JOURNAL OF COMPUTER AND SYSTEM SCIENCES 11, 118–128 (1975)

Space Bounds for Processing Contentless Inputs*

A. R. FREEDMAN

Department of Mathematics, Simon Fraser University, Burnaby 2, British Columbia, Canada

AND

R. E. LADNER

Department of Computer Science, University of Washington, Seattle, Washington 98195

Received December 12, 1973; revised September 16, 1974

The space and time bounds of Turing machines which process contentless inputs, i.e., inputs of the form a^n are investigated. There is such a Turing machine which uses space bounded by $\log \log n$ but not space bounded by any constant. Properties of this processor are given. The general properties of Turing machines processing contentless inputs are discussed. Any nontrivial processor can be transformed into a recognizer of a nonregular language in the same input alphabet and using exactly the same space. Finally, a theorem which establishes a hierarchy of contentless languages whose recognizers require at least $\log n$ space is given.

INTRODUCTION

We investigate the space requirements of Turing machines and in particular those that have their inputs restricted to a single letter alphabet, that is, those that process contentless inputs. Such machines have a read only input with end markers and an infinite storage tape. We only consider machines that eventually halt on every input and the read head is allowed to move in two directions. A set of strings in a single letter alphabet is called a contentless language. Stearns, Hartmanis, and Lewis [5] show that any Turing machine using unbounded space must use $\log \log n$ space for infinitely many n. Lewis, Stearns, and Hartmanis [4] exhibit a nonregular set that can be recognized by a Turing machine in space bounded by $\log \log n$, namely the set $C = \{cw_1c \cdots cw_kc : N(w_i) = i \text{ for } 1 \leqslant i \leqslant k\}$, where $N(w)$ is the integer represented by the binary string $w \in 1\{0, 1\}^*$. The recognition of C by a $\log \log n$

* Research supported by N.S.F. Grant No. GJ-34745X and by N.R.C. Grant No. 741 001.

space bounded machine seems to depend heavily on the "content" of the words being processed. This research was originally motivated by the question of whether or not there is a nonregular contentless language that can be recognized by a $\log \log n$ space bounded Turing machine.

In Section 1 we exhibit a Turing machine processor of contentless inputs that uses space bounded by $\log \log n$ but not bounded by any constant. The processor can be transformed into a $\log \log n$ space bounded recognizer of a nonregular contentless language. We notice that the time used by the machine is bounded above by $n \log n / \log \log n$ and that the processor realizes this bound on infinitely many inputs. The Turing machine must use bounded space on certain infinite sets of the inputs; in fact, any set of inputs for which the machine does not use bounded space on an infinite subset must itself be very sparse.

In Section 2 we discuss, among other things, the time required by space bounded Turing machines processing contentless inputs. We show that an $s(n)$ space bounded machine that uses unbounded space must use $n \log n / s(n)$ time for infinitely many n. We use techniques developed by Lewis, Stearns, and Hartmanis [4].

In Section 3 we first show that any Turing machine processing contentless inputs that uses unbounded space can be transformed into a recognizer of a nonregular contentless language without changing the space used. Second, we show that if $r(n)$ and $s(n)$ are exactly the spaces used, respectively, by two Turing machines processing contentless inputs with $r(n) \geqslant \log n$ and $\liminf_n (r(n)/s(n)) = 0$, then there is a contentless language which can be recognized in $s(n)$ space but not in $r(n)$ space. This result parallels a similar result of Stearns, Hartmanis, and Lewis [5] but is not directly implied by their theorem because they appeal to languages with content.

The reader may consult Hopcroft and Ullman [3, Chap. 10] for the basic definitions of off-line and on-line tape bounded Turing machines. All our Turing machines are off-line unless otherwise specified. We fix ¢ and \$ as the left and right end markers, respectively. Let T be any Turing machine (with arbitrary input alphabet). Define $s_T(n)$ to be the maximum number of storage tape cells scanned by T on an input of length n. Further, $t_T(n)$ is the maximum number of steps made by T on an input of length n. Also, define $m_T(n)$ to be the maximum number of times that the input head leaves an end marker and proceeds to the other end marker, without returning to the first end marker in the interim, on an input of length n. A function s is *tape constructable* if there is a Turing machine T such that $s(n) = s_T(n)$ for all n. A function s is *uniformly tape constructable* if there is a Turing machine T processing contentless inputs such that $s(n) = s_T(n)$ for all n. It follows that a function s is uniformly tape constructable if and only if there is a Turing machine T such that T uses exactly $s(n)$ space on every input of length n.

If $a(n)$ is a sequence of real numbers, then $\limsup_n a(n)$ and $\liminf_n a(n)$ have their standard meanings.

We say that a Turing machine T *uses unbounded space* if $\limsup_n s_T(n) = \infty$ and *uses space bounded by* $s(n)$ if $\liminf_n(s(n)/s_T(n)) > 0$. A language *can be recognized in space* $s(n)$ if there is a Turing machine T which uses space bounded by $s(n)$ and recognizes L. A language *requires space* $s(n)$ if, whenever T recognizes the language, $\limsup_n(s_T(n)/s(n)) > 0$. The meaning is clear should we add the adverb "on-line" in the appropriate places or replace "space" by "time" in the above definitions.

We assume that all logarithms are base 2 unless otherwise specified. Finally, for real number x, integers a and b, and string $w \in \Sigma^*$, we define $\lfloor x \rfloor$, $\lceil x \rceil$, $a \mid b$, $a \nmid b$, $|w|$ to be the greatest integer $\leqslant x$, the least integer $\geqslant x$, a divides b, a does not divide b, and the length of w (number of symbols in the string w), respectively.

1. Log Log n Space Bounded Contentless Processors

We describe a Turing machine T_1 with one storage tape which processes inputs of the form a^n and whose properties are presented in the theorems below. The Turing machine T_1 finds the first prime number p which does not divide n on input a^n. The action of T_1 is as follows. First, the number 2 is written in binary on the storage tape. Then the head is made to scan the entire input, a^n, from left to right to determine whether or not $2 \mid n$. If $2 \nmid n$ the machine halts. If $2 \mid n$ the number 3 replaces 2 on the storage tape (3 is the next prime number). In general, if a prime p has just been calculated and written in binary on the storage tape, T_1 then checks the input to see whether or not $p \mid n$. If $p \nmid n$, T_1 halts. If $p \mid n$, the next larger prime is calculated and written in binary on the storage tape, replacing p. More particularly, we want the read head of T_1 to behave as follows. It should remain unmoving on either the ¢ or \$ while the calculation of the next higher prime p is being made. Computing whether or not a binary string is a prime can be done in space equal to the length of the binary string. Then, in checking whether or not $p \mid n$, we want the read head to scan the input a^n, reaching the opposite end marker in exactly $n + 1$ machine moves (i.e., in time $n + 1$). This will involve the use of a "real time counter of count p." Such a device will allow T_1 to "count off" p consecutive moves of the machine and, by repeating the process, to determine whether or not $p \mid n$ in $n + 1$ machine moves.

The heart of a real time counter of count p is a subroutine called *pseudocounter* described as follows. The tape contains initially $0^B 0^{\lceil \log p \rceil - 2} 0^E$ with the head on 0^B. The head makes repeated cycles over the string of length $\lceil \log p \rceil$. Each cycle consists of $2\lceil \log p \rceil$ moves, the head traversing from 0^B to 0^E, then back to 0^B. On each such cycle the current binary string of length $\lceil \log p \rceil$ is changed to the next string in lexicographical order. To be specific, if the cycle begins with $\sigma_1{}^B \sigma_2 \cdots \sigma^E_{\lceil \log p \rceil}$ on the tape, then on the left to right pass each leading 1 is changed to a 0, and the first 0 is changed to a 1. Nothing else is changed during the remainder of the cycle. The *pseudocount* p is what remains on the tape after exactly p moves of the pseudocounter together with an indication of the head position and direction, and whether the head is among the leading 1's or the first 0. Also, if the pseudocounter has finished making changes, then the first 1 is marked. So the pseudocount looks like $\sigma_1{}^B\sigma_2 \cdots \sigma_i{}' \cdots \sigma_j{}^X \cdots \sigma^E_{\lceil \log p \rceil}$, where j indicates the head position, $\sigma_i{}'$ is the first 1, and X is a two-bit binary string, the first bit indicating the head direction and the second bit indicating whether or not the head is among the leading 1's or first 0. An important feature of the pseudocounter is that it can be run in *reverse*. Given a pseudocount p with the head positioned on the letter marked with a binary bit string, the process of pseudo-counting can be run in reverse so that after exactly p moves the tape consists of $0^B 0^{\lceil \log p \rceil - 2} 0^E$ with the head on 0^B. The changes in the reverse process are made as the head moves from *right* to *left*. The last 1 (reading right to left) is changed to a 0 and the remaining 0's are changed to 1's. Of course, the last 1 reading right to left is the first one reading left to right, so that it can be marked on each left to right pass in order to be located on the return trip.

A *real time counter of count* p consists of a tape of length $\lceil \log p \rceil$ with three tracks. The first track contains p written in binary. By simulating a pseudocounter construct the pseudocount of count p on track 2. Track 3 acts now as a pseudocounter and reverse pseudocounter which successively counts up to p, then down to zero. As the pseudocounter is running, a comparison of tracks 2 and 3 can be made (without time loss) to detect when track 3 has attained the pseudocount p. Likewise, as the reverse pseudocounter is running, it can detect (without time loss) whether the count is 0. Each time the pseudocount on track 3 reaches 0 or p a signal is sent indicating that exactly p moves have been made since the last signal.

Now it is clear that $p \mid n$ can be checked in $n + 1$ moves. Notice that the construction of the real time counter of count p uses no more space than the number p written in binary. Finally, we note that T_1 halts on every input a^n since there is a p such that $p \nmid n$, and, since all the storage tape calculations are done in space $\lceil \log q_n \rceil$, where q_n is the first prime not dividing n, that $s_{T_1}(n) = \lceil \log q_n \rceil$.

THEOREM 1. *The Turing machine T_1 uses unbounded space and space bounded by* $\log \log n$.

Proof. To show that T_1 uses unbounded space, it suffices to take $n_q = \prod_{p<q} p$, where q is a prime. Then $s_{T_1}(n_q) = \lceil \log q \rceil$ and so $\limsup s_T(n) = \infty$.

To show that T_1 uses space bounded by $\log \log n$ we notice, since each prime $p < q_n$ divides n, that $\prod_{p<q_n} p \mid n$ so that $\prod_{p<q_n} p \leqslant n$. Taking ln ($= \log_e$) of this inequality we get $\sum_{p<q_n} \ln p \leqslant \ln n \leqslant c \log n$ for some constant $c > 0$. The well-known result of Tchebycheff (1850), namely, that there exist positive constants a and b such that $ax \leqslant \sum_{p \leqslant x} \ln p \leqslant bx$ (see [1, Theorem 414]) implies that $q_n \leqslant (1/a) \sum_{p<q_n} \ln p$. Hence $q_n \leqslant (c/a) \log n$ whence $\liminf_n (\log \log n / s_{T_1}(n)) > 0$. ▮

COROLLARY. *There exists a nonregular subset of* $\{a^n \mid n \geqslant 1\}$ *which can be recognized in space bounded by* $\log \log n$.

Proof. This result is implied by Theorem 5 below in view of Theorem 1, but we give here an independent proof by actually exhibiting a nonregular set satisfying the requirements.

Let $B = \{a^n: q_n \equiv 1 \pmod 4\}$. Now B can be recognized in space bounded by $\log \log n$ by slightly modifying the machine T_1 to check, after q_n is found, whether or not $q_n \equiv 1 \pmod 4$ (which does not require any additional storage space), and accepting a^n if $q_n \equiv 1 \pmod 4$ and rejecting otherwise.

We have to show that B is nonregular. Suppose that B is regular. Then there is an integer b, residues $r_1, r_2, \ldots, r_k \pmod b$ $(0 \leqslant r_i < b)$, and two finite sets F, D such that

$$B = [\{a^n: n \equiv r_1 \pmod b\} \cup \cdots \cup \{a^n: n \equiv r_k \pmod b\} \cup F] - D.$$

Let α be the greatest number such that for some prime p, p^α divides b. Consider the infinitely many numbers of the form $m_q = \prod_{p<q} p^\alpha$ where $q \equiv 1 \pmod 4$, $q > \max\{|x| : x \in F \cup D \cup \{a^b\}\}$ and b divides m_q. Each $m_q \in B$ and therefore $m_q \equiv r_i \pmod b$ for some r_i. Hence $r_i = 0$ for some i. Hence $jb \in B$ for all sufficiently large j. But this is a contradiction since we may take $j = \prod_{p<q} p$ where q is a sufficiently large prime and $q \equiv 3 \pmod 4$. ∎

We next prove an upper bound for the time function $t_{T_1}(n)$ of the machine T_1. As will be indicated in Theorem 4 below, this bound is minimal.

THEOREM 2. *T_1 uses time bounded by* $n \log n/\log \log n$.

Proof. The Turing machine T_1 operates as follows. The read head sits on ¢ or \$ until it is ready to traverse the input in $n + 1$ moves. The number of traverses is clearly $\pi(q_n)$ where q_n is the first prime not dividing n and $\pi(x)$ is the number of primes not exceeding x. Since the space is bounded by $\log \log n$, by elementary considerations we can show that, to avoid T_1 looping, there is an $r > 0$ such that no more than $(\log n)^r$ machine moves are possible while the read head remains stationary. Hence there is a constant $c > 0$ such that the total time used in processing the input a^n is

$$t_{T_1}(n) \leqslant (n + 1 + (\log n)^r) \cdot \pi(q_n) + (\log n)^r \leqslant cn\pi(q_n).$$

By the well-known result [1, Theorem 7], there are positive constants a and b such that $an/\log n \leqslant \pi(n) \leqslant bn/\log n$. By the argument in Theorem 1 there is a constant $d > 0$ such that $q_n \leqslant d \log n$. Since $x/\log x$ is increasing for $x > 1$, then

$$\pi(q_n) \leqslant bq_n/\log q_n \leqslant bd \log n/\log(d \log n).$$

Whence $\liminf_n[(n \log n/\log \log n)/s_{T_1}(n)] > 0$. ∎

We observe that $s_{T_1}(n) = 2$ for any odd number n and in general $s_{T_1}(n) = \lceil \log q \rceil$ for any number of the form $m \cdot \prod_{p<q} p$ (q prime, $q \nmid m$). Hence $s_{T_1}(n)$ is bounded on infinite sets, and this will be the case for any Turing machine T with a single letter input alphabet and $\limsup_n(s_T(n)/\log n) = 0$ (Theorem 4(i) below). For T_1 we have the stronger result which follows.

THEOREM 3. *If* $\{n_i\}$ *is a sequence of integers such that* $\lim_i s_{T_1}(n_i) = \infty$, *then* $\lim_i(i/n_i) = 0$.

Proof. If $s_{T_1}(n) > K$, then, letting p_K be the largest prime with $\lceil \log p_K \rceil \leqslant K$, we have $\prod_{p \leqslant p_K} p \mid n$. Hence for all but finitely many of the n_i we have $n_i \equiv 0$ (mod $\prod_{p \leqslant p_K} p$) so that $\limsup_i(i/n_i) \leqslant 1/\prod_{p \leqslant p_K} p$. But the right-hand side becomes arbitrarily small as K approaches ∞. ∎

We conjecture that Theorem 3 holds for any contentless processor T with $\limsup_n(s_T(n)/\log n) = 0$.

To conclude the section, we remark that we do not know of any contentless language which is recognizable in space $s(n)$ where $\limsup_n(s(n)/\log n) = 0$ but also requires space $s'(n)$ where $\liminf_n(\log \log n/s'(n)) = 0$. A candidate follows. $\{a^n : p \equiv 1$ (mod 4), where p is the smallest prime such that $p^p \nmid n\}$. This set is nonregular. If we construct a Turing machine T_2 in imitation of T_1 writing p^p in binary on the storage tape the space used is $s_{T_2}(n) = \lceil p_n \log p_n \rceil$, where p_n is the smallest prime such that $p_n^{p_n} \nmid n$. Using techniques like those of Theorem 1 and the result that there exist positive constants a and b such that $an^2 \leqslant \sum_{p \leqslant n} p \log p \leqslant bn^2$, we can show that there exist $c > 0$ such that $s_{T_2}(n) \leqslant c(\log \log n)(\log n)^{1/2}$ and also

$$\liminf_n(\log \log n/s_{T_2}(n)) = 0.$$

2. TIME AND OTHER CONSIDERATIONS

Stearns, Hartmanis, and Lewis [5] have shown that any Turing machine using unbounded space processing on-line requires space $s(n)$, where $\liminf_n(s(n)/\log n) > 0$. This might lead one to suspect that a Turing machine processing contentless inputs, using unbounded space and space $s(n)$, where $\limsup_n(s(n)/\log n) = 0$, must make many "passes" over the input. This suspicion is verified by the following.

THEOREM 4. *Let T be a Turing machine processing contentless inputs such that T uses unbounded space and* $\lim \sup_n (s_T(n)/\log n) = 0$.

(i) $\lim \inf_n s_T(n) < \infty$,

(ii) $\lim \sup_n (s_T(n)\, m_T(n)/\log n) > 0$,

(iii) $\lim \sup_n (s_T(n)\, t_T(n)/n \log n) > 0$.

Proof. Let T be given with, say, k storage tapes, q states, and r storage tape symbols. Define $p(n) = q(s_T(n))^k r^{ks_T(n)}$. The number $p(n)$ represents the maximum number of storage states that can be achieved by T on the input of length n. A *storage state* of T is a triple (v, h, γ), where v is a state, h is a k-dimensional vector of positive integers (representing the k head positions on the k storage tapes) and γ is a k-dimensional vector of strings from the storage tape alphabet (representing the contents of the k storage tapes). (i) Since $\lim \sup_n (s_T(n)/\log n) = 0$, then there exists an m such that $p(m) < m$. Hence on every pass across the entire input of length m by the read head the machine T must enter the same storage state at least twice. We must have $s_T(m) = s_T(m + jm!)$ for all $j \geq 0$. (See Lewis, Stearns, and Hartmanis [4] for elaboration.) (ii) Let n_i be the least number such that $s_T(n_i) \geq i$. Such a number exists for each i because T uses unbounded space. Let $l_1, l_2, \ldots, l_m$, where $m = m_T(n_i)$, be such that l_j represents the shortest length of input on which T passes twice through the same storage state on the jth pass over the entire input. Each l_j is $\leq p(n_i)$. Hence, if $(p(n_i))^{m_T(n_i)} < n_i$, then a string of length $\prod_{j=1}^{m_i} l_j$ can be removed from the original input of length n_i to form a new input of shorter length on which T uses the same amount of space. This is impossible. We conclude that $n_i \leq (p(n_i))^{m_T(n_i)}$, that is, $n_i \leq [q(s_T(n_i))^k r^{ks_T(n_i)}]^{m_T(n_i)}$. There is a constant $c > 0$ such that $\log n_i \leq c s_T(n_i)\, m_T(n_i)$. Hence $\lim \sup_n (s_T(n)\, m_T(n)/\log n) > 0$. (iii) Since each pass over the input of length n_i takes at least n_i moves then $n_i \log n_i \leq c s_T(n_i)\, t_T(n_i)$. We have

$$\lim_n \sup (s_T(n)\, t_T(n)/n \log n) > 0. \quad \blacksquare$$

Theorem 4(iii) implies that any Turing machine T processing contentless inputs, using unbounded space and space bounded by $\log \log n$, satisfies

$$\lim_n \sup [t_T(n)/(n \log n/\log \log n)] > 0.$$

In other words, T requires $n \log n/\log \log n$ time for infinitely many n. This implies that the process T_1 described in Section 1 is optimal in the sense that no other process on contentless inputs, using unbounded space and space $\log \log n$, can run any faster than T_1.

Using the example of a $\log \log n$ recognizer of C (see Introduction), we see that there is a Turing machine T and positive constants a and b such that $a \log \log n \leq s_T(n) \leq b \log \log n$. By Theorem 4(i), $s_T(n)$ cannot be uniformly tape constructable.

Hence not every function that is tape constructable is uniformly so. It would be interesting to know whether or not every tape constructable function s with $\liminf_n(s(n)/\log n) > 0$ is uniformly tape constructable. It may be of some help to know that every uniformly tape constructable function s with $\liminf_n(s(n)/\log n) > 0$ is uniformly tape constructable by an on-line Turing machine processing contentless inputs. This is accomplished by using a binary counter track on the storage tape to simulate the position of the read head.

3. Diagonal Constructions

We begin by showing that every Turing machine processing contentless inputs and using unbounded space can be transformed into another that uses the same space exactly and recognizes a nonregular contentless language.

Theorem 5. *If s is uniformly tape constructable and unbounded then there exists a Turing machine U recognizing a nonregular contentless language with $s_U = s$.*

Proof. Let T be a Turing machine with input alphabet $\{a\}$ and with $s_T = s$. It is possible to obtain a uniform method of describing all deterministic finite state automata as strings in the alphabet $\{0, 1\}$. If $d \in \{0, 1\}^*$, then it is possible in $|d|$ space to check whether d is a well-formed description of a finite state automaton. If d is well formed and written on a track of the storage tape, then the simulation of the automaton described by d on the input can be done in just the space where d is written. Let d_i be the ith member of $\{0, 1\}^*$ in the natural ordering $(\lambda, 0, 1, 00, 01, \ldots)$. We construct U in such a way that U uses exactly $s_T(n)$ space and for each i there is a number n such that d_i is written on the storage tape on input a^n, d_i is checked to see if it is well formed, and if it is, d_i is simulated on a^n with U accepting a^n if and only if d_i rejects a^n.

We give a brief description of U. Each variable used can be stored on a track of the storage tape. Let n be arbitrary. On input a^n,

1. Simulate T in order to construct $s_T(n)$ space. Set $p = 0$, $m = 0$, $k = s_T(0)$, $d = d_0$.

2. Set $p = p + 1$. If $p \leqslant \min\{s_T(n), n - 1\}$ and $s_T(p) < s_T(n)$ then go to 3. Otherwise, see if d is well formed. If not, reject. If so, simulate d on input a^n, and accept if and only if d rejects the input.

3. If $s_T(p) > k$ and $s_T(p) \geqslant m$ then set $m = p$, $k = s_T(p)$, and $d = d_{i+1}$ if d is currently d_i. Go to 2.

It is not difficult to see that this procedure can be done in space $s_T(n)$.

The result is verified if we can show that for all i, d_i is realized as the final description written on some input. Define $n_0 = 0$ and n_{i+1} = the least $n > n_i$ such that $s_T(n) > s_T(n_i)$ and $s_T(n) \geqslant n_i$. On input a^{n_i} we may show by induction that m steps through the values $n_0, n_1, n_2, \ldots, n_{i-1}$ exactly. Hence the final value of d is d_i on input a^{n_i}.

We now show that there is a fine hierarchy of tape complexity classes of contentless languages above $\log n$.

THEOREM 6. *If r and s are uniformly tape constructable with* $\lim \inf_n(r(n)/\log n) > 0$ *and* $\lim \inf_n(r(n)/s(n)) = 0$, *then there is a Turing machine U recognizing a contentless language L such that $s_U = s$ and L cannot be recognized in space $r(n)$.*

Proof. Let T and R be Turing machines with input alphabet $\{a\}$ that construct s and r, respectively. The proof is similar to the proof of Theorem 5 except that we must contend with possible looping and with the fact that the simulation of an arbitrary storage tape alphabet by a single storage tape alphabet introduces a constant factor. Let d_i be as before except that d_i may be a well-formed description of a Turing machine with input alphabet $\{¢, \$, a\}$ and tape alphabet $\{0, 1, B\}$ (see Hopcroft and Ullman [3]). Let e_i be the ith member of the sequence $d_0, d_0, d_1, d_0, d_1, d_2, \ldots$. Using two tracks it is not difficult to generate e_i in space bounded by $\max\{| e_j |: 0 \leqslant j \leqslant i\}$.

The Turing machine U can be described as follows. On input a^n:

1. Simulate T in order to construct $s_T(n)$ space. Set $p = 0$, $m = 0$, $k = \lfloor s_T(0)/s_R(0) \rfloor$, $e = e_0$.

2. Set $p = p + 1$. If $p \leqslant \min\{\lfloor s_T(n)/s_R(n) \rfloor, \; n - 1\}$ and $\lfloor s_T(p)/s_R(p) \rfloor < \lfloor s_T(n)/s_R(n) \rfloor$ then go to 3. Otherwise see if e is a well formed description. If not, reject. If so, simulate e on the input for no more than $2^{s_T(n)}$ moves with a universal simulator (using a binary counter track) and without letting the space used exceed $s_T(n)$. Accept the input unless the simulation halts in an accepting state, in which case reject.

3. If $\lfloor s_T(p)/s_R(p) \rfloor > k$ and $\lfloor s_T(p)/s_R(p) \rfloor \geqslant m$ then set $m = p$, $k = \lfloor s_T(p)/s_R(p) \rfloor$, and $e = e_{i+1}$ if e is currently e_i. Go to 2.

It is not difficult to show that $s_U = s_T$. Let L be the contentless language recognized by U. Suppose that L is also recognized by a Turing machine V in space bounded by $r(n)$. This will prove to be impossible. There is a description $d \in \{0, 1\}^*$ and a constant $c > 0$ (depending on V) of a Turing machine with tape alphabet $\{0, 1, B\}$ that recognizes L in space bounded by $cr(n)$.

Define $n_0 = 0$ and n_{i+1} = the least $n > n_i$ such that $\lfloor s(n)/r(n) \rfloor > \lfloor s(n_i)/r(n_i) \rfloor$ and $\lfloor s(n)/r(n) \rfloor \geqslant n_i$. Such an infinite sequence exists because $\lim \inf_n(r(n)/s(n)) = 0$. As in Theorem 5 we can show that on input a^{n_i} the final value of e is e_i. Since

$\lim_i(r(n_i)/s(n_i)) = 0$, d occurs infinitely often as an e_i, and $\lim\inf_n(r(n)/\log n) > 0$, then there must be an i such that

(i) $d = e_i$,

(ii) $s(n_i) > cr(n_i)$,

(iii) $2^{s(n_i)} > n_i qcr(n_i) 3^{cr(n_i)}$,

where q is the number of states represented in d. On input a^{n_i}, U eventually successfully simulates d. Hence U accepts a^{n_i} if and only if V rejects a^{n_i}. This is impossible since both U and V presumably recognize L. ▮

Let Σ be a fixed finite alphabet. A function s is said to be Σ-tape constructable if there is a Turing machine T processing inputs in the alphabet Σ such that $s_T = s$. A Σ-language is simply a subset of Σ^*. The two theorems of this section can be proved in an identical way for Σ-tape constructable functions and Σ-languages.

Stearns, Hartmanis, and Lewis [5] show that if s is tape constructable then there is a language L recognizable in space $s(n)$, but if r is any function with $\lim\inf_n(r(n)/\log n) > 0$ and $\lim\inf_n(r(n)/s(n)) = 0$, then L is not recognizable in space $r(n)$. To obtain such a strong result they define the alphabet of L by subscripting the input alphabet of the Turing machine T that constructs s. A word of length n in the alphabet of L encodes a word of length n in the input alphabet of T and some Turing machine description. Words in the alphabet of L have much "content." We do not know whether or not it is necessary to obtain their result that the alphabet of L be larger than the input alphabet of T. However, using techniques like those of this section we can show that if s is Σ-tape constructable then there is a Σ-language L recognizable in space $s(n)$ but not in space $r(n)$ if $\lim\inf_n(r(n)/\log n) > 0$ and $\lim\inf_n(r(n)/s(n)) = 0$.

Hopcroft and Ullman [2] have shown that an infinite hierarchy of space complexity classes exists even below $\log n$. Again their proof relies heavily on the fact that words are allowed content. We do not even know whether or not every contentless language recognizable in space s with $\lim\sup_n(s(n)/\log n) = 0$ is not already recognizable in space $\log\log n$.

Note Added in Proof. Hartmanis and Berman have recently shown that there is an infinite hierarchy of space complexity classes of contentless languages even below $\log n$.

Bibliography

1. G. H. Hardy and E. M. Wright, "An Introduction to the Theory of Numbers," Oxford Press, 1960.
2. J. E. Hopcroft and J. D. Ullman, Some results on tape-bounded Turing machines, *J. Assoc. Comput. Mach.* **16** (1967), 168–177.

3. J. E. Hopcroft and J. D. Ullman, "Formal Languages and Their Relation to Automata," Addison-Wesley, Reading, Mass., 1969.
4. P. M. Lewis II, R. E. Stearns, and J. Hartmanis, Memory bounds for recognition of context free and context sensitive languages, *in* "IEEE Conf. Record on Switching Circuit Theory and Logical Design," 1965, pp. 191–202.
5. R. E. Stearns, J. Hartmanis, and P. M. Lewis II, Hierarchies of memory limited computations, "IEEE Conf. Record on Switching Circuit Theory and Logical Design," 1965, pp. 179–190.