数理解析研究所講究録
第 458 巻 1982 年 211-220

A Note on Alternating On-Line Turing Machines

Katsushi Inoue, Itsuo Takanami, and Hiroshi Taniguchi

(Faculty of Engineering, Yamaguchi University)

1. Introduction

Recently, alternating (off-line) Turing machines were introduced in [1] as a generalization of nondeterministic Turing machines and as a mechanism to model parallel computation. In papers [2-6], several fundamental investigations of alternating machines have been continued. It seems to us, however, that there are many problems about alternating machines to be solved in the future . In this short paper, we introduce a simple, natural complexity measure for alternating on-line Turing machines, called "leaf-size", and we provide a spectrum of complexity classes based on leaf-size bounded computations. Leaf-size is a useful abstruction which provides a spectrum of complexity classes intermediate between nondeterminism and full alternation.

2. Preliminaries

An alternating on-line Turing machine is like a nondeterministic on-line (or one-way) Turing machine [9] with its state set divided into two disjoint sets, the set of universal states and the set of existential states.

Definition 2.1. An alternating on-line Turing machine (AONTM) is a seven tuple $M=(Q,U,\Gamma,\Sigma,\delta,q_0,F)$, where (1) Q is a finite set of states, (2) $U\subseteq Q$ is the set of universal states, (3) Γ is a finite storage tape alphabet ($B\in\Gamma$ is the blank symbol), (4) Σ is a finite input alphabet ($\$\notin\Sigma$ is the right endmarker), (5) $\delta\subseteq(Q\times(\Sigma\cup\{\$\})\times\Gamma)\times(Q\times(\Gamma-\{B\})\times\{\text{no move, right}\}\times\{\text{left, no move, right}\})$ is the next move relation, (6) $q_0\in Q$ is the initial state, and (7) $F\subseteq Q$ is the set of accepting states. A state q in $Q-U$ is said to be existential.

The machine M has a read-only input tape with the right endmarker \$ and one semi-infinite storage tape, initially blank. A step of M consists of reading one symbol from each tape, writing a symbol on the storage tape, moving the input and storage heads in specified directions (note that the input head can only move to the right), and entering a new state, in accordance with the next move relation δ. Note also that as usual the machine cannot write the blank symbol B.

Definition 2.2. An instantaneous description (ID) of an AONTM $M=(Q,U,\Gamma,\Sigma,\delta,q_0,F)$ is an element of

$$\Sigma^*\times N\times S_M,$$

where $S_M=Q\times(\Gamma-\{B\})^*\times N$, and N denotes the set of all positive integers. The first and second components, x and i, of an ID $I=(x,i,(q,\alpha,j))$ represent the input (excluding the right endmarker \$) and the input head position, respectively. The third component (q,α,j) ($\in S_M$) of I represents the state of the finite control, the nonblank contents of the storage tape, and the storage head position. An element of S_M is called a "storage state" of M. If q is the state associated with an ID I, then I is said to be a universal (existential, accepting) ID if q is a universal (existential, accepting) state. The initial ID of M on input x is $I_M(x)=(x,1,(q_0,\lambda,1))$, where λ is the null word.

Definition 2.3. Given $M=(Q,U,\Gamma,\Sigma,\delta,q_0,F)$, we write $I \vdash I'$ and say that I' is a successor of I if an ID I' follows from an ID I in one step, according to the transition rules δ. A computation path of M on input x is a sequence $I_0 \vdash I_1 \vdash \ldots \vdash I_n$ $(n \geq 0)$, where $I_0 = I_M(x)$. A computation tree of M is a finite, non-empty labeled tree with the properties

(1) each node π of the tree is labeled with an ID, $\ell(\pi)$,

(2) if π is an internal node (a non-leaf) of the tree, $\ell(\pi)$ is universal and $\{I \mid \ell(\pi) \vdash I\} = \{I_1,\ldots,I_k\}$, then π has exactly k children $\rho_1,\ldots,\rho_k$ such that $\ell(\rho_i) = I_i$,

(3) if π is an internal node of the tree and $\ell(\pi)$ is existential, then π has exactly one child ρ such that $\ell(\pi) \vdash \ell(\rho)$.

A computation tree of M on input x is a computation tree of M whose root is labeled with $I_M(x)$. An accepting computation tree of M on x is a computation tree of M on x whose leaves are all labeled with accepting ID's. We say that M accepts x if there is an accepting computation tree of M on x. Define $T(M) = \{x \in \Sigma^* \mid M \text{ accepts } x\}$.

With each AONTM M, we associate a space complexity function SPACE which takes ID's to natural numbers. That is, for each ID $I=(x,i,(q,\alpha,j))$, let SPACE(I) be the length of α. We say that M is L(n) space-bounded if for each n and for each input x of length n, each computation tree of M on x is such that for each node π of the tree, $\text{SPACE}(\ell(\pi)) \leq \lceil L(n) \rceil$[‡]. By AONTM(L(n)) we denote an L(n) space-bounded AONTM. Define

$$\mathcal{L}[\text{AONTM}(L(n))] = \{T \mid T=T(M) \text{ for some AONTM}(L(n))\ M\}.$$

Nondeterministic on-line (or one-way) Turing machines are special cases of AONTM's. That is, a nondeterministic on-line Turing machine is an AONTM which has no universal state. By ONTM(L(n)), we denote an L(n) space-bounded non-deterministic on-line Turing machine. Define

‡ $\lceil r \rceil$ means the smallest integer greater than or equal to r.

$$\mathcal{L}[\text{ONTM}(L(n))]=\{T \mid T=T(M) \text{ for some ONTM}(L(n))\ M\}.$$

We next present a simple, natural complexity measure for AONTM's, called "leaf-size" [6,7]. (In [6], the term "branching" is adopted instead of the term "leaf-size".) Basically, the "leaf-size" used by an AONTM on a given input is the number of leaves of an accepting computation tree with fewest leaves. Leaf-size, in a sense, reflects the minimal number of processors which run in parallel in accepting a given input. One motivation for introducing leaf-size bounded computations is to provide a restriction of an AONTM which is intermediate in power between nondeterministic and (full) alternating computations. Let $Z(n):N \rightarrow R$ be a function with one variable n, where R denotes the set of all non-negative real numbers. For each tree t, let LEAF(t) denote the leaf-size of t (i.e., the number of leaves of t). We say that an AONTM M is Z(n) leaf-size bounded if for each n and for each input x of length n, if x is accepted by M then there is an accepting computation tree t of M on x such that $\text{LEAF}(t) \leq \lceil Z(n) \rceil$. By AONTM(L(n),Z(n)), we denote a Z(n) leaf-size bounded AONTM(L(n)). That is, an AONTM(L(n),Z(n)) is a simultaneously L(n) space- and Z(n) leaf-size bounded AONTM. Define

$$\mathcal{L}[\text{AONTM}(L(n),Z(n))]=\{T \mid T=T(M) \text{ for some AONTM}(L(n),Z(n))\ M\}.$$

As easily seen, it follows that $\mathcal{L}[\text{ONTM}(L(n))]= \mathcal{L}[\text{AONTM}(L(n),1)]$.

3. Results

The main purpose of this section is to provide Theorem 3.1 below, which shows that there is a spectrum of (low-level) complexity classes based on leaf-size bounded computations.

To give Theorem 3.1, we need a few definitions.

Definition 3.1. A function $Z(n):N \rightarrow R$ is log-space countable if there is a deterministic off-line (or two-way) Turing machine M which, when given a string of length n, halts after its read-write head has written down the k-adic ($k\geq 2$) notation of the number $\lceil Z(n)\rceil$ by using at most $\lceil \log n+1\rceil$ cells[‡] of the storage tape, where M has a read-only input tape and one semi-infinite storage tape [9,10].

Definition 3.2. Let $x=a_1a_2\ldots a_n$ be a word of length n ($n\geq 1$). For each i,j ($1\leq i\leq j\leq n$), $x(i,j)$ denotes the subword $a_ia_{i+1}\ldots a_j$.

Theorem 3.1. Let $L(n):N\rightarrow R$, $Z_1(n):N\rightarrow R$, and $Z_2(n):N\rightarrow R$ be a functions such that

(i) $L(n)\geq \log n$ ($n\geq 1$),

(ii) $L(n)$ is fully space constructible [11],

(iii) $Z_2(n)$ is log-space countable,

(iv) $\lceil L(n)\rceil\lceil Z_2(n)\rceil\leq n/2$ ($n\geq 1$),

(v) $Z_1(n)\leq Z_2(n)$ ($n\geq 1$), and

(vi) $\lim_{n\to\infty}[Z_1(n)/Z_2(n)]=0$. Then

$$\mathcal{L}[\text{AONTM}(L(n),Z_1(n))]\subsetneq\mathcal{L}[\text{AONTM}(L(n),Z_2(n))].$$

Proof. Let $T[L,Z_2]$ be the following set depending on the functions $L(n)$ and $Z_2(n)$ in the theorem.

$$T[L,Z_2]=\{x2w\mid (x,w\in\{0,1\}^+)\ \&\ \exists n\geq 1[|x|=|w|=2n\ \ \& \\ w(1,\lceil L(2n)\rceil\lceil Z_2(2n)\rceil)=w(n+1,n+\lceil L(2n)\rceil\lceil Z_2(2n)\rceil)]\}^{\#}.$$

(Note that, from the condition (iv) in the theorem, this set can be well-defined.) The set $T[L,Z_2]$ is accepted by an AONTM$(L(n),Z_2(n))$ M which acts as follows. Suppose that an input x2w with $|x|=|w|=2n$ ($n\geq 1$) and with $x,w\in\{0,1\}^+$ is presented to M. While reading the initial subword x, M first stores the length, 2n, of x in binary notation on the storage tape. By using this length 2n, M marks off exactly $\lceil L(2n)\rceil$ cells of the storage tape, and then

[‡] Below, let the base of logarithms be 2.
[#] For any word x, let $|x|$ denote the length of x.

writes down the k-adic notation (for some $k\geq 2$) of the number $\lceil Z_2(2n)\rceil$ on one track of the storage tape with its input head staying on the symbol "2" immediatly following x. (These actions are all possible because of conditions (i), (ii), and (iii) in the theorem.) After that, M universally tries to check that, for each $1\leq i\leq \lceil Z_2(2n)\rceil$, $w((i-1)\lceil L(2n)\rceil+1, i\lceil L(2n)\rceil) = w(n+(i-1)\lceil L(2n)\rceil+1, n+i\lceil L(2n)\rceil)$. That is, on the symbol $w((i-1)\lceil L(2n)\rceil+1)$[‡] $(1\leq i\leq \lceil Z_2(2n)\rceil)$, M enters a universal state to choose one of two further actions. One action is to pick up and store the subword $w((i-1)\lceil L(2n)\rceil+1, i\lceil L(2n)\rceil)$ on some track of the storage tape (of course, M uses the cells marked off above of the storage tape), to move its input head to the symbol $w(n+(i-1)\lceil L(2n)\rceil+1)$, to compare the subword stores above with the subword $w(n+(i-1)\lceil L(2n)\rceil+1, n+i\lceil L(2n)\rceil)$, and to enter an accepting state if both subwords are identical. The other action is to continue moving to the symbol $w(i\lceil L(2n)\rceil+1)$ (in order to pick up the next subword $w(i\lceil L(2n)\rceil+1, (i+1)\lceil L(2n)\rceil)$ and compare it with the corresponding subword $w(n+i\lceil L(2n)\rceil+1, n+(i+1)\lceil L(2n)\rceil)$). Note that the number of pairs of subwords which should be compared each other in the future can be seen by updating the k-adic notation of $\lceil Z_2(2n)\rceil$. It will be obvious that the input x2w is in $T[L,Z_2]$ if and only if there is an accepting computation tree of M on x2w with $\lceil Z_2(2n)\rceil$ leaves. (Any input which is not of the form x2w ($|x| = |w| = 2n$, $n\geq 1$, $x,w\in\{0,1\}^+$) can easily be rejected by M.) Thus, $T[L,Z_2]$ is in $\mathcal{L}[\mathrm{AONTM}(L(n), Z_2(n))]$.

We next show that $T[L,Z_2]$ is not in $\mathcal{L}[\mathrm{AONTM}(L(n), Z_1(n))]$. Suppose that there is an $\mathrm{AONTM}(L(n), Z_1(n))$ M accepting $T[L,Z_2]$. We assume without loss of generality that M enters an accepting state only on the right endmarker \$. Let r and s be the numbers of states (of the finite control) and storage tape symbols of M, respectively. For each accepting computation tree t of M, let SS(t)

[‡] For any word x and any i $(1\leq i\leq |x|)$, x(i) denotes the i-th symbol (from the left) on x.

be a "multi-set" of storage states of M defined as follows:

$SS(t) = \{(q,\alpha,j) \in S_M \mid I = (x2w, i, (q,\alpha,j))$ is a node label of t, and

I is an ID of M just after the point where the input head

left the former half of w$\}$,

where x2w ($\in T[L,Z_2]$) is the input associated with t.

For each input $0^{2n}2w$ with $|w| = 2n$ ($n \geq 1$), let ACT(w) be the set of all accepting computation trees of M on $0^{2n}2w$ whose leaf-sizes are at most $Z_1(4n+1)$.

For each $n \geq 1$, let

$$V(n) = \{0^{2n}2w \mid |w| = 2n \ \& \ w \in \{0,1\}^+ \ \& \ w(1, \lceil L(2n)\rceil \lceil Z_2(2n)\rceil) = w(n+1, n+\lceil L(2n)\rceil \lceil Z_2(2n)\rceil) \ \& \ w(\lceil L(2n)\rceil \lceil Z_2(2n)\rceil+1, n) = w(n+\lceil L(2n)\rceil \lceil Z_2(2n)\rceil+1, 2n) \in \{0\}^*\}$$

and for each $0^{2n}2w$ in V(n), let $C(w) = \{SS(t) \mid t \in ACT(w)\}$. (Clearly, each tape in V(n) is in $T[L,Z_2]$, and so it is accepted by M. Thus, it follows, since we assumed that M enters an accepting state only on the right endmarker \$, that for each $0^{2n}2w$ in V(n) C(w) is not empty.)

Then the following proposition must hold.

Proposition 3.1. For any two different words $0^{2n}2u$, $0^{2n}2v$ in V(n),

$$C(u) \cap C(v) = \emptyset \quad \text{(empty set)}.$$

[For otherwise, suppose that $C(u) \cap C(v) = \emptyset$. Then there exist accepting computation trees t and t' in ACT(u) and ACT(v), respectively, such that SS(t) = SS(t'). We consider the word $0^{2n}2w$ (with $|w| = 2n$) satisfying the following two conditions:

(i) $w(1,n) = u(1,n)$;

(ii) $w(n+1,2n) = v(n+1,2n)$.

Recalling that for any accepting conputation tree t_1 of M $SS(t_1)$ is a "multi-set", it is easily seen that one can construct, from the trees t and t', an accepting computation tree of M on $0^{2n}2w$ whose leaf-size is at most $Z_1(4n+1)$. Thus, it follows that $0^{2n}2w$ is in T(M). This contradicts the fact that $0^{2n}2w$ is not in $T[L,Z_2]$.]

Let p(n) be the number of possible storage states of M just after the input head left the former halves of w's in words $0^{2n}2w$'s (in V(n)). Then we have

$$p(n) \le rL(4n+1)s^{L(4n+1)}.$$

(Note that for each $0^{2n}2w$ in V(n) $|0^{2n}2w| = 4n+1$.) Since for each $0^{2n}2w$ in V(n) and for each t in ACT(w) LEAF(t) is at most $Z_1(4n+1)$, it follows that for each $0^{2n}2w$ in V(n) and for each t in ACT(w)

$$|SS(t)| \le Z_1(4n+1).^{\ddagger}$$

Therefore, letting $S(n) = \{SS(t) \mid t \in ACT(w) \text{ for some } 0^{2n}2w \text{ in } V(n)\}$, it follows that for some constant c,

$$\begin{aligned}|S(n)| &\le cp(n)^{Z_1(4n+1)}\\ &\le cr^{Z_1(4n+1)}L(4n+1)^{Z_1(4n+1)}s^{L(4n+1)Z_1(4n+1)}. \qquad (1)\end{aligned}$$

It also follows, from conditions (iv) and (v) in the theorem, that for some constants c' and c", $Z_1(4n+1) \le c'Z_1(2n)$ and $L(4n+1) \le c''L(2n)$. From this and from (1) above, we have for some constant c_1

$$\log |S(n)| \le c_1L(2n)Z_1(2n). \qquad (2)$$

As is easily seen, $|V(n)| = 2^{\lceil L(2n)\rceil \lceil Z_2(2n)\rceil}$, and so

$$\log |V(n)| = \lceil L(2n)\rceil \lceil Z_2(2n)\rceil. \qquad (3)$$

From (2) and (3) above, and from condition (vi) in the theorem, we have $|S(n)| < |V(n)|$ for large n. Therefore, it follows that for large n there must be different words $0^{2n}2u$, $0^{2n}2v$ in V(n) such that $C(u) \cap C(v) \ne \emptyset$. This contradicts Proposition 3.1, and thus it follows that $T[L,Z_2] \notin \mathcal{L}[\text{AONTM}(L(n), Z_1(n))]$. From condition (v) in the theorem, it directly follows that $\mathcal{L}[\text{AONTM}(L(n),Z_1(n))] \subseteq \mathcal{L}[\text{AONTM}(L(n),Z_2(n))]$. This completes the proof of the

‡ For any set A, let |A| denote the number of elements in A.

theorem. Q.E.D.

Remarks 3.1. It is well-known [8,9] that if $\lim_{n\to\infty}[L(n)/\log n] = 0$, then ONTM(L(n))'s accept only regular sets. Recently, it is shown [12] that

(1) $\mathcal{L}$[AONTM(log log n)] properly contains the class of regular sets, and

(2) if $\lim_{n\to\infty}[L(n)/\log\log n] = 0$, then AONTM(L(n)'s accept only regular sets.

Let $L_c = \{w2w \mid w \in \{0,1\}^*\}$. As is easily seen, L_c is accepted by an AONTM(log n). On the other hand, it can be shown that L_c is not in $\mathcal{L}$[AONTM(L(n))] for any L(n) such that $\lim_{n\to\infty}[L(n)/\log n] = 0$. Therefore, it follows that $\mathcal{L}$[AONTM(L(n))] $\subsetneq \mathcal{L}$[AONTM(log n)] for any L(n) such that $\lim_{n\to\infty}[L(n)/\log n] = 0$.

It will be interesting to investigate whether a similar fact to Theorem 3.1 holds for L(n)'s such that $\log\log n \leq L(n) \leq \log n$.

REFERENCES

[1] A.K.Chandra, D.C.Kozen, and L.J.Stockmeyer, Alternation, J.ACM., Vol.28, No.1, 114-133 (1981).

[2] R.E.Ladner, R.J.Lipton, and L.J.Stockmeyer, Alternating pushdown automata, Proc. 19th IEEE Symp. on Foundations of Computer Science, Ann Arbor, Mich., (1978).

[3] W.L.Ruzzo, Tree-size bounded alternation, J.Comput.Syst.Sci., Vol.21, 218-235 (1980).

[4] W.Paul and R.Reischuk, On alternation, Acta Informat., Vol.14, 243-255 (1980).

[5] W.Paul and R.Reischuk, On alternation II, Acta Informat., Vol.14, 391-403 (1980).

[6] K.N.King, Measures of parallelism in alternating computation trees, Proc. 13th Ann. ACM Symp. on Theory of Computing, 189-201 (1981).

[7] K.Inoue, I.Takanami, and H.Taniguchi, Two-dimensional alternating Turing machines, To appear in the 14th ACM Symp. on Theory of Computing, (May 1982).

[8] J.Hartmanis, R.Sterns, and P.M.Lewis, Hierarchies of memory limited computations, IEEE Conf.Record on Switching Circuit Theory and Logical Design, 179-190 (1965).

[9] J.E.Hopcroft and J.D.Ullman, Some results on tape-bounded Turing Machines, J.ACM., Vol.16, No.1, 168-177 (1967).

[10] J.E.Hopcroft and J.D.Ullman, Formal languages and their relation to automata, Addison-Wesley, Reading, Mass., (1969).

[11] J.E.Hopcroft and J.D.Ullman, Introduction to automata theory, languages, and computation, Addison-Wesley, Reading, Mass., (1979).

[12] I.H.Sudborough, Efficient algorithms for Path System Problems and applications to alternating and time-space complexity classes, Proceedings of the 21st Annual Symp. on Foundations of Computer Science, (1980).