LABORATORY FOR COMPUTER SCIENCE
(formerly Project MAC)

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

MIT/LCS/TM-75

K+1 HEADS ARE BETTER THAN K

ANDREW C. YAO

RONALD L. RIVEST

SEPTEMBER 1976

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

MIT/LCS/TM-75

K+1 HEADS ARE BETTER THAN K

A.C. Yao
R.L. Rivest

September 1976

Massachusetts Institute of Technology
Laboratory for Computer Science
(formerly Project MAC)
Cambridge, Massachusetts
02139

K+1 Heads are Better than K†

Andrew C. Yao* and Ronald L. Rivest**
Massachusetts Institute of Technology, Cambridge, Mass. 02139
April 1976

Abstract:

There are languages which can be recognized by a deterministic (k+1)-headed one-way finite automaton but which cannot be recognized by a k-headed one-way (deterministic or non-deterministic) finite automaton. Furthermore, there is a language accepted by a 2-headed nondeterministic finite automaton which is accepted by no k-headed deterministic finite automaton.

Keywords: multihead finite automata.

† This report was prepared with the support of the National Science Foundation grant no. DCR74-12997-A01, National Science Foundation grant no. MCS76-14294, and the Office of Naval Research N00014-76-C-0366.

* Department of Mathematics and Laboratory for Computer Science.
** Laboratory for Computer Science.

1. Introduction and Definitions

We consider the class of languages recognized by k-headed one-way finite automata (k-FA's). These devices consist of a finite-state control, a single read-only input tape with an endmarker \$, and k one-way reading heads which begin on the first square of the input tape and independently move towards the endmarker under the finite-state control. The language accepted by a k-FA is precisely the set of words x such that there is some computation of the k-FA beginning with x\$ on the input tape and ending with the k-FA halting in an accepting state. The deterministic variety of k-FA's will be denoted as k-DFA's. The notion of a multihead finite automaton was apparently first described by Piatkowski [6], and was soon thereafter extensively studied by Rosenberg [1,7].

We assume that the finite control cannot detect coincidence of the heads. Such a capability increases the class of languages recognized by multihead automata somewhat. For example, the language $\{0^{n^2} \mid n \geq 1\}$ can be recognized by a 3-DFA which can detect coincidence (this was pointed out to the authors by A.R. Meyer), but cannot be recognized by any k-FA without this capability [3]. As it turns out, however, our proof that k+1 heads are more powerful than k heads holds even if the devices are allowed to detect coincidence.

Let R_k (respectively R_k^D) denote the class of languages recognized by k-FA's (respectively, k-DFA's). It is well-known that $R_1 = R_1^D$, and easy to see that $R_1^D \subsetneq R_2^D$ (consider the language $\{x2x \mid x \in \{0,1\}^*\}$). Rosenberg [1] claimed that $R_k^D \subsetneq R_{k+1}^D$ for $k \geq 1$, but Floyd [2] pointed out that Rosenberg's informal proof was incomplete. Subsequently, Sudborough [3,4], and later Ibarra and Kim [5], proved that $R_2 \subsetneq R_3$ and $R_2^D \subsetneq R_3^D$. The main result of this paper is that $R_k^D \subsetneq R_{k+1}^D$ and $R_k \subsetneq R_{k+1}$ (actually, that $R_{k+1}^D - R_k \neq \emptyset$) for all $k \geq 1$. That is, we show that "k+1 heads are better than k" in the sense that there is for each k, a language L which can be recognized by a (k+1)-DFA which can be recognized by no k-FA (even if the k-FA can detect coincidence). Our proof uses a counting argument and some observations due to Rosenberg about possible sequences of head movements.

We also show that $R_k^D \subsetneq R_k$ for $k \geq 2$; adding nondeterminism to multihead finite automata strictly increases the class of languages they can recognize.

We actually show that

$$R_2 - \left(\bigcup_{1\le k<\infty} R_k^D \right) \ne \emptyset;$$

there is a language recognized by a 2-FA but by no k-DFA.

2. The Hierarchy Theorem

Consider the language L_b, defined for positive integers b, over the alphabet $\{0,1,*\}$:

$$L_b = \{w_1 * w_2 \ldots * w_{2b} \mid (w_i \in \{0,1\}^*) \wedge (w_i = w_{2b+1-i}) \text{ for } 1 \le i \le 2b\}.$$

Theorem 1. The language L_b is recognizable by a k-FA if and only if $b \le \binom{k}{2}$.

Proof: Rosenberg has demonstrated this in the "if" direction; as the first head traverses $w_{2b+2-k},\ldots,w_{2b}$ the remaining k-1 heads can be used to compare these words with $w_{k-1},\ldots,w_1$, respectively. These k-1 heads can then be positioned at the beginning of w_k and the same procedure used inductively to verify that $w_k * \ldots * w_{2b+1-k}$ is in L_{b+1-k}. Note that this procedure is deterministic.

To prove the theorem in the other direction, we derive a contradiction by assuming that a k-FA $\mathfrak{M}$ accepts every word in L_b^n for $b > \binom{k}{2}$ and n sufficiently large, where L_b^n is the language

$$L_b^n = \{w_1 * w_2 \ldots * w_{2b} \mid (w_i \in \{0,1\}^n) \wedge (w_i = w_{2b+1-i}) \text{ for } 1 \le i \le 2b\}.$$

Specifically, we show that if $\mathfrak{M}$ accepts every word in L_b^n then $\mathfrak{M}$ accepts some word not in L_b. Since $L_b \supseteq \bigcup_n L_b^n$ the contradiction follows.

A configuration of the k-FA $\mathfrak{M}$ is a (k+1)-tuple $(s,p_1,\ldots,p_k)$ where s is the state of the finite control and p_i is the position of the ith head (where the left-most tape square is position number 1). The type of a configuration $(s,p_1,\ldots,p_k)$ is the k-tuple $(\lceil p_1/(n+1)\rceil,\ldots,\lceil p_k/(n+1)\rceil)$; the ith element q_i of the type specifies that the ith head of $\mathfrak{M}$ is on w_{q_i} or its following delimiter in this configuration when scanning a word in L_b^n.

Let $c_1(x),c_2(x),\ldots,c_{\ell_x}(x)$ be the sequence of configurations of the k-FA $\mathfrak{M}$ during an (arbitrarily selected) accepting computation of a word $x \in L_b^n$. Here ℓ_x is the length of this computation. Let $d_1(x),\ldots,d_{\ell'_x}(x)$ be the subsequence obtained by selecting $c_1(x)$ and all subsequent $c_i(x)$ such that type $(c_i(x)) \neq$ type$(c_{i-1}(x))$. Call $d_1(x),\ldots,d_{\ell'_x}(x)$ the <u>pattern</u> of x. (While the pattern of x depends on which accepting computation of x was selected, this does not matter to our proof; we require only that each word $x \in L_b^n$ be associated with one pattern in this fashion). The pattern of x describes the computation of $\mathfrak{M}$ on input x in a rough fashion - we select only those configurations where some head has just moved to the first character of some subword w_i of x. Using the fact that $\ell'_x \leq k\cdot(2b-1)+1$, we see that the number P of possible patterns is less than

$$(s\cdot(2b(n+1))^k)^{k\cdot(2b-1)+1}$$

where s is the number of states in $\mathfrak{M}$'s finite-state control.

Now we classify the words in L_b^n according to their patterns. There must exist a pattern $\hat{d}_1,\ldots,\hat{d}_{\hat{\ell}}$ which corresponds to a set S_0 of at least $2^{bn}/P$ words.

Rosenberg observed that if $b > \binom{k}{2}$ then for any computation of $\mathfrak{M}$ on an $x \in L_b^n$ there exists an index i such that w_i* and $w_{2b+1-i}*$ (or $w_{2b}\$$ if $i = 1$) are never being read simultaneously. (If a pair of heads is reading such a matched pair of subwords at some point during the computation, then at no other time during the computation could that pair of heads read some other matched pair of subwords. The observation follows since there are only $\binom{k}{2}$ pairs of heads to consider.) The possible values for i are determined entirely by the pattern of the computation. Let i_0 be such a value for the pattern $\hat{d}_1,\ldots,\hat{d}_{\hat{\ell}}$.

Partition the words in S_0 into classes according to the string

$$w_1*w_2*\ldots*w_{i_0-1}*w_{i_0+1}*\ldots*w_{2b-i_0}*w_{2b+2-i_0}*\ldots$$

of characters they contain, exclusive of the matched pair of subwords w_{i_0} and w_{2b+1-i_0}. Let S_1 be a class which contains at least $|S_0|/2^{n(b-1)} \geq 2^n/P$ words, and assume n is large enough so that $|S_1| \geq 2$.

Let $x = x_1*x_2*\dots*x_{2b}$ and $y = y_1*\dots*y_{2b}$ be two distinct words in S_1. By assumption, then

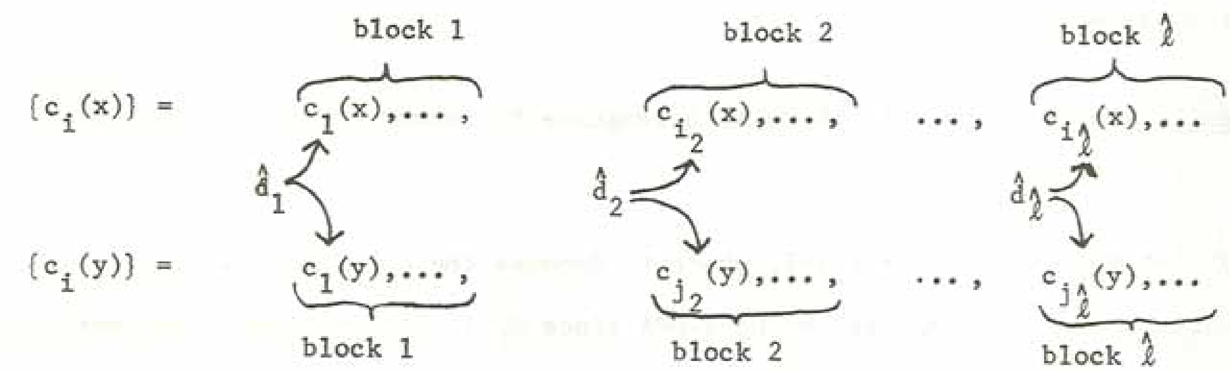
$$(x_i = y_i) \Leftrightarrow i \notin \{i_0, 2b+1-i_0\}.$$

We claim that the word

$$\begin{aligned} z &= z_1*\dots*z_{2b} \\ &= x_1*x_2*\dots*x_{2b-i_0}*y_{2b+1-i_0}*x_{2b+2-i_0}*\dots, \end{aligned}$$

obtained by replacing y_{2b+1-i_0} for x_{2b+1-i_0} in x, will be accepted by $\mathfrak{M}$. However, $z \notin L_b^n$ since $z_{i_0} \neq z_{2b+1-i_0}$, the desired contradiction.

To prove that $\mathfrak{M}$ accepts z we use a "cutting and pasting" argument on the sequence of configurations $c_1(x),\dots$ and $c_1(y),\dots$, to obtain a sequence of configurations for $\mathfrak{M}$ on z such that $\mathfrak{M}$ accepts z. By construction, both $c_1(x),\dots$ and $c_1(y),\dots$ contain the pattern $\hat{d}_1,\dots,\hat{d}_{\hat{\ell}}$ as a subsequence. Divide the sequences $c_1(x),\dots$ and $c_1(y),\dots$ into $\hat{\ell}$ blocks each by beginning a new block with each occurrence of an element $\hat{d}_i$, as in the following figure.



By definition of $\hat{d}_1,\dots$, the subwords of x or y being read change only at the interblock transitions; during any block they remain fixed, and since $\{c_i(x)\}$ and $\{c_i(y)\}$ have the same pattern during the ith block the heads are reading corresponding subwords of x and y.

We construct an accepting computation for $\mathfrak{M}$ of z by selecting successive blocks from $\{c_i(x)\}$, except when $\mathfrak{M}$ during that block would be reading x_{2b+1-i_0} ($\neq z_{2b+1-i_0}$), in which case we select the corresponding block from $\{c_i(y)\}$ (since $y_{2b+1-i_0} = z_{2b+1-i_0}$). This sequence forms a valid computation for z since the last configuration in block i for either $\{c_i(x)\}$ or $\{c_i(y)\}$ yields $\hat{d}_{i+1}$ as the next configuration of $\mathfrak{M}$, and by construction $\mathfrak{M}$ is never reading subwords i_0 and $2b+1-i_0$ simultaneously, so that as far as $\mathfrak{M}$ is concerned, at any instant it cannot distinguish between z and one of x or y. □

In summary, the preceding theorem states that

$$L_{\binom{k+1}{2}} \in R^D_{k+1} - R_k,$$

so that $R^D_k \subsetneq R^D_{k+1}$ and $R_k \subsetneq R_{k+1}$.

3. Consequences of the Hierarchy Theorem.

We present several results which follow more or less directly from the Hierarchy theorem.

Theorem 2. For every $k \geq 1$, there is a language M_k recognized by a 2-FA but by no k-DFA.

Proof. Let $M_k = \overline{L}_b$ for $b = \binom{k}{2}+1$, where $\overline{L}_b$ denotes the complement of L_b. By theorem 1, M_k is recognized by no k-DFA since R^D_k is closed under complementation. However, a 2-FA can recognize M_k by guessing which matched pair of subwords w_i, w_{2b+1-i} are unequal and then verifying this. □

Let

$$M = \{w_1 * w_2 * \ldots * w_{2b} \mid (b \geq 1) \wedge (w_i \in \{0,1\}^* \text{ for } 1 \leq i \leq 2b) \wedge (\exists i)(w_i \neq w_{2b+1-i})\}.$$

Theorem 3. The language M is recognizable by a 3-FA but by no k-DFA.

Proof. To recognize M, send heads one and two to the beginning of some (nondeterministically chosen) subword w_i. Using head one to count the number of words between w_i and the endmarker, simultaneously position head three at the beginning of w_{2b+1-i}. Use heads two and three now to check that $w_i \neq w_{2b+1-i}$.

On the other hand, if $M \in R_k^D$, then for any fixed b, the language

$$M_b = M \cap \{w_1 * \ldots w_{2b} \mid (w_i \in \{0,1\}^* \text{ for } 1 \le i \le 2b)\}$$

would be in R_k^D as well, since this only involves counting up to 2b in addition. But then for any b the language L_b of Theorem 1 would be in R_k^D, since L_b is just the complement of M_b with respect to the regular set $\{w_1 * \ldots * w_{2b} \mid (w_i \in \{0,1\}^*$ for $1 \le i \le 2b\}$, contradicting Theorem 1. ⌐

The theorem can in fact be strengthened as follows:

Theorem 4. There is a language L which can be recognized by a 2-FA but by no k-DFA, for any k. That is, $(R_2 - \bigcup_k R_k^D) \neq \emptyset$.

Proof: We just present the main idea here and leave the details to the reader, as they are quite similar to those of the proof of Theorem 1.

Let $L = \{w_1 * w_2 * \ldots * w_{2b} \mid ((\forall i, 1 \le i \le 2b)$

$$(w_i \in \{0,1\}^* \not{c} \{0,1\}^*)) \wedge [(\exists i,j)(w_i = x \not{c} y \wedge w_j = x \not{c} z \wedge y \neq z)], \text{ for any } b \ge 1\}.$$

That is, each w_i consists of a "tag" field w_i' and a "value" field w_i'' so that $w_i = w_i' \not{c} w_i''$. A word $w_1 * \ldots$ is in L iff there is a pair of words with the same tag field but different value fields. Clearly $L \in R_2$.

To show $L \notin \bigcup_k R_k^D$, consider the subset of L such that the tag field of w_i is the binary representation of min(i,2b+1-i). As in the proof of Theorem 1, there can be constructed a word in this subset of L which the k-DFA will reject, using the fact that there are many words having this tag structure such that $w_i = w_{2b+1-i}$ for $1 \le i \le b$ (and thus not in L). ⌐

Acknowledgements

We would like to thank Albert Meyer and Jeffrey Jaffe for several helpful discussions, and Zvi Galil for pointing out an error in section 3 of an earlier draft of this paper. Greg Nelson at Harvard has also proven a version of the hierarchy theorem, using entirely different methods [8].

References

[1] Rosenberg, A.L., On mutli-head finite automata, IBM J.R. and D., 10, (1966), 388-394.

[2] Floyd, R.W., Review 14, 352 (of above paper by Rosenberg), Computing Reviews, 9, 5 (May 1968), p. 280.

[3] Sudborough, I.H., Computation by multi-head finite automata, IEEE Conf. Record of 12th Annual Symp. on Switching and Automata Theory, Mich. (1971), 105-113.

[4] Sudborough, I.H., Computation by multi-head writing finite automata, Penn. State Univ., Ph.D. thesis, 1971.

[5] Ibarra, O.H. and Kim, C.E., On 3-head versus 2-head finite automata, Acta Informatica 4 (1975), 193-200.

[6] Piatkowski, T.F., N-head finite state machines, (Ph.D. thesis) University of Michigan (1963).

[7] Rosenberg, A.L., Nonwriting extensions of finite automata, (Ph.D. thesis), Report No. BL-39, The Computation Laboratory, Harvard University (1965).

[8] Nelson, C.G., One-Way Automata on Bounded Languages , Harvard Center for Research and Computing Technical Report TR14-76 (July 1976).