A NOTE ON ONE-WAY AND TWO-WAY AUTOMATA

J. Hartmanis

Technical Report

No. 69-43

September 1969

Department of Computer Science
Cornell University
Ithaca, New York 14850

A NOTE ON ONE-WAY AND TWO-WAY AUTOMATA

J. Hartmanis

Department of Computer Science
Cornell University
Ithaca, New York 14850

ABSTRACT

The purpose of this note is to show that there exist non-regular languages whose memory requirements for recognition by one-way and two-way automata differ by a double exponential and that this difference cannot be exceeded.

INTRODUCTION

Early in the study of finite automata [1] it was shown that permitting the finite automaton to move its read-head in both directions on the input tape did not increase the automaton's computational capabilities. That is, for any two-way finite automaton which accepts a language L there exists a one-way automaton which accepts the same language. Furthermore, it was shown [2] that any two-way automaton with n states could be replaced by a one-way automaton with no more than $(n+1)^{n+1}$ states. Thus for regular languages the memory requirements for one-way and two-way recognizers could not differ by more than an exponential. In a later paper Karp [3] investigated the problem of approximating non-regular languages by finite automata, by determining the memory required for a finite automaton to recognize correctly the strings of a non-regular language up to length n. Again the same exponential relationship was established between the memory requirements for one-way and two-way automata which approximate a non-regular language up to length n.

These results seem to suggest (as is sometimes assumed) that the difference in memory requirements between one-way and two-way recognizers cannot exceed an exponential amount even for non-regular languages. In the next section we show that for non-regular languages the memory required by one-way and two-way recognizers can differ by a double exponential and observe that it can not differ by more than this amount.

THE DOUBLY EXPONENTIAL MEMORY GAP

In this section we establish the existence of a doubly exponential memory gap between one-way and two-way recognizers of non-regular languages.

In the following discussions we are considering automata with a read-only input tape on which we place the input sequences over some finite alphabet. The one-way automaton can scan the input tape only once, say from left-to-right, whereas the two-way automaton can move its reading head in either direction on the input tape.

The specific organization of the memory and the state-to-state transitions of the automaton are of little interest to us in this note and we define the automaton only for the sake of completeness.

A <u>one-way automaton</u> (called a sequential machine in [3]) is a 5-tuple $M = (A, Q, q_o, \delta, T)$, where A is the finite input alphabet, Q is the (possibly infinite) set of states, q_o is the starting state, $\delta: Q \times A \to Q$ is the next state function, and T is the set of accepting states. A sequence w , $w \in A^*$, is <u>accepted</u> by M if and only if M started on the left end symbol of w leaves the right end of w in an accepting state (without loss of generality we assume that M moves one square to the right with each state transition). A set of sequences D , $D \subseteq A^*$, is said to be <u>accepted</u> by M provided w is accepted by M if and only if w is in D .

A <u>two-way automaton</u> is a 6-tuple $N = (A,Q,q_o,\delta,m,T)$ which is defined in the same way as a one-way automaton, with the addition of $m:Q\times A \to \{-1,0,1\}$ which at each state transition determines the number of tape squares the reading head is to be moved to the right.

Our <u>measure of memory</u> used in processing a string w, of length n, is the logorithm of the number of different states entered by M (on N) during the computation. With each machine M (or N) we associate the memory count:

$$M(n) = \log_2[\text{number of different states entered by } M \text{ in processing all } w \text{ in } A^* \text{ such that } \ell(w) \le n] .$$

We now define the set D on which a doubly exponential memory gap is achieved between one-way and two-way recognizers. The set D is defined on a six-symbol alphabet,

$$A = \left\{ {0 \atop 0}, {0 \atop 1}, {1 \atop 0}, {1 \atop 1}, {a \atop 0}, {a \atop 1} \right\} ,$$

and we think of sequences over A as two sequences written (one above the other) on two separate tracks of the input tape:

$$D = \left\{ \begin{matrix} 1 & a10a11 & a & 1 & 0 & 00a10 & 1 & a & \ldots a10^k & a \\ x_1 x_2 & \ldots & x_k & x_1 & x_2 & & x_k & z_1 & \ldots & z_r \end{matrix} \middle| \, k = 1,2,3,\ldots \right\} .$$

Thus the "upper" sequences in D all are sequences consisting of binary representations of the integers $1,2,3,\ldots,2^k$, separated by the marker "a" :

$$1a10a11a100a\ldots a10^k a \ ;$$

the corresponding lower part is any binary sequence

$$u = wwt$$

such that $\ell(w) = k$.

Theorem. The set D cannot be recognized by any one-way automaton with

$$M(n) < n \ .$$

There exists a two-way automaton N which recognizes D with

$$N(n) \leq C[\log_2 \log_2 n] \quad \text{for} \quad n \geq 4 \ .$$

Proof. First, we prove that for a one-way recognizer M of D we must have $M(n) \geq n$. This is done by formalizing the intuitive observation that, since M does not know how long the input sequence will be, it must remember the lower part of any prefix read in order to compare it with what follows. There are 2^n different prefixes of length n in D and thus M must have entered at least 2^n different states in processing these sequences. But then $M(n) \geq n$. More precisely, assume that M has scanned the two prefixes,

$$\begin{array}{l} 1\ a10a \ldots a \\ x_1x_2 \ldots x_n \end{array}$$

and

$$\begin{array}{l} 1\ a10a \ldots a \\ y_1y_2 \ldots y_n \end{array}$$

with

$$x_1x_2 \ldots x_n \neq y_1y_1 \ldots y_n \ .$$

Then there exists a sequence which can be concatenated with these prefixes so that the first sequence is in D and the second sequence is not in D ;

$$\begin{array}{l} 1\ a10a \ldots a \quad \ldots \quad a10^n\ a \\ x_1x_2 \ldots x_nx_1x_2 \ldots x_nz_1z_2 \ldots z_r \end{array}$$

is in D and

$$\begin{array}{l} 1\ a10a \ldots a \quad \ldots \quad a10^n\ a \\ y_1y_2 \ldots y_nx_1x_2 \ldots x_nz_1z_2 \ldots z_r \end{array}$$

is not in D . If M is in the same state after scanning these two different prefixes, then, we conclude, M will be in the same state after scanning the extended sequences. Since only one of these is in D M would not recognize D . This implies that after scanning any two different prefixes (of sequences in D) the automaton recognizing D must be in different states. Since there are 2^n such different prefixes, we conclude that

$$M(n) \geq \log_2 2^n = n .$$

This proves the first part of the theorem.

Next we show that D can be recognized by a two-way automaton N with

$$N(n) \leq C \cdot \log_2 \log_2 n , \text{ for large } n .$$

First, note that D is so constructed that by successively counting away from two consecutive "a" markers an automaton with three counters (and a few control states) can check digit by digit whether the upper sequence is properly constructed. At each stage N checks whether the two binary sequences under inspection represent the integers r and r + 1 , respectively. Since the length of the representation of r is less than or equal to k + 1 and since

$$k + 1 < \log_2 n ,$$

for any sequence in D , we see that none of the counters will ever exceed the count of $\log_2 n$. After M has found that the upper sequence is properly formed up to the representation of a power of two, say, 2^s , it checks, using the three counters, whether the lower sequence has the form

$$u = wwz , \ \ell(w) = s .$$

After this is determined, N resumes the further checking of the upper sequence making sure that it leaves the last symbol of the previously varified segment in an accepting state if $u = wwz$, $\ell(w) = s$, and not in an accepting state otherwise. This insures that N leaves sequences in D in an accepting state and all other sequences in non-accepting states. Thus D can be recognized by a two-way device entering no more than

$$C_1 + C_2k^3 \leq C_1 + C_2[\log n]^3$$

states while processing input sequences of length n (C_1 accounts for the handling of very short sequences, k^3 comes from the fact that M has three counters counting up to k and C_2 accounts for the control states). But then

$$N(n) \leq \log_2(C_1 + C_2[\log n]^3) \leq C \log_2\log_2 n ,$$

for $C > 0$ and for $n \geq 4$. This completes the second part of the proof. The two parts jointly establish the existence of a doubly exponential memory gap between one-way and two-way automata recognizing the set D .

The next result will be used to show that the doubly exponential memory gaps between one-way and two-way recognizers cannot be exceeded.

Theorem. Let N be a two-way automaton with an unbounded $N(n)$. Then

$$\sup_{n\to\infty} \frac{N(n)}{\log_2 \log_2 n} > 0 .$$

Proof. The proof is just a slight generalization of a proof in [4] or a proof in Section 10.5 of [5], and involves a simple counting argument on crossing sequences.

We now observe that any F , $F \subseteq A^*$, can be recognized by a one-way automaton M with $M(n) \leq Cn$. Thus we see that for one-way and two-way automata the memory gap cannot exceed the double exponential one and that for the set D this gap is achieved.

Note. The fact that any F , $F \subseteq A^*$, can be accepted by a one-way automaton with $M(n) \leq Cn$ (even if F is not recursively enumerable) may not be intuitively satisfying and we may expect that for more "realistic" models the doubly exponential memory gap could be exceeded. It turns out that this is not the case for "realistic" models. For example, for tape-bounded computations (as defined in [4] or [5]), the difference in memory requirements between one-way and two-way automata disappears for tape length $L(n) \geq n$. Thus again, for tape-bounded computations there exists a doubly exponential memory gap between one-way and two-way models and this gap cannot be exceeded.

REFERENCES

1. M. O. Rabin and D. Scott, "Finite Automata and their Decision Problems," in Sequential Machines: Selected Papers, E. F. Moore, Ed., Addison-Wesley, Reading, Mass., 1969, pp. 63-91.

2. J. C. Shepherdson, "The Reduction of Two-Way Automata to One-Way Automata," in Sequential Machines: Selected Papers, E. F. Moore, Ed., Addison-Wesley, Reading, Mass., 1969, pp. 92-97.

3. R. M. Karp, "Some Bounds on the Storage Requirments of Sequential Machines and Turing Machines," JACM 14, 3(July 1967), pp. 478-489.

4. J. Hartmanis, P. M. Lewis, and R. E. Stearns, "Hierarchies of Memory Limited Computations," 1965 IEEE Conference Record on Switching Circuit Theory and Logical Design, Ann Arbor, Michigan, pp. 179-190.

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