A NOTE ON TAPE BOUNDS FOR SLA LANGUAGE PROCESSING*

J. Hartmanis and L. Berman

TR 75-242

May 1975

Department of Computer Science
Cornell University
Ithaca, New York 14853

* This research as been supported in part by National Science Foundation Grants GJ-33171X and DCR75-09433.

A NOTE ON TAPE BOUNDS FOR SLA LANGUAGE PROCESSING*

J. Hartmanis and L. Berman
Department of Computer Science
Cornell University, Ithaca, NY 14850

Abstract

In this note we show that the tape bounded complexity classes of languages over single letter alphabets are closed under complementation. We then use this result to show that there exists an infinite hierarchy of tape bounded complexity classes of sla languages between log n and log log n tape bounds. We also show that every infinite sla language recognizable on less than log n tape has infinitely many different regular subsets, and, therefore, the set of primes in unary notation, P, requires exactly log n tape for its recognition and every infinite subset of P requires at least log n tape.

---

* This research has been supported in part by National Science Foundation Grants GJ-33171X and DCR75-09433.

Introduction

In this paper we investigate the properties of languages over a single letter alphabet (sla) which can be recognized with small amounts of memory. In particular, we are concerned with sla languages recognizable on $L(n)$ tape for $L(n) \leq \log n$. Clearly, for tape bounds $L(n)$ properly below $\log n$, the recognition device does not have enough memory to count up to the length of the input. Nevertheless, the main result of this note shows that we can still diagonalize over tape bounds in this range and get sla languages with very exact tape requirements for their recognition, thus, extending diagonalization methods to a range where up until now only ad hoc crossing sequence arguments had been used. On the other hand, we also show that diagonalization methods have very severe limitations in this low complexity range. For example, we cannot diagonalize over the regular sets with small amounts of memory and therefore, every sla language recognizable with small amounts of memory must contain infinitely many different infinite regular sets. This observation is then applied to get more results about the memory requirements for the recognition of primes in unary notation.

The recognition device used in this study is a two tape Turing machine, Tm, with a two-way, read-only input tape and a two-way, read-write work tape [4,6]. The input is placed between special end markers on the input tape and the read head cannot go past the end markers, nor can it change the input. The number of tape squares used on the work tape is our measure of computational complexity. We say that a language A, $A \subseteq \Sigma^*$, is accepted on $L(n)$ tape iff there exists a Tm M_i

which accepts A and never visits more than $L(n)$ different tape squares on its work tape for an input of length n. The set of tapes accepted by a Tm M_i is denoted by $T(M_i)$; the family of languages accepted on $L(n)$ tape is denoted by TAPE[L(n)], and the family of single letter alphabet languages accepted on $L(n)$ tape is designated by SLATAPE[L(n)]. For a given Tm M_i let $L_i(n)$ denote the maximum number of tape squares visited on its work tape for all inputs of length n. We say that $L(n)$ is tape constructable iff $L(n) = L_i(n)$ for some Tm M_i. Since in this note we deal primarily with sla languages, $L_i(n)$ will denote the amount of tape used for the input of length n and $L(n)$ is tape constructable if it is tape constructable for sla inputs.

It is known that there exist non-regular languages which can be recognized on

$$L(n) = \log \log n$$

tape and, furthermore, that this is the least amount of tape used for recognition of non-regular sets, since

$$\lim_{n\to\infty} \frac{L(n)}{\log \log n} = 0$$

implies that TAPE[L(n)] is the family of regular sets [6]. As a matter of fact, there does not exist an unbounded tape constructable $L(n)$ such that

$$\lim_{n\to\infty} \frac{L(n)}{\log \log n} = 0.$$

It is simply impossible for a Tm to lay off small amounts of tape; either it lays off only an amount bounded by a constant or the amount laid off must reach C loglogn infinitely often, for a fixed $C > 0$.

Furthermore, it is known that if L(n) is an unbounded, tape constructable function and

$$\lim_{n\to\infty} \frac{L_1(n)}{L(n)} = 0$$

then there exists a language acceptable on L(n) tape but not on $L_1(n)$ tape. This result is obtained by diagonalization for $L(n) \geq \log n$ and by a crossing sequence argument for L(n) below log n [3,6]. Furthermore, in the range below log n the proof required that the languages are over alphabet Σ with $|\Sigma| > 1$ [3].

Recently a proof was published showing that there exist non-regular sla languages which can be accepted on log log n tape, and it was pointed out that it is not yet known whether there exists a hierarchy of sla languages between log log n and log n [1].

In this paper we solve this problem by showing that there is indeed a rich hierarchy of complexity classes of sla languages in the range between log log n and log n.

To prove the existence of different complexity classes of sla languages below the log n tape bound, we need several results which permit us to carry out a diagonalization argument. We do this by first showing that if an sla language A, $A \subseteq a^*$, is in TAPE[L(n)] then so is its complement, $\bar{A} = a^* - A$. From the proof of this result it follows that there is a recursive mapping σ such that for all i, $M_{\sigma(i)}$ is equivalent to M_i, uses no more tape than M_i, and halts for all inputs for which M_i uses a finite amount of tape. After proving a further technical result about tape constructable languages for sla inputs, we prove that if L(n) is tape constructable and for some infinite recursively enumerable set of integers $\{n_1, n_2, \ldots\}$

$$\lim_{k\to\infty} \frac{L_1(n_k)}{L(n_k)} = 0,$$

then there exists A, $A \subseteq a^*$, in TAPE[L(n)] but not in TAPE[L_1(n)]. From this result, using the fact that there exist non-regular sla languages acceptable on log log n tape, it follows that for any tape constructable $F(n) \geq n$,

$$\lim_{n\to\infty} \frac{L_1(n)}{F[\log \log n]} = 0$$

implies that

$$\text{SLATAPE}[L_1(n)] \subsetneq \text{SLATAPE}[F(\log \log n)].$$

We observe that it still is not known whether for alphabets Σ, $|\Sigma| > 1$, the family of languages TAPE[L(n)] is closed under complementation for all L(n). This is the case for $L(n) \geq \log n$, but the standard proof breaks down for $L(n) < \log n$. As we show in this paper, the restriction to sla languages permits us to prove this result for all L(n) for SLATAPE[L(n)].

We also note that if an infinite set A, $A \subseteq a^*$, is recognized on L(n) tape with

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0,$$

then A must contain infinitely many infinite regular subsets. Thus, we see that for the sla languages requiring less than log n tape we cannot diagonalize over the infinite regular sets (this can be done for $L(n) \geq \log n$). It is also interesting to observe that there exists an infinite language B, $B \subseteq \{0,1,2\}^*$, recognizable on log log n tape which contains no infinite regular subsets [1,3,6]. On the other hand, if

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0$$

then any language C, $C \subseteq \Sigma^*$, recognized on L(n) tape is such that either C or $\Sigma^* - C$ contains infinite regular subsets.

Finally, we observe that the set of primes in unary notation, P, requires exactly log n tape for its recognition and that no infinite subset of P can be recognized on less than log n tape. The corresponding question for the recognition of primes in binary notation, P_B, has not yet been solved completely. So far we only know that the recognition of P_B requires at least log n tape [2]. If P_B could be recognized on log n tape then we would have a deterministic polynomial time algorithm for testing whether a number is or is not a prime. A recent result shows that if the Generalized Riemann Hypotheses holds that then P_B can be recognized in deterministic polynomial time [5]. We conjecture that P_B cannot be recognized on log tape.

Tape Bounds for SLA Language Recognition

In this section we show that there is an infinite set of essentially different tape constructable functions in the log log n to log n range and that for each constructable tape bound L(n) there exists an sla language whose recognition in essence requires L(n) tape.

For the sake of completeness we first show that there exists a non-regular sla language recognizable on log log n tape [1]. This will guarantee the existence of many other sla tape constructable functions below log n. In the following proof, we make use of several well-known results from number theory, which are summarized below.

Let p_1, p_2, ... denote the prime numbers in increasing order. Then it is easily seen that the set of primes in binary notation represents a set which can be recognized on linear tape (it is a deterministic csl) [2]. Furthermore, for $k \geq 1$, there exists a prime p such that $k < p \leq 2k$. Thus, if we have p_i written on tape in binary notation, we need at most one more tape square to write down p_{i+1}. Also, every arithmetic progression a + bk, k = 1, 2,..., contains infinitely many primes if (a,b) = 1. Finally, if

$$\theta(x) = \sum_{p_i \leq x} \ln p_i$$

then

$$\lim_{x \to \infty} \frac{\theta(x)}{x} = 1,$$

and therefore, for every $\varepsilon > o$ and sufficiently large x

$$e^{(1-\varepsilon)x} \le \prod_{p_i \le x} p_i \le e^{(1+\varepsilon)x}$$

Consider now the sla language

$A_o = \{a^n \mid p_1, p_2, \ldots, p_t$ divide n but $p_1^2, p_2^2, \ldots, p_t^2$ and p_{t+1} do not divide $n\}$.

Theorem 1: A_o is a non-regular language recognizable on $L(n) = \log \log n$ tape. Furthermore, A_o is not recognizable on $L_1(n)$ tape if

$$\lim_{n\to\infty} \frac{L_1(n)}{\log \log n} = 0.$$

Proof: To see that A_o is non-regular, assume that A_o is recognized by a finite automaton with k_o states. Then for $a^n \in A$ with $n > k_o$, we see that for all $q \geq 0$ (by the pumping lemma)

$$a^{n+k_o!q} \in A_o,$$

and therefore

$$a^{n[1+k_o!q]} \in A_o.$$

Since $1 + k_o!q$ forms an arithmetic progression for $q = 1, 2, \ldots,$ we know that for some p_i and q_o, $1 + k_o!q_o = p_i$. But then by choosing $n = \prod_{p_j \le 2p_i} p_j$, we have $a^n \in A_o$ and $a^{n[1+k_o!q_o]} \in A_o$, and also that $n[1+k_o!q_o]$ is divisible by $p_1, p_2, \ldots, p_{i-1}, p_i^2, p_{i+1}, \ldots, p_t$, which implies that $a^{n[1+k_o!q_o]}$ is not in A_o. Thus, we conclude that A_o cannot be accepted by a finite automaton.

To see that A_o can be accepted on log log n tape, consider the following recognizer M*. For input a^n, M* checks whether n is divisible by p_1 and not p_1^2, p_2 and not $p_2^2, \ldots,$ until a p_{t+1} is found which does not divide n. To check for the input a^n whether n is divisible by p_t, we need only log p_t tape, and to check that n is not divisible by p_t^2, we need only 2 log p_t tape (or a richer tape alphabet and log p_t tape). Finally, to check whether n is not divisible by p_{t+1}, we need only one additional tape square because between k and 2k lies a new prime. Thus, the divisibility can be checked on log p_t tape and since

$$n \geq p_1, p_2, \ldots, p_t,$$

we get, by our previously mentioned result, that

$$\log \log n \geq \log p_t$$

for sufficiently large n. Thus, we see that A_o is recognizable on log log n tape.

The last statement of the theorem follows from the general result [6], mentioned in the Introduction, that recognition of non-regular sets requires that there exists a C, C > 0, such that for infinitely many n

$$L(n) \geq C \log \log n.$$

This completes the proof.

Note that in the recognition process of A_o, the recognizer M* lays off the same amount of tape for infinitely many inputs. Since if a^n is accepted and p_{t+1} was the largest prime used in this computation, then $a^{n \cdot p_s}$, $p_s > p_{t+1}$, is also accepted and the same amount of tape is laid off by M*. As shown in [1], this is a property of all sla language recognizers: if Tm M_i uses $L_i(n)$ tape on sla inputs and

$$\lim_{n\to\infty} \frac{L_i(n)}{\log n} = 0$$

then for some m_o all $L_i(n) > m_o$ will be achieved infinitely often.

Next we show that all tape bounded complexity classes of sla languages are closed under complement and then use this result to show that there exists a hierarchy of sla language tape bounded complexity classes below log n.

Theorem 2: If $A \subseteq a^*$ and $A \in \text{TAPE}[L(n)]$ then $\bar{A} = a^* - A \in \text{TAPE}[L(n)]$.

Proof: Consider a Tm M which has q states, a work tape alphabet of k symbols and which runs on L(n) tape. Then for an input of length n, this Tm can not enter more than

$$s(n) = q \cdot L(n) \cdot k^{L(n)}$$

different configurations while scanning a square of the input tape. The factor q represents the number of possible states of M, L(n) the possible head positions on the work tape and $k^{L(n)}$ the possible patterns which can be written on the work tape. For a suitable r, depending on q and k only,

$$r^{L(n)} \geq 2 \cdot q \cdot L(n) \cdot k^{L(n)},$$

from which it will follow that on L(n) tape another Tm can count high enough to detect cycling of M.

To do this construct Tm M' as follows: M' has a five-track working tape such that on each track on t tape squares it can count higher than r^t. On track 1, M' simulates M and if M ever halts, M' rejects the input if M accepts and vice versa.

On track 2 and track 3, M' counts the number of times the input head of M hits the left and right end marker, respectively. If either of these counts grow so large that they try to use more tape than so far used by M, M' rejects the input since M is cycling.

On track 4, M' counts the number of moves M has performed since its input head last encountered an end marker. If this count tries to use more tape than M has used so far (i.e. the count is at least twice the number of configurations M can enter), then M has entered a configuration twice since encountering an end marker and M either is heading for an end marker or cycling near one end of the input. M' now records the configuration that M is in on track 5 and counts on track 4 the displacement of the input head of M from its present position until the recorded configuration of M is repeated (which we know must happen in less steps than we can count on the available tape). If the displacement is zero, then M is cycling and M' accepts the input; if the displacement is not zero, then the input head of M will eventually hit an end marker, up the end marker count, and the process starts all over. Since M uses L(n) tape, M' will eventually halt, accept a^n iff M does not, and use no more tape than M. Thus, $T(M) = a^* - A \in TAPE[L(n)]$, as was to be shown.

An inspection of the proof of Theorem 2 shows that for every Tm we can effectively construct an equivalent Tm which uses no more tape and never cycles on a finite amount of tape.

Corollary 3: There exists a recursive function σ, such that for all Tm's M_i with sla inputs:

1. $T(M_i) = T(M_{\sigma(i)})$,
2. $L_i(n) = L_{\sigma(i)}(n)$,
3. if $L_i(n) < \infty$, then $M_{\sigma(i)}$ halts on input a^n.

Proof: By a slight modification of the proof of Theorem 2. It is also interesting to note, σ can be so chosen, that there exists a constant $C > 0$, such that $C \cdot |M_i| \geq |M_{\sigma(i)}|$, where $|M_i|$ denotes the length of the description of M_i and such that the set $\{M_{\sigma(i)}\}$ is a deterministic csl.

Next we establish the existence of infinitely many different tape bounded complexity classes of sla languages below the log n tape bound.

Theorem 4: Let $L(n)$ be tape constructable and let $D = \{n_1, n_2, \ldots\}$ be an infinite recursively enumerable set of integers such that $L_1(n_k) \geq$ and

$$\lim_{k\to\infty} \frac{L_1(n_k)}{L(n_k)} = 0.$$

Then there exists an sla language A in TAPE[$L(n)$], but not in TAPE[$L_1(n)$]

Proof: We first clarify the use of the set $\{n_1, n_2, \ldots\}$ in our theorem. Since for tape constructable $L(n)$ such that

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0$$

all (sufficiently large) values $L(n)$ are reached for infinitely many different inputs, we see that the condition

$$\lim_{n\to\infty} \frac{L_1(n)}{L(n)} = 0$$

implies that $L_1(n) = 0$ infinitely often. Thus, the use of this condition in the Theorem would lead to a weak result. As we will show, in this proof, it suffices that $L(n)$ outgrows $L_1(n)$ infinitely often to guarantee that we can construct an A computable on $L(n)$ tape but not on $L_1(n)$ tape.

To construct A we will diagonalize over all Tm's which can be simulated on $L(n)$ tape. To do this we will use the fact that we can detect when a Tm cycles, without using more tape, and we need a method to insure that we simulate every Tm on infinitely many inputs to make sure that A cannot be computed on $L_1(n)$ tape from some point on. We now give details of this construction.

Let $M_{\sigma(1)}, M_{\sigma(2)}, \ldots,$ be the list of Tm's guaranteed by Corollary 3, which never cycle using a finite amount of tape. Thus, if M_i runs on $L_i(n)$ tape then $M_{\sigma(i)}$ halts for all inputs, and also runs on $L_i(n)$ tape. Furthermore, $\{M_{\sigma(i)}\}$ is a deterministic csl. Let $L(n)$ be an unbounded tape constructable function and let

$$C = \{m \mid (\exists n_k \in D)\ [L(n_k) = m]\}.$$

The set C is clearly recursively enumerable, as D is; say $C = T(M_C)$. Let ρ be a recursive function, $\rho: N \to N$, such that for every i there exists infinitely many j for which $\rho(j) = i$, and such that $\rho(j)$ can be computed on j tape squares. Then there exists a recursive function t such that

a) $(\forall_j)[t(j) \in C]$

b) if $T(j)$ is the amount of tape used to compute $t(j)$, then $T(j) > |M_{\sigma \circ \rho(j)}|$ and $(\forall j > 1)\ (\exists m \in C)\ [t(j) > m > t(j-1) + T(j-1)]$.

We now exhibit a Tm, M_t, which computes a function t satisfying the above conditions:

M_t has t(1) stored in its finite control and it uses tape $T(1) > |M_{\sigma \circ \rho(1)}|$ before producing t(1). To compute t(j), M_t computes t(j-1), counts T(j-1) and stores t(j-1) + T(j-1), and calls M_C to enumerate C until an element, m, is found such that $m \geq t(j-1) + T(j-1)$. This element is stored and more of C is enumerated until an element m' of C is found such that $m' > |M_{\sigma \circ \rho(j)}|$ and m' > m. This element is t(j).

To construct the desired set A in SLATAPE[L(n)] and not in SLATAPE[L_1(n)], we consider Tm M_A: for input a^n, L(n) tape is laid off and the largest j is determined such that

$$L(n) > t(j) + T(j),$$

call this map of $L(n) \to j$ ψ; if no j can be found the input is rejected. Otherwise, M_A finds $M_{\sigma \circ \rho(j)}$, which can be dome on L(n) tape, and simulates $M_{\sigma \circ \rho(j)}$ on input a^n. If the simulation tries to use more than L(n) tape, then a^n is accepted. Otherwise, by the construction of $M_{\sigma \circ \rho(j)}$, we know know that $M_{\sigma \circ \rho(j)}$ will halt and M_A accepts if $M_{\sigma \circ \rho(j)}$ rejects and vice versa. Clearly $T(M_A)$ is L(n) tape acceptable, since M_A operates in L(n) tape. Furthermore, if $M_{\sigma(i)}$ runs in $L_1(n) \geq 1$ tape and

$$\lim_{k \to \infty} \frac{L_1(n_k)}{L(n_k)} = 0,$$

then M_A can simulate $M_{\sigma(i)}$ on tape $cL_1(n_k)$, for some c > 0, and for sufficiently large n_k, the limit condition implies that

$$cL_1(n_k) < L(n_k).$$

Since ρ maps infinitely many j onto i, we conclude that for some sufficiently large n_k we have $\psi(L(n_k)) = j$, $\rho(j) = i$ and therefore $M_{\sigma \circ \rho(j)} \equiv M_{\sigma(i)}$ and M_A has enough tape to find out what $M_{\sigma(i)}$ does and do the opposite. Thus, $T(M_A) \neq T(M_{\sigma(i)})$. Therefore, we conclude that A is in SLATAPE[L(n)] and not in SLATAPE[L_1(n)], as was to be shown.

The next result shows how we can easily get infinitely many different tape bounded classes of sal language in the range below log n.

<u>Corollary 5</u>: Let F(n), $n \leq F \leq 2^n$, be a tape constructable function. Then

$$\lim_{n\to\infty} \frac{L_1(n)}{F[\log \log n]} = 0$$

implies that

$$\text{SLATAPE}[L_1(n)] \subsetneq \text{SLATAPE}[F(\log \log n)].$$

<u>Proof</u>: Clearly, the containment follows from the limit condition. To show that the containment is proper, we proceed as follows. Using the construction in Theorem 1, lay-off L(n) tape which reaches log log n infinitely often. Now compute F of this amount of tape and diagonalize as in the proof of Theorem 4. This shows that there exists an A in SLATAPE[F(log log n)] and not in SLATAPE[L_1(n)].

Next we show that the sla languages requiring small amounts of tape for their recognition must contain infinite regular subsets.

Lemma 6: Let A be an infinite sla language in TAPE[L(n)] with

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0.$$

Then A contains infinite regular subsets.

Proof: Let M_i accept A on L(n) tape. Then M_i can enter no more than

$$q \cdot L(n) \cdot k^{L(n)}$$

configurations and, because of the limit condition, for large n

$$q \cdot L(n) \cdot k^{L(n)} < n.$$

Thus, in traversing the input a^n, for large n, M_i must repeate its configuration and therefore, if a^n is accepted, so is

$$a^{n+t\cdot n!}, \ t = 0, 1, 2, \dots .$$

(For details of this analysis see [1] or [6]). But then we know that the regular set

$$\{a^n | n = n + t\cdot n!, \ t = 0, 1, 2 \dots \}$$

is a subset of A. Thus, A contains infinitely many different , infinite regular sets, as was to be shown

It is interesting to note that Lemma 6 does not hold for tape alphabets with more than one letter. There exists infinite log log n recognizable languages which contain no infinite regular subsets; one such language is [6]:

$$\{\#b_1\ \#b_2\ \#b_3\ \#\ldots\#b_k\ \#\,|\,b_i \text{ is binary representation of } i\}.$$

On the other hand, we can prove that if A is in TAPE[L(n)] and

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0$$

then either A or Σ^* - A contains an infinite regular subset. To see this, note that either A or Σ^* - A must contain an infinite set over a single letter alphabet. Therefore, either A or Σ^* - A contains an infinite regular subset of this infinite sla subset (by the same proof as used for Lemma 6).

In conclusion, we apply this result to the recognition of prime numbers. Let

$$P = \{a^n \mid n \text{ is a prime number}\}.$$

Theorem 7: The set P of prime numbers in unary notation is in TAPE[log n] and every infinite subset of P requires at least log n tape of its recognition.

Proof: The set P can be recognized on log n tape by a Tm which first counts up to n and records n on the tape in binary notation. After that, on the available work tape, it checks if n is or is not a prime. Conversely, it is easily shown that every infinite subset of P requires at least log n tape for its recognition.

To see this let $S \subseteq P$ be an infinite subset of P and assume that S is in TAPE[L(n)] with

$$\lim_{n\to\infty} \frac{L(n)}{\log n} = 0.$$

Then S contains an infinite regular subset by Lemma 6, say T, $T \subseteq S$. Since T can be recognized by a finite automaton, there exists a k_0 such that $a^p \in T$ and $p > k_0$ implies that $a^{p + t \cdot k_0!} \in T$ for $t = 0, 1, 2, \ldots$. Clearly, $p + t \cdot k_0!$ cannot be a prime for all t since there exists arbitrarily large gaps between consecutive prime numbers. Thus, we see that no infinite subset of P can be recognized on less than log n tape; as was to be shown.

We recall that the corresponding problem for the recognition of the set of primes in binary notation, P_B, is not yet completely solved. The best result to date shows that at least log n tape is required, but it is not known whether this is sufficient for the recognition of P_B [2].

Acknowledgement

The authors wish to thank A.R. Freedman and R.E. Ladner for reawakening their interest in low complexity sla languages.

References

1. A.R. Freedman and R.E. Ladner, "Space Bounds for Processing Contentless Inputs", to appear in JCSS.

2. J. Hartmanis and H. Shank, "Two Memory Bounds for the Recognition of Primes by Automata", M.S.T., Vol. 3. (1969), 125-129.

3. J.E. Hopcroft and J.D. Ullman, "Some Results on Tape-Bounded Turing Machines", JACM, 16(1967), 168-177.

4. J.E. Hopcroft and J.D. Ullman, Formal Languages and Their Relation to Automata, Addison-Wesley, Reading Mass., 1969.

5. G.L. Miller, "Riemann's Hypotheses and Tests for Primality", Proceedings of the 7th Annual ACM Symposium on Theory of Computing (1975), 234-239.

6. R.E. Stearns, J. Hartmanis, and P.M. Lewis, II, "Hierarchies of Memory Limited Computations", IEEE Conference Record on Switching Circuit Theory and Logical Design (1965), 191-202.