϕ into Φ_1 if either the hyperplane defined by ϕ (if ϕ is an equation) is parallel to V or the halfspace defined by ϕ has the edge parallel to V. Otherwise, ϕ is in Φ_2. Let

$$V_i=\{x\in\mathbb{Q}^{2m}\colon \forall\phi\in\Phi_i\ \phi(x)\}$$

for $i=1,2$. Obviously, $U_{l_0}=V_1\cap V_2$ and

(i) all facets of V_1 are parallel to V,

(ii) no facet of V_2 is parallel to V.

Before we proceed let us notice the following property.

Property 2.15. *If a halfline $p\subseteq\mathbb{Q}^{2m}$ contains a point from X', then it contains infinitely many of them, each two subsequent points staying at a constant distance (depending on p).*

The property easily follows from the fact that we are working in $\mathbb{Q}^{2m}$, not in $\mathbb{R}^{2m}$.

Lemma 2.16. $V_1\cap X'\subseteq V$.

Proof. We know that $V_1\cap V_2\cap X'=U_{l_0}\cap X'\subseteq V$. Assume that there is a point $x\in(V_1\cap X')\setminus V$.

Claim 2.17. *There is a halfline $p\subseteq\mathscr{L}(W_{l_0}, x_0)$ with the beginning x_0 not parallel to any facet of V_2.*

Proof of Claim 2.17. To show the claim note that for each facet F of V_2 the set $\{y\in\mathscr{L}(W_{l_0}, x_0)\colon \overrightarrow{x_0y}$ is parallel to $F\}$ has dimension less than m. A finite union of sets of dimension less than m cannot cover polyhedron $\mathscr{L}(W_{l_0}, x_0)$ of dimension m. Hence, there is a point $y\in\mathscr{L}(W_{l_0}, x_0)$ such that vector $\overrightarrow{x_0y}$ is not parallel to any facet of V_2. Take p to be the halfline beginning with x_0 and containing point y. □

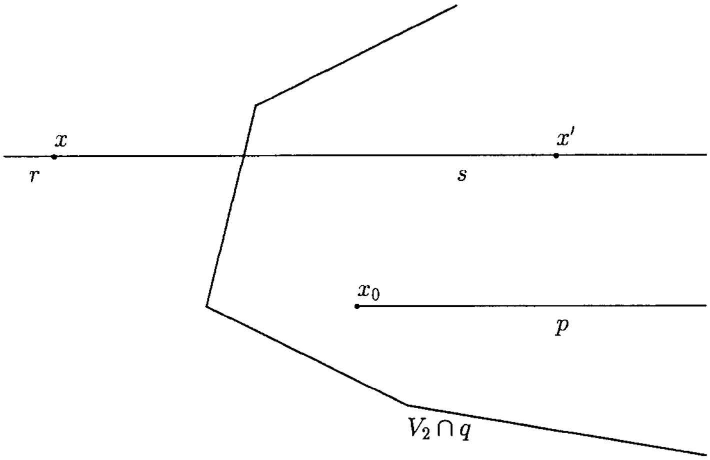

Fig. 2. The situation on plane q.

Proof of Lemma 2.16 (*conclusion*): Let q be the plane containing halfline p and point $\boldsymbol{x}$ (Fig. 2). Then, by Claim 2.17 $V_2 \cap q$ is a polygon with edges not parallel to p. Since $p \subseteq U_{l_0}$, $p \subseteq U_{l_0} \cap q \subseteq V_2 \cap q$. Let r be the line parallel to p containing point $\boldsymbol{x}$. It is geometrically evident that $(V_2 \cap q) \cap r$ is a halfline, say s. Since $\boldsymbol{x} \in r$, it follows from Property 2.15 that s contains infinitely many points from X'. On the other hand, line r is parallel to p so r is parallel to V. Polyhedron V_1 has facets parallel to V, line r contains point $\boldsymbol{x}$ from V_1, so $r \subseteq V_1$. Hence, $s \subseteq V_1 \cap V_2$. We have noticed previously that $s \cap X' \neq \emptyset$. So there is a point $\boldsymbol{x}' \in s \cap X' \subseteq V_1 \cap V_2 \cap X' = X'_{l_0}$. Leaf l_0 is accepting, so $\boldsymbol{x}' \in V$. We know that $x' \in V$, $p \subseteq V$ and V is an affine space. So $q \subseteq V$ and hence, $\boldsymbol{x} \in V$ contrary to the assumption about $\boldsymbol{x}$. □

Polyhedron V_1 has interesting properties: $V \subseteq V_1$ and $V_1 \cap X' \subseteq V$. Let $g_1, g_2, \ldots, g_\eta$ be all reduced case conditions defining V_1 (i.e. $\Phi_1 = \{g_1, g_2, \ldots, g_\eta\}$).

Lemma 2.18. *Each g_j is a condition of the form*

$$h_j(x_1, \ldots, x_{2m}) < c \quad or \quad h_j(x_1, \ldots, x_{2m}) = c',$$

where $c' = 0$ and

$$h_j(x_1, \ldots, x_{2m}) = \sum_{i \leqslant m} \alpha_{i,j} x_i - \sum_{i \leqslant m} \alpha_{i,j} x_{2m-i+1}$$

for some $\alpha_{1,j}, \ldots, \alpha_{m,j} \in \mathbb{Q}$.

Proof. We know that each h_j is a linear combination of $x_1, \ldots, x_{2m}$. Essentially, Lemma 2.18 says that in $h_j(x_1, \ldots, x_{2m})$ the coefficients of x_i and x_{2m-i+1} are the same except for their signs, which are different. Let $h_j(x_1, \ldots, x_{2m}) = \sum_{i \leqslant m} \alpha_i x_i - \sum_{i \leqslant m} \beta_i x_{2m-i+1}$. Now let us fix some i. Consider the points $n \cdot (a_1, a_2, \ldots, a_{2m})$, where $a_t = 1$ for $t = i, 2m-i+1$ and $a_t = 0$ otherwise. For $n \in \mathbb{Z}$ all these points are elements of V and g_j holds for them. But g_j takes for these points the form $n\cdot(\alpha_i-\beta_i)<c$ or $n\cdot(\alpha_i-\beta_i)=c'$. Because it holds for every n, α_i and β_i must be equal. In the last case we must have $c'=0$. □

It is a simple observation that if g_j takes the form $h_j(x_1,\ldots,x_{2m})<c$, then $c>0$. Indeed, g_j holds for $x_1=\cdots=x_{2m}=0$ and $h_j(0,\ldots,0)=0$.

Lemma 2.19. *There are m linearly independent polynomials among h_1, h_2, ..., h_η.*

Proof. Consider the set $S=\{x\in\mathbb{Q}^{2m}\colon \forall i\leqslant\eta\ h_i(x)=0\}$. Clearly, $V\subseteq S$ and $S\subseteq V_1$. Assume that $\dim S>m$. Then there is a point $x\in S\setminus V$, $x\in X'$. On the other hand, $V_1\cap X'\subseteq V$: a contradiction. So $\dim S=m$ and the number of linearly independent polynomials among h_1, ..., h_η is equal to $2m-\dim S=m$. □

We may assume that h_1, ..., h_m are linearly independent. Let condition g_i be generated at the beginning of stage s_i. By the critical moment of g_i we mean the beginning of stage s_i. We may assume that $s_1<s_2<\cdots<s_m$.

Definition 2.20. Suppose H, H' are heads of M. We say that a pair (H,H') is dead at some moment of computation of M if for some $l\leqslant 2m$ head H stands on the right side of block B_l while head H' stands on the right side of block B_{2m-l+1}.

Note that if a pair (H,H') is dead at some moment, then it will remain dead for the rest of the computation. Intuitively, if a pair (H,H') is dead then the heads H and H' cannot be used to check that any two corresponding blocks are equal. However, it is not perfectly true. If variable x_j occurs in g_j, then block B_j must be read by some head before the critical moment of g_j. However, the length of B_j may be recorded by some other head by its position in some other block. In turn, this information can be transmitted elsewhere and so on. Therefore, information about block B_j can be used long after reading block B_j. This makes the analysis complex.

Now we shall show that $m\leqslant\frac{1}{2}k^3$. First consider g_1. Suppose that x_j occurs in g_1 with a nonzero coefficient. Hence, there is a head H which reads block B_j before the critical moment of g_1. The coefficient of x_{2m-j+1} in g_1 is the same by Lemma 2.18, so also not equal to zero. Hence, before the critical moment of g_1 some head H' reads block B_{2m-j+1}. We see that at the critical moment of g_1 the pair (H,H') is dead. Take a look what happens next. We show that after at most k next critical moments a new pair of heads becomes dead.

Assume that H'' is the head which reaches a new block at the beginning of stage s_1. Let $f_1,\ldots,f_{k-1}$ denote the distances of the other heads from the ends of the blocks they were in (some of f_i's might be equal to 0). If after this moment any reduced case condition is generated then it takes the form

$$z+\sum\alpha_i f_i<c \quad \text{or} \quad z+\sum\alpha_i f_i=c,$$

where $\alpha_i \in \mathbb{Q}$ and z is an expression depending only on the blocks read *after* the beginning of stage s_1. Consider some g_j for $j>1$. Then $h_j = z + \sum \alpha_i f_i$, where z is as above. Suppose $z \neq 0$. Then z contains some variable x_t standing for the length of block B_t read by some head, say $H^{(3)}$, after reaching stage s_1. The corresponding block B_{2m-t+1} was read by some head $H^{(4)}$, not necessarily after reaching stage s_1, may be before. Note that if block B_{2m-t+1} was read by head $H^{(4)}$ after reaching stage s_1, then the pairs $(H^{(3)}, H^{(4)})$ and (H, H') must be different. Indeed, it is a consequence of the fact that the pair (H, H') is dead after reaching stage s_1 and these heads cannot read any corresponding blocks.

Let g_t be the first case condition such that at the critical moment of g_t a new pair of heads becomes dead. Each h_j for $1<j<t$ takes the form

$$h_j = z_j + \sum_{i=1}^{k-1} \alpha_{i,j} f_i,$$

where z_j contains only variables denoting the lengths of the blocks read after the beginning of stage s_1. As we have noticed, z_j cannot contain simultaneously variables x_i and x_{2m-i+1} ($i \leqslant m$) since otherwise a new pair of heads would be dead at the critical moment of g_j.

Lemma 2.21. *Functions* $u_j = \sum_{i=1}^{k-1} \alpha_{i,j} f_i$ *for* $1<j<t$ *are linearly independent.*

Proof. Assume that these functions are linearly dependent, i.e.

$$\gamma_2 u_2 + \gamma_3 u_3 + \cdots + \gamma_{t-1} u_{t-1} = 0$$

for some $\gamma_2, \ldots, \gamma_{t-1} \in \mathbb{Q}$, not all equal 0. We show that $\gamma_2 h_2 + \cdots + \gamma_{t-1} h_{t-1} = 0$. Consider variables x_s and x_{2m-s+1} ($s \leqslant 2m$). Only one of them, say x_s, can be used in expressions z_j ($1<j<t$). Variables x_s and x_{2m-s+1} stand in u_j with some coefficients λ_j and Λ_j. Then

$$\sum_{j=2}^{k-1} \gamma_j \lambda_j$$

is the coefficient of x_s in $\gamma_2 u_2 + \gamma_3 u_3 + \cdots + \gamma_{t-1} u_{t-1}$. So $\sum \gamma_j \lambda_j = 0$. Similarly, $\sum \gamma_j \Lambda_j = 0$. Note that now the coefficient of x_{2m-s+1} in h_j is the same as in u_j, i.e. Λ_j, since $h_j = z_j + u_j$ and x_{2m-s+1} does not occur in z_j. The coefficient of x_s in h_j is $-\Lambda_j$ by Lemma 2.18. But $h_j = z_j + u_j$ and in u_j variable x_s has coefficient λ_j. So the coefficient of x_s in z_j is equal to $-\Lambda_j - \lambda_j$. On the other hand, we have

$$\begin{aligned}\gamma_2 h_2 + \cdots + \gamma_{t-1} h_{t-1} &= \gamma_2(z_2 + u_2) + \cdots + \gamma_{t-1}(z_{t-1} + u_{t-1}) \\ &= (\gamma_2 z_2 + \cdots + \gamma_{t-1} z_{t-1}) + (\gamma_2 u_2 + \cdots + \gamma_{t-1} u_{t-1}) \\ &= \gamma_2 z_2 + \cdots + \gamma_{t-1} z_{t-1}.\end{aligned}$$

The last expression does not contain x_{2m-s+1} since no z_j contains x_{2m-s+1}. In turn the coefficient of x_s is there equal to $\sum \gamma_j(-\Lambda_j - \lambda_j) = -\sum \gamma_j \Lambda_j - \sum \gamma_j \lambda_j = 0$. Hence,

$\gamma_2 h_2 + \cdots + \gamma_{t-1} h_{t-1}$ does not contain x_s and x_{2m-s+1}. Number s was arbitrary, so $\gamma_2 h_2 + \cdots + \gamma_{t-1} h_{t-1} = 0$ and $h_2, \ldots, h_{t-1}$ are linearly dependent: a contradiction. □

It follows from Lemma 2.21 that we have $t-2$ linearly independent functions u_j. Each u_j is a linear combination of $f_1, \ldots, f_{k-1}$. Hence, $t-2 \leqslant k-1$, so $t \leqslant k+1$.

We have just proved that at the critical moment of g_{k+1} a new pair of heads, different from (H, H') must be dead. The above proof can be repeated virtually without change to show that for each n at least one pair of heads becomes dead after the critical moment of g_n and no later than the critical moment of g_{n+k}. There are only $k^2/2$ pairs of heads (we allow the first and the second elements in a pair to be the same). Hence, m, the number of critical moments is not greater than $k \cdot k^2/2 = k^3/2$. It completes the proof of Theorem 2.2.

3. Recognizing P_m

Recall that $P_{m,c}$ (for $c \in \mathbb{N}$) is a sublanguage of language P_m defined as follows:

$$P_{m,c} = \{1^{a_1} * \cdots * 1^{a_{2m}} \in P_m \colon \forall n, l \leqslant 2m \ a_n < c \cdot a_l\}.$$

We show in this section that for each c we can recognize language $P_{m,c}$ using an automaton with about $\sqrt[3]{m}$ heads. It shows that in some sense the bound given in Theorem 2.2 is stringent. However, $P_{m,c}$ is only a sublanguage of P_m and we do not know such an algorithm for P_m.

Theorem 3.1. *Let $k, c \in \mathbb{N}$. There is a k-head 1-MFA M recognizing a language P' such that $P_{m,c} \subseteq P' \subseteq P_m$ and $m \geqslant k^3/24$.*

Proof. We consider only inputs of the form $1^{a_1} * \cdots * 1^{a_{2m}}$, where $a_l < c \cdot a_n$ for each $n, l \leqslant 2m$. By blocks of such an input word we mean the subwords $1^{a_1}, 1^{a_2}, \ldots, 1^{a_{2m}}$. These blocks will be denoted by $B_1, B_2, \ldots, B_{2m}$. Let $|B_i|$ stand for the length of block B_i. So M has to check that for each $i \leqslant m$, $|B_i| = |B_{2m-i+1}|$.

First we describe how M can remember the length of some block B_l. For that purpose we need three heads, say H_1, H_2, H_3, with H_1 positioned at the beginning of B_l and two other heads placed at the beginning of some block B_n. Information about the length of B_l will be stored inside B_n. More precisely, we record only $[|B_l|/c]$ and $r(|B_l|, c)$ is to be remembered by the finite memory of M. Firstly, H_1 and H_2 move simultaneously until H_1 reaches the end of block B_l. For each c moves of H_1, head H_2 makes only one move to the right. So when H_1 reaches the end of B_l then the distance between H_2 and H_3 is $[|B_l|/c]$. Head H_2 is still inside block B_n since $|B_l| < c \cdot |B_n|$. Now we move simultaneously heads H_2 and H_3 with the same speed until H_2 reaches the end of B_n. The distance between H_2 and H_3 remains unchanged, so finally, H_3 is placed at the distance $[|B_l|/c]$ from the end of block B_n. If later we have to check that $|B_l| = |B_{2m-l+1}|$, then we can use head H_3 and some other head H placed at the beginning of B_{2m-l+1}. We start both of them, H making c moves for each single move of H_3. If $|B_l|=|B_{2m-l+1}|$ then there are exactly $r(|B_l|, c)$ symbols left inside B_{2m-l+1} in front of head H at the moment when H_3 reaches the end of B_l. This can be easily verified using the finite memory of M.

If W is one of the blocks $B_1, B_2, \ldots, B_{2m}$, say B_l, then let W' be the corresponding block B_{2m-l+1}. The computation of M consists of $k-1$ different stages. For each stage of execution there is a corresponding group of blocks. If W belongs to such a group for stage j then during stage j it is checked whether $|W|=|W'|$. Let the blocks corresponding to stage j be denoted by $B_{j,1}, B_{j,2}, \ldots, B_{j,\sigma(j)}$. Each input word which we consider therefore takes the form

$$\underbrace{B_{1,1}*B_{1,2}*\cdots*B_{1,\sigma(1)}}_{\text{for stage } 1}*\underbrace{B_{2,1}*\cdots*B_{2,\sigma(2)}}_{\text{for stage } 2}*\cdots*\underbrace{B_{k-1,1}*\cdots*B_{k-1,\sigma(k-1)}}_{\text{for stage } k-1}*$$

$$\underbrace{B'_{k-1,\sigma(k-1)}*\cdots*B'_{k-1,1}}_{\text{for stage } k-1}*\underbrace{B'_{k-2,\sigma(k-2)}*\cdots*B'_{k-2,1}}_{\text{for stage } k-2}*\cdots*\underbrace{B'_{1,\sigma(1)}*\cdots*B'_{1,1}}_{\text{for stage } 1}$$

Values $\sigma(j)$ shall be determined later.

Now we describe one stage of computation, say stage j. It has the following important properties.

(i) Only heads $H_1, \ldots, H_{k-j+1}$ are in use, all of them initially placed at the beginning of block $B_{j,1}$.

(ii) When stage j ends, head H_{k-j+1} is at the end of $B'_{j,1}$ and will not be used during the next stages, the remaining heads $H_1, \ldots, H_{k-j}$ are moved to the beginning of $B_{j+1,1}$.

For the sake of simplicity put $B_{j,i}=D_i$. Also let $t=k-j+1$. Stage j is divided into $[(t-2)/2]$ substages plus one additional "final" substage.

$$\cdots\underbrace{D_1*D_2*}_{\text{final substage}}\cdots\underbrace{\quad\cdots\quad}_{\substack{\text{substage}\\ [(t-2)/2]}}\cdots\underbrace{\quad\cdots\quad}_{\text{substage } 2}\underbrace{*D_{\sigma(j)}}_{\text{substage } 1}\cdots$$

$$\underbrace{D'_{\sigma(j)}*\cdots}_{\text{substage } 1}\underbrace{\quad\cdots\quad}_{\text{substage } 2}\cdots\underbrace{\quad\cdots\quad}_{\text{substage } [(t-2)/2]}\underbrace{\cdots*D'_1}_{\text{final substage}}\cdots$$

First substage: During this substage M checks if the lengths of blocks $D_{\sigma(j)}, D_{\sigma(j)-1}, \ldots, D_{\sigma(j)-[(t-2)/2]+1}$ match with the lengths of the corresponding blocks. At the beginning, head H_t moves to the end of $D_{\sigma(j)}$. In the meantime it reads blocks

$$D_{\sigma(j)-[(t-2)/2]+1}, \ldots, D_{\sigma(j)}$$

and uses heads $H_2, H_3, \ldots, H_{t-1}$ to record their lengths inside D_1. It is possible since $2[(t-2)/2]+1 \leqslant t-1$. Head H_1 is left unmoved at the beginning of D_1 and will stay there until the final substage. Then head H_t reads blocks

$$D'_{\sigma(j)}, D'_{\sigma(j)-1}, \ldots, D'_{\sigma(j)-[(t-2)/2]+1}$$

and simultaneously, using information stored by the heads lying inside D_1 their lengths are checked.

Second substage: The second substage looks like the first one except for few details. At the beginning, head H_{t-1} moves to the end of $D_{\sigma(j)}$. In the meantime it reads blocks

$$D_{\sigma(j)-[(t-2)/2]-[(t-4)/2]+1}, \ldots, D_{\sigma(j)-[(t-2)/2]}$$

and uses heads $H_3, H_4, \ldots, H_{t-2}$ to record their length inside D_2. Again, it is possible since $2[(t-4)/2]+2 \leqslant t-2$. Head H_2 is left unmoved at the beginning of D_2 and will stay there until the final substage. Then head H_t reads blocks

$$D'_{\sigma(j)-[(t-2)/2]}, \ldots, D'_{\sigma(j)-[(t-2)/2]-[(t-4)/2]+1}$$

and simultaneously, using information stored by the heads lying inside D_2, automaton M checks the block lengths.

During subsequent substages M works similarly. Each time one head is moved to the end of $D_{\sigma(j)}$ and one is left unmoved for the final substage. Also head H_t reads some number of blocks. The remaining heads move one block forward because of the length checking. The number of such heads decreases by two each substage. Note that there must be at least four of them at the beginning of a substage. It follows that there are $[(t-2)/2]$ of these substages.

Final substage: After the last nonfinal substage there are heads left at the beginnings of $D_1, D_2 \ldots, D_{[(t-2)/2]}$. Also in front of block $D_{[(t-2)/2]+1}$ there are at least two heads. We move one of them to the beginning of $D_{[t/2]+1}$. During this substage M checks that the blocks $D_1, D_2, \ldots, D_{[t/2]+1}$ have the same length as the corresponding blocks. It can be easily done since the number $\sigma(j)$ is chosen so that after the last nonfinal substage the head H_t stands in front of $D'_{[t/2]+1}$. After the checking is done M moves all heads (except H_t and the other heads not already in use) to the end of $D_{\sigma(j)}$.

Now we count for how many blocks D_i automaton M checks that $|D_i| = |D'_i|$ during stage j. During the final substage M checks $[t/2]+1$ blocks. During the first substage M checks $[(t-2)/2]$ blocks, during the second one only $[(t-4)/2]$ of them, then $[(t-6)/2]$, $[(t-8)/2]$, ... So during stage j automaton M checks together

$$\sigma(j) = (1+[t/2]) + ([(t-2)/2] + [(t-4)/2] + \cdots + 1)$$

blocks. Then

$$\begin{aligned}\sigma(j) &= ([t/2] + [(t-2)/2] + [(t-4)/2] + \cdots + 1) + 1\\ &= \tfrac{1}{2} \cdot [t/2] \cdot ([t/2]+1) + 1\\ &\geqslant \frac{1}{2} \cdot \frac{t-1}{2} \cdot \left(\frac{t-1}{2} + 1\right) + 1\\ &= \frac{(t-1)(t+1)}{8} + 1\\ &= \frac{t^2-1}{8} + 1 \geqslant \frac{t^2}{8} = \frac{(k-j+1)^2}{8}.\end{aligned}$$

The number of pairs of blocks checked during all stages can be estimated as follows:

$$\sum_{j=1}^{k-1} \sigma(j) \geqslant \frac{k^2}{8} + \frac{(k-1)^2}{8} + \cdots + \frac{2^2}{8}$$

$$= \frac{1}{8}(k^2 + (k-1)^2 + \cdots + 1^2) - \frac{1}{8}$$

$$= \frac{1}{8 \cdot 6} k \cdot (k+1) \cdot (2k+1) - \frac{1}{8} \geqslant \frac{k^3}{24}. \quad \square$$

The algorithm presented works effectively for large k. It can be improved slightly by combining it with a straightforward algorithm used for recognizing languages L_m ([16]).

References

[1] M. Chrobak, Hierarchies of one-way multihead automata languages, *Theoret. Comput. Sci.* **48** (1986) 153–181, also in: *Proc. ICALP'85*, Lecture Notes in Computer Science, Vol. 194 (Springer, Berlin, 1985) 101–110.

[2] M. Chrobak and W. Rytter, Remarks on string-matching and one-way multihead automata, *Inform. Process. Lett.* **24** (1987) 325–329.

[3] R.W. Floyd, Review 14353 of the paper [13], *Comput. Rev.* **9** (1968) 280.

[4] Z. Galil, Open problems in stringology, in: A. Apostolico and Z. Galil, eds., *Combinatorial Algorithms on Words* (Springer, Berlin, 1974) 350–359.

[5] J. Hromkovič, Closure properties of the family of languages recognized by one-way two-head deterministic finite state automata, in: *Proc. 10th MFCS*, Lecture Notes in Computer Science, Vol. 118 (Springer, Berlin, 1981) 304–313.

[6] J. Hromkovič, One-way multihead deterministic finite automata, *Acta Inform.* **19**(4) (1983) 377–384.

[7] O.H. Ibarra and C.E. Kim, On 3-head versus 2-head finite automata, *Acta Inform.* **4** (1975) 173–200.

[8] M. Kutyłowski, One-way multihead finite automata and 2-bounded languages, *Math. Systems Theory* **23** (1990) 107–139.

[9] M. Li, Lower bounds by Kolmogorov complexity, in: *Proc. ICALP'85*, Lecture Notes in Computer Science Vol. 194 (Springer, Berlin, 1985) 383–393.

[10] C.G. Nelson, One-way automata on bounded languages, Tech. Report TR 14–76, Harvard Center for Res. and Computing, 1976.

[11] T.F. Piatkowski, *N*-head finite state machines, Ph.D. Thesis, University of Michigan, 1965.

[12] A.L. Rosenberg, Nonwriting extensions of finite automata, Ph.D. Thesis, Harvard University, 1965.

[13] A.L. Rosenberg, On multihead finite automata, *IBM J. Res. Develop.* **10** (1966) 388–394.

[14] I.H. Sudborough, Computation by multi-head writing finite automata, Ph.D. Thesis, Pennsylvania State University, 1971.

[15] I.H. Sudborough, One-way multihead writing finite automata, *Inform. Control* **30** (1976) 1–20.

[16] A.C. Yao and R.L. Rivest, $K+1$ heads are better than K, *J. Assoc. Comput. Mach.* **25** (1978) 337–340.