The Complexity of Decision Problems for Finite-Turn Multicounter Machines

by

Eitan M. Gurari*
Department of Computer Science
SUNY at Buffalo
Amherst, NY 14226

and

Oscar H. Ibarra†
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Abstract

We exhibit a large class of machines with polynomial time decidable containment and equivalence problems. The machines in the class accept more than the regular sets. We know of no other class (different from the finite-state acceptors) for which the containment and equivalence problems have been shown polynomially decidable. We also discuss the complexity of other decision problems.

1. Introduction

It is well-known that the equivalence problem[1] for (one-way) deterministic finite-turn pushdown automata is decidable [17]. Equivalence is also decidable for deterministic one-counter machines [18]. On the other hand, for both cases, the containment problem is undecidable [16].

In this paper, we investigate the decidable properties of a large subclass of two-way multicounter machines. For positive integers m, r, k, let NCM(m,r,k) be the class of two-way nondeterministic m-counter machines M (with input endmarkers ¢ and $) [1,4,8,13] satisfying the following conditions for each input[2] ¢x$:

(1) Any computation on ¢x$ (accepting or not)[3] leads to a halting state.

(2) In any computation on ¢x$, M makes at more r turns (i.e. alternations between increasing and decreasing modes, and vice-versa) in each of the m counters.

*Work supported in part by NSF Grant MCS79-09967.

†Work supported in part by NSF Grant MCS78-01736.

[1] Let C be a class of machines. The emptiness, disjointness, containment, and equivalence problems are the problems of deciding for arbitrary machines M_1 and M_2 in C whether $T(M_1)=\emptyset$, $T(M_1)\cap T(M_2)=\emptyset$, $T(M_1)\subseteq T(M_2)$, and $T(M_1)=T(M_2)$, respectively, where T(M) denotes the set of inputs accepted by M.

[2] For convenience, we include the endmarkers as part of the input.

[3] Since M is nondeterministic, there may be more than one computation (i.e., sequence of moves).

(3) In any computation on ¢x$, no boundary (between input symbols) is crossed by the input head more than k times. (Note that the number of reversals the input head makes may be unbounded.)

The deterministic class is denoted by DCM(m,r,k). For convenience NCM(m, r,1) and DCM(m,r,1) are written NCM(m,r) and DCM(m,r), respectively. When the input is two-way unrestricted, k=∞.

The following are the main results of the paper:

(1) Let m,r, and k be fixed positive integers. Then the containment and equivalence problems for DCM(m,r,k) are decidable in polynomial time.

As far as we know, DCM(m,r,k) is the first class of one (input)-tape machines which accept more than the regular sets for which the containment and equivalence problems can be shown polynomially decidable. An example of a nonregular language accepted by a machine in DCM(1,1,5) is the set

L={¢x$ | x in $\{a,b,c,d\}^+$, the sum of lengths of all runs of c's occurring between symbols a and b (in this order) equals the number of d's}

We note that recently, a polynomial time algorithm for deciding equivalence of two-tape deterministic finite automata was shown in [5].

(2) Let m, r, and k be fixed positive integers. Then the emptiness and disjointness problems for NCM(m,r,k) are decidable in polynomial time.

(3) Let m, r, and k be fixed positive integers. Then the nonemptiness, nondisjointness, noncontainment, and inequivalence problems for DCM(m,r,k) are nondeterministic log-space complete. (See [14] for the definition of log-space complete.)

(4) The nonemptiness problem for $\bigcup_r$ DCM(2,r) is NP-hard, even for machines with unary input alphabet. The result also holds for $\bigcup_m$ DCM(m,1). (See [7] for the definition of NP-hard.)

(5) The nonemptiness problem for $\bigcup_{m,r}$ DCM(m,r) is PSPACE-hard, even for machines with unary input alphabet. (See [7] for the definition of PSPACE-hard.)

(6) The languages accepted by machines in $\bigcup_{m,r}$ NCM(m,r,∞) and $\bigcup_{m,r}$ DCM(m,r,∞) are in NSPACE(log n) and DSPACE(log n), respectively. Thus, these languages can be accepted by polynomial time-bounded deterministic Turing machines.

The decidability of the emptiness and disjointness problems (respectively, containment and equivalence problems) for two-way nondeterministic (respectively, deterministic) multicounter machines with finite-turn counters and reversal-bounded inputs have already been shown in [13]. However, no complexity analysis have been shown in [13]. The key constructions in this paper which are different from those in [13] permit us to give sharp complexity bounds. In particular, the construction in Lemma 2 uses a technique similar to the one developed in [1] and later generalized in [10]. We conclude this section with the following lemma which is easily verified (see e.g., [1]).

Lemma 1. Let M be in NCM(m,r,k) and n be the size (i.e., length of the representation) of M. We can construct, in time polynomial in n, m, and r, a machine M' in NCM $(m\lceil(r+1)/2\rceil,1,k)$ such that T(M')=T(M). (The result also holds for the deterministic class.)

2. One-way Multicounter Machines

Recall that machines in classes NCM(m,r) and DCM(m,r) are one-crossing bounded and therefore have one-way input tape. In this section, we show that for fixed m and r, DCM(m,r) has polynomial time decidable containment and equivalence problems.

We begin with the following important lemma that sharpens a similar result in [10].

Lemma 2. There is a fixed positive constant c with the following property: Let M be in NCM(m,1). Let s be the number of transition rules of M. Then $T(M)\neq\emptyset$ if and only if M accepts some input within time (i.e. number of moves) $(ms)^{cm}$.

Proof. The "if" part is obvious. To prove the "only if" part, let $M=\langle K,\Sigma,\delta,q_0,F\rangle$ be in NCM(m,1), where K, Σ, and F are finite nonempty sets of states, input alphabet, and accepting states, respectively. q_0 in K is the start state, and each transition rule in δ is a mapping from $K\times\Sigma\times\{=0,\neq 0\}^m$ to $K\times\{0,+1\}\times\{-1,0,+1\}^m$. For $(q,u,\pi_1,\ldots,\pi_m)$ in $K\times\Sigma\times\{=0,\neq 0\}^m$, if $\delta(q,u,\pi_1,\ldots,\pi_m)$ contains $(p,d,\lambda_1,\ldots,\lambda_m)$ and M in state q scanning "u" on the input tape and for each $1\le i\le m$ the value of the i'th counter is 0 if and only if π_i is =0, then M may move its input head d (=0 or +1) position to the right, change the value of the i'th counter, $1\le i\le m$, by λ_i, and enter state p. In addition, $abs(\lambda_1)+\ldots+abs(\lambda_m)\le 1$. Without loss of generality it is assumed that on entering an accepting configuration all of the counters are zero. At any given instant of the computation, each counter is in one of three modes: zero mode, increasing mode, or decreasing mode depending on whether it is zero, it is nonzero and the last change in its value was by +1, or it is nonzero and the last change in its value was by -1, respectively. A counter-mode vector is a vector representing the modes of the m counters. Now let T by any accepting computation. Clearly, T can be decomposed into at most 3m+1 subcomputations $T_1,\ldots,T_\ell$. Each T_i corresponds to a distinct counter mode vector v_i, and the sequence of moves (i.e., transition rules) used in T_i are all made with the counters having mode v_i.

Now consider any T_i. T_i is just a sequence of (not necessarily distinct) transition rules, say, $\sigma_1,\sigma_2,\ldots,\sigma_u$. For each transition rule that appears in the sequence $\sigma_1,\sigma_2,\ldots\sigma_u$ arbitrarily mark exactly one of its occurrences. Clearly, there will be at most s(= number of transition rules in M) marked positions. Suppose that between two marked transition rules there is a sequence of unmarked transition rules $\alpha_1,\ldots\alpha_k$, where $k>1$, $\alpha_1=\alpha_k$, and $\alpha_2,\ldots,\alpha_k$ are distinct. (Thus $k\le s+1$.) By deleting the sequence $\alpha_2,\ldots\alpha_k$ from T_i a new sequence of transition rules is obtained. The new sequence preserves the proper order of transition rules corresponding to counter-mode vector v_i. However, the new sequence does not correspond to a valid computation if the counters contain "improper" values. The above process of elimination of transition rules from T_i can be iterated until no further deletion can be done. Let S_i be the multiset containing exactly the sequences (of transition rules) deleted from T_i and $\hat{T}_i$ be the remainder of T_i after all the deletions have been made. Note that $\hat{T}_i$ has at most $(s+1)^2$ transition rules and contains all the marked transition rules. In general, a sequence may occur several times in S_i. Define an equivalence relation on the sequences in S_i as follows: Two sequences in S_i are equivalent if and only if they start with the same transition rules and end with the same transition rules, and they have the same net effect on the counters. All the sequences in S_i correspond to the same counter-mode vector v_i and therefore for each counter the net effects are either all nonnegative or all nonpositive. Moreover, the absolute values of these net effects are no greater than s because each sequence in S_i has length at most s. It follows that the equivalence relation induces a partition of S_i into equivalence classes whose cardinality is no greater than (number of possible distinct transition rules at the start of the sequence)* (number of possible distinct transition rules at the end of the sequence)* (1+maximum of the absolute value of the net effect of the sequences on each of the counter values)$^m=s^2(s+1)^m$.

From the partition of $S_1,\ldots,S_\ell$ an ordered set of, say, p distinct equivalence classes can be obtained. Clearly, $p\le \ell s^2(s+1)^m$. For $1\le j\le p$, choose a fixed sequence of transition rules t_j in the j'th equivalence class. Then the net change in the q'th counter, $1 \le q \le m$, due to T_i, $1 \le i \le \ell$, is

$$b_{iq} + \sum_{j=1}^{p} a_{ijq}(\hat{x}_{ij}+1)$$

where

(1) If S_i contains a t which is equivalent to t_j, then let a_{ijq} be the net change in the q'th counter of M due to t_j. If no such t appears in S_i then let $a_{ijq}=0$.

(2) b_{iq} is the net change in the q'th counter due to $\hat{T}_i$.

(3) $\hat{x}_{ij}+1$ equals the number of times sequences that are equivalent to t_j appear in S_i. (If $a_{ijq}=0$ then let the value of $\hat{x}_{ij}+1$ be equal to 1.)

Moreover, for each $1 \le q \le m$

$$\sum_{i=1}^{\ell} [b_{iq}+\sum_{j=1}^{p} a_{ijq}(\hat{x}_{ij}+1)]=0.$$

Hence, the system

$$\sum_{i=1}^{\ell} [b_{iq}+\sum_{j=1}^{p} a_{ijq}(x_{ij}+1)]=0, \qquad 1 \le q \le m$$

has a nonnegative integral solution. Then, by [11] (see also [2]), the system also has a nonnegative integral solution $(\tilde{x}_{11},\ldots,\tilde{x}_{1p},\ldots,\tilde{x}_{\ell 1},\ldots,\tilde{x}_{\ell p})$ in which each $\tilde{x}_{ij}$ is no greater than $3d\Delta^2$, where d is the number of variables in the system and Δ is the maximum of the absolute values of all subdeterminants of the augmented matrix formed by the system. Now in the above system, $d \le \ell^2 s^2(s+1)^m$, $\ell \le 3m+1$, $|a_{ijq}| \le s$, and $|b_{iq}| \le (s+1)^2$. Thus, $\tilde{x}_{ij} \le 3d\Delta^2 \le (ms)^{c_1 m}$ for some fixed positive constant c_1.

The desired shorter accepting computation $\tilde{T}$ can now be constructed as a sequence of ℓ subcomputations: $\tilde{T}_1,\ldots,\tilde{T}_\ell$. Each $\tilde{T}_i$ is constructed from $\hat{T}_i$ as follows: For each t_j, if a t equivalent to t_j appears in S_i then $\tilde{x}_{ij}+1$ copies of t_j are inserted directly to the right of the marked transition rule in $\hat{T}_i$ which is identical to the last transition rule in t_j. Clearly, $\tilde{T}$ has length at most $(ms)^{cm}$ for some fixed positive constant $c \ge c_1$.

By construction, $\tilde{T}_i$ corresponds to a proper order of transition rules for moves on counter-mode vector v_i. In addition, $\tilde{T}_i$ includes a transition rule that increases (respectively, decreases) the q'th counter, $1 \le q \le m$, if and only if T_i includes a (possible different) transition rule that increases (respectively, decreases) the q'th counter. However the net change in the q'th counter due to $\tilde{T}$ is zero. Therefore, $\tilde{T}_i$ is in zero mode, increasing mode, or decreasing mode if and only if T_i is so. Hence, $\tilde{T}$ is a valid computation. ∎

We can now use Lemma 2 to prove the following result.

Theorem 1. Let m and r be fixed positive integers. Then the emptiness problem for NCM(m,r) is decidable in polynomial time.

Proof. By Lemma 1, it is sufficient to prove the result for the class NCM(m,1). We describe a nondeterministic Turing machine (NTM) Z which when given (the representation of) M of size n accepts M if and only if $T(M) \neq \emptyset$. Moreover, Z is $O(\log n)$-tape bounded.

Given M, Z simulates the computation of M on some input string ¢x$ by guessing x symbol by symbol. By Lemma 2, $T(M) \neq 0$ if and only if M accepts some input within time $(ms)^{cm}$. It follows that in the computation on such an input, the maximum integer stored in any counter of M is no greater than $(ms)^{cm}$. Hence, Z can do the simulation on ¢x$ using at most space $\log((ms)^{cm}) = O(m \log n)$, since $n \ge m,s$. For fixed m, $O(m \log n) = O(\log n)$. Thus, Z is $O(\log n)$-tape bounded. From Z, we can construct a deterministic Turing machine Z' which accepts M if and only if $T(M) = \emptyset$, and Z' operates in p(n) time for some polynomial p [12]. ∎

Corollary 1. Let m and r be fixed positive integers. The disjointness problem for NCM(m,r) is decidable in polynomial time.

Proof. Clearly, given two machines M_1 and M_2 in NCM(m,r), we can construct (in time polynomial in the sum of the sizes of M_1 and M_2) a machine M in NCM(2m,r) such that $T(M) = T(M_1) \cap T(M_2)$. The result follows from Theorem 1. ∎

Next, we have

Theorem 2. Let m and r be fixed positive integers. Then the containment and equivalence problems for DCM(m,r) are decidable in polynomial time.

Proof. Clearly, we only need consider the containment problem. Given two machines M_1 and M_2 in DCM(m,r), we can easily construct (in polynomial time) a machine M in DCM(2m,r) which accepts an input string ¢x$ if and only if ¢x$ is in $T(M_1) - T(M_2)$. Then $T(M) = \emptyset$ if and only if $T(M_1) \subseteq T(M_2)$. Hence, from Theorem 1, containment is decidable in polynomial time. ∎

Remark. The equivalence problem for NCM(1,1) is undecidable. In fact, the problem of determining if a machine M in NCM(1,1) accepts the set ¢$\Sigma^*$$ (Σ is the input alphabet of M) is undecidable [1,9].

It follows from Theorem 1 and Corollary 1 that the emptiness and disjointness problems for $\bigcup_{m,r} NCM(m,r)$ are decidable. Also, from Theorem 2, the containment and equivalence problems for $\bigcup_{m,r} DCM(m,r)$ are decidable. Although the problems are decidable, the next two results show that polynomial time algorithms are unlikely.

<u>Theorem 3</u>. The nonemptiness problem for the class $\bigcup_{r} DCM(2,r)$ is NP-hard, even for machines with <u>unary</u> input alphabet. The result also holds for the class $\bigcup_{m} DCM(m,1)$.

<u>Proof</u>. The construction is similar to one given in [6]. ∎

The nonemptiness problem for one-way deterministic two-counter machines over a unary input alphabet (whose counters are <u>not</u> finite-turn) is undecidable [15]. On the other hand, for one-way nondeterministic one-counter machines, emptiness is decidable in polynomial time. In fact, the emptiness problem for one-way nondeterministic pushdown automata is decidable in polynomial time [12].

Our next result concerns the class $\bigcup_{m,r} DCM(m,r)$.

<u>Theorem 4</u>. The nonemptiness problem for $\bigcup_{m,r} DCM(m,r)$ is PSPACE-hard, even for machines with <u>unary</u> input alphabet.

<u>Proof</u>. Let Z be a deterministic linear bounded automaton with state set Q and tape alphabet Γ ($\not{c}$ and \$ are not in Γ). Assume that $Q \cap (\Gamma \cup \{\not{c}, \$\}) = \emptyset$. Let $s=|Q|$, $t=|\Gamma|+2$ and $k=s+t$. Also assume that Z halts on <u>all</u> inputs. For any input $\not{c}a_1,\ldots,a_{n-2}\$$ to Z, we can construct in polynomial time (in n) a machine M in $DCM(m,r)$ which accepts some input string if and only if Z accepts $\not{c}a_1,\ldots,a_{n-2}\$$. Moreover, m and r can be chosen to be $\lceil \log_2 k \rceil (n+1)$ and $s \cdot n \cdot t^{n-2}$, respectively. The construction of M is straightforward. M encodes each tape symbol of $Q \cup (\Gamma \cup \{\not{c}, \$\})$ as a binary string of length $\lceil \log_2 k \rceil$. Thus, the initial instantaneous description (ID) $q_0\not{c}a_1,\ldots,a_{n-2}\$$ of Z can be encoded as a binary string of length $\lceil \log_2 k \rceil (n+1)$. This binary string can be represented in M's counters. (M has exactly $\lceil \log_2 k \rceil (n+1)$ counters.) The computation of Z can then be simulated by M using its counters to construct successive ID's of Z. Since Z is deterministic and always halts, the number of ID's in the sequence is at most $s \cdot n \cdot t^{n-2}$. (The end markers are not changed.) Hence, each counter of M makes at most $s \cdot n \cdot t^{n-2}$ turns. It is easy to verify that M's size is at most p(n) for some polynomial p. ∎

3. Two-Way Multicounter Machines

The results of Section 2 can be generalized to hold for the classes NCM(m,r,k) and DCM(m,r,k). The key result is the following theorem.

Theorem 5. Let M be in NCM(m,l,k). We can effectively construct a machine M' in NCM(2m,1,1) (i.e. M' is one-way) such that T(M')=T(M). If M has size n, then M' has size $n' \leq cn^k$ for some fixed positive constant c independent of M. Moreover, M' can be constructed in time polynomial in n'.

Proof. Every accepting computation of M can be described by a time-input graph which shows the sequence of transition rules used during the computation (see Figure 1(a)). A node is at coordinate (ζ,μ) in the graph if and only if it corresponds to the transition rule associated with the ζ'th move in the computation and just before this move the input head of M was at the μ'th symbol of the input string.

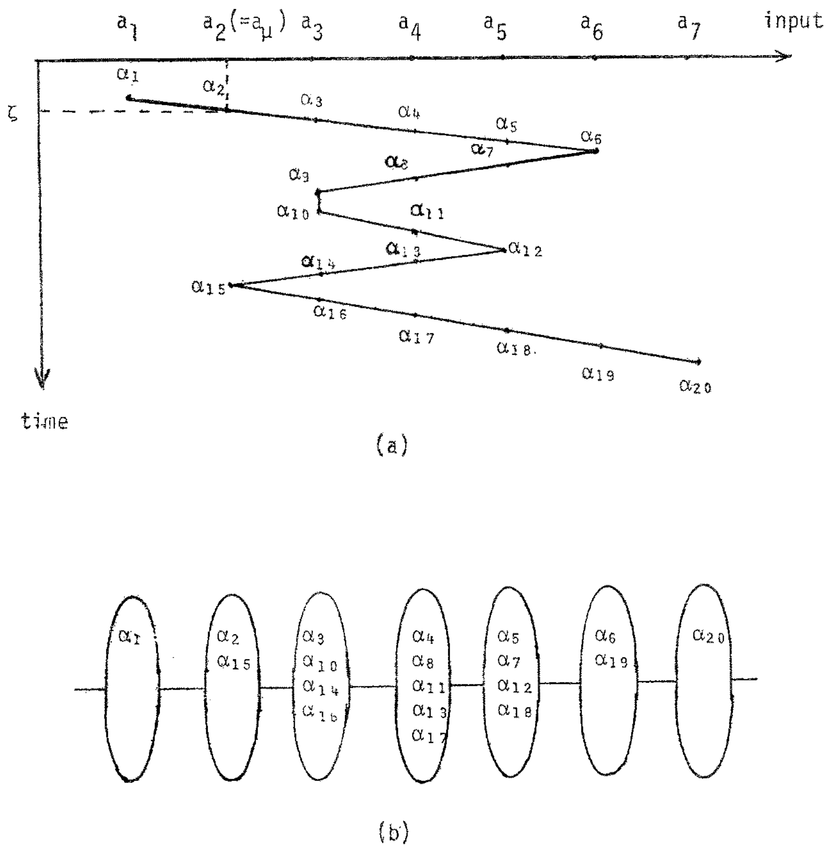
Now, consider any accepting computation of M. Then a linear tree, say, T, which describes the computation, can also be constructed (see Figure 1 (b)). Each node in T corresponds to an ordered set of at most k transition rules. The i'th node in T is associated with the i'th symbol of the input string, say, a_i, where the ordered set of transition rules are exactly those used to move the input head from a_i (in the given order).

Thus in simulating an accepting computation of M the device M' need only nondeterministically determine a sequence of ordered sets of transition rules, where the sequence (of these sets) corresponds to the linear tree which describes the desired computation. (Note that the number of distinct nodes in T does not exceed the value s^k, where s is the number of transition rules of M.) Corresponding to each counter, say, C of M the counter machine M' uses two counters, say $\tilde{C}$ and $\hat{C}$. $\tilde{C}$ is used to record the increases in C while $\hat{C}$ is used to record the decreases in C. Thus once M' completes the simulation of an accepting computation of M both $\tilde{C}$ and $\hat{C}$ must contain the same value. ∎

Theorem 5 does not hold for DCM(m,l,k). For example, the language L={¢x$|x in $\{a,b,c,d\}^+$, the sum of the lengths of all runs of c's occurring between symbols a and b (in this order) equals the number of d's} is in DCM(1,1,5). However, L cannot be accepted by a deterministic multicounter machine with finite-turn counters, even if the input head can make a fixed number of reversals.

From Lemma 1, Theorems 1 and 5, and the construction in Corollary 1, we have

Theorem 6. Let m, r, and k be fixed positive integers. Then the emptiness and disjointness problems for NCM(m,r,k) are decidable in polynomial time.



<u>Figure 1.</u> A description of an accepting computation of a nondeterministic two-way multicounter machine by (a) a graph; and (b) a linear tree.

Theorem 6 and the construction in Theorem 2 yield

<u>Theorem 7.</u> Let m, r, and k be fixed positive integers. Then the containment and equivalence problems for DCM(m,r,k) are decidable in polynomial time.

<u>Remark.</u> It follows from Theorem 6 that the emptiness and disjointness problems for $\bigcup_{m,r,k}$ NCM(m,r,k) are decidable. Also, from Theorem 7, the containment and equivalence problems for $\bigcup_{m,r,k}$ DCM(m,r,k) are decidable. When the input is two-way unrestricted (i.e. $k=\infty$) or the input is one-way but the counters are unrestricted (i.e. $k=1$ and $r=\infty$), the problems are undecidable. In fact, we can show the following using the proof

techniques in [13] and [15], respectively:

(1) For some fixed r, the emptiness problem for DCM(2,r,∞) is undecidable.

(2) The emptiness problem for DCM(2,∞,1) is undecidable.

Theorem 8. Let m, r, and k be fixed positive integers. Then the nonemptiness, nondisjointness, noncontainment, and inequivalence problems for DCM(m,r,k) are nondeterministic log-space complete.

Proof. From Lemmas 1 and 2 and Theorem 5 and constructions similar to those in Theorem 1, Corollary 1, and Theorem 2, the problems are solvable in nondeterministic log-space. The theorem now follows because the nonemptiness problem for (one-way) deterministic finite automata is nondeterministic log-space complete [14]. ∎

Finally, using Lemma 2 we have

Theorem 9. The languages accepted by machines in $\bigcup_{m,r}$ NCM(m,r,∞) and $\bigcup_{m,r}$ DCM(m,r,∞) are in NSPACE(log n) and DSPACE(log n), respectively.

Thus, these languages can be accepted by polynomial time-bounded deterministic Turing machines.

Remark. Theorem 9 has also been shown in [3] using a different technique.

References

1. Baker, B. and Book, R., Reversal-bounded multipushdown machines, J. Comptr. and Syst. Sci. 8 (1974), 315-332.
2. Borosh, I. and Treybig, L., Bounds on positive integral solutions of linear Diophantine equations, Proc. Amer. Math. Soc. 55 (1976), 299-304.
3. Chan, T., Ph.D. Thesis (Cornell University), 1980.
4. Fischer, P., Meyer, A., and Rosenberg, A., Counter machines and counter languages, Math. Syst. Theory 2 (1968), 265-283.
5. Friedman, E. and Greibach, S., A polynomial time algorithm for deciding the equivalence problem for 2-tape deterministic finite state acceptors, to appear.
6. Galil, Z., Hierarchies of complete problems, Acta Informatica 6 (1976), 77-88.
7. Garey, M. and Johnson, D., "Computers and Intractability: A Guide to the Theory of NP-Completeness", H. Freeman, San Francisco, 1978.
8. Greibach, S., Remarks on the complexity of nondeterministic counter languages, Theoretical Comptr. Sci. 1 (1976), 269-288.
9. Greibach, S., An infinite hierarchy of context-free languages, J. ACM 16 (1969), 91-106.
10. Gurari, E., Transducers with decidable equivalence problem, TR-CS-79-4, University of Wisconsin-Milwaukee, (1979).
11. Gurari, E. and Ibarra, O., An NP-Complete number-theoretic problem, J. ACM 26 (1979), 567-581.
12. Hopcroft, J. and Ullman, J., "Introduction to Automata Theory, Languages, and Computation", Addison-Wesley, Reading, MA, 1979.
13. Ibarra, O., Reversal-bounded multicounter machines and their decision problems, J. ACM 25 (1978), 116-133.
14. Jones, N., Space-bounded reducibility among combinatorial problems, J. Comptr. and Syst. Sci. 11 (1975), 68-85.
15. Minsky, M., Recursive unsolvability of Post's problem of Tag and other topics in the theory of Turing machines, Annals of Math 74 (1961), 437-455.
16. Valiant, L., Decision procedures for families of deterministic pushdown automata, Theory of Computation-Report No. 1, University of

Warwick, Coventry, Great Britain, 1973.

17. Valiant, L., The equivalence problem for deterministic finite-turn pushdown automata, Inform. and Contr. 25 (1974), 123-133.

18. Valiant, L. and Patterson, M., Deterministic one-counter automata, J. Comptr. and Syst. Sci. 10 (1975), 340-350.