ON THE POWER OF REAL-TIME TURING MACHINES UNDER VARYING SPECIFICATIONS*
(Extended abstract)

Paul M.B. Vitányi
Mathematisch Centrum
2e Boerhaavestraat 49
Amsterdam, The Netherlands

ABSTRACT

We investigate the relative computing power of Turing machines with differences in the number of work tapes, heads pro work tape, instruction repertoire etc. We concentrate on the k-tape, k-head and k-head jump models as well as the 2-way multihead finite automata with and without jumps. Differences in computing power between machines of unlike specifications emerge under the real-time restriction. In particular it is shown that k+1 heads are more powerful than k heads for real-time Turing machines.

1. INTRODUCTION

Since the first Turing machine appeared in 1936, there have been many advances in the field. In the late 1950's the *multitape* Turing machine was introduced, often equiped with a separate read-only input tape. Since then we saw the arrival of the *multihead* Turing machine, Turing machines with a *fast rewind square* (also called limited random-access machines) and Turing machines with *head-to-head jumps*, and many others. One common feature in this abundance of models is that they all have a finite control and an unrestricted read-write storage facility. This allows each model, whatever its specification, to compute all recursive functions. Differences in capabilities become apparent if we impose time limitations, and in particular when we demand the machines to operate in *real-time*. As a standard in this area we may take the class of *real-time definable languages* R, which is the class of all languages accepted by multitape Turing machines in real-time, ROSENBERG [1967]. It has been shown that all of the above mentioned variations of Turing machines accept in real-time precisely R. Hence we observe that, within the world of real-time Turing machine-like devices, R plays somewhat the same role as the class of recursively enumerable languages in the world of computability at large. Like in this wider setting, we shall impose restrictions on the machines and observe what happens. In the province of real-time computations, differences in computing power amongst unlike Turing

---

* The results in sections 2 and 3 are taken from VITÁNYI [1979]. The present paper is registered as Mathematical Centre Technical Report IW 132.

machines may come out under variations in instruction repertoire, amount or type of storage devices, in short, under different *specifications*.

The class of real-time definable languages is remarkably extensive (e.g. the set of unmarked palindromes is in R, GALIL [1978]). To prove that a given language is not in R is often hard. Proofs usually rely on an information-capacity argument, see HARTMANIS & STEARNS [1965] and ROSENBERG [1967].

Real-time computations of Turing machines are especially interesting because of their intrinsic feasibility. Originally, they were defined relative to the multitape Turing machines. Most algorithms, however, are more naturally stated in terms of computing models which allows faster memory access. A k-head tape unit consists of a Turing machine with a single storage tape on which k read-write heads operate. P. FISCHER, MEYER & ROSENBERG [1972] proved that one can simulate a k-head tape unit in real-time by a multitape Turing machine with 11k-9 tapes. Later, LEONG & SEIFERAS [1977] improved this to 4k-4 tapes. RABIN [1963] has observed that 2-tape Turing machines are more powerful in real-time than 1-tape Turing machines. (Recall that a 1-tape Turing machine has one input tape and one storage tape with a single head.) AANDERAA [1974] demonstrated that k+1 tapes are more powerful than k tapes in real-time. Together with the LEONG & SEIFERAS' result this shows that more heads will yield additional power in real-time. Specifically, it follows that a (4k-3)-head tape unit is more powerful in real-time than a k-head tape unit. We shall show that AANDERAA's result implies that a (k+1)-head tape unit is more powerful than a k-head tape unit in real-time, section 2.

In ROSENBERG [1967] several closure properties of R are investigated. We investigate such questions for the classes R(k) (languages recognized by k-tape real-time Turing machines), $R^H(k)$ (languages recognized by k-head real-time Turing machines) and $R^J(k)$ (languages recognized by k-head real-time Turing machines with head-to-head jumps). Furthermore, we shall consider the relations between R(k), $R^H(k)$ and $R^J(k)$, sections 3 and 5.

In SAVITCH & VITÁNYI [1977] it was shown that a k-head jump Turing machine can be simulated in linear time by an (8k-8)-tape Turing machine. KOSARAJU [1979] has claimed a proof that jump Turing machines can be simulated in real-time by multitape Turing machines at the cost of many tapes in the latter pro head in the former machine. In section 4 we show that the analog of this result does not hold if we restrict ourselves to 2-way multihead finite automata. The sample languages we use to prove this result are interesting in their own right, since they give once more an indication how wrong our intuition can be with respect to which languages belong to R and which languages do not.

But for RABIN's and AANDERAA's results, all results in the area of models of real-time Turing machines are about feasibility of simulating one type of machine by another one. Virtually nothing is known about the nonfeasibility of certain computations, which are possible on a machine of specification A, by a machine of specifica-

tion B. Obvious open problems in this area of specified Turing machines are, for instance:
$R(2) \subset R^H(2)$; $R^H(k) \subset R^H(k+1)$; $R^J(k) \subset R^J(k+1)$; $R(k) \subset R^H(k)$; $R(k) \subset R^J(k)$;
$R^H(k) \subset R^J(k)$? Some of these questions we shall decide, or alternatively, show some interdependence among seemingly unrelated questions.

For formal definitions and so on concerning multitape- and multihead Turing machines, real-time computations, etc. we refer to ROSENBERG [1967], FISCHER, MEYER & ROSENBERG [1972] and LEONG & SEIFERAS [1977]. In this paper we do not give all proofs; complete proofs and additional results shall be provided in a final version to appear elsewhere.

2. k+1 HEADS ARE BETTER THAN k HEADS IN REAL-TIME

AANDERAA [1974] proved by a very complicated argument that there is, for each $k \geq 0$, a language A_{k+1} which can be recognized by a (k+1)-RTTM but not by a k-RTTM. For completeness we define A_{k+1} below by a real-time algorithm which accepts it using k+1 pushdown stores. The input alphabet is $\Sigma_{k+1} = \{0_i, 1_i, P_i \mid 1 \leq i \leq k+1\}$. The algorithm is as follows:

```
"ACCEPTENABLED := TRUE;
Initialize k+1 stacks to empty;
REPEAT FOREOVER
  CASE NEXTINPUTLETTER OF
  $0_i$: Push 0 in stack i
  $1_i$: Push 1 on stack i
  $P_i$: IF stack i empty
      THEN ACCEPTENABLED := FALSE and reject input
      ELSE BEGIN
           pop stack i;
           IF element popped was 1
           AND ACCEPTENABLED
           THEN accept input
           ELSE reject input
           END
  ENDCASE"
```

The strategy used to prove that k+1 heads are more powerful in real-time than k heads (on a single tape) is, by a judicious choice of input, to force the heads so far apart that for a given recognition problem the k-head unit must act like a k-tape Turing machine since the heads will never read each others writing.

THEOREM 2.1. *There is a language which is recognized by a* k+1 *head real-time Turing machine but not by any* k *head real-time Turing machine.*

PROOF. By induction on the number of heads. (k=0 is obvious).

k=1. The language A_2 cannot be recognized by a 1-tape (= 1-head) real-time Turing machine, but can be recognized by a 2-tape (and hence by a 2-head) RTTM. Set $H_2 = A_2$.

$k > 1$. Suppose the theorem is true for all $j < k$. Hence, in particular there is a language H_k such that H_k is recognized by a k-head RTTM but not by a (k-1)-head RTTM. Define H_{k+1} as follows:

$$H_{k+1} = H_k \cup H_k * A_{k+1}$$

where $*$ is a special symbol not in the alphabet of A_i, $i \geq 2$.

Let M_k be a k-head RTTM claimed to recognize H_{k+1}. Present M_k with strings of the form

$$w = a_1^{(2)} a_2^{(2)} \ldots a_{n_2}^{(2)} * a_1^{(3)} a_2^{(3)} \ldots a_{n_3}^{(3)} * \ldots * a_1^{(k+1)} a_2^{(k+1)} \ldots a_{n_{k+1}}^{(k+1)}$$

$$= w_2 * w_3 \ldots * w_{k+1}$$

such that w_i is over the alphabet of A_i, $2 \leq i \leq k+1$. During the processing of w_2, M_k must recognize A_2. Since A_2 cannot be recognized by a 1-head RTTM, the distance between the outermost heads on the storage tape of M_k must grow larger than any given constant c_2 for a suitable choice of w_2. Hence, subsequent to the processing of w_2, we can single out a tapesegment of length at least c_2/k tape squares, contained by the tapesegment delineated by the outermost heads, such that no tape square of the former segment is scanned by a head. Choose c_2 later so that $c_2/k > 2 \Sigma_{i=3}^{k+1} (n_i+1)$. Therefore, for the remainder of the computation on w, M_k consists in effect of at best a $k_1^{(1)}$-head and a $k_2^{(1)}$-head tape unit, $k_1^{(1)}, k_2^{(1)} \geq 1$ and $k_1^{(1)} + k_2^{(1)} = k$, where $k_1^{(1)}$ is the number of heads left of the unscanned tapesegment and $k_2^{(1)}$ is the number of heads right of it, at the end of processing w_2. Now M_k is presented with w_3. Since $w_3 \in A_3$ cannot be decided in real-time by 2 single-headed tapes, M_k must use one, or both, of its remaining tape units in an essential way during the processing of w_3. I.e., for at least one of the tape units (and one containing more than one head), say the $k_1^{(1)}$-head unit, the distance between the outermost heads must grow larger than any given constant c_3 for a suitable choice of w_3. Hence, subsequent to the processing of w_3, we can single out a tapesegment, no square of which is scanned by a head and of length at least $c_3/k_1^{(1)}$, which is in between the outermost heads of this $k_1^{(1)}$-head tape unit. Now choose c_3, and hence w_3, later so that $c_3/k_1^{(1)} > 2 \Sigma_{i=4}^{k+1}(n_i+1)$. Similar to before, we now divide the $k_1^{(1)}$ heads into $k_1^{(2)}$ and $k_2^{(2)}$ heads to the left and right, respectively, of the latter nonscanned tapesegment, and we observe that, for the remainder of the computation on w, M_k now consists in effect of a $k_1^{(2)}$-head-, a $k_2^{(2)}$-head- and a $k_3^{(2)}$-head tape unit, $k_1^{(2)}, k_2^{(2)}, k_3^{(2)} \geq 1$, $k_1^{(2)} + k_2^{(2)} + k_3^{(2)} = k$, $k_1^{(2)} + k_2^{(2)} = k_1^{(1)}$ and $k_3^{(2)} = k_2^{(1)}$.

Repeating the argument we can choose $w_4, \ldots, w_k$ such that after the processing of

w_k we are left in effect with a k-tape RTTM which is required to determine whether $w_{k+1} \in A_{k+1}$. According to AANDERAA [1974], for each k-tape RTTM claimed to recognize A_{k+1} we can construct a word v which fools the machine. Let w_{k+1} be such a word, and choose $c_k, w_k, c_{k-1}, w_{k-1}, \ldots, c_2, w_2$, in that order, so that the above inequalities and conditions are satisfied. Hence w is accepted by M_k iff $w \notin H_{k+1}$ which contradicts the assumption that M_k recognizes H_{k+1}. (The above argument seemingly contains a circularity which might invalidate it. The word v which fools the machine trying to recognize A_{k+1} does not only depend on the finite control but *also* on the initial tape contents. Thus the argument seems to become circular: w_{k+1} depends on $w_2 * w_3 * \ldots * w_k *$, while $w_2, w_3, \ldots, w_k$ depend on the length of w_{k+1}. As it happens, AANDERAA's argument does not need to make any assumptions about the initial tape contents of the k-RTTM assumed, by way of contradiction, to accept A_{k+1}. Hence he proves in fact that for all k-RTTM M there exists a positive integer n such that for all initial tape contents of M there exists a word v of at most length n which fools M. The existence of such a bound n eliminates the apparent circularity from the above argument.) It is easy to see that k+1 pushdown stores can recognize H_{k+1} in real-time. □

Surprisingly, an argument like "H_k is not accepted by a (k-1)-head RTTM and hence $H_{k+1} = H_k \cup H_k * A_{k+1}$ is not accepted by a k-head RTTM" does not work, since we cannot assume a priori that in a k-head RTTM recognizing H_k all heads get pairwise arbitrarily far apart for some input. We could only conclude that all k heads are necessary, but it might very well be that for each time t some heads are near to each other. Then we could be stuck with a set of tape units, one of which is a multihead one, for which AANDERAA's proof might not work.

The situation we have in mind is exemplified by, e.g., the languages E_k, $k \geq 4$, in section 5 (although AANDERAA's proof technique fails there for another reason, as shall be pointed out). As an example of a language which can be recognized by a 4-head RTTM in which there are always 2 heads together, and which probably cannot be recognized by a 4-RTTM, or a 3-head RTTM, we give the language L below. Clearly, we cannot conclude from $L \notin R^H(3)$ (if that is the case) that $L \cup L * A_5 \notin R^H(4)$ just because $A_5 \notin R(4)$. We would need to show at least that A_5 cannot be recognized by a RTTM with one 2-head tape and 2 1-head tapes as storage.

$$L' = \{u_1 w w^R u_2 v v^R u_3 2\ 0^{|u_1 w|} 2\ 0^{2|w|} 2\ 0^{|u_3 v|} 2\ 0^{|v|} \mid u_1 w u_2 v u_3 \in \{0,1\}^*\};$$
$$L\ = \{x \in \{0,1,2\}^* \mid x \text{ is a prefix of a word in } L'\}.$$

For suppose we want to recognize L by a 3-head or a 4-head RTTM. Essentially, up to reading the marker 2 on the input tape, it would seem that we can do nothing more than record the input prefix over {0,1} on the storage tape.

Now if we take $|w|, |v| \in \Theta(n^{2/3})$, $|u_2| \in \Theta(n)$, $|u_1|, |u_3| \in \Theta(n^{2/3})$, where n is the length of the input word, we need 2 heads to check ww^R (since to check ww^R with 1 head takes time $\Theta(n^{4/3})$) and 2 heads to check vv^R (for the same reason). To cross

u_2 with some head takes time $\Theta(n)$, but upon meeting the first letter 2 we have only time $\Theta(n^{2/3})$ left. Hence 4 heads seem necessary, although there always are 2 together. If this conjecture is true, then $L \in R^H(4) - R^H(3)$. But in this case $L \in R^H(4) - R^H(3)$ together with $A_5 \notin R(4)$ does not, without additional considerations, imply $L \cup L * A_5 \notin R^H(4)$.

By the proof method of Theorem 2.1 we precluded this flaw in the argument. Due to the form of A_{k+1}, the line of reasoning works also for A_{k+1} itself. Hence, $A_{k+1} \in R(k+1) - R^H(k)$.

COROLLARY 2.2. *There is a language which can be recognized by* k+1 *pushdown stores in real-time* (*and hence by a* (k+1-RTTM)) *but not by any* k-*head* RTTM.

The relation between tapes and pushdown stores is direct; clearly 2k pushdown stores can simulate k tapes in real-time. Hence from AANDERAA's result we have: (if $R^P(k)$ denotes the class of languages recognizable by k pushdown stores in real-time)

$$\begin{aligned} &R^P(k+1) - R(k) \neq \emptyset;\\ &R^P(k) \subset R^P(k+1) \quad ;\\ &R(k) \subset R(k+1) \quad ;\\ &R(k) \subset R^P(2k) \quad . \end{aligned}$$

By the result above it follows that we can replace R by R^H in the first formula above. It also follows that

$$\begin{aligned} &R(k+1) - R^H(k) \neq \emptyset;\\ &R^H(k) \subset R^H(k+1). \end{aligned}$$

By using LEONG & SEIFERAS' [1977] result we obtain

LEMMA 2.3. $R(k) \subseteq R^H(k) \subset R(4k-4)$.

3. CLOSURE PROPERTIES OF R(k)

In ROSENBERG [1967] several closure properties of the class R of languages accepted by real-time Turing machines were investigated. It appeared that R is closed under union as well as intersection, complementation, suffixing with a regular set, inverse real-time transducer mapping, and minimization. R is not closed under concatenation, Kleene star, reversal, (nonerasing) homomorphism, inverse nondeterministic sequential machine mapping, quotient with a regular set, maximization and prefixing with a regular set.

When we restrict the number of tapes the picture gets different: R(k) is closed under complementation, union as well as intersection with regular sets, suffixing with regular sets, inverse gsm mapping and minimization. R(1) is not closed under union or intersection, nor under inverse real-time transducer mapping.

In this section we will investigate some more closure properties of (number of) tape restricted real-time languages. It will e.g. appear that R(k) is closed under several marked operations; furthermore it often happens that the closure under certain operations of R(k) is in R(2k) but not in R(2k-1). (Proofs to be provided later).

LEMMA 3.1. R(k) *is closed under marked union, marked concatenation and marked Kleene star.*

LEMMA 3.2. *Let* k_1, k_2 *be positive integers such that* $k_1 + k_2 \geq 1$.

(i) R(k) *is not closed under union or intersection, for* $k > 0$. *If we take* $A \in R(k_1)$ *and* $B \in R(k_2)$ *then* $A \cup B, A \cap B \in R(k_1+k_2)$, *but not necessarily* $A \cup B, A \cap B \in R(k_1+k_2-1)$.

(ii) *If* $A \in R(k_1)$ *and* $B \in R(k_2)$ *and the alphabets of* A *and* B *are disjoint, then shuffle* (A,B) $\in R(k_1+k_2)$ *but shuffle* (A,B) *does not need to belong to* $R(k_1+k_2-1)$. *Hence* R(k) *is not closed under shuffle over disjoint alphabets.*

(iii) R(k) *is not closed under inverse real-time transducer mapping. The closure of* $R(k_1)$ *under inverse* k_2-RTTM *mapping is contained in* $R(k_1+k_2)$ *but not in* $R(k_1+k_2-1)$.

(iv) (i)-(iii) *hold also if we replace everywhere* "R" *by* "R^H".

The results in Lemma 3.2 are obtained by reducing the problems to the recognition problem of $A_{k_1+k_2}$.

LEMMA 3.3. *If* $A \in R(0)$ *and* $B \in R(1)$ *then shuffle* (A,B) *does not need to belong to* R. *I.e.,* R *is not closed under shuffle.*
($L = \{\Sigma^* x \Sigma^* 2x^R \mid \Sigma = \{0,1\}, x \in \Sigma^*\} \notin R$ *and an isomorphic language can be obtained as a shuffle of languages in* R(0) *and* R(1).)

According to FISCHER, MEYER & ROSENBERG [1972], the family of multihead RTTM languages equals R and hence the (non) closure properties mentioned before apply. If we look at multihead RTTM languages in $R^H(k)$ the situation is different. Here not more was known than we could readily deduce from the results on R(k) and simulations like LEONG & SEIFERAS [1977]. With the preceding results we obtained more. Also, $R^H(k)$ is closed under complementation, union and intersection with regular sets, suffixing with regular sets, inverse gsm mapping and minimization. Lemma 3.2 holds even if we denote by k only the total number of heads on the storage tapes, and don't take into account the way in which the heads are distributed.

Clearly, $R^H(k)$ is closed under marked union. The markers in an input, due to marked concatenation or marked Kleene star, serve to indicate the beginning of a new task. Accordingly, it seems reasonable to assume that recognizing RTTMs *ignore*, subsequent to reading such a marker, the garbage left on the storage tapes by the preceding computation segment. Under this assumption we can prove Conjectures 3.4 and 3.5.

CONJECTURE 3.4. $R^H(k)$ is closed under marked concatenation iff $R^H(k)$ is closed under marked Kleene star iff $R^H(k) = R(k)$.

A k-head jump Turing machine (cf. SAVITCH & VITÁNYI [1977]) is a k-head Turing machine where at each step the k heads may be redistributed over the scanned tape squares. In SAVITCH & VITÁNYI [1977] it was shown that a k-head jump Turing machine can be simulated in linear time by a (8k-8)-tape Turing machine. KOSARAJU [1979] has claimed that, by a complicated simulation, a k-head jump Turing machine can be simulated in real-time by a multitape Turing machine. It is at present unresolved whether k heads are more powerful than k tapes in real-time. A possibly easier problem is to show that k heads with jumps are more powerful than k tapes in real-time. We will show that these matters are related.

It is easy to see that $R^J(k)$ (the class of languages accepted in real-time by k-head jump Turing machines) is closed under marked concatenation and marked Kleene star. By first feeding A_k, we can always reduce a k-head RTTM to a k-tape RTTM. This, however, is not the case for a k-head jump RTTM. Hence, k jump heads are more powerful than k tapes iff k jump heads are more powerful than k heads. Similarly, if k heads are more powerful than k tapes then k jump heads are more powerful than k heads. Hence we have

<u>CONJECTURE 3.5</u>.
(i) $R(k) \subset R^J(k)$ *iff* $R^H(k) \subset R^J(k)$;
(ii) *if* $R(k) \subset R^H(k)$ *then* $R^H(k) \subset R^J(k)$.

4. REAL-TIME 2-WAY MULTIHEAD FINITE AUTOMATA WITH AND WITHOUT JUMPS

Recall that we saw before that KOSARAJU [1979] has shown that the jump Turing machine as defined in SAVITCH & VITÁNYI [1977] may be simulated in real-time by multitape Turing machines. Hence $R^J = R$ (where $R^J = \bigcup_{k=1}^{\infty} R^J(k)$). In this section we show that for 2-way multihead finite automata the head-to-head jump facility does extend the class of languages accepted in real-time. Incidentally, this shows also that the class of languages accepted by real-time 2-way multihead finite automata is strictly included in R. To obtain the result, we give several example languages which are acceptable in real-time by 2-way 2-head finite automata with jumps, but not by any real-time 2-way multihead finite automaton without jumps. Hence these languages belong to R, and constitute nontrivial examples of the power of the head-to-head jump option. Let in the following $h: \{0,1,\bar{0},\bar{1}\}^* \to \{0,1\}^*$ be a homomorphism which is defined by $h(\bar{a}) = h(a) = a$ for $a \in \{0,1\}$.

$$L_1 = \{\overline{wva}av^R \mid \bar{w}\bar{v} \in \{0,1,\bar{0},\bar{1}\}^*,\ v \in \{0,1\}^*,\ a \in \{0,1\},\ h(\bar{v}) = v\};$$

$$L_2 = \{\bar{w}b\bar{u}cva \mid \bar{w}\bar{u} \in \{0,1,\bar{0},\bar{1}\}^*,\ v \in \{0,1\}^*,\ c \in \{\bar{0},\bar{1}\},\ |\bar{u}| = |v|,$$
$$a \in \{0,1\},\ b \in \{0,1,\bar{0},\bar{1}\},\ h(b) = a\}.$$

The reader will easily figure out more complicated examples along these lines.

Note that L_1, L_2 are linear context free but not deterministic context free.

LEMMA 4.1. L_1, L_2 *are accepted by real-time 2-way 2-head finite automata with jumps.*

PROOF. Let M be a 2-way 2-head finite automaton with jumps as follows. The front head reads from left to right one letter at a time. Whenever this first head reads a barred letter it calls the second head to its present position. This second head starts reading from right to left one letter at a time. So M is able to recognize L_1. A minor variation of M can recognize L_2. □

LEMMA 4.2. L_1, L_2 *are not accepted by any real-time 2-way multihead finite automaton.*

PROOF. Along the same lines as the proof of Theorem 2.1. □

Hence we have:

THEOREM 4.3. (i) *There are languages accepted by real-time 2-way 2-head finite automata with jumps which are not accepted by any real-time 2-way multihead finite automaton without jumps.*
(ii) *The class of languages accepted by real-time 2-way k-head finite automata with jumps properly includes the class of languages accepted by such automata without jumps.*

Computations of 1-way multihead finite automata have been considered by YAO & RIVEST [1978]. They show that k+1 heads are better than k heads for both the deterministic and the nondeterministic versions of the machine. Furthermore, they show that the k-head nondeterministic variety is strictly more powerful than the k-head deterministic one. Recently, JANIGA [1979] studied the analog questions for 2-way real-time multihead deterministic (resp. nondeterministic) finite automata, from now on called 2DRTFA and 2NRTFA, respectively. He obtained, mutatis mutandis, the same results for the 2-way real-time machines as did YAO and RIVEST for the 1-way (no time limit) variety. Whereas the latter used "palindromes" of $\binom{k}{2}$ strings to obtain their result, for the 2-way real-time case the former employed strings of k palindromes. E.g., let PALM be the set of palindromes in $\{0,1\}^*\{2\}\ \{0,1\}^*$. Let $P_k = (\mathrm{PALM}\{*\})^k$. Then P_k is recognized by a (k+1)-head 2DRTFA but not by any k-head 2NRTFA. $\{0,1,2,*\}^*-P_k$ is accepted by a 2-head 2NRTFA but not by any k-head 2DRTFA. Now consider the language $P = \cup_{k=1}^{\infty} P_k$. It is easy to see that P is recognized by a 2-head 2DRTFA with jumps, but that P is not accepted by any multihead 2NRTFA without jumps because of JANIGA's result. Therefore we have:

THEOREM 4.4. *The class of languages accepted by* k*-head* 2NRTFA *with jumps properly includes the class of languages accepted by* k*-head* 2NRTFA *without jumps,* $k \geq 2$. *The same holds for* 2DRTFA*'s (i.e. Theorem* 4.3).

Another matter which we would like to decide is the power of jumps versus nondeterminism for the machines.

THEOREM 4.5. *There is a language acceptable by a 2-head* 2NRTFA *which is not acceptable by any multihead* 2DRTFA *with jumps.*

PROOF. The language L in the proof of Lemma 3.3 was not in R, and hence, by KOSARAJU's [1979] result, is not acceptable by any multihead 2DRTFA with jumps. It is easy to see how L can be accepted by a 2-head 2NRTFA. □

The only question remaining seems to be whether (k+1)-head 2DRTFA's with jumps are more powerful than k-head 2DRTFA's with jumps, and the same matter for the nondeterministic versions. For a proof we might use the language J_k over the alphabet

$$\Sigma = \{0,1\} \times F \times M \times Q,$$

where

$$F = \{f \mid f \text{ is a total function } f\colon \{0,1\}^k \times Q \to \{0,1\}\},$$

$$M = \{m \mid m \text{ is a total function } m\colon \{1,2,\ldots,k\} \times Q \to$$
$$\to \{\text{left},\text{right},\text{no move}\} \text{ and } m(1,q) = \text{right}$$
$$\text{for all } q \in Q\}.$$

The interpretation is as follows. J_k is recognized by a k-head 2DRTFA M with state set Q. Suppose M has an input $s_1s_2\ldots s_is_{i+1}\ldots s_n$ on its tape, $s_i = (a_i,f_i,m_i,q_i) \in \Sigma$, $1 \le i \le n$. At the i-th step the vanguard head 1 of M reads s_i in state $q_{i-1} \in Q$ and outputs $f_i(a_{j1},a_{j2},\ldots,a_{jk},q_{i-1})$ where a_{jh} is the first element of the symbol read by the head h at that moment, $1 \le h \le k$. Subsequently, M repositions head h according to $m_i(h,q_i)$, $1 \le h \le k$, and enters state q_i.

THEOREM 4.6. J_{k+1} *is accepted by a* (k+1)*-head* 2DRTFA *but not by any* k*-head* 2NRTFA *with jumps. Hence* (k+1)*-head* 2DRTFA (2NRTFA) *with jumps are strictly more powerful than* k*-head* 2DRTFA (2NRTFA) *with jumps.*

If we take J'_k equal to J_k but without "left" in the range of $m \in M$ we can similarly prove:

COROLLARY 4.7. J'_{k+1} *is accepted by a* (k+1)*-head* 1DRTFA *but not by any* k*-head* 1NRTFA *with jumps. This implies that all inclusions according to the number of heads in the* 1XRTFA *are proper, where* X $\in$ {D,N,D *with jumps,* N *with jumps*}.

All results in this section hold whether or not we assume end markers, or that the heads can detect coincidence.

We think that Theorem 4.3 also holds for the corresponding Turing machine versions which are allowed to modify the contents of each square on the storage tapes but a bounded number of times, for some fixed constant bound.

5. ON THE RELATIVE POWER OF TAPES, HEADS AND JUMP HEADS IN REAL-TIME TURING MACHINES

One of the major drawbacks in the game of showing a difference in power between two very similar machine types A and B such as considered in this paper, apart from the difficulties involved in giving a proof, is to find some likely candidates for showing a difference between type A and type B. RABIN's [1963] language in $R(2) - R(1)$ did not generalize in an obvious way to show a difference between $R(k+1)$ and $R(k)$, $k > 1$. AANDERAA [1974] provided a uniform construction for a language in $R(k+1) - R(k)$, $k \geq 1$. No likely candidates for showing the difference between, e.g., $R(k)$ and $R^H(k)$ or $R^H(k)$ and $R^J(k)$ have been proposed, except possibly $\{xy2x \mid xy \in \{0,1\}^*\}$ for showing a difference between $R^H(2)$ and $R(2)$. In the present section we propose to fill this gap, besides proving some facts about the candidates. The only languages known to be in $R - R(k)$ are $A_{k'}$, $k' > k$, put unfortunately these languages are not in $R^H(k)$ either. SEIFERAS [personal communication] claims to have proven that $A_{k'} \notin R^J(k)$, and we will proceed on this assumption. Hence the only candidates of which we have negative results are not acceptable either by placing all heads on the same tape nor by adding the jump option. From the existing simulation results it is also clear that there cannot be a single language L which is acceptable by some k-head (jump) RTTM but not by any multitape (multihead) RTTM, thus proving the required results by a single example as in section 4. Now consider a language which is like A_k but with the extra requirement that at all times during the processing of the input w by a k stack machine at least 2 of the stacks are of equal length for w to be accepted. More formally, if $|v|_i$ denotes the number of 0_i's and 1_i's subtracted by the number of P_i's in v, then:

$$E_k = \{w \in \Sigma_k^* \mid w \in A_k \ \& \ \forall v \in \text{prefix}(w) \ \exists i,j (i \neq j \text{ and } 1 \leq i,j \leq k) \ [|v|_i = |v|_j + \delta, -1 \leq \delta \leq +1]\}.$$

<u>LEMMA 5.1</u>. $E_k \notin R(k-2)$, $R^H(k-2)$, $R^J(k-2)$.

<u>PROOF</u>. Suppose, by way of contradiction, that the (k-2)-RTTM M accepts E_k. Now change M to a (k-2)-RTTM M^* which accepts A_{k-1} by having the finite control of M, for every letter $0_{k-1}, 1_{k-1}, P_{k-1}$ read $0_{k-1}0_k, 1_{k-1}1_k, P_{k-1}P_k$, respectively, and speed up the storage handling as much as required. Then A_{k-1} is accepted by the (k-2)-RTTM M^* contradicting known results. $E_k \notin R^H(k-2)$ then follows by Theorem 2.1 and for $E_k \notin R^J(k-2)$ see the introduction of this section. □

(The case $k = 2$ above is obvious since E_2 is not regular.) Note that AANDERAA's proof does not show that $E_k \notin R(k-1)$ since the subset $S\Sigma_k^*$ used in AANDERAA's proof (which in fact shows that no k-RTTM can distinguish between $S\Sigma_k^* \cap A_k$ and $S\Sigma_k^* \cap (\Sigma_k^* - A_k)$) is disjoint from E_k.

<u>LEMMA 5.2</u>. $E_2 \in R(1)$, $E_3 \in R^H(2)$.

PROOF. $E_2 \in R(1)$ is obvious. $E_3 \in R^H(2)$: keep the 3 stacks on different tracks of the recognizing 2-head RTTM M. Whenever there is a change in pairs of equal size stacks, all 3 stacks must be of equal length, otherwise we reject the input. Both heads of M therefore come together with everything to the right of them blank, and therefore the role of the "fat" head, maintaining 2 tracks, can change. □

We conjecture that $E_3 \notin R(2)$. To prove this conjecture would also prove that $R(2) \subset R^H(2)$, a well-known open problem. In general we conjecture that $E_k \notin R(k)$, $k \geq 3$, which for the case $k = 3$ would show that the LEONG-SEIFERAS simulation is optimal for 2 heads. By Lemma 5.1 and the fact that a multihead machine can detect coincidence we have that

LEMMA 5.3. $E_k \in R^H(k) - R^H(k-2)$.

LEMMA 5.4. $E_k \in R^J(k-1)$ *for all* $k > 1$.

COROLLARY 5.5. $E_k \in R^J(k-1) - R^J(k-2)$.

We conjecture that E_k cannot be recognized by a (k-1)-head RTTM for $k \geq 4$. A proof of this fact would show that $R^H(k) \subset R^J(k)$ for $k \geq 3$, leaving open the case $k = 2$. Although we have an upper bound on the recognition of E_k by multihead RTTM's (with respect to the number of heads needed) we have not yet a good upper bound for recognition by multitape RTTM's, except by the crude $E_k \in R(4k-4)$ offered by Lemma 5.3 and the LEONG-SEIFERAS' result.

LEMMA 5.6. $E_2 \in R(1)$; $E_3 \in R(4)$; $E_k \in R(2k-2)$, $k \geq 3$.

We can generalize the above approach in several directions. For instance, by requiring that i of the k stacks have the same height at all times during the processing of the input, Formally,

$$E_{\binom{k}{i}} = \Big\{ w \in \Sigma_k^* \mid w \in A_k \ \&\ \forall v \in \text{prefix}(w)\ \underset{j_1<j_2\cdots<j_i}{\exists j_1, j_2, \ldots, j_i} \in \{1,\ldots,k\} \; [\, \big| |v|_{j_\ell} - |v|_{j_m} \big| \leq 3 \text{ for all } j_\ell, j_m \in \{j_1, j_2, \ldots, j_i\}] \Big\}.$$

These languages are especially suited to jump Turing machines since it is easily seen that:

LEMMA 5.7. $E_{\binom{k}{i}} \in R^J(k-i+1)$.

Furthermore, we can easily show that $E_{\binom{k}{i}} \in R^H(k-i+1)$ provided $i > k/2$; $E_{\binom{k}{i}} \notin R(k-i), R^H(k-i), R^J(k-i)$; and $E_{\binom{k}{i}} \in R^H(k)$ for $i < k/2$. (Some border cases for $i \geq k/2$: $E_{\binom{5}{3}} \in R^H(3)$ and $E_{\binom{5}{4}} \in R^H(2) \subset R(4)$.)

Looking at the above we see there is a relation between the optimality of the real-time simulations of jump heads by heads and heads by tapes and how many tapes or heads are needed to recognize $E_{\binom{k}{i}}$. Let f(k) be the minimum number of tapes (heads) needed for simulating k jump heads in real-time. Then, if we need at least k tapes (heads) for accepting $E_{\binom{k}{i}}$, $i < k/2$, then

$$f(k-i+1) \geq k.$$

Hence the conjecture that we need k or more tapes (heads) to recognize $E_{\binom{k}{i}}$ for $i < k/2$ can be dissolved if we can improve KOSARAJU's result to "less than 2 k tapes (heads) are necessary for the real-time simulation of k jump heads". From the real-time simulation of heads by tapes it follows that $E_{\binom{k}{i}} \in R(4(k-i))$ for $i > k/2$, and therefore e.g. $E_{\binom{k}{3k/4}} \in R(k)$.

Yet another language sequence we might consider is $A_k - E_k$, $k \geq 1$. Since $A_k - E_k$ contains AANDERAA's subset $A_k \cap S\Sigma_k^*$, it follows that $A_k - E_k \notin R(k-1), R^H(k-1), R^J(k-1)$. We also see that $A_k - E_k \in R^H(k), R^J(k)$. With respect to acceptance by k-RTTM's the same upper bounds apply as argued for E_k. This is not so for the languages $A_k - E_k'$, where E_k' is like E_k but the condition of two stack heights being equal only holds at the end of the processing of the input word, i.e.,

$$E_k' = \{w \in \Sigma_k^* \mid w \in A_k \;\&\; \exists_{i \neq j} i,j \in \{1,\ldots,k\}[\,||w|_i - |w|_j| \leq 3]\}.$$

Here we have that $A_2 - E_2' \in R(3)$ but, presumably, that $A_2 - E_2' \notin R(2)$. By the now familiar reasoning, if the latter case is affirmative then $A_2 * (A_2 - E_2') \in R^J(2) - R^H(2)$, settling the question whether or not $R^H(2) \subset R^J(2)$.

Some of the candidates to try for solving the various questions met are given in the table below.

	$R(k) \subset R^H(k)$?	$R^H(k) \subset R^J(k)$?
k = 2 :	$L = \{xy2x \mid xy \in \{0,1\}^*\}$ E_3, $A_2 - E_2'$	$A_2 * (A_2 - E_2')$
arbitrary $k \geq 3$:	E_k, $A_k - E_k'$	E_{k+1}

Acknowledgements. *J. SEIFERAS pointed out to me that the earlier version of the proof of Theorem* 2.1 *may have been prone to circularity of the argument. Discussions with W. SAVITCH were valuable for section* 4.

REFERENCES

AANDERAA, S.O. (1974), *On k-tape versus (k-1)-tape real time computation*, SIAM AMS Proceedings, Vol. 7 (Complexity of Computation), 75-96.

FISCHER, M.J. & A.L. ROSENBERG (1968), *Limited random access Turing machines,* Proceedings 9-th IEEE-SWAT, 356-367.

FISCHER, P.C., A.R. MEYER & A.L. ROSENBERG (1972), *Real-time simulation of multihead tape units,* JACM 19, 590-607.

GALIL, Z. (1978), *Palindrome recognition in real time on a multitape Turing machine,* J. Comp. Syst. Sci. 16, 140-157.

HARTMANIS, J. & R.E. STEARNS (1965), *On the computational complexity of algorithms,* Trans. AMS 117, 285-306.

JANIGA, L. (1979), *Real-time computations of two-way multihead finite automata,* in: Fundamentals of Computation Theory (FCT '79) (L. Budach ed.), Akademie Verlag, Berlin, 214-218.

KOSARAJU, R. (1979), *Real-time simulation of concatenable double-ended queues by double-ended queues,* Proceedings 11-th ACM-STOC, 346-351.

LEONG, B. & J. SEIFERAS (1977), *New real-time simulations of multihead tape units,* Proceedings 9-th ACM-STOC, 239-248.

RABIN, M.O. (1963), *Real-time computation,* Israel Journal of Mathematics 1, 203-211.

ROSENBERG, A.L. (1967), *Real-time definable languages,* J. ACM 14, 645-662.

SAVITCH, W.J. & P.M.B. VITÁNYI (1977), *Linear time simulation of multihead Turing machines with head-to-head jumps,* Lecture Notes in Computer Science (ICALP 4) 52, Springer-Verlag, Berlin, 453-464.

VITÁNYI, P.M.B. (1979), *Multihead and multitape real-time Turing machines.* Technical Report IW 111, Mathematisch Centrum, June 1979.

YAO, A. & R.RIVEST (1978), k+1 *heads are better than* k, J. ACM 25, 337-340.