Juraj Hromkovič

Reversals-space-parallelism tradeoffs for language recognition

Mathematica Slovaca, Vol. 41 (1991), No. 2, 121--136

Persistent URL: http://dml.cz/dmlcz/129539

Math. Slovaca 41, 1991, No. 2, 121—136

REVERSALS — SPACE — PARALLELISM TRADEOFFS FOR LANGUAGE RECOGNITION

JURAJ HROMKOVIČ

ABSTRACT. This paper is devoted to the development of a lower bound proof technique for the general model of alternating computations. The produced, combinatorial technique enables to obtain $\Omega(n^{1/3}/\log_2 n)$ lower bound on the tradeoff of complexity measures *REVERSALS·SPACE·PARALLELISM* for the recognition of a specific language on a general alternating machine with the multihead access to the input and an arbitrary organization of the memory.

1. Introduction

One of the hardest problems in the theory of computations and computational complexity is to prove *nontrivial lower bounds on different complexity measures of specific problems* (i.e., on the inevitable amount of computer resources for computing a given computing task). In spite of much effort the results so far have not been satisfactory. Thus, nobody has been able to prove any nonlinear lower bound on the combinational complexity of a specific Boolean function in spite of the well-known fact that almost all Boolean functions of n variables have exponential combinational complexity [7], and we are not able to prove any higher than a quadratic lower bound on the time complexity of Turing machines recognizing a specific language in NP [6, 14, 16]. Because of this unpleasant situation there is much endeavour now to develop some new lower bound proof techniques or to improve the old ones because several computer scientists believe that the way of the gradual development of proof techniques may lead to the solution of the fundamental open problems (like P? NP) in the complexity theory.

Our paper represents also a contribution to the development of lower bound proof techniques. It is a continuation of our paper [9], where the *first technique for proving lower bounds on some complexity measures of alternating devices* was

AMS Subject Classification (1985): Primary 03D15, 68Q05, 68Q10, 68Q45.
Key words: Alternation, Tradeoffs of complexity measures, Lower bounds.

introduced. Since alternating devices represent some parallel computing models on the one hand, and on the other hand are a generalization of nondeterministic computing models, the lower bounds on the complexity measures of alternating machines are valid also for a wide scale of computing devices obtainable from alternating machines by various restrictions.

Our lower bound proof techniques belong among the so-called *combinatorial techniques* based on the following combinatorial considerations: Let M be a *machine recognizing a language* $L(M)$. First one has to find some "essential features" of each accepting computation of M that characterize the computation in some sense. The choice of the essential features is usually connected with the structure of the input words. Then an upper bound on the amount of the computing resources of the computer M is assumed in order to enumerate how many computations on the inputs of a fixed length differing in the essential features may exist. The aim is to show that the number of "essentially" different computations on words of a fixed length is smaller than the number of distinct words in $L(M)$ of the considered length. This fact is later used to construct an accepting computation on a word that does not belong to $L(M)$, which creates, a contradiction with the above assumed restriction on the amount of computing resources. The crucial point of this combinatorial technique is to find the "essential features" of computations, because it means to comprehend the essence of the structure of the considered computing problem. Also the subtlety of the choice of the essential features plays an important role, since the quality of the obtained lower bounds depends crucially on it.

One of the most used combinatorial techniques is the *crossing sequence technique* (see, for example [3, 4, 5, 6, 8–12, 17], where some steps in the development of this technique for various computing devices have been made). We have developed the crossing sequence technique for proving lower bounds on parallel processing in [9, 11]. The main contribution of this paper is to bring some new ideas in the procedures of finding the "essential features" of computations and in enumerating the number of essentially different computations. The consequence of our new considerations is the lower bound $\Omega(n^{1/3} \log_2 n)$ on the complexity measure $REVERSALS \cdot SPACE \cdot PARALLELISM$ for the recognition of a specific language. An important point is also that we prove this lower bound for a very general computing model, the so-called *k-head alternating machine* in order to obtain a lower bound valid for almost all computing models used in the computation theory. Note that we consider *alternation* in the same way as it was introduced in [2] and the *complexity measures of alternating devices* are considered as defined in [9, 11, 13, 15].

Before giving the formal definition of our machine model and its complexity measures let us describe the *k-head alternating machine*, $AM(k)$, in an informal way. *An* $AM(k)$ *consists of a separate input tape with k two-way read-only heads*, *and a countable state control.* The *countable set of states of the AM*(k) is partitioned into two disjoint sets K_E (*the set of existential states*) and K_U (*the set of universal states*) with the same meaning as in all alternating devices [2]. A *step of AM*(k) M is made according to the state of M and the k symbols read by the k heads on the input tape. Using this information M can branch the computation into a finite number of computations and independently, for each branch of the computation, change the state and the positions of the heads by 1. We give only one restriction on M, namely that there must be a constant d_M such that branching from any universal state of M is bounded by d_M.

Clearly, the *multihead alternating machines* (*MAM*s) include a large number of different types of computing models. For example, a *MAM* is the generalization of the *alternating multitape Turing machine* (*ATM*), *multihead automata*, *multipushdown* and *multicounter machines*, *RAM*s, *Kolmogorov–Uspensky machine*, etc. More precisely, our *MAM* includes all models with a constant number of two-way heads on the input tape and an arbitrary organization of the memory (in fact, *MAM* can see the whole contents of its memory and change it in every single step of the computation).

Let $\mathbf{N}$ denote the set of positive integers.

Definition 1. *A k-head alternating machine AM*(k) *is a* 8-*tuple* $M = (K, \Sigma, K_U, \delta, q_0, F, d, k)$, *where*

(1) *K is the nonempty, countable set of states* (*internal configurations*);
(2) *$q_0 \in K$ is the initial state;*
(3) *$K_U \subseteq K$ is the set of universal states, $K_E = K - K_U$ is the set of existential states;*
(4) *$F \subseteq K$ is the set of accepting states;*
(5) *Σ is a finite, nonempty set called the input alphabet, ¢ and $\$ \notin \Sigma$ are the endmarkers;*
(6) *$\delta \subseteq (K \times (\Sigma \cup \{¢, \$\})^k) \times (K \times \{-1, 0, 1\}^k)$ is the next-move relation, where -1, $+1$, and 0 denote the direction of the head move* (*left, right, stationary, respectively*); *for $((q, (a_1, \ldots, a_k)), (p, (\gamma_1, \gamma_2, \ldots, \gamma_k))) \in \delta$ the following is required: if $a_j \equiv ¢$ for some $j \in \{1, \ldots, k\}$, then $\gamma_j \in \{0, 1\}$, if $a_i \equiv \$$ for some $i \in \{1, \ldots, k\}$, then $\gamma_i \in \{-1; 0\}$;*
(7) *d is a positive integer such that, for $\forall q \in K_U$, $\forall x \in (\Sigma \cup \{¢, \$\})^k$, there exist at most d different tuples (p, α), where $p \in K$, $\alpha \in \{-1, 0, 1\}^k$, such that $((q, x), (p, \alpha)) \in \delta$.*

Definition 2. *A descriptional configuration of an AM*(k) *machine $M = (K, \Sigma, K_U, \delta, q_0, F, d, k)$ is any element $(w, q, (i_1, \ldots, i_k))$ from*

$$\Sigma^* \times K \times (\mathbf{N} \cup \{0\})^k$$

with $0 \leqslant i_j \leqslant |w| + 1$ for each $j \in \{1, \ldots, k\}$.

Informally, a descriptional configuration $(w, q, (i_1, i_2, \ldots, i_k))$ *describes the situation in which the* $AM(k)$ *is in the state* q, *has the word* w *on the input tape, and the* j*th head is on the* i_j*th position of the input tape involving* ¢w\$.

Definiton 3. *A configuration of an* $AM(k)$ $M = (K, \Sigma, K_U, \delta, q_0, F, d, k)$ *is an element from* $K \times (\mathbf{N} \cup \{0\})^k$. *For all* $x \in \Sigma^*$, $I_M(x) = (x, q_0, (0, 0, \ldots, 0))$ *is the initial descriptional configuration. We shall say that the descriptional configuration* $(x, q, (i_1, \ldots, i_k))$ *is universal, existential, and accepting, respectively, if* q *is a universal, existential, and accepting state, respectively.*

In what follows we define the notions "*step*" and "*computation*" of multihead alternating machines.

Definition 4. *Let* $M = (K, \Sigma, K_U, \delta, q_0, F, d, k)$ *be an* $AM(k)$. *Let* C *and* C *be two descriptional configurations. We shall say that* M *can go from* C *to* C' *in one step,* $C \vdash C'$, *if* C' *can be obtained from* C *by applying the next-move relation* δ. *A sequential computation of* M *on* x *is a sequence* $C_0 = I_M(x) \vdash C_1 \vdash \vdash \ldots \vdash C_m$, $m \geqslant 0$. *In what follows we shall often write* $C_0, C_1, \ldots, C_m$ *only.*

A computation (*computation tree* if we want to draw attention to the structure of the computation) *of* M *on a word* x is a finite, nonempty, labelled tree with the following properties:

(1) each node v of the tree is albelled by a descriptional configuration $l(v)$;
(2) if v is an internal node (a non-leaf) of the tree, $l(v)$ is universal, and $\{C \mid l(v) \vdash C\} - \{C_1, \ldots, C_m\}$, then v has exactly m children $u_1, \ldots, u_m$ such that $l(u_i) = C_i$;
(3) if v is an internal node of the tree and $l(v)$ is existential, then v has exactly one child u such that $l(v) \vdash l(u)$.

An accepting computation (*tree*) *of* M *on an input word* x is a computation (tree) whose root is labelled with $I_M(x)$ and whose leaves are all labelled with accepting descriptional configurations. We say that *M accepts x* if there is an accepting computation (tree) of M on input x. We define $L(M) = \{x \in \Sigma^* \mid M$ *accepts* $x\}$ as the language accepted by M.

In what follows we shall often consider the computation as a tree labelled by configurations instead of descriptional configurations. It will cause no confusion because it will be clear which input word is considered. For the recognition of different languages we shall define the notion "*prominent configuration*" according to the given language. If V is the set of prominent configurations, then we define, for each accepting computation, the *pattern of the accepting computation* as a tree U with the following properties:

(1) the root of U is the root of D;
(2) the rest nodes are the nodes of D labelled by the prominent configurations from V;

(3) the nodes u and v are connected by an edge in U iff D involves a path from u to v that involves no node labelled by a prominent configuration.

The notions prominent configuration and pattern are important for our lower bound technique because they are the formal representations of the above mentioned "essential features" of computations.

Now, let us define the *complexity measures for multihead alternating machines*. Let A be an $AM(k)$ accepting a language $L(A)$.

The space complexity of A is a function of the input word length $S_A(n) = \log_2(C_A(n))$, where $C_A(n)$ is the number of all different states (internal configurations) used in all accepting computations on words from $L(A) \cap \Sigma^n$. We note that the number of all configurations used in accepting computations on inputs with the length n can be at most $(n+2)^k C_A(n)$, where $(n+2)^k$ is the number of all different positions of the heads on the input tape.

For an accepting computation D of A we denote by $T_A(D)$ ($R_A(D)$) the *maximum number of steps (head reversals) performed in the sequential computations from the root of D to the leaves of D*. The *time* and *reversal complexity measure* respectively are defined in the usual way as the following function: $X_A(n) = \max\{X_A(D) \mid D$ *is an accepting computation on an input of the length* $n\}$, where $X \in \{T, R\}$.

The *parallel complexity measure* is defined as introduced in [9] for alternating devices. The definitions of a *similar complexity measure* called leaf-size can be found in [15]. Let $P_A(D)$ denote the *number of universal states in the accepting computation D*. Clearly, $P_A(D)$ is an upper bound on branchings in D. The *parallel complexity of A* is the function $P_A(n) = \max\{P_A(D) \mid D$ *is an accepting computation on an input of the length* $n\}$.

Let $\mathscr{R}$ denote the set of all positive, real numbers. For arbitrary functions f and g from $\mathbf{N}$ to $\mathscr{R}$, $f(n) \in \Omega(g(n))$ is equivalent to $\exists c \in \mathscr{R}$, $\exists m \in \mathbf{N}$, such that, for $\forall n \geqslant m$, $f(n) \geqslant cg(n)$, and $f(n) \in O(g(n))$ is equivalent to $\exists c \in \mathscr{R}$, $\exists m \in \mathbf{N}$ such that, for $\forall n \geqslant m$, $f(n) \leqslant cg(n)$. If $f(n) \in \Omega(g(n))$ and $f(n) \in O(g(n))$, then we shall write $f(n) = \Theta(g(n))$. The fact $\lim_{n \to \infty} f(n)/g(n) = 0$ will be denoted by $f(n) = o(g(n))$. The *cardinality* of a set K will be denoted by $|K|$. $\lfloor d \rfloor$ for a $d \in \mathscr{R}$ is the *greatest nonnegative integer m such that $d \geqslant m$*. If we write, for example, $REVERSALS \cdot SPACE \cdot PARALLELISM \in \Omega(n^a)$ in what follows, then it means that $R_A(n)\, S_A(n)\, P_A(n) \in \Omega(n^a)$ for each device A of the computing model considered.

Now, giving some restrictions on MAMs we define multihead deterministic and nondeterministic machines.

Definition 5. *Let $A = (K, \Sigma, K_U, \delta, q_0, F, d, k)$ be an $AM(k)$. We shall say that A is a k-head nondeterministic machine, $NM(k)$, if $K_U = \emptyset$. We shall say that A is a k-head deterministic machine, $DM(k)$, if the next-move relation is a function.*

The structure of this paper is as follows: In Section 2 we introduce the language L and prove $RSP \in \Omega(n^{1/3}/\log_2 n)$ for L. The immediate corollaries for a nondeterministic sequential model are formulated in Section 2 too. In Section 3 we shall prove the stronger lower bound $RSP \in \Omega(n^{(1+2\varepsilon)/3})$ for the case $SPACE \geqslant n^{\varepsilon}$ $(0 < \varepsilon < 1)$. The lower bounds for alternating and nondeterministic multihead finite automata are involved in Section 4. In Section 5 it is shown that the simulation of $AM(2)$ by an alternating machine with one two-way, read-only head and k two-way blind heads on the input tape can require more than the $n/\log_2 n$ times increase of the complexity RSP. Further, several results for multihead simple finite automata are established as consequences of this result.

2. The main theorem and its consequences

Let us consider the languages L_r, and $L_r(m)$ introduced by Yao and Rivest [17] for arbitrary positive integers r, and m: $L_r = \{w_1 c w_2 c \ldots c w_r + w_r c \ldots c w_2 c w_1 \mid w_i \in \{0, 1\}^*$ for $i = 1, 2, \ldots, r\}$, $L_r(m) = \{w_1 c w_2 c \ldots c w + w_r c \ldots c w_2 x w_1 \mid w_i \in \{0, 1\}^m$ for $i = 1, 2, \ldots, r\} \subsetneq L_r$. We put $R = \bigcup_{r \in \mathbf{N}} L$. and $L = \{x_1 \# x_2 \# \ldots \# x_k \# 0^j \mid k \geqslant 0,\ j \geqslant 0,\ x_i \in R$ for $i = 1, \ldots, k\}$. For some functions f and g from $\mathbf{N}$ to $\mathbf{N}$ such that $f^2(n) g(n) \leqslant n$ for all $n \in \mathbf{N}$, we define $L(f, g) = \{x_1 \# x_2 \# \ldots \# x_{f(n)} \# 0^j \mid n \geqslant 1$, for all $i \in \{1, 2, \ldots, f(n)\}$ $x \in L_{f(n)}(\lfloor g(n)/2 - 1 \rfloor)$ and, $j = n - f^2(n) \lfloor g(n)/2 \rfloor\} \subseteq L$. In what follows, for a computing device A, $L(A)$ denotes the language recognized by A.

Now, let us formulate our main theorem.

Theorem 1. *Let f, and g be functions from $\mathbf{N}$ to $\mathbf{N}$ such that $f(n) = g(n) = \lfloor n^{1/3} \rfloor$, and let A be an $AM(k)$ for a $k \in \mathbf{N}$. If $L(f, g) \subseteq L(A) \subseteq L$ then*

$$R_A(n)\, S_A(n)\, P_A(n) \in \Omega(n^{1/3}/\log_2 n).$$

Proof. First we note that we do not need two functions f and g in Theorem 1 because $f(n) = g(n)$. But, we shall use both f, and g for two reasons. First, it increases the readability of this proof, and secondly, the proof of Theorem 1 will be useful to prove some tradeoffs in Section 3, where languages $L(f, g)$ for different f and g will be considered.

To prove Theorem 1 we shall follow the idea of Yao and Rivest [17] used to fool one-way multihead finite automata. This idea was modified in [8, 12] to prove that some languages cannot be recognized by one-way deterministic multihead finite automata, and real-time two-way multihead finite automata. It led to the development of some stronger proof techniques used to fool more powerful devices as time-space restricted sequential computing models in [4], one-way alternating multihead finite automata with bounded parallelism in [9], and reversal-bounded two-way nondeterministic sensing multihead finite automata in [10]. Now, generalizing these proof techniques we fool parallel computing models with bounded *RSP*.

The proof is done by contradiction. We assume that there exists an $AM(k)$ A with $L(f, g) \subseteq L(A) \subseteq L$ and $R_A(n) S_A(n) P_A(n) \notin \Omega(n^{1/3}/\log_2 n)$. In what follows, we shall show that if $L(f, g) \subseteq L(A)$, then there is a word in $L(A) - L$, which will be a contradiction. Let $A = (K, \Sigma, K_U, \delta, q_0, F, d, k)$.

Since $R_A(n) S_A(n) P_A(n) \notin \Omega(n^{1/3}/\log_2 n)$ we can assume that there is a positive integer m with the property

$$(1) \qquad 27k^2 d R_A(m) P_A(m) S_A(m) < \lfloor m^{1/3}/\log_2 m \rfloor .$$

Let $L_m(f, g) = \{x \in L(f, g) \mid |x| = m\}$. Now, we start to determine the "essential features" of computations of A on the words in $L_m(f, g)$ in terms of prominent configurations. But we do not define the prominent configurations directly according to the structure of inputs as it was usually done (see, for example [8, 12, 17]). First we make some combinatorial considerations which help us to choose such important configurations that the number of different patterns of computations on words in $L_m(f, g)$ enumerated later will be the smallest possible.

Let $x = x_1 \# x_2 \# \dots \# x_{f(m)} \# 0^s$, where $x_j = w_{j1} c w_{j2} c \dots c w_{jf(m)} + w_{jf(m)} c \dots \dots c w_{j2} c w_{j1}$ for $j \in \{1, \dots, f(m)\}$, be a word in $L_m(f, g)$. We shall say that the twins of subwords w_{ij} of x are compared in an accepting computation D of A iff there exists a configuration in D such that one of the heads is positioned on the "first" twin w_{ij} of x, and another head is positioned on the "second" twin w_{ij} of x in this configuration.

Now, we prove an important fact for our considerations which claims that the machine A is not able to compare all twins of subwords of words in $L_m(f, g)$ *because* A *has not enough computing resources.*

Fact 1. *Let D be an accepting computation of A on an $x \in L_m(f, g)$. Then there exist positive integers b, $r \in \{1, 2, \dots, f(m)\}$ such that the twins of subwords w_{br} of x are not compared in D.*

The proof of Fact 1. Let $C = C_1 C_2 \dots C_z$ be a sequential computation of D from the initial configuration to an accepting one, where for $u = 1, 2, \dots, z$, C_u is the part of the sequential computation C which involves no head reversal. Clearly, $z \leqslant R_A(m) + 1$.

Let us first consider the twins of subwords which can be compared by pairs of heads moving in the opposite direction in a C_u. Each pair of heads moving in the opposite direction can compare at most the twins of subwords of one subword x_i of x (i.e. no pair of heads moving in the opposite direction can compare two twins of subwords w_{pq} and w_{vc} for $p \neq v$ in a C_u). Since the number of head pairs moving in the opposite direction in each C_u can be bounded by k^2 there are at most $k^2(R_A(m)+1)$ subwords x_i whose twins of subwords w_{ij} can be compared by the pairs of heads moving in the opposite direction in a sequential computation C. Realizing that the computation D involves at most $dP_A(m)$ sequential computations (from the initial configuration to an accepting one) we obtain that there are at most $k^2d(R_A(m)+1)\,P_A(m)$ subwords x_i whose twins of subwords w_{ij} can be compared by the pairs of heads moving in the opposite direction in the entire computation D.

Since (1) holds for m we obtain that

$$k^2d(R_A(m)+1)\,P_A(m) < \lfloor m^{1\,3} \rfloor /2 = f(m)/2, \quad \text{i.e.,} \tag{2}$$

there exists a natural number b such that no twins of subwords w_{bj} of x_b are compared by a pair of heads moving in the opposite direction in D.

Obviously, each s-tuple of heads moving in the same direction can compare at most $\binom{s}{2} < s^2$ twins w_{bj} of x_b in a part C_u (without reversal) of a sequential computation C of D (if a pair of heads is reading twins w_{bj} at the same point during the computation part C_u, then at no other time during the computation part C_u that pair of heads could read some other twins w_{bi}, where $j \neq i$). So, during the entire computation D the machine A can compare at most $dk^2(R_A(m)+1)\,P_A(m)$ twins w_{bj} of x_b. Realizing (2) we have that there exists a positive integer $r \in \{1, \ldots, f(m)\}$ such that the twins w_{br} of x are not compared in D.

The proof of Theorem 1 continued. In what follows we shall consider for any input word $x \in L_m(f, g)$ a fixed accepting computation D_x. Clearly, the number of words in $L_m(f, g)$ is

$$2^{f^2(m)(\lfloor g(m)\ 2 \rfloor - 1)}.$$

Following Fact 1 we obtain that there exist two positive integers p and q such that the twins w_{pq} are not compared in the accepting computations on at least

$$2^{f^2(m)\lfloor g(m)-2)\ 2\rfloor}/f^2(m)$$

different words in $L_m(f, g)$. Let $\bar{L}_m(f, g)$ denote the set of such words.

Now, let us define the notion "prominent configuration" according to the fixed numbers p and q. A prominent configuration is a configuration of the accepting computation D_x on $x \in L_m(f, g)$ from which A moves one of its heads on the symbol c immediately preceeding or following the subwords w_{pq}.

Now, for any $x \in L_m(f, g)$, $\bar{D}_x$ be the pattern of x defined as the pattern of the accepting computation D_x on x according to the above defined prominent configurations.

Fact 2. *The number of all different patterns of the words in $\bar{L}_m(f, g)$ is bounded by*

$$e(m) = 2^{(k \log_2 m + S_A(m)) 4dk R_A(m) P_A(m)}.$$

The proof of Fact 2. Each pattern can be transformed to a sequence containing the concatenation of all (at most $dP_A(m)$) paths from the root of the pattern to the leaves of the pattern. We note that having such a sequence of prominent configurations we can unambiguously construct the original pattern.

Since there are at most $4k$ prominent configurations in each part of computations without reversals the length of every sequence corresponding to a pattern is bounded by $4kdR_A(m) P_A(m)$. Realizing that the number of all different, prominent configurations is bounded by $(2 + m)^k C_A(m) \leqslant 2^{(k \log_2 (m+2)) + S_A(m)}$ the proof of Fact 2 is completed.

The proof of Theorem 1 continued. Following (1) we obtain $e(m) < |\bar{L}_m(f, g)| - 1$. It thus follows that, for $v_1 \neq v_2$, $z_1 = x_1 \# x_2 \# \ldots \# \# x_{p-1} \#$, and $z_2 = \# x_{p+1} \# \ldots \# x_{f(m)} \# 0^s$, there are two different words

$$y_1 = z_1 w_{p1} c \ldots c w_{pq-1} c v_1 c \ldots c w_{pf(m)} + w_{pf(m)} c \ldots c v_1 c w_{pq-1} c \ldots c w_{p1} z_2$$

$$y_2 = z_1 w_{p1} c \ldots c w_{pq-1} c v_2 c \ldots c w_{pf(m)} + w_{pf(m)} c \ldots c v_2 c w_{pq-1} c \ldots c w_{p1} z_2$$

in $L_m(f, g)$ with the same pattern X of the accepting computations D_{y_1} and D_{y_2} resp. in which the twins v_1 of y_1 and the twins v_2 of y_2 resp. are not compared.

Now, we shall construct an accepting computation of A on the word

$$y = z_1 w_{p1} c \ldots c w_{pq-1} c v_1 c \ldots c w_{pf(m)} + w_{pf(m)} c \ldots c v_2 c w_{pq-1} c \ldots c w_{p1} z_2 .$$

Since y does not belong to L the proof of Theorem 1 will be completed.

The construction of an accepting computation (tree) on y is based on the fact that during the computation on the words y_1 and y_2 the $AM(k)$ A did not read the twins of subwords v_i in y_i at the same time. Let us construct an accepting computation on y from the pattern X in the following way. For each node u in the pattern X, let X_u^1, and X_u^2 resp. be the subtrees of the accepting computations of D_{y_1}, and D_{y_2} resp. from u (i.e. with the root u) to the prominent configurations in which an edge leads from u in X. Then, for every node u in X, we replace the node u with the edges leading from u by one of the subtrees X_u^1, X_u^2. The determination which of X_u^1, X_u^2 is chosen is given below.

If some head is positioned on the word v_1, then X_u^1 is chosen. If some head is positioned on v_2, then X_u^2 is chosen. We have already shown that the situation in which one of the heads is positioned on v_1 and another head on v_2 does not occur. In the case when none of the heads is positioned on v_1 or v_2, it is not important which X_u^i we choose.

So, we have shown that if A accepts all words in $L(f, g)$, then it has to accept a word not in L which proves Theorem 1. □

Corollary 1. *Let A be an $AM(k)$, for a $k \in \mathbf{N}$, such that $L(A) = L$, and $S_A(n) \geqslant c \log_2 n$ for a constant c. Then*

$$R_A(n)\, S_A(n)\, P_A(n) \in \Omega(n^{1/3}).$$

Proof. The proof is the same as that of Theorem 1. The new assumption $S_A(n) \geqslant c \log_2 n$ is used to show that

$$(k \log_2 n + S_A(n))\, 4kdR_A(n)\, P_A(n) \in O(S_A(n)\, R_A(n)\, P_A(n)).$$

All other considerations of the proof of Theorem 1 hold without any change. □

Now, let us consider the same nondeterministic machine $NM(k)$, for a $k \in \mathbf{N}$, as Ďuriš and Galil in [4]. Obviously, a $NM(k)$ is an $AM(k)$ having no universal state. So the following theorem is an immediate consequence of Theorem 1, and Corollary 1.

Theorem 2. *Let A be an $NM(k)$, for a $k \in \mathbf{N}$, such that $L(\lfloor n^{1/3} \rfloor, \lfloor n^{1/3} \rfloor) \subseteq$ $\subseteq L(A) \subseteq L$. Then $R_A(n)\, S_A(n) \in \Omega(n^{1/3}/\log_2 n)$. In the case that $S_A(n) \geqslant c \log_2 n$, for a constant c, $R_A(n)\, S_A(n) \in \Omega(n^{1/3})$.*

We have proved the lower bound $\Omega(n^{1/3}/\log_2 n)$ on the complexity measure RSP for the recognition of the language L. We are not able to prove any tight upper bound to this lower bound, and we believe that no such upper bound exists. We conjecture that the recognition of L requires still more computing resources, and so our lower bound can be improved. We make some improvements in the following section only by adding some additional assumptions. Thus, to prove a higher lower bound on RSP of the recognition of L remains as the main open problem left in our paper.

3. Improved lower bound for polynomial space

The aim of this section is to improve the lower bound obtained in Theorem 1 in the case when our computing model uses at least polynomial space.

Theorem 3. *Let ε be a real number such that $0 < \varepsilon < 1$. Let f_ε and g_ε be functions from $\mathbf{N}$ to $\mathbf{N}$ such that $f_\varepsilon(n) = \lfloor n^{(1-\varepsilon)/3} \rfloor$, and $g_\varepsilon(n) = \lfloor n^{(1+2\varepsilon)/3} \rfloor$. Let V_ε be a language such that $L(f_\varepsilon, g_\varepsilon) \subseteq V_\varepsilon \subseteq L$. Let A be an $AM(k)$, for a $k \in \mathbf{N}$, such that $L(A) = V_\varepsilon$, and $S_A(n) \geqslant n^\varepsilon$. Then*

$$R_A(n)\, S_A(n)\, P_A(n) = \Omega(n^{(1+2\varepsilon)/3}).$$

Proof. We prove Theorem 3 following the proof of Theorem 1. Let $L_n(f_\varepsilon, g_\varepsilon) = \{x \in L(f_\varepsilon, g_\varepsilon) \mid |x| = n\}$, $S_A(n) \geqslant n^\varepsilon$, and $R_A(n) S_A(n) P_A(n) \notin \Omega(n^{(1+2\varepsilon)3})$. Since, for sufficiently large n,

$$k^2 d(R_A(n) + 1) P_A(n) \leqslant 2k^2 d R_A(n) S_A(n) P_A(n)/n^\varepsilon < \lfloor n^{(1-\varepsilon)3} \rfloor = f_\varepsilon(n)$$

(see the inequality (2) of Fact 1) Fact 1 of the proof of Theorem 1 holds in this proof too.

Let the notions "prominent configurations" and "pattern" be defined in the same way as in the proof of Theorem 1. So, following Fact 2 the number of all different patterns of the words in $\bar{L}_n(f_\varepsilon, g_\varepsilon)$ is bounded by

$$e(n) = 2^{4dkS_A(n) R_A(n) P_A(n)}$$

for sufficiently large numbers $n \in \mathbf{N}$. Since $4dkS_A(n) R_A(n) P_A(n) \notin \Omega(n^{(1+2\varepsilon)3}) = \Omega(g(n))$ we obtain that

$$e(n) < |\bar{L}_n(f_\varepsilon, g_\varepsilon)| - 1.$$

It implies that there are two different words $y_1, y_2 \in \bar{L}_n(f_\varepsilon, g_\varepsilon)$ differing only in some twins of subwords of the length $\lfloor (g(n) - 2)/2 \rfloor$ which have the same pattern. So, the proof of Theorem 3 can be completed in the same way as the proof of Theorem 1. □

Now, we apply the assertion of Theorem 3 to $NM(k)$'s in order to obtain a polynomial lower bound on the number of reversals for fixed space.

Theorem 4. *Let A be a $NM(k)$, for a $k \in \mathbf{N}$, such that $L(A) = V_\varepsilon$, and $S_A(n) \geqslant n^\varepsilon$ for sufficiently large numbers $n \in \mathbf{N}$. Then*

$$R_A(n) S_A(n) \in \Omega(n^{(1+2\varepsilon)3}).$$

Corollary 2. *Let A be a $NM(k)$, for a $k \in \mathbf{N}$, such that $L(A) = V_\varepsilon$, and $S_A(n) = \Theta(n^\varepsilon)$. Then*

$$R_A(n) \in \Omega(n^{(1-\varepsilon)3}).$$

4. Lower bounds for multihead finite automata

Multihead finite automata are computation devices which have no additional working space, i.e. MAMs with constant space. Let, for any $k \in \mathbf{N}$, $2AFA(k)$, $2NFA(k)$, and $2DFA(k)$, resp., denote the class of *two-way k-head alternating, nondeterministic* and *deterministic*, resp. *multihead finite automata*. Let $1AFA(k)$, $1NFA(k)$, and $1DFA(k)$, resp., be the class of *one-way versions* (i.e. without any reversal in the computations) of $2AFA(k)$, $2NFA(k)$, and $2DFA(k)$, respectively. Let, for a class of devices M, $\mathscr{L}(M)$ be the *family of languages recognized by* $A \in M$.

Multihead finite automata were extensively studied for several reasons (see for example [4, 8, 9, 10, 11, 12, 13, 15, 17, 18]). One of the most important properties of them according to the complexity theory is that they characterize the basic complexity classes [13] in the following way:

$$P - \bigcup_{k \in \mathbf{N}} \mathscr{L}(2AFA(k)), \quad DLOG = \bigcup_{k \in \mathbf{N}} \mathscr{L}(2DFA(k)),$$

and

$$NLOG = \bigcup_{k \in \mathbf{N}} \mathscr{L}(2NFA(k)),$$

where *P is the family of languages recognized by deterministic Turing machines in polynomial time*, and *DLOG* (*NLOG*) *is the family of languages recognized by deterministic* (*nondeterministic*) *Turing machines with logarithmic space*. The basic open problems concerning the relation between these complexity classes can be formulated as equivalent problems in the terms of the family of languages defined by multihead finite automata. So, it is interesting to study different complexity measures for these computing devices. In the following theorem (which is an immediate consequence of Theorem 1), we give the first, nontrivial lower bound for the complexity measure *REVERSALS·PARALLELISM* for two-way alternating multihead finite automata.

Theorem 5. *Let* $A \in \bigcup_{k \in \mathbf{N}} 2AFA(k)$, *and let* $L(A) = L$. *Then* $R_A(n)\, P_A(n) \in \Omega(n^{1\,3}/\log_2 n)$.

Corollary 3. *Let* $A \in \bigcup_{k \in \mathbf{N}} 1AFA(k)$, *and let* $L(A) = L$. *Then* $P_A(n) \in \Omega(n^{1\,3}\ \log_2 n)$.

Corollary 4. *Let* $A \in \bigcup_{k \in \mathbf{N}} 2NFA(k)$, *and let* $L(A) = L$. *Then* $R_A(n) \in \Omega(n^{1\,3}\ \log_2 n)$.

Corollary 5. $L \notin \bigcup_{k \in \mathbf{N}} \mathscr{L}(1NFA(k))$.

We note that, for the language R, a stronger lower bound $P_A(n) \in \Omega((n \log_2 n)^{1\,2})$ for one-way alternating multihead finite automata is proved in [9]. A similar result as in Corollary 4 is established in [10] too.

Now, let us consider the language $L^C = \{0, 1, c, +, \#\}^* - L$. It is no problem to construct an automaton $B \in 1NFA(3)$ which, guessing the twins of different subwords (or guessing that the form of the input word is different from the form of words in L), recognizes the language L^C. Denoting by $M - R(f) - P(g)$, for each automaton class M and functions f, g from $\mathbf{N}$ to $\mathbf{N}$, the *automaton subclass of* M *such that* $B \in M$ *iff* B *uses in its accepting computations on words of the length* n *at most* $f(n)$ *reversals and* $g(n)$ *universal configurations* we can formulate the following consequences of Theorem 5.

Corollary 6. *Let f, and g be some functions from* $\mathbf{N}$ *to* $\mathbf{N}$ *such that* $f(n)g(n) \in \Omega(n^{1/3}/\log_2 n)$. *Then, for every integer* $k \geq 3$, *the families of languages* $\mathscr{L}(2AFA(k) - R(f) - P(g))$, $\mathscr{L}(1AFA(k) - P(g))$, $\mathscr{L}(1NFA(k))$, *and* $\mathscr{L}(2NFA(k) - R(f))$ *are not closed under complementation.*

In the case that a class $\mathscr{L}(2DFA(k) - R(f))$, for a function $f(n) \leq n^{1/3}/\log_2 n$ is closed under complementation (for example if $f(n) = c$ for a constant $c \in \mathbf{N}$) one can obtain a separation result between nondeterminism and determinism for reversal-bounded two-way multihead finite automata.

5. Two read-only heads versus one read-only head and k blind heads

Cobham [3] proved that a sequential computing model with one read-only head on the input tape recognizing the language $L_R = \{wcw^R \mid w \in \{0, 1\}^*\}$ has $TIME \cdot SPACE \in \Omega(n^2)$. The fact that the same sequential model with two read-only heads on the input tape can recognize L_R with $TIME \cdot SPACE \in O(n)$ implies that one read-only head cannot be compensated for by $o(n)$ increase of $TIME \cdot SPACE$. In [11] it is shown that the recognition of $L' = \{w2^iw \mid i \geq 1, w \in \{0, 1\}^*\}$ by an $AM(1)$ requires $TIME \cdot SPACE \cdot PARALLELISM \in \Omega(n^2)$. So, one read-only head on the input tape cannot be compensated for by $o(n)$ increase of $TIME \cdot SPACE \cdot PARALLELISM$.

Now, we shall show that one read-only head on the input tape cannot be compensated by $o(n/\log_2 n)$ increase of $REVERSALS \cdot SPACE \cdot PARALLELISM$, and k blind heads on the input tape (a blind head recognizes only the endmarkers on the input tape, see [15] for details). To prove this result we shall consider the language $L_1 = \{w + w \mid w \in \{0, 1\}^*\}$ which belongs to $\mathscr{L}(1DFA(2))$. *The parallel computing model* $AM(1)$ *with k additional blind heads on the input will be denoted by* $AM(1, k)$.

Theorem 6. *Let k be a natural number, and let A be an* $AM(1, k)$ *recognizing* L_1. *Then*

$$R_A(n)\, S_A(n)\, P_A(n) \in \Omega(n/\log_2 n).$$

Proof. Let us follow the proof of Theorem 1 assuming that $R_A(n)\, S_A(n) \cdot P_A(n) \notin \Omega(n^2/\log_2 n)$, and $L(A) = L_1$. Clearly, A cannot compare the twins of subwords w of an input word $w + w$ in its computation because A has only one reading head on the input tape. Let, for each $w \in \{0, 1\}^*$, D_w be a fixed accepting computation of A on $w + w$. The prominent configuration of D_w is the initial configuration, and every configuration C in which the reading head is adjusted on $+$, and one of the following conditions holds.

(i) The reading head crossed the "first" twin w in the sequential computation between the immediately preceding prominent configuration of C and C, and in the following step the reading head will be adjusted on the first symbol of the "second" twin w.

(ii) The reading head crossed the "second" twin w in the sequential computation between the immediately preceding prominent configuration of C and C, and in the following step the reading head will be adjusted on the last symbol of the "first" twin w.

So, at least one reversal of the reading head has to be done between two prominent configurations (different from the initial configuration).

Let, for odd, positive integers n, $L_1(n) = \{x \in L_1 \mid |x| = n\}$, and let the pattern of a word in L_1 be defined according to the above specified prominent configurations. Following Fact 2 we see that the number of different patterns of words in $L_1(n)$ is bounded by

$$(n^k C_A(n))^{dR_A(n) P_A(n)} \leqslant 2^{d(k \log_2 n + S_A(n)) R_A(n) P_A(n)},$$

where *d is the maximal possible branching from a universal state.*

Since $d(k \log_2 n + S_A(n)) R_A(n) P_A(n) \notin \Omega(n)$, and the number of different words in $L_1(n)$ is $2^{(n-1)/2}$ there are, for $w_1 \neq w_2$, two words $w_1 + w_1$ and $w_2 + w_2$ with the same pattern. Obviously, the proof can be completed in the same way as the proof of Theorem 1. □

Corollary 7. *Let A be an $AM(1)$ recognizing L_1. Then*

$$R_A(n) S_A(n) P_A(n) \in \Omega(n).$$

Proof. The proof is the same as the proof of Theorem 6. It suffices to put $k = 0$, which implies that the number of patterns is bounded by $2^{dS_A(n) R_A(n) P_A(n)}$. □

Concluding this paper we call attention to some consequences of Theorem 6 concerning the multihead simple finite automata. Let $2ASFA(k)$ be the class of two-way k-head simple multihead finite automata (k-head denote one reading and $k-1$ blind heads). Analogously, we shall consider the classes $1ASFA(k)$, $2NSFA(k)$.

In [9, 15] it is proved that $\mathscr{L}(1ASFA(k)) = \mathscr{L}(1AFA(k))$, and $\mathscr{L}(2ASFA(k)) = \mathscr{L}(2AFA(k))$. But Hromkovič [9] stated that the simulation of a $B \in 1AFA(k)$ by a $C \in 1ASFA(k)$ can require $n/\log_2 n$ increase of parallel complexity. Here, we obtain a more general result.

Theorem 7. *For functions f, and g from* **N** *to* **N** *such that $f(n) g(n) = o(n \log_2 n)$,*

$$\mathscr{L}(1DFA(2)) - \mathscr{L}\left(\bigcup_{k \in \mathbf{N}} 2ASFA(k) - R(f) - P(g)\right) \neq \emptyset.$$

Corollary 8. *For every function f from* $\mathbf{N}$ *to* $\mathbf{N}$ *such that* $f(n) = o(n/\log_2 n)$:

$$\mathscr{L}(1DFA(2)) - \mathscr{L}\Big(\bigcup_{k \in \mathbf{N}} 1ASFA(k) - P(f)\Big) \neq \emptyset.$$

$$\mathscr{L}(1DFA(2)) - \mathscr{L}\Big(\bigcup_{k \in \mathbf{N}} 2NSFA(k) - R(f)\Big) \neq \emptyset.$$

King [13] formulated some open problems concerning the fact whether two-way alternating (simple) multihead finite automata are more powerful than one-way ones. We are not able to solve this problem but for parallel-bounded versions we can separate these classes.

Corollary 9. *Let $k \geqslant 3$ be an integer. Let f and g be functions from* $\mathbf{N}$ *to* $\mathbf{N}$ *such that* $f(n) = o(n^{1\,3}/\log_2 n)$, $g(n) = o(n/\log_2 n)$. *Then*

(a) $$\mathscr{L}(1AFA(k) - P(f)) \subsetneq \mathscr{L}(2AFA(k) - P(f))$$

(b) $$\mathscr{L}(1ASFA(k) - P(g)) \subsetneq \mathscr{L}(2ASFA(k) - P(g)).$$

Proof. The assertion (a) is the immediate consequence of Corollary 3 and of the fact $L \in \mathscr{L}(2DFA(2)) \subseteq \mathscr{L}(2AFA(k) - P(f))$. The assertion (b) is the consequence of Corollary 8 and of the fact $L_1 \in \mathscr{L}(2DSFA(3)) \subseteq \subseteq \mathscr{L}(2ASFA(k) - P(g))$. □

We note that using other languages similar results as in Corollary 9 were obtained in [9] too.

It is simple to see that $(L_1)^C = \{0, 1, +\}^* - L_1$ belongs to $\mathscr{L}(1NFA(2))$. So, the reader can formulate for some families of languages that they are not closed under complementation.

REFERENCES

[1] BORODIN, A. B. COOK, S. A.: A time-space tradeoff for sorting on a general model of computation. In: Proc. 12th Annual ACM STOC, Los Angeles, ACM 1980, pp. 294 301.

[2] CHANDRA, A. K. KOZEN, D. C. STOCKMEYER, L. J.: Alternation. J. ACM, 28, 1981, 114—133.

[3] COBHAM, A.: The recognition for perfect squares. In: Proc. 7th Annual IEEE Symp. on SWAT, Berkeley 1966, pp. 78 87.

[4] ĎURIŠ, P. GALIL, Z.: A time-space tradeoff for language recognition. Math. Systems Theory, 17, 1984, 3 12.

[5] ĎURIŠ, P. GALIL, Z. PAUL, W. REISCHUK, R.: Two nonlinear lower bounds. In: Proc. 15th Annual ACM STOC, ACM 1983, pp. 127 132.

[6] FREIVALDS, R.: Quadratic lower bound for nondeterministic Turing machines. Unpublished communication at the 11th MFCS '84, Prague 1984.

[7] LUPANOV, O. B.: Ob odnom metode sinteza schem. (Russian.) Izv. Vuzov Radiofiz., 1, 1958, 120 140.

[8] HROMKOVIČ, J.: One-way multihead deterministic finite automata. Acta Inform., 19, 1983, 377 384.

[9] HROMKOVIČ, J.: On the power of alternation in automata theory. J. Comput. Syst. Sci., 31, 1985, 28 39.

[10] HROMKOVIČ, J.: Fooling a two-way multihead automaton with reversal number restriction Acta Inform., 22, 1985, 589 594.

[11] HROMKOVIČ, J.: Tradeoffs for language recognition on alternating machines. Theor. Comput. Sci., 63, 1989, 203 221.

[12] JANIGA, L.: Real-time computations of two-way multihead finite automata. In: Proc. FCT '79, ed. L. Budach. Academic Verlag, Berlin 1979, pp. 214 219.

[13] KING, K. N.: Alternating multihead finite automata. In: Proc. 8th ICALP '81. Lecture Notes in Computer Science 115. Springer-Verlag, Berlin 1981, pp 506 520.

[14] MAASS, W.: Quadratic lower bounds for deterministic and nondeterministic one-tape Turing machines. In: Proc. 16th Annual ACM STOC, ACM 1984, pp. 401—408.

[15] MATSUNO, H. INOUE, K. TANIGUCHI, H. TAKANAMI, I.: Alternating simple multihead finite automata Theor. Comput. Sci., 36, 1985, 299 308.

[16] LI, M.: On one tape versus two stacks. Technical Report, 84 591, January 1984, Dept. of Computer Science, Cornell University, Ithaca, New York.

[17] RIVEST, R. L. YAO, A. C.: $k + 1$ heads are better than k. J. ACM, 25, 1978, 337 340.

[18] SUDBOROUGH, I. H.: Bounded-reversal multihead finite automata languages. Informat. Control, 25, 1974, 317 328

Received July 3, 1987

Katedra teoretickej kybernetiky
Matematicko-fyzikálna fakulta
Univerzita Komenského
842 15 Bratislava