Some Results on Tape-Bounded Turing Machines

J. E. HOPCROFT* AND J. D. ULLMAN

Bell Telephone Laboratories, Inc., Murray Hill, New Jersey

ABSTRACT. Classes of tape-bounded Turing machines similar to the on-line and off-line Turing machines, but without the restrictions that each machine halt and be deterministic, are studied. It is shown that the lower bounds on tape complexity of [1] depend on neither the halting assumption nor determinism. The existence of a dense hierarchy of complexity classes likewise does not depend on the halting assumption, and it is shown that below log n tape complexity there exists a dense hierarchy of complexity classes for two-way nondeterministic devices. It is also shown that the complexity classes of one-way, nondeterministic machines below linear tape complexity are not closed under complementation and are larger than the corresponding deterministic complexity class.

KEY WORDS AND PHRASES: Turing machine, computational complexity, tape complexity, off-line Turing machine, on-line Turing machine, nondeterministic Turing machine, transition matrix

CR CATEGORIES: 5.22

1. *Introduction*

Hartmanis, Lewis, and Stearns [1, 2] have considered the classification of languages according to the amount of memory necessary for recognition by a Turing machine. In [1] and [2], two Turing machine models are considered. Each model has a read-only input tape and an infinite working tape. One model, the off-line Turing machine, has a two-way input head with endmarkers. It accepts or rejects by entering one of several designated accepting or rejecting states, and it is assumed not to loop. The other model, the on-line Turing machine, has a one-way input, and it must enter an accepting or rejecting state before each shift right. This model is also assumed not to loop.

A main result of [1] is the existence of a hierarchy of recognizable languages. A set is said to be $L(n)$-recognizable by an off-line or on-line Turing machine if every word of length n can be recognized using at most $L(n)$ cells of working tape. For the off-line Turing machine, the hierarchy exists for $L(n) \gtrsim \log \log n$.[1] Unless $L(n) \gtrsim \log \log n$, the off-line Turing machine accepts only regular sets; in fact $L(n)$ is a constant in that case. For the on-line Turing machine, the hierarchy exists for $L(n) \gtrsim \log n$. Specifically, for $L_1(n)$ and $L_2(n) \gtrsim \log \log n$ (alt. $\log n$), if $L_1(n) \gg L_2(n)$ then there is some language which is $L_1(n)$-recognizable by an off-line (alt. on-line) Turing machine but not $L_2(n)$-recognizable by an off-line (alt. on-line) Turing machine.

In this note we extend the results of Hartmanis, Lewis, and Stearns in two direc-

* Present address: Cornell University, Ithaca, New York.

[1] Throughout this paper we shall write $L_1(n) \gtrsim L_2(n)$ if $\sup_{n\to\infty} (L_1(n)/L_2(n)) > 0$; $L_1(n) \gg L_2(n)$ if $\inf_{n\to\infty} (L_2(n)/L_1(n)) = 0$. Also since constant factors in a function $L(n)$ are seen in [1] not to affect the complexity class defined, we will write $\log n$ to mean $\log_b n$ for some b.

tions. First we determine the effect of removing the assumption that the Turing machine halts for every input. Second we extend some results to nondeterministic Turing machines. The latter are machines which may have a choice of moves in a given configuration.

Our results concern a class of machines which is in some cases broader than the class of on-line or off-line Turing machines, although there are obvious relationships between our models and the latter models. In particular, the halting restriction is removed from our machines. We shall designate our machines as one-way or two-way.

Formally, we define a *nondeterministic Turing machine* (NTM) M to be a 6-tuple $(K, T, I, \delta, q_0, F)$ where

K is the finite set of states;
T is the finite set of storage tape symbols;
I is the finite set of input tape symbols;
q_0, in K, is the start state;
F is the set of final states, $F \subseteq K$;
δ is a map from $K \times I \times T$ to subsets of $K \times T \times \{-1, 0, +1\} \times \{-1, 0, +1\}$.

If a quadruple (p, X, d_1, d_2) is in $\delta(q, a, Z)$, p and q in K, X and Z in T, a in I, and d_1 and d_2 chosen from among -1, 0, and $+1$, then the machine in state q, scanning input symbol a and storage symbol Z, has the option of replacing the Z by X on the storage tape, going to state p, and moving its input and storage heads d_1 and d_2 cells right, respectively. (-1 indicates a move left.)

Initially the NTM is started with some string of input symbols on a finite length input tape, with its input head at the leftmost input symbol, in state q_0, and with a particular storage symbol (called the "blank") occupying each storage tape cell.

We reserve two symbols, ¢ and \$, to be left and right endmarkers, respectively, for all Turing machines. We say that a NTM $M = (K, T, I, \delta, q_0, F)$ is a *two-way nondeterministic Turing machine* (2NTM) if

(1) I contains ¢ and \$, and

(2) for no p and q in K, X and Y in T, and d in $\{-1, 0, +1\}$ does $\delta(q, ¢, X)$ contain $(p, X, -1, d)$, or does $\delta(q, \$, X)$ contain $(p, X, +1, d)$. (M cannot move left from ¢ nor right from \$.)

The *language defined by* the 2NTM M is the set of w in $(I - \{¢, \$\})^*$ such that M enters an accepting state when started with ¢w\$ on its input.

We say a NTM $M = (K, T, I, \delta, q_0, F)$ is a *one-way nondeterministic Turing machine* (1NTM) if

(1) I contains \$,

(2) for each q in K and X in T, $\delta(q, \$, X)$ is empty, and

(3) for each q in K, a in I, and X in T, (p, Y, d_1, d_2) in $\delta(q, a, X)$ implies that $d_1 \neq -1$. (M cannot move left.)

The *language defined* by the 1NTM is the set of w in $(I - \{\$\})^*$ such that when started with w\$ on its input tape, M enters an accepting state upon moving its input head to \$.

A NTM $M = (K, T, I, \delta, q_0, F)$ is *deterministic* if for no q in K, a in I, and X in T does $\delta(q, a, X)$ contain more than one element. The one-way and two-way varieties will be denoted 1DTM and 2DTM, respectively.

A Turing machine M is of *tape complexity* $L(n)$ if for each input of length n (excluding endmarkers, if any), M uses at most $L(n)$ cells of its storage tape. A language is of tape complexity $L(n)$ for some machine model if it is defined by a Turing machine of that model which is of tape complexity $L(n)$. A nondeterministic Turing machine is *halting* if for each input there is a bound on the length of any allowable sequence of moves.

2. *The Halting Property*

Observe that if a one-way or two-way Turing machine is halting, it can be easily converted to an equivalent on-line or off-line Turing machine. We naturally ask under what circumstances a machine of one of the models we have defined can be modified to be halting without changing the language defined.

For the one-way machines, the answer is "always." Let a 1NTM M have s states and t storage tape symbols. If M has, at some point in its computation, used k storage cells and made more than skt^k moves without using a new storage cell or shifting its input head, then some configuration has repeated. If M accepts its input, then there is a shorter sequence of moves leading to acceptance. Therefore M may halt if it makes more than skt^k moves without shifting its input head or using a new storage cell.

Modify M to count in base $b = 2st$ to b^k on a track of its storage tape. The counter is reset to zero when M shifts the input head or uses a new storage cell. M will halt if the counter overflows. Since $b^k \geq skt^k$, the modification causes the machine to halt only when a configuration has repeated.

The construction for the case where M is two-way works in almost the same manner. However we need the condition that M is not $L(n)$-bounded if $\log n > L(n)$. The number of configurations of M with an input of length n and k storage cells used is $sk(n+2)t^k$. The new machine counts in base $b = 2st$ to $b^{k+\log_b(n+2)}$ and resets the count only when M uses a new storage cell. The count requires at most $L(n) + \log_b(n+2)$ cells. Since it is not true that $\log n > L(n)$, constants c and n_0 can be found such that for all $n \geq n_0$, $\log_b(n+2) \leq cL(n)$. The new machine recognizes all inputs of length less than n_0 in its finite control. For longer inputs it uses at most $(1+c)L(n)$ storage cells. By Theorem 1 of [1], an equivalent machine of tape complexity $L(n)$ can be found. Thus we have:

THEOREM 1. *Given one of the machines below, one can find an equivalent halting machine of the same model and complexity class.*

(1) *A 1NTM.*
(2) *A 1DTM.*
(3) *A 2NTM of tape complexity $L(n)$ unless $\log n > L(n)$.*
(4) *A 2DTM of tape complexity $L(n)$ unless $\log n > L(n)$.*

It is an open question whether a 2NTM or 2DTM of tape complexity $L(n)$ can be modified to halt if $\log n > L(n)$. We conjecture that the answer is no and offer the language S_1 (described below) as a candidate for a language which is defined by a 2DTM of tape complexity $\log \log n$ but by no halting 2NTM of tape complexity $L(n)$ if $\log n > L(n)$. We, of course, have no proof. The best way to describe S_1 is in terms of a 2DTM M_1 defining it. S_1 consists of words of the form $x_1cx_2c \cdots cx_mccw_2c \cdots cw_k$ where x_i, $1 \leq i \leq m$, is the binary representation of the integer i with no leading 0's. M_1 uses $x_1, x_2, \cdots, x_m$ to mark off a block of length log log m on its storage tape. (See [2] for an explanation of how this is done.) $w_1, w_2, \cdots, w_k$ are binary numbers equal to or less than m. M_1 then does the following:

(1) M_1 stores w_1 in the block on its storage tape and performs step 2.

(2) If M_1 has w_i in the storage block and its input head at w_i, M_1 moves its input head right, searching for some w_j equal to w_i, $i < j \leq m$. If none is found by the time M_1 reaches the right endmarker, M_1 accepts. If M_1 finds $w_j = w_i$, M_1 places w_{j-1} in the storage block and performs step 3.

(3) If M_1 has w_i in the storage block and its input head at w_i, M_1 moves its input head left, searching for some w_j equal to w_i, $1 \leq j < i$. If none is found by the time M_1 reaches $\cent$, M_1 halts without accepting. If M_1 finds $w_j = w_i$, M_1 places w_{j+1} in the storage block and performs step 2.

Observe that M_1 may enter a loop and never move off either end of the w's. These loops seem not to be detectable if only log log n memory is used with words of length n.

3. *Lower Bounds on Tape Growth*

We now show that the results of [1] concerning lower bounds on the rate of growth of tape-complexity functions do not depend on determinism or the halting assumption. To begin we need a few definitions.

A *storage state* of a NTM is the combination of state, contents of the storage tape, and position of storage tape head.

Suppose that with input w, the 2NTM M uses k storage cells. Let r be the number of storage states using up to k storage cells, and assume these states are numbered from 1 to r. For this M and w, and for each suffix u of w, we define the transition matrix T_u to be the $r \times r$ matrix whose elements t_{ij}, $1 \leq i, j \leq r$, have the following properties:

(1) $t_{ij} = 00, 01, 10,$ or 11.

(2) $t_{ij} = 1x$, $x = 0$ or 1, if starting in storage state i with the input head at the leftmost symbol of u, M can enter storage state j before its input head leaves u. Otherwise, $t_{ij} = 0x$ for some x.

(3) $t_{ij} = x1$, $x = 0$ or 1, if starting as in (2), M can enter storage state j immediately upon moving its input head left from u, never having previously moved from u. $t_{ij} = x0$ for some x otherwise.

THEOREM 2. *Let M be a 2NTM of complexity class $L(n)$. Then if there is no constant c such that $L(n) \leq c$ for all n, we must have $L(n) \geq$ log log n.*

PROOF. Suppose that for some k there is a word $w = \cent a_1a_2 \cdots a_n\$$ which is the shortest input causing M to use exactly k storage cells. Since M is of complexity class $L(n)$, but there is no constant upper bound on length of tape used, there are an infinity of such w and k.

Define w_i to be $a_ia_{i+1} \cdots a_n\$$, $1 \leq i \leq n$, and suppose $T_{w_i} = T_{w_j}$ for some $1 \leq i < j \leq n$. Then we claim that $w' = \cent a_1a_2 \cdots a_{i-1}a_ja_{j+1} \cdots a_n\$$ is a shorter input causing M to use exactly k storage cells.

To see the above, observe that if M can move right from position $i - 1$ when in a given storage state and return to position $i - 1$, it can return to position $i - 1$ in the same storage states whether w or w' is the total input. Thus if w causes M to enter a storage state using k cells, with the input head at one of the first $i - 1$ input symbols, w' will do likewise. Furthermore if w causes M to enter a storage state using k cells while the input head is scanning one of the symbols of w_i, w' will do likewise since $T_{w_i} = T_{w_j}$. Thus assuming that $T_{w_i} = T_{w_j}$ leads to the contradiction that w' is a shorter word using k storage cells than the assumed shortest word w. Hence T_{w_i} cannot equal T_{w_j} for any i and j, and there must be a different transition matrix for each of the n suffixes. Consequently, the number of possible different transition matrices must equal or exceed n.

If M has s states and t tape symbols, then r, the number of storage states using up to k storage cells, is at most skt^k. Hence the number of possible transition matrices for terminal subwords of a word using k storage cells is 4^{r^2}, which equals $4^{s^2k^2t^{2k}}$. We require that

$$4^{s^2k^2t^{2k}} \geq n.$$

Taking logarithms twice,

$$1 + 2 \log_2 s + 2 \log_2 k + 2k \log_2 t \geq \log_2 \log_2 n. \qquad (1)$$

Since $k \geq 1$, and $\log_2 x \leq x$ for all x, the left-hand side of (1) is not greater than $k[2 \log_2 t + 2 \log_2 s + 3]$. Hence

$$k \geq \frac{1}{\log_2 8t^2s^2} \log_2 \log_2 n,$$

from which it immediately follows that $L(n) \geq \log \log n$.

THEOREM 3. *If M is a 1NTM with no constant upper bound on the amount of storage tape used for any input, and M is of complexity class $L(n)$, then $L(n) \geq \log n$.*

PROOF. The proof is essentially the same as that used in [1] for the deterministic, on-line Turing machine. Suppose $w = a_1a_2 \cdots a_n\$$ is a word of shortest length causing M to use k storage cells. The configuration using k storage cells must not be reached before the input head scans a_n. Suppose that for some i and j, $1 \leq i < j < n$, there is a storage state which M may enter immediately before shifting the input head from a_i to a_{i+1} or from a_j to a_{j+1}. Then $a_1a_2 \cdots a_ia_{j+1}a_{j+2} \cdots a_n\$$ is a word of shorter length using k storage cells.

We must therefore have

$$n - 1 \leq skt^k$$

where s and t are the number of states and tape symbols of M, respectively.

Taking logarithms,

$$\log_2 (n - 1) \leq \log_2 s + \log_2 k + k \log_2 t. \qquad (2)$$

As in the proof of Theorem 2, from (2) we can derive

$$k \geq \frac{1}{\log_2 2st} \log_2 (n - 1),$$

from which $L(n) \geq \log n$ follows immediately.

4. *Nondeterministic Versus Deterministic Models*

One would suspect that the nondeterministic devices are in general more powerful than deterministic devices of the same tape complexity. We were able to show this statement true only for some one-way classes, and the proof is presented here.

If for some 1NTM the amount of storage tape used is bounded, only regular sets are accepted. In this case the deterministic and nondeterministic models are equivalent [3]. If the working tape is not bounded, by Theorem 3 it must grow at least as $\log n$. The next theorem exhibits a language which is accepted by a 1NTM of tape complexity $\log n$, but which cannot be accepted by any 1DTM of less than linear tape complexity.

THEOREM 4. *Let S_2 be the language consisting of all words of length $n = 2^r$ of the form $0^{2^{r-1}} w_1 w_2 \cdots w_k 0^m$ where:*

(1) *Each w_i, $1 \leq i \leq k$, is in $\{1, 2\}^r$.*

(2) *k is the integer part of $2^{r-1}/r$.*

(3) *m is the remainder when 2^{r-1} is divided by r.*

(4) *For some $1 \leq i < j \leq k$, $w_i = w_j$.*

S_2 is accepted by a 1NTM of tape complexity $\log n$ but by no 1DTM of tape complexity $L(n)$ unless $L(n) \geq n$.

PROOF. One can construct a 1NTM of tape complexity $\log n$ which counts the 0's on its tape, checks that this number is a power of 2, computes k and m, and then scans the words w_i, $1 \leq i \leq k$. While reading these it checks to see that there are exactly k of these, of proper length, followed by m 0's. Nondeterministically, it chooses one word to store on the tape it has used and compares it with subsequent words. If a match is found, the machine accepts. This 1NTM accepts S_2.

Now consider M, a 1DTM accepting S_2. Suppose that M has s states and t tape symbols and that after reading p input symbols, M uses at most $L(p)$ tape cells. Consider inputs of the form $0^{2^{r-1}} w_1 w_2 \cdots w_{k-1}$ such that each w_i, $1 \leq i < k$, is in $\{1, 2\}^r$; for no $1 \leq i < j < k$ does $w_i = w_j$; and for $1 \leq i \leq k - 2$, the integer value of w_i is less than that of w_{i+1}. Since there are n possible values for w_i, the total number of inputs of this form is $n(n - 1) \cdots (n - k + 2)/(k - 1)!$ For each of these inputs the internal configuration of the machine must be unique; otherwise, we could choose w_k so that one word was in S_2 and the other not, while M would accept both or neither. If we let $p = n/2 + (k - 1)r$, then

$$sL(p)t^{L(p)} \geq n(n - 1) \cdots (n - k + 2)/(k - 1)!$$

Taking logarithms,

$$L(p)\log_2 t + \log_2 L(p) + \log_2 s \geq \sum_{i=0}^{k-2} \log_2 \left(\frac{n - i}{k - i - 1}\right).$$

The left-hand side of the above is bounded above by $cL(p)$ where $c = 1 + \log_2 t + \log_2 s$. Also the right-hand side is bounded below by $(k - 1)\log_2[(n - k)/k]$. Thus

$$cL(p) \geq (k - 1) \log_2 \left(\frac{n - k}{k}\right).$$

For large n, $k - 1$ is at least $n/(3 \log_2 n)$, and $(n - k)/k \geq n/(2k)$. Also $\log_2 [n/(2k)] \geq \frac{1}{2} \log_2 n$ for large n. Thus

$$cL(p) \geq \frac{n}{6 \log_2 n} \log_2 n.$$

But $n \geq p$, so

$$L(p) \geq \frac{p}{6c}.$$

There are an infinity of integers p for which a word of length p, $0^{2^{r-1}} w_1 w_2 \cdots w_{k-1}$, is a prefix of a word in S_2. Therefore $L(p) \geq p$. We may conclude that if M accepts S_2, then the amount of tape used by M grows at least linearly.

For the two-way case, if the storage tape is bounded then only regular sets are accepted, and thus the deterministic and nondeterministic models are equivalent. However for two-way Turing machines of all other complexity classes it is an open problem as to whether or not the deterministic and nondeterministic models are equivalent.

5. *Hierarchy of Complexity Classes Below Log n*

The "diagonalization" proof of a dense hierarchy of complexity classes that was given in [1] breaks down when the complexity classes are below log n. We offer a proof of the existence of such a hierarchy in this region, not only for off-line Turing machines but for the classes of 2NTM and 2DTM which are not restricted to be halting.

Say a function $L(n)$ is *constructible* if there is some halting 2DTM M which is of tape complexity $L(n)$ but of no tape complexity that is smaller than $L(n)$ for any value of n. Say M *constructs* $L(n)$.

THEOREM 5. *If $L(n)$ is a constructible function and $L(n) \leq \log_2(n/2)$ for all $n > 1$, then there is a set S such that S is accepted by a halting 2DTM of tape complexity $L(n)$ but by no 2NTM of tape complexity $Q(n)$ if $L(n) > Q(n)$.*

PROOF. Let M construct $L(n)$, and let the input alphabet of M be A. Let A' be the alphabet consisting of a_b, a_c, and a_d for each a in A. For each a_x in A', let $h_1(a_x) = a$ and $h_2(a_x) = x$. Extend the homomorphisms h_1 and h_2 to words over A' in the obvious way. Let S be the language consisting of those words w over A' having the following property: If w is of length n and $h_1(w)$ causes M to use k storage cells, then $h_2(w)$ is of the form $ud^{n-2^{k+1}}u$ for some word u of length 2^k over $\{b,c\}$.

It is easy to see that S is accepted by a halting 2DTM of tape complexity $L(n)$. The machine first lays off k cells of memory by simulating M on $h_1(w)$, then checks the format of w (to see that $h_2(w)$ is of the correct form), and finally compares the initial and terminal strings of b's and c's symbol by symbol, using its storage tape to measure positions within the initial and terminal subwords of b's and c's.

Suppose S were accepted by a 2NTM of tape complexity $Q(n)$ where $L(n) > Q(n)$.

Let N be such a Turing machine with s states and t tape symbols that accepts S. The number of storage states available for inputs of length n or less is at most $sQ(n)t^{Q(n)}$. Thus there are at most $4^{[sQ(n)t^{Q(n)}]^2}$ transition matrices for words in alphabet A' of length up to n.

Let y be a word of length n over alphabet A such that M uses exactly $L(n)$ storage cells when processing y. Then consider all w in L such that $h_1(w) = y$. There are clearly $2^{2^{L(n)}}$ such w. For each of these words, the terminal string of b's and c's must have a unique transition matrix, or else a word not in L would be accepted. Hence

$$2^{2^{L(n)}} \leq 4^{[sQ(n)t^{Q(n)}]^2}.$$

Take logarithms twice:

$$L(n) \leq 1 + 2(\log_2 s + \log_2 Q(n) + Q(n) \log_2 t).$$

From the above it is immediate that there is a constant k such that $L(n) \leq kQ(n)$. But then $L(n) > Q(n)$ could not be true. We may conclude that N does not exist, proving the existence of hierarchies in the range desired for both deterministic and nondeterministic two-way Turing machines.

6. *Closure Under Boolean Operations*

The sets of languages of particular complexity classes for the various models are sometimes closed under the operations of union, intersection, and complementation. (We shall say hereafter that the complexity classes, rather than the set of languages accepted by them, are closed.)

The complexity classes for nondeterministic models are easily seen to be closed under union. A machine is designed which "guesses" which of two other machines to simulate and accepts if either accepts.

For the two-way Turing machines, closure under union and intersection is shown by simulating two machines one at a time and accepting if either or both, respectively, accept. For union one obviously requires that the machines simulated are halting. The complexity classes of 2DTM that can be modified to halt are also closed under complementation, as can easily be shown.

The one-way classes can all be modified to be halting, and closure under complementation is easily shown for the 1DTM. The classes for the 1NTM and 1DTM are also closed under union and intersection. A machine is constructed to simulate two others on two tracks of its storage tape. If both are halting they can be simulated in turn while the input head rests at each input symbol. These results are summarized in the following theorem.

THEOREM 6. (a) *All complexity classes for the 1DTM are closed under union, intersection, and complementation.*

(b) *All complexity classes for the 1NTM and 2NTM are closed under union and intersection.*

(c) *All complexity classes for the 2DTM are closed under intersection. Complexity class $L(n)$ is closed under union and complementation, unless, perhaps, if $\log n > L(n)$.*

There is no reason to suspect that the results not stated in Theorem 6 are true. However we have only been able to show a negative result in one simple case.

THEOREM 7. *Let S_3 be the language $\{wcw \mid w \text{ in } \{0,1\}^*\}$. Then $\bar{S}_3 = \{0, 1, c\}^* - S_3$ is accepted by a 1NTM of tape complexity $\log n$, but S_3 is not accepted by any 1NTM of tape complexity $L(n)$ unless $L(n) \geq kn$ for some constant k. Hence if $n > L(n)$ but $L(n) \geq \log n$ for all n, then complexity class $L(n)$ for the 1NTM is not closed under complementation.*

PROOF. A word w in $\{0, 1, c\}^*$ is in $\bar{S}_3$ for one of three reasons:

(1) w does not contain exactly one c.

(2) w is of the form w_1cw_2, w_1 and w_2 in $\{0, 1\}^*$, but w_1 and w_2 are of different lengths.

(3) w is of the form w_1cw_2, w_1 and w_2 in $\{0, 1\}^*$, and w_1 and w_2 are of the same length, but $w_1 \neq w_2$.

We can design a 1NTM M_3 of tape complexity $\log n$ which checks (1) in its finite control. If w contains one c, M_3 uses its storage tape as a binary counter to check (2). While counting, M_3 "guesses" a position within w_1, recording the symbol there and the distance of that position from the left end of w_1. That symbol is checked against the corresponding symbol of w_2. If unequal, M_3 determines that (3) is satisfied.

Now suppose that there is a 1NTM M accepting S_3. Suppose there are words u_1 and u_2 in $\{0, 1\}^*$, and a storage state Q of M such that if M is in storage state Q, and either $u_1\$$ or $u_2\$$ is the remaining portion of input tape, then T would accept. Then Q could not be accessible from the initial configuration of M for any input sequence, or M would accept a word not in S_3.

Thus for any storage state Q, there is at most one input sequence leading to acceptance from Q. Moreover for any word u in $\{0, 1\}^*$, there must be some storage state Q from which, if $u\$$ is the remaining portion of input, M will enter an accepting state. Also Q must be accessible from the initial configuration of T when uc is the initial portion of the input tape.

We may conclude that the number of storage states accessible from the initial state with an input of the form uc where u is in $\{0, 1\}^n$ is at least 2^n. If M has s states and t storage symbols, we must have

$$sL(n)t^{L(n)} \geq 2^n,$$

or

$$\log_2 s + \log_2 L(n) + L(n) \log t \geq n.$$

It immediately follows that there is a constant k such that $L(n) \geq kn$.

7. *Conclusions*

We have demonstrated the following results concerning tape-bounded Turing machines.

(1) The lower bounds on tape complexity for nonregular sets given in [1] for the off-line and on-line machines do not depend on either the halting assumption or determinism.

(2) There exist dense hierarchies of both the deterministic and nondeterministic two-way devices in the range between log log n and log n.

(3) Tape complexity class $L(n)$ for the one-way nondeterministic Turing machine is larger than complexity class $L(n)$ for the one-way deterministic Turing machine if $L(n)$ is not bounded above by a constant and varies with n at a rate less than linear.

(4) Tape complexity class $L(n)$ for the 1NTM is not closed under complementation if $L(n) \geq \log n$ and $n > L(n)$.

There are many unanswered questions proposed by the simple theorems we have presented. Among them are: Can 2NTM's and 2DTM's of tape complexity between log n and log log n be modified to be halting? Are the 2NTM's and 2DTM's of given complexity class equivalent? Are complexity classes for 2NTM's closed under complementation?

It might be asked why proving (3) and (4) for the 1NTM was so easy while answering the last two questions appears difficult. Intuitively, the answer lies in the fact that there is only one known way to prove that a language is not acceptable by a one-way machine of tape complexity $L(n)$. If $L(n)$ is small, it can be shown that there is not enough storage space to store an initial portion of the input so that each initial portion has a unique representation in storage. If the language is defined by some rule which involves the comparison of initial and final portions of the input, the machine might not be able to recognize the language because it does not know precisely what the initial portion of the input was when it is time for the final portion to be scanned. This argument breaks down if $L(n) \geq n$, since in that case the entire input may be stored on the working tape.

There is a similar argument used for showing certain sets not acceptable by a 2NTM of tape complexity $L(n)$ if $\log n > L(n)$. If $\log n > L(n)$ then not all suffixes of all inputs can have distinct transition matrices. In this case the accepting device cannot always determine what the terminal portion of the input is and carry this information to the initial portion of the input. The argument breaks down unless $\log n > L(n)$, because it is then possible for each suffix to have its own unique transition matrix.

REFERENCES

1. Hartmanis, J., Stearns, R., and Lewis, P. M. Hierarchies of memory limited computations. *IEEE Conf. Record on Switching Circuit Theory and Logical Design*, 1965, pp. 179–190.
2. Lewis, P. M., Stearns, R., and Hartmanis, J. Memory bounds for the recognition of context free and context sensitive languages. *Ibid.*, 1965, pp. 191–202.
3. Rabin, M. O., and Scott, D. Finite automata and their decision problems. *IBM J. Res. Develop. 3* (1959) 115–125.

RECEIVED JULY, 1967; REVISED JULY, 1968