Theoretical Computer Science 88 (1991) 297–311
Elsevier

Separation of deterministic, nondeterministic and alternating complexity classes

Andrej Bebják and Ivana Štefáneková

Department of Theoretical Cybernetics, Comenius University, 842 15 Bratislava, Czechoslovakia

Communicated by G. Mirkowska
Received May 1988
Revised September 1989

Abstract

Bebják, A. and I. Štefáneková, Separation of deterministic, nondeterministic and alternating complexity classes, Theoretical Computer Science 88 (1991) 297–311.

In this paper nondeterministic and deterministic complexity classes as well as alternating and nondeterministic complexity classes for one-head and multihead Turing machines are separated, while as a complexity measure the combined complexity measure – the product of time and space – is considered. Further, hierarchies are stated within a given type of the computation model according to some complexity measures.

1. Introduction

The question of the nature of the relationship between nondeterminism and determinism, and between alternation and nondeterminism is one of the most investigated questions in complexity theory. In this work we separate nondeterminism from alternation and determinism from nondeterminism for one-head and multihead Turing machines (TM). In doing so, combined complexity measure – the product of time and space – is considered as a complexity measure.

Namely, for one-head TM we shall prove that

$$(\text{NTM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{DTM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset,$$

$$(\text{ATM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{NTM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset,$$

where DTM(1)-TIME $\times$ SPACE($f(n)$), NTM(1)-TIME $\times$ SPACE($f(n)$), ATM(1)-TIME $\times$ SPACE($f(n)$) are the families of languages recognized by one-head deterministic, nondeterministic and alternating Turing machines, respectively, for which the product of the time and space complexity is at most $c \cdot f(n)$.

The consequence of this assertion is the separation of the classes DLOG, NLOG and ALOG with regard to the time complexity

$$(\text{NLOG-TIME}(n)) - (\text{DLOG-TIME}(o(n^2/\log n))) \neq \emptyset$$

$$(\text{ALOG-TIME}(n)) - (\text{NLOG-TIME}(o(n^2/\log n))) \neq \emptyset.$$

It follows from these facts that even if DLOG = NLOG (NLOG = ALOG), there exists a language for which deterministic TM (nondeterministic TM) working in the logarithmic space has to use a substantially larger time than nondeterministic (alternating) TM working in the same space.

In the second part of this paper some hierarchies within one type of model are proved. The main result obtained is the following. Let M be one of the DTM(1), NTM(1), ATM(1) models, let $a(n)$ be a computable function and let $b(n)$ be a function such that $b(n) = o(a(n)^2)$. Then,

$$(M\text{-TIME} \times \text{SPACE}(a^2(n))) - (M\text{-TIME} \times \text{SPACE}(b(n))) \neq \emptyset.$$

Again, similar results are obtained for multihead TMs.

This paper is organized as follows. Section 2 involves some basic definitions. Hierarchies for one-head computation models are given in Section 3.1, and those for multihead computation models in Section 3.2. Results concerning hierarchies valid within one computation model are formulated for one-head models in Section 4.1 and for multihead models in Section 4.2.

2. Definitions

2.1. Computation models

Hierarchies for deterministic, nondeterministic and alternating computation models are obtained by proving particular lower and upper bounds of the time–space complexity for specific languages. Rudiments of lower bound proof technique appear in [6] and have thereafter, been used in [2, 3, 5] for automata and in [1, 4] for TMs. The idea of the proof is based on the interchange of certain substrings of the input string which the given model under certain assumptions is not able to distinguish. While doing so, upper bounds are stated for a TM model but, on the other hand, lower bounds are obtained for a computation model which is more general than the TM one. It enables us to prove the hierarchies for a wide range of computation models. We describe this general model only informally. Exact definition can be found in [4].

A k-heads alternating machine AM(k) consists of an input tape and a state control unit. Read-only heads can move on the input tape in both directions by one field in each step. The control unit is not a finite-state one, but it can contain an infinite number of states (internal configurations) which are divided into two disjoint sets K_E (existentional states) and K_U (universal states) with generally used meaning as in alternating models. One computational step of AM(k) depends on a k-tuple of symbols read by the heads and on the internal configuration. According to this information, AM(k) can branch its computation on a finite number of parallel computations changing its state and moving its heads. We require the existence of a constant $d \in N$ such that the branching from universal state is bounded from above by this constant.

By adding certain bounds to alternating machines we obtain some other models — k-heads nondeterministic machines NM(k) if $K_U = \emptyset$, and k-heads deterministic machines if $K_U = \emptyset$ and transition relation is a function.

Obviously, an alternating machine is a generalization of an alternating multitape Turing machine in the following sense.

(1) An alternating machine can have k heads on the input tape, a Turing machine can have only one.

(2) It has an arbitrary organization of memory, not just a linear one as a Turing machine has.

(3) One computational step of AM(k) depends on the whole content of the memory, while one computational step of a Turing machine depends only on a constant number of symbols scanned by the heads on the worktapes.

The state and the content of worktapes of TM form the internal configuration. Alternating, nondeterministic and deterministic multitape Turing machines are, thus, particular cases of AM(k), NM(k) and DM(k). We shall denote them as ATM(k), NTM(k) and DTM(k). Particular cases of k-heads machines with a finite state control are alternating, nondeterministic and deterministic finite automata (we shall denote them as AFA(k), NFA(k) and DFA(k), respectively).

2.2. Complexity measures

As complexity measures we consider the measures TIME, SPACE and PARALLELISM. Let A be a machine, then its space complexity $S_A(n) = \log_2(C_A(n))$, where $C_A(n)$ is the number of states used in accepting computations on words of the length n. The time complexity $T_A(n)$ and the parallel complexity $P_A(n)$ are defined as follows. For every accepting computation D of the machine A let $T_A(D)$ be equal to the maximal number of steps performed in the sequential computations of D. Then, $T_A(n) = \max\{T_A(D) | D \text{ is an accepting computation of } A \text{ on an input of length } n\}$. Further, let $P_A(D)$ be the number of universal configurations in the computation D. Then we define $P_A(n) = \max\{P_A(D) | D \text{ is an accepting computation of } A \text{ on an input of length } n\} + 1$. Let the values of $T_A(n)$, $S_A(n)$ and $P_A(n)$ be undefined if there is no

word of length n accepted by A. For nondeterministic and deterministic computations the parallel complexity is equal to 1.

Besides these measures, we consider combined complexity measures TIME·SPACE·PARALLELISM for alternating models and TIME·SPACE for the other ones. Let us define M-TIME × SPACE × PAR$(f(n))$ as a family of languages recognized by those devices A of type M for which $T_A(n)S_A(n)P_A(n)=O(f(n))$ holds. The families M-TIME × SPACE$(f(n))$, M-TIME × PAR$(f(n))$ and M-TIME$(f(n))$ are defined in the same way.

The symbols O and Ω have the following meaning. Let N' be a subset of N. Let g,f be functions $g:N\to N$, $f:N'\to N$. Then, $f(n)=O(g(n))$ $(f(n)=\Omega(g(n)))$ iff $\exists a\in R^+$, $\exists m\in N$: $\forall n\in N'$, $n\geqslant m$, $f(n)\leqslant a\cdot g(n)$ $(f(n)\geqslant a\cdot g(n))$.

2.3. Additional notions

For a function $h:N\to N$ such that $h(n)\leqslant n$ and any functions $f:N\to N$, $g:N\to N$ let us define the concept of a computable function: The function $h(n)$ is called $(f(n), g(n))$-computable by a computation model M iff there is a machine A of type M such that:

(1) every computation of A stops in an accepting configuration K;

(2) in the configuration K the first reading head points to the $h(n)$th field on the input tape;

(3) $T_A(n)=O(f(n))$, $S_A(n)=O(g(n))$.

The computation will often be considered as a tree with vertices – the configurations. For the description of the computation on a word we shall use the notion of a prominent configuration, which is dependent on a particular language. If V is a set of prominent configurations, let us define a pattern of computation D regarding V as a tree U with these following properties:

(1) A source of U is a source of D.

(2) The other vertices of U are those ones from D which belong to V.

(3) Vertices u and v are joined by an edge in U iff there exists a path from u to v in the computation D involving no prominent configuration.

The lower and upper bounds are proved in this paper for the following languages over the alphabet $\{0,1,2\}$:

$$L=\{x2^m x \mid x\in\{0,1\}^m,\ m\geqslant 1\},$$

$$\mathrm{PL}=\{x_1 2^m x_2 \mid x_1, x_2\in\{0,1\}^m,\ m\geqslant 1,\ x_1\neq x_2\}.$$

Let f, g be functions $f, g:N\to N$; $f(n)=\lfloor n^{1/2}\rfloor$, $g(n)=\lfloor f(n)/2\rfloor$, then

$$S(f,g)=\left\{ x_1 2^{g(n)} x_2 2^{g(n)} \ldots 2^{g(n)} x_{f(n)} 2^{g(n)+z(n)} \,\middle|\, x_i\in\{0,1\}^{g(n)},\ n\geqslant 1,\ \sum_{i=1}^{f(n)} \oplus x_i = 0^{g(n)},\ z(n)=n-2f(n)g(n) \right\},$$

where

$$\sum_{i=1}^{r} \oplus x_i = ((x_{11} \oplus x_{21} \oplus \cdots \oplus x_{r1}) \cdots (x_{1m} \oplus x_{2m} \oplus \cdots \oplus x_{rm})),$$

$$x_i = x_{i1} x_{i2} \ldots x_{im} \quad \text{for} \quad 1 \leqslant i \leqslant r,$$

$$\mathrm{PS}(f, g) = \left\{ x_1 2^{g(n)} x_2 2^{g(n)} \ldots 2^{g(n)} x_{f(n)} 2^{g(n)+z(n)} \,\middle|\, x_i \in \{0,1\}^{g(n)}, \right.$$

$$\left. n \geqslant 1, \sum_{i=1}^{f(n)} \oplus x_i \neq 0^{g(n)}, z(n) = n - 2f(n)g(n) \right\}.$$

3. Hierarchies for deterministic, nondeterministic and alternating Turing machines

3.1. Hierarchies for one-head devices

In this part we specify the hierarchies for deterministic, nondeterministic and alternating one-head (Turing) machines. As a special case of this hierarchy we obtain a hierarchy for the families DLOG, NLOG and ALOG.

In the first place the relation between deterministic and nondeterministic devices is formulated. For the language PL the lower bound $\Omega(n^2)$ of TIME·SPACE complexity on DM(1) machines and the upper bound $O(n \cdot \log n)$ on NTM(1) are proved.

Lemma 3.1. *Let A be a* DM(1) *such that* $L(A) = \mathrm{PL}$. *Then*

$$T_A(n) S_A(n) = \Omega(n^2).$$

Proof. Let A be a DM(1), $L(A) = \mathrm{PL}$ and let $T_A(n)S_A(n) = \Omega(n^2)$ be not true, i.e. $\forall a \in R^+$, $\forall m \in N$: $\exists s \geqslant m$, $T_A(s)S_A(s) < as^2$. We show that there is a word from *PL* rejected by A. Let us consider the set $L_n = \{x 2^m x \mid x \in \{0,1\}^m,\ 3m = n\}$ for each n divisible by 3. For the rejecting computation D_y on a word $y \in L_n$ (it is the only one) we define a prominent configuration as follows:

(1) The initial configuration is a prominent configuration.

(2) The configuration K, in which the reading head reads the first or the last symbol 2 from the subword 2^m, is a prominent configuration iff the reading head had run over the whole subword 2^m between the immediately preceding prominent configuration and the configuration K.

Let us denote as V a set of prominent configurations reached by A during the computation D_y in time $\leqslant T_A(n)$. If the length of D_y is greater than $T_A(n)$ (it is possible because D_y is a rejecting computation), we add to the set V the first prominent configuration reached by A during D_y in time $> T_A(n)$ (if such exists). As the pattern of the word y we consider the pattern of the computation D_y regarding V.

Proposition 3.2. *For every $n \in N$ the number of different patterns of words from L_n is limited by $2^{8T_A(n)S_A(n)/n}$.*

Proof of Proposition 3.2. For the cardinality of V it holds that $|V| \leqslant 3T_A(n)/n$ since between two prominent configurations A has to do at least m steps. The pattern of the computation D_y involves at most $|V|+1$ prominent configurations. The number of prominent configurations can be limited by the number $2^{S_A(n)}$. If any other configuration is used in the rejecting computation, the computation can be stopped. Thus, the number of all patterns of words from L_n is $2^{S_A(n)}+(2^{S_A(n)})^2+\cdots+(2^{S_A(n)})^{((3T_A(n)/n)+1)} \leqslant 2^{8T_A(n)S_A(n)/n}$. □

Proof of Lemma 3.1 (*conclusion*). The number of words from L_n is $2^{n/3}$. According to the assumption, there exists a number s such that $S_A(s)T_A(s)<s^2/24$, i.e. $2^{8T_A(s)S_A(s)/s}<2^{s/3}$. Hence, there are two different words $v=x2^{s/3}x$ and $v'=x'2^{s/3}x'$ from L_s having the same pattern P. We show that there is a word from PL rejected by A. We distinguish two cases:

(1) The last prominent configuration K of the pattern P is reached in time $\leqslant T_A(s)$ by at least one of $D_v, D_{v'}$ (for instance by D_v). Let the position of the head in K be $s/3+1$. Let us consider the computation on a word $w=x2^{s/3}x'$. According to the determinism of the machine A, the following assertion is valid. All prominent configurations of the pattern P are reached in D_w by A in the same order. Further, let K_1 and K_2 be two prominent configurations of the pattern joined by an edge in P. Let the position of the head in the configuration K_1 be $s/3+1$ (or 1 if K_1 is initial) $[2s/3]$, and in the configuration K_2, $2s/3$ $[s/3+1]$. Then, the part of D_w between K_1 and K_2 is identical with the part D_v $[D_{v'}]$ between K_1 and K_2. The computation D_w after reaching the last prominent configuration K of P is identical with D_v after K (in this part the symbols on the right-hand side from $2^{s/3}$ are not being read). Thus, the word $w \in$ PL is rejected by A (v was rejected). If the position of the head is $2s/3$ in K, the word $w'=x'2^{s/3}x$ from PL is rejected by A.

(2) The last prominent configuration K of the pattern P is reached by both computations D_v and $D_{v'}$ in time $>T_A(s)$. Let us consider deterministic computations on the words $w=x2^{s/3}x'$ and $w'=x'2^{s/3}x$. The assertion from the point (1) is valid for them too. Besides, $T_A(D_w(K))>T_A(s)$ or $T_A(D_{w'}(K))>T_A(s)$, where $T_A(D_z(K))$ is the time in which the computation D_z reached K,

$$2T_A(s)<T_A(D_v(K))+T_A(D_{v'}(K))=T_A(D_w(K))+T_A(D_{w'}(K)).$$

Let $T_A(D_w(K))>T_A(s)$. Then, even if the word w were accepted by A, the time greater than $T_A(s)$ would be necessary for its acceptance. □

Lemma 3.3. *There is a* NTM(1) *machine A such that $L(A)=$ PL and $T_A(n)S_A(n)=O(n\cdot\log n)$.*

Proof. In the first phase of its computation, A verifies whether an input word $y \in \{0,1\}^m \cdot 2^m \cdot \{0,1\}^m$ for some $m \in N$. It takes time $O(n)$ because the binary counting up to m, which is equivalent to the searching of binary tree with m vertices, can be performed within $O(n)$ steps and space $O(\log n)$. In the second phase, A decides nondeterministically which couple of the corresponding symbols is not equal. Time $O(n)$ and space $O(\log n)$ are sufficient for A to verify its decision.

To separate nondeterminism and alternation we consider the language L.

Lemma 3.4. *Let A be a* NM(1) *such that $L(A)=L$.* Then, $T_A(n)S_A(n)=\Omega(n^2)$.

The lower bound was proved by Hromkovič in [4].

Lemma 3.5. *There is an* ATM(1) *machine A such that $L(A)=L$ and $T_A(n)S_A(n)=O(n \cdot \log n)$.*

Proof. Let us informally describe the computation of A:

(1) the machine A verifies whether an input word $y \in \{0,1\}^m \cdot 2^m \cdot \{0,1\}^m$ for some $m \in N$;

(2) the machine A sets its head on the first symbol of the word y;

(3) the machine A branches the computation on two parallel computations;

(4) in the first one A moves the head H by one symbol to the right. If H reads symbol 2, then A goes into an accepting state, else it proceeds by point 3;

(5) in the second of the two parallel computations A remembers the read symbol and moves the head H by $2m$ symbols to the right. If the symbol just read is equal to the remembered one, then A goes into an accepting state.

Clearly, the time complexity of A is not greater than $O(n)$ and space $O(\log n)$. □

Theorem 3.6.

$$(\text{NM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{DM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset,$$

$$(\text{AM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{NM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset.$$

Theorem 3.7.

$$(\text{NTM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{DTM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset,$$

$$(\text{ATM(1)-TIME} \times \text{SPACE}(n \cdot \log n)) - (\text{NTM(1)-TIME} \times \text{SPACE}(o(n^2))) \neq \emptyset.$$

Proofs follow from the previous assertions.

For Turing machines working in the logarithmic space, we have the following consequence from Theorem 3.7.

Consequence 3.8.

$$(\text{NLOG-TIME}(n))-(\text{DLOG-TIME}(o(n^2/\log n)))\neq\emptyset,$$

$$(\text{ALOG-TIME}(n))-(\text{NLOG-TIME}(o(n^2/\log n)))\neq\emptyset.$$

3.2. Hierarchies for multihead models

The hierarchies for multihead models are proved in the same way as for one-head models. To separate nondeterminism and determinism let us consider the language $\text{PS}(f, g)$.

Lemma 3.9. *For arbitrary natural number $k>1$, let A be a $\text{DM}(k)$ such that $L(A)=\text{PS}(f, g)$. Then, $T_A(n)S_A(n)=\Omega(n^{3/2}/\log n)$.*

Proof. Let A be a $\text{DM}(k)$, $L(A)=\text{PS}(f, g)$ and let $T_A(n)S_A(n)=\Omega(n^{3/2}/\log n)$ be not true, i.e. $\forall a\in R^+$, $\forall m\in N$: $\exists s\geqslant m$ $T_A(s)S_A(s)<a\cdot s^{3/2}/\log s$. From this follows the inequality $T_A(s)S_A(s)<a\cdot s\cdot 4\cdot \lfloor \lfloor s^{1/2} \rfloor/2 \rfloor/\log s$. For $a=1/(4\cdot 96\cdot k^3)$ the inequality has the form $96\cdot k^3\cdot T_A(s)S_A(s)<s\cdot g(s)/\log s$. We show that there is a word from $\text{PS}(f, g)$ rejected by A. Let us consider the set $S_s(f, g)=\{w\in S(f, g)\,||w|=s\}$. For the rejecting computation D_y on a word $y\in S_s(f, g)$ (it is the only one) let us say that A compares a pair of subwords (x_i, x_j) in D_y iff D_y involves a configuration in which one of the heads on the input tape points to the subword x_i and some other one to the subword x_j. Below, every configuration in which one of the reading heads points to the first or to the last symbol of the subword x_v after this head had run over the whole subword $2^{g(s)}$, is denoted as a v-prominent configuration for $v\in\{1, \ldots, f(s)\}$. Every v-prominent configuration is a prominent configruation. Let us denote V as the set of prominent configurations reached by A during the computation D_y in time $\leqslant T_A(n)$. If the length of D_y is greater than $T_A(n)$, we add to the set V the first prominent configuration reached by A during D_y in time $>T_A(n)$ (if such exists).

Proposition 3.10. *For the rejecting computation D_y on a word $y\in S_s(f, g)$ there is a pair of indices (i, j), $1\leqslant i<j\leqslant f(s)$ such that*

(a) *the subwords x_i and x_j are not compared in D_y, and*

(b) *V comprises at most $16kT_A(s)/s$ i-prominent configurations and $16kT_A(s)/s$ j-prominent configurations.*

Proof of Proposition 3.10. The cardinality of the set V is at most $(kT_A(s)/g(s))+1<2kT_A(s)/g(s)$. There are, therefore, at least $f(s)/2$ subwords x_h of the word y such that D_y involves at most $4kT_A(s)/g(s)f(s)$ h-prominent configurations. The

number of pairs selected from these $f(s)/2$ subwords is $\binom{f(s)/2}{2} > s/48$. In that part of the computation which is between the two prominent configurations at most k^2 pairs of subwords are compared. Owing to the cardinality of the set V, at most $2k^3 T_A(s)/g(s)$ pairs of subwords of the word y can be compared during D_y within time $T_A(s)$. According to the assumption about s,

$$96k^3 T_A(s) \leqslant 96k^3 T_A(s) S_A(s) < sg(s)/\log s \leqslant sg(s),$$

i.e. $2k^3 T_A(s)/g(s) < s/48$. Thus, there are two subwords among the considered $f(s)/2$ ones which are not compared in D_y. □

Proof of Lemma 3.9. (*continued*). The number of words in $S_s(f, g)$ is $2^{g(s)(f(s)-1)}$. According to Proposition 3.10 there are two natural numbers h and r, $1 \leqslant h < r \leqslant f(s)$ such that for the pair (h, r) conditions (a) and (b) in Proposition 3.10 hold for at least $2^{g(s)(f(s)-1)}/f^2(s)$ computations on words from $S_s(f, g)$. From among these words $2^{g(s)(f(s)-1)}/(f^2(s)2^{g(s)(f(s)-2)}) = 2^{g(s)}/f^2(s)$ words can be chosen such that they differ only in the subwords x_h and x_r. Let V' involve only h-prominent and r-prominent configurations from V. As the pattern of the word y, we consider the pattern of the computation D_y regarding V'. □

Proposition 3.11. *The number of different patterns of words form $S_s(f, g)$ is limited by $e(s)$, where*

$$e(s) = 2^{(S_A(s) + k \cdot \log_2 s) \cdot 32kT_A(s)/s}.$$

Proof. The number of prominent configurations can be limited by the number $s^k \cdot 2^{S_A(s)} = 2^{(S_A(s) + k \cdot \log_2 s)}$, and the cardinality of V' by the number $32kT_A(s)/s$ (Proposition 3.10). □

Proof of Lemma 3.9. (*conclusion*). It holds that

$$2^{(S_A(s) + k \cdot \log_2 s) \cdot 32kT_A(s)/s} \cdot f^2(s) \leqslant 2^{(3 \cdot 32k^3 T_A(s) S_A(s) \log_2 s)/s} < 2^{g(s)},$$

i.e. $e(s) < 2^{g(4s)}/f^2(s)$. Hence, there are two words

$$v = x_1 2^{g(s)} \ldots x_{h-1} 2^{g(s)} u_h \ldots x_{r-1} 2^{g(s)} u_r \ldots x_{f(s)} 2^{g(s)+z(s)},$$

$$v' = x_1 2^{g(s)} \ldots x_{h-1} 2^{g(s)} u'_h \ldots x_{r-1} 2^{g(s)} u'_r \ldots x_{f(s)} 2^{g(s)+z(s)},$$

from $S_s(f, g)$ with the following properties:

(1) $u_h \neq u'_h$ and $u_r \neq u'_r$;

(2) v and v' have the same pattern P;

(3) the couple of the subwords (u_h, u_r) of v and (u'_h, u'_r) of v'

are not compared in D_v and $D_{v'}$.

We show that there is a word from PS(f, g) rejected by A. We distinguish two cases in a similar way as in the proof of Lemma 3.1.

Case 1: The last prominent configuration K of the pattern P is reached in time $\leqslant T_A(s)$ by at least one of D_v, $D_{v'}$ (for instance D_v). At most one of the subwords u_h, u_r (for instance u_h) is being read in K. Then, the computation on the word

$$w = x_1 2^{g(s)} \dots x_{h-1} 2^{g(s)} u_h \dots x_{r-1} 2^{g(s)} u_r' 2^{g(s)} \dots x_{f(s)} 2^{g(s)+z(s)}$$

is identical with D_v after reaching K. Thus, the word $w \in \mathrm{PS}(f, g)$ is rejected by A. In the other cases it is analogous.

Case 2: The last prominent configuration K of the pattern P is reached by both D_v and $D_{v'}$ in time $> T(s)$. Let us consider the deterministic computation on words w and w' given by

$$w' = x_1 2^{g(s)} \dots x_{h-1} 2^{g(s)} u_h' \dots x_{r-1} 2^{g(s)} u_r \dots x_{f(s)} 2^{g(s)+z(s)}.$$

$T(D_w(K)) > T(s)$ or $T(D_{w'}(K)) > T(s)$. Let $T(D_w(K)) > T(s)$. Then, even if the word $w \in \mathrm{PS}(f, g)$ were accepted by A, a time greater than $T(s)$ would be necessary for its acceptance.

Lemma 3.12. *There is a* NFA(k) A *for which* $L(A) = \mathrm{PS}(f, g)$ *and* $T_A(n) = \mathrm{O}(n)$.

Proof. A verifies whether the input word has the desired form: it finds out the number of groups of symbols 2; it computes the second power of this number and compares it with the length of the input word. Then the machine A nondeterministically decides which sum modulo 2 of which $f(n)$-tuple of corresponding symbols is equal to 1. The verification of this decision takes linear time.

To separate nondeterminism and alternation, once again, we utilize the lower bound for language $S(f, g)$ proved in [4].

Lemma 3.13. *For arbitrary natural number* $k > 1$, *let* A *be an* AM(k) *such that* $L(A) = S(f, g)$. *Then,* $T_A(n) S_A(n) P_A(n) = \Omega(n^{3/2}/\log n)$.

Lemma 3.14. *There is an* AFA(k) A *such that* $L(A) = S(f, g)$ *and* $T_A(n) = \mathrm{O}(n)$.

Proof. Let us informally describe the computation of the machine A:

(1) A verifies whether the lengths of all subwords $x_i 2^j$ are $g(n)$ and their number is $f(n)$.

(2) A sets its head H on the first symbol of the input word.

(3) A branches the computation on two parallel computations.

(4) In the first one A moves the head H by one symbol to the right. If H reads symbol 2, then A goes into an accepting state, else it proceeds by point (3).

(5) In the second of the two parallel computations A remembers the read symbol and moves the head H by $2 \cdot g(n)$ symbols to the right. The machine A adds the symbol just read to the remembered one. This is repeated $f(n)$ times. If the remembered symbol is equal to 0, A goes into an accepting state. □

Theorem 3.15.

$$(\mathrm{NTM}(k)\text{-TIME} \times \mathrm{SPACE}(n)) - (\mathrm{DTM}(k)\text{-TIME} \times \mathrm{SPACE}(o(n^{3/2}/\log n))) \neq \emptyset,$$

$$(\mathrm{ATM}(k)\text{-TIME} \times \mathrm{SPACE}(n)) - (\mathrm{DTM}(k) - \mathrm{TIME} \times \mathrm{SPACE}(o(n^{3/2}/\log n))) \neq \emptyset.$$

The same hierarchy holds for the machines $\mathrm{AM}(k)$, $\mathrm{NM}(k)$ and $\mathrm{DM}(k)$. As a consequence of the above assertions, results related to the finite multihead automata are obtained.

Theorem 3.16.

$$(\mathrm{NFA}(k)\text{-TIME}(n)) - (\mathrm{DFA}(k)\text{-TIME}(o(n^{3/2}/\log n))) \neq \emptyset,$$

$$(\mathrm{AFA}(k)\text{-TIME}(n)) - (\mathrm{NFA}(k)\text{-TIME}(o(n^{3/2}/\log n))) \neq \emptyset.$$

The language $S(f, g)$ is acceptable by a deterministic Turing machine in linear time and by a deterministic multihead finite automaton in time $O(n^{3/2})$. This, together with Lemma 3.13, enables us to compare the computational power of deterministic and alternating devices.

Theorem 3.17.

$$(\mathrm{DTM}(1)\text{-TIME}(n)) - (\mathrm{AM}(k)\text{-TIME} \times \mathrm{SPACE} \times \mathrm{PAR}(o(n^{3/2}/\log n))) \neq \emptyset,$$

$$(\mathrm{DFA}(k)\text{-TIME}(n^{3/2})) - (\mathrm{AM}(k)\text{-TIME} \times \mathrm{SPACE} \times \mathrm{PAR}(o(n^{3/2}/\log n))) \neq \emptyset.$$

4. Hierarchies referred to one computational model

Whereas in the preceding Section it was shown that nondeterministic (alternating) Turing machines with a certain bound on TIME·SPACE complexity are more powerful than the deterministic (nondeterministic) ones with greater bounds on

TIME·SPACE complexity, in this part we shall demonstrate a distinction between the complexity power of one type of a model with different bounds on TIME·SPACE complexity (TIME·SPACE·PARALLELISM complexity for alternating models). Again, at first we consider one-head models and in the second part the multihead ones.

The hierarchies are obtained by proving lower bounds on the computation complexity of some languages recognized on AM(1) (AM(k)) machines and upper bounds on DTM(1) (DFA(k)). The same technique as in the preceding section is used.

4.1. One-head models

Let us consider a language $L(a)=\{x2^{a(n)}xu \mid x\in\{0,1\}^{a(n)}, u\in\{0,1\}^{n-3a(n)}\}$ for any function $a(n)$ $a:N\to N$, $3a(n)\leqslant n$.

Lemma 4.1. *Let A be an* AM(1) *such that $L(A)=L(a)$. Then, $T_A(n)S_A(n)P_A(n)=\Omega(a(n)^2)$.*

Proof. Let us suppose that there is an AM(1) A such that $L(A)-L(a)$ and the equality $T_A(n)S_A(n)P_A(n)=\Omega(a(n)^2)$ is not true. Let us consider a set $L_n^u(a)$ consisting of all words from $L(a)$ of the length n having the same postfix u. For every accepting computation we define a prominent configuration in such a way as in the proof of Lemma 3.1. For a word $y\in L_n^u(a)$ let D_y be a fixed accepting computation on y. Let V be a set of prominent configurations reached by A during the computation D_y. As the pattern of the word y, we consider the pattern of the computation D_y regarding V. Then, the number of different patterns of words from $L_n^u(a)$ can be limited (in the same way as in Proposition 3.2) by the number

$$e(n)=2^{d\cdot S_A(n)T_A(n)P_A(n)/a(n)},$$

where d is the constant which bounds the branching from the universal states of the machine A. According to the assumption, there is an $s\in N$ for which $e(s)<2^{a(s)}=|L_s^u(a)|$. From this inequality follows the existence of two words $v=x2^{a(s)}xu$ and $v'=x'2^{a(s)}x'u$ from $L_s^u(a)$ having the same pattern P. Hence, the word $x2^{a(s)}x'u\notin L(a)$ is accepted, too. □

Lemma 4.2. *Let a be a function, $a:N\to N$, where $3\cdot a(n)\leqslant n$ and $a(n)$ is $(a(n), a(n))$-computable by* DTM(1). *Then, there exists a* DTM(1) *machine A recognizing the language $L(a)$ such that $T_A(n)S_A(n)=O(a(n)^2)$.*

Proof. In the first phase of its computation the machine A computes the value $3\cdot a(n)$ and the reading head points to the $3\cdot a(n)$ and the reading head points to the $3\cdot a(n)$th symbol of the input word. In the second phase A moves the input head to the left and copies the read symbols on the working tape until the input head reads the symbol 2.

Then, A compares the contents of the working tape with the prefix of the input word. The second phase takes time $3 \cdot a(n)$ and space $a(n)$. □

As the consequence of Lemmas 4.1. and 4.2. the following assertions hold.

Theorem 4.3. *Let M be any device from* DTM(1), NTM(1), ATM(1), DM(1), NM(1), and AM(1). *Let $a(n), b(n)$ be functions $a, b: N \to N$, $3 \cdot a(n) \leqslant n$, $3 \cdot b(n) \leqslant n$, $a(n)$ is $(a(n), a(n))$-computable by M and $b(n) = o(a(n)^2)$. Then,*

$$(M\text{-TIME} \times \text{SPACE} \times \text{PAR}(a(n)^2)))$$
$$-(M\text{-TIME} \times \text{SPACE} \times \text{PAR}(b(n))) \neq \emptyset.$$

4.2. Multihead models

Once again, we state the lower and the upper bounds of TIME · SPACE · PARALLELISM complexity, this time for the language

$$S(f, g)(a) = \left\{ vu \;\middle|\; |vu| = n,\ |v| = a(n),\ u \in \{0,1\}^{n-a(n)},\right.$$
$$v = x_1 2^{g(a(n))} x_2 2^{g(a(n))} \ldots x_{f(a(n))} 2^{g(a(n)) + z(a(n))},$$
$$x_i \in \{0,1\}^{g(a(n))},\ \sum_{i=1}^{f(a(n))} \oplus x_i = 0^{g(a(n))},$$
$$\left. z(a(n)) = a(n) - 2f(a(n))g(a(n)) \right\}$$

for any function $a: N \to N$, $a(n) \leqslant n$.

Lemma 4.4. *Let A be an* AM(k), $k \in N$, $k > 1$ *such that $L(A) = S(f, g)(a)$.* Then,

$$T_A(n) S_A(n) P_A(n) = \Omega(a(n)^{3/2}/\log n). \tag{1}$$

Proof. Let us suppose that there is such an AM(k) A that $L(A) = S(f, g)(a)$ and the equality (1) does not hold, i.e.

$$\forall c \in R^+,\ \forall m \in N\ \exists s \geqslant m:\ T_A(s) S_A(s) P_A(s) < c a(s)^{3/2}/\log s.$$

From this it follows that

$$T_A(s) S_A(s) P_A(s) < c a(s) \cdot 4 \cdot \lfloor \lfloor a(s)^{1/2} \rfloor / 2 \rfloor / \log s.$$

For $c = 1/(4 \cdot 96 \cdot k^3)$,

$$96 k^3 T_A(s) S_A(s) P_A(s) < a(s) g(a(s))/\log s.$$

Let us consider a set $S_s^u(f, g)(a)$ consisting of all words from $S(f, g)(a)$ of the length s having the same postfix u. For every accepting computation we define a prominent configuration in such a way as in the proof of Lemma 3.9. For a word $y \in S_s^u(f, g)(a)$ let D_y be an accepting computation on y. Let V be a set of prominent configurations reached by A during D_y. the following assertion can be proved in the same way as in the Proposition 3.10. □

Proposition 4.5. *For every accepting computation D_y on the word $y \in S(f, g)(a)$ there is a pair of indices i, j, $1 \leqslant i < j \leqslant f(a(s))$, such that*

(a) *the subwords x_i, x_j are not compared in D_y,*

(b) *D_y involves at most $16dkT_A(s)\, P_A(s)/a(s)$ i-prominent and $16dkT_A(s)P_A(s)/a(s)$ j-prominent configurations.*

For every $y \in S_s^u(f, g)(a)$ let D_y be a fixed accepting computation on y. The cardinality of the set $S_s^u(f, g)(a)$ is

$$c(s) = 2^{g(a(s))(f(a(s))-1)}.$$

It follows then from Proposition 4.5 that there are two natural numbers h, r $1 \leqslant h < r \leqslant f(a(s))$ such that the conditions (a) and (b) hold for the pair (h, r) in at least $c(s)/f(a(s))^2$ computations D_y on words from $S_s^u(f, g)(a)$. Further, let the pattern of the word y be the pattern of the fixed accepting computation D_y regarding the r-prominent and h-prominent configurations from V. The number of different patterns of words from $S_s^u(f, g)(a)$ is limited by

$$e(s) = 2^{(S_A(s)+k\cdot\log s)\,32dkT_A(s)P_A(s)/a(s)}.$$

The existence of the word, which does not belong to $S(f, g)(a)$ but is accepted by the machine A, follows from these facts in such a way as in the proof of Lemma 3.9.

Lemma 4.6. *Let a be a function $a: N \to N$, $a(n) \leqslant n$; $a(n)$ is $(a(n)^{3/2}, 1)$-computable by* DFA(k). *Then, there exists a* DFA(k) *machine A recognizing the language $S(f, g)(a)$ such that $T_A(n) = O(a(n)^{3/2})$.*

Proof. In the first phase the values $a(n)$, $g(a(n))$, $f(a(n))$ are computed by A. Then, A checks whether the lengths of all subwords x_i, 2^j, are $g(a(n))$ and their number is $f(a(n))$. Finally, A checks whether the sum modulo 2 is zero. □

As a consequence of Lemmas 4.4 and 4.6, we obtain the following theorem.

Theorem 4.7. *Let M be any device from* DFA(k), NFA(k), AFA(k), DTM(k), NTM(k), ATM(k), DM(k), NM(k), AM(k). *Let $a(n)$, $b(n)$ be functions $a, b: N \to N$,*

$a(n) \leqslant n$, $b(n) \leqslant n$; $a(n)$ *is* $(a(n)^{3/2}, 1)$*-computable by* M, $b(n) = o(a(n)^{3/2}/\log n)$. *Then,*

$$(M\text{-TIME} \times \text{SPACE} \times \text{PAR}(a(n)^{3/2}))$$
$$-(M\text{-TIME} \times \text{SPACE} \times \text{PAR}(b(n))) \neq \emptyset.$$

References

[1] P. Ďuriš and Z. Galil, A time-space tradeoff for language recognition, *Math. Systems Theory* **17** (1984) 3–12.

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

[3] J. Hromkovič, On the power of alternation in automata theory, *J. Comput. System Sci.* **31** (1) (1985) 28–39.

[4] J. Hromkovič, Tradeoffs for language recognition on parallel computing models, in *Proc. 13th ICALP'86*, Lecture Notes in Computer Science Vol. 226 (Springer, Berlin, 1986) 157–166.

[5] L. Janiga, Real-time computations of two-way multihead finite automata, in: L. Budach, ed., *Proc. FCT 1979*, (Akademie Verlag, Berlin, 1979) 214–219.

[6] R.L. Rivest and A.C. Yao, $K+1$ heads are better than K, *J. ACM* **25**(2) (1978) 337–340.