ON THE TIME AND TAPE COMPLEXITY OF LANGUAGES, I

H. B. Hunt III

TR 73 - 156

January, 1973

Department of Computer Science
Cornell University
Ithaca, New York 14850

ON THE TIME AND TAPE COMPLEXITY OF LANGUAGES, I

H. B. Hunt III†
Cornell University
Ithaca, New York

Abstract:

We investigate the relationship between the classes of languages accepted by deterministic and nondeterministic polynomial time bounded Turing machines and the relationship between the classes of languages accepted by deterministic polynomial time bounded and by nondeterministic polynomial tape bounded Turing machines. In both cases we study generators of the nondeterministic class that generate it by operations that the deterministic class is closed under.

†This research was supported by a National Science Foundation Fellowship in Computer Science.

Section 1: Introduction

Before describing our results we need several definitions and facts.

Definition 1.0: For all positive integers k, dtape(n^k) is the class of all languages (over some countably infinite alphabet $\bar{\Sigma}$) recognized by some deterministic n^k - tape bounded Turing machine.

Definition 1.1: PTIME is the class of all languages (over some countably infinite alphabet $\bar{\Sigma}$) recognized by some deterministic polynomial time bounded Turing machine.

Definition 1.2: NPTIME is the class of all languages (over some countably infinite alphabet $\bar{\Sigma}$) recognized by some non-deterministic polynomial time bounded Turing machine.

Definition 1.3: PTAPE is the class of all languages (over some countably infinite alphabet $\bar{\Sigma}$) recognized by some deterministic or nondeterministic polynomial tape bounded Turing machine.

In [15] Savitch showed that every $L(n)$-tape bounded non-deterministic Turing machine can be simulated by an $(L(n))^2$-tape bounded deterministic Turing machine, provided $L(n) \geq \log_2(n)$. In particular, this implies that the class of languages accepted by deterministic and nondeterministic polynomial tape bounded Turing machines are the same.

We sketch the results that appear in this paper. In Section 2 we investigate the relationship between PTIME and NPTIME. We find several necessary and sufficient conditions for PTIME to equal NPTIME. In particular, we find proper subproblems of Karp's 3 p-hard problems that are p-complete (See either Section 2 or Karp [10].) We also present an ϵ-free homomorphism h_0 and a language $L_0 \in$ PTIME such that $h_0(L_0) \in$ PTIME implies PTIME = NPTIME.

In Section 3 we extend Greibach's two undecidability theorems (See Section 3 or Greibach [6]). We give several examples where our theorems apply and those of Greibach do not. We also show that our extensions lead naturally to several theorems about the minimal deterministic time complexity of many properties of the regular sets.

In Section 4 we study the classes PTAPE, PTIME and NPTIME. We extend the Simulation Lemma in Meyer [12]; and we are able to show that each member of a broad class of problems about the regular expressions is at least as hard to decide as it is to recognize <u>any</u> context-sensitive language. We prove a similar result for a class of problems about the regular expressions with squaring of Meyer and Stockmeyer [13]. We also extend a recently announced result of L. Stockmeyer to show that several problems about the extended regular expressions are not elementary recursive. Section 5 is a brief conclusion.

Section 2: The Classes PTIME and NPTIME

The main results of this section are several necessary and sufficient conditions for PTIME to equal NPTIME. In particular, we show that the following are equivalent:

(1) PTIME = NPTIME;

(2) all linear time nondeterministic 2-tape Tm languages are elements of PTIME;

(3) all $n \log n$ time nondeterministic single-tape Tm languages are elements of PTIME;

(4) the set of pairs of equivalent star-free regular expressions over $\{0,1\} \in$ PTIME, i.e., $\{(\alpha,\beta) \mid \alpha$ and β are regular expressions over $\{0,1\}$ with no occurrence of * and $L(\alpha) = L(\beta)\} \in$ PTIME;

and (5) the set of pairs of equivalent nondeterministic finite automata whose accepted languages are finite is an element of PTIME, i.e., $\{(\alpha,\beta) \mid \alpha$ and β are n.f.s.a., $L(\alpha) = L(\beta)$, and $L(\alpha)$ and $L(\beta)$ are finite$\} \in$ PTIME.

We also show that PTIME = NPTIME if and only if PTIME is closed under ϵ-free homomorphism. In fact, we present an ϵ-free homomorphism h_0 and a language $L_0 \in$ PTIME such that PTIME = NPTIME if and only if $h_0(L_0) \in$ PTIME.

Two different types of Turing machines are studied in this section. The first is the single-tape Turing machine with a two-way infinite tape. The second is the multitape Turing machine. We assume the reader is familiar with the single-tape device; we will informally describe the multitape device.

Following Hopcroft and Ullman [8] pp. 94 and 95, a multitape Tm consists of a finite control with k tape heads and k tapes. Each tape is 2-way infinite. Initially, the input appears on the first tape and the other tapes are blank. Let "-" mean a blank tape. We denote such a tape configuration by $(w,-,-,\ldots,-)$. Let M be in its start state with its first tape-head left - justified on w. Then $w \in T^*$ is an element of L(M), if M when applied to the tape configu ation $(w,-,-,\ldots-)$ eventually halts in an accepting state.

To prove these results we need two technical results:

Lemma 2.1: $(\forall k \geq 1)$ $(\exists$ a 2-tape Tm $M_k)(\forall w \in \Sigma^+)$

$[L(M_k) = \{x \mid x = w \text{¢} 1^{|w|^{2^k}} \$ \text{ with ¢, } \$ \notin \Sigma\}$.

Moreover, M_k operates within linear time.].

Lemma 2.2: $(\forall k \geq 1)$ $(\exists$ a single-tape Tm $M_k)(\forall w \in \Sigma^+)$

$[L(M_k) = \{x \mid x = w \text{¢} 1^{|w|^{2^k}} \$ \text{ with ¢, } \$ \notin \Sigma\}$.

Moreover, M_k operates within time $O(|x| \log |x|)$.][1]

---

[1] In Lemma 2.1 and 2.2 we allow arbitrarily large tape alphabets. If we restrict ourselves to the tape alphabet Σ {0,1,b̸}, then we could let w be an arbitrary string in $\{0,1\}^+$ and let the ¢ and \$ signs be replaced by b̸. Otherwise, we could encode w, ¢, and \$ as strings in $\{0,1\}^+$.

The proofs of Lemmas 2.1 and 2.2 are standard and the experienced reader can proceed directly to Theorem 2.3.

Proof of 2.1: The lemma states that for any $k \geq 1$, there is a 2-tape linear time Tm M_k whose accepted language

$$L(M_k) = \{x \mid x = w ¢1^{|w|^{2^k}} \$ \text{ with } ¢, \$, \notin T\}.$$

Only the details will be sketched. The techniques used are illustrated in more detail in Davis [5] or Hopcroft and Ullman [8].

The proof is by induction on k. Let k = 1. The first tape of M_1 contains the input string x. The second tape has 2 tracks; a counter of length $|w|$ is kept on each track. The $|w|^2$ 1's are divided into $|w|$ sets; each set contains $|w|$ 1's. The bottom counter records which set the machine is checking-off on the first-tape. The top counter records which 1 in a given set the machine is checking-off on the first-tape. The time used is $O(|x|)$.

Assume that the lemma holds for k = i - 1. The proof for k = i is now similar to that for k = 1 with $|w|$ replaced by $|w|^{2^{i-1}}$.

Proof of 2.2: The method of proof follows that of Lemma 2.1 with the following modifications:

1) the counters are on the same tape as the input,
2) they are kept in base 2 and hence have maximum size $O(\lceil \log |x| \rceil)$, and
3) they are moved one cell to the right each time a new symbol is checked. Again the time required

is $O(\ell og|x|)$. Hence, the total time required is $O(|x|\ \ell og\ |x|)$.

Finally, the following three definitions are needed in the proof of the equivalence of (1), (4), and (5).

1) A Boolean form f is in D_3 if f is the conjuction of clauses $c_1, c_2, \ldots, c_p$, where each clause is the disjunction of at most three literals.

2) The D_3 Tautology problem is defined to be
[D_3 TAUTOLOGY
INPUT: Clauses $c_1, c_2, \ldots, c_p$ such that each c_i is the disjunction of at most three literals
PROPERTY: The conjunction of the given clauses is a tautology.]

3) Given a regular expression β (or a nondeterministic finite state automaton β), $L(\beta)$ is the language defined (or recognized) by β.

<u>Theorem 2.3</u>: The following are equivalent:

1) PTIME = NPTIME;
2) all linear time nondeterministic 2-tape Tm languages are elements of PTIME;
3) all $n\ \ell og\ n$ time nondeterministic single tape Tm languages are elements of PTIME;
4) the set $\{(\alpha,\beta)\ |\ \alpha$ and β are regular expressions over $\{0,1\}$ with no occurrence of $*$ and $L(\alpha) = L(\beta)\} \in$ PTIME;

and 5) the set $\{(\alpha,\beta) \mid \alpha$ and β are nondeterministic finite state automata, $L(\alpha) = L(\beta)$, and $L(\alpha) = L(\beta)$ are finite$\} \in$ PTIME;

Proof of 2.3: The proof of the equivalence of (1), (2), and (3) is based upon a padding technique which we will use several times in this paper. We illustrate two examples of this technique:

1) Given M an n^k time bound nondeterministic single tape Tm with tape alphabet T, the padding technique produces a nondeterministic 2-tape linear time Tm M_* such that $(\forall w \in T^+)[w \in L(M)$ if and only if $y \in L(M_*)$, where $y = w ¢ 1^{|w|^{2^i}}\$$ with ¢, \$, $1 \notin T]$.

2) Given M as above, the padding technique can also produce a nondeterministic single tape $n \log n$ time bound Tm M_* such that $(\forall w \in T^+)[w \in L(M)$ if and only if $y \in L(M_*)$, where $y = w ¢ 1^{|w|^{2^i}}\$$ with ¢, \$, $1 \notin T]$.

(1) => (2): It is well-known that every language L, recognizable by a nondeterministic 2-tape Tm in polynomial time, is also recognizable by a nondeterministic single tape Tm in polynomial time. (See Hopcroft and Ullman [8] pp. 139 and 140.) Hence by (1) $L \in$ PTIME.

(2) => (1): Let M be a nondeterministic polynomail time single-tape Tm. Given (2) we show how to construct a deterministic single tape polynomial time Tm M_* such that $L(M_*) = L(M)$. First, we construct a nondeterministic 2-tape linear time Tm M_1 such that

$$L(M_1) = \{x \mid x = w ¢ 1^{|w|^{2^i}} \$ \text{ where } w \text{ is in } L(M), 2^i \geq k, \text{ and } ¢, \$, \notin T\},$$

that is M_1 accepts a padded version of $L(M)$. Then (2) implies that there is a deterministic single tape polynomial time Tm M_2 such that $L(M_2) = L(M_1)$. M_* behaves as follows:

a) Given input w, M_* pads it, that is M_* extends w to $w ¢ 1^{|w|^{2^i}} \$$,

b) M_* applies M_2 (as a subroutine) on the string $w ¢ 1^{|w|^{2^i}} \$$, and

c) M_* accepts w if and only if M_2 accepts $w ¢ 1^{|w|^{2^i}} \$$. $L(M_*) = L(M)$ and M_* operates within polynomial time.

It remains to construct M_1 given M. From Lemma 2.1 there is a nondeterministic 2-tape linear time Tm M_1, which accepts a string x if and only if $x = w ¢ 1^{|w|^{2^i}} \$$ with $2^i \geq k$. We modify M_1 to do the following:

<u>STEP1</u>: M_1 verifies its input x is of the form $x = w ¢ 1^{|w|^{2^i}} \$$. If not, M_1 rejects x.

STEP2: Otherwise, M_1 erases the counters on its second tape and copies w there.

STEP3: M_1 simulates M on its second tape. M_1 is linear time bound since STEP1 is $O(|x|)$ from Lemma 1.1, STEP2 is $O(|x|)$ since the length of the counters is less than $|x|$, and STEP3 is $O(|x|)$ since $|x| = |w|^{2^i} + |w| + 2 \geq |w|^{n^k}$.

Clearly (1) => (3) since every nlogn time nondeterministic single-tape Tm language is an element of NPTIME. Modifying the proof of (2) => (1) above, by using Lemma 2.2 instead of Lemma 2.1, we have (3) => (1).

(1) <= (4): Let $C_1, C_2, \ldots, C_p$ be a set of clauses and let $f = \bigvee_{i=1}^{p} c_i$. Cook [4] has shown that PTIME = NPTIME if and only if D_3 TAUTOLOGY is in PTIME. Thus we need only show how to construct a regular expression β using only $\cup$ and $\cdot$ such that $L(\beta) = \{0,1\}^n$ if and only if f is a tautology, by a construction requiring time bounded by a polynomial in $|C_1| + \ldots + |C_p|$.

Let the set of variables appearing in f be $\{x_1, x_2, \ldots, x_n\}$. Then the set of literals appearing in f is contained within the set $\{x_1, x_2, \ldots, x_n, \bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$. Without loss of generality, we assume that each clause C_i does not contain a complementary pair of literals.

That is, for $1 \leq i \leq p$ $c_i = y_j\, y_k\, y_\ell$ where $1 \leq j,h, \ell \leq n$ and $y_j \in \{x_j, \bar{x}_j\}$, $y_k \in \{x_k, \bar{x}_k\}$, and $y_\ell \in \{x_\ell, \bar{x}_\ell\}$. (The construction of β is unaffected if β contains only one or two literals.)

The regular expression will be the union of β_i's where each $L(\beta_i)$ is a subset of $\{0,1\}^n$.

For $1 \leq i \leq p$, $\beta_i = \beta_{i1} \cdot \beta_{i2} \cdot \ldots \cdot \beta_{in}$.

For $1 \leq j \leq n$, $\beta_{ij} = \begin{cases} (0 \cup 1) & \text{if } x_j \text{ and } \bar{x}_j \text{ are not literals in } C_i, \\ (0) & \text{if } \bar{x}_j \text{ is a literal in } C_i, \text{ or} \\ (1) & \text{if } x_j \text{ is a literal in } C_i \end{cases}$

Clearly, $L(\beta) \subset \{0,1\}^n$. Let $y = y_1 y_2 \ldots y_n \in \{0,1\}^n$. Then $y \in L(\beta)$ iff $y \in L(\beta_i)$ for some i. But $y \in L(\beta_i)$ iff $(y_1, y_2, \ldots, y_n)$ satisfies c_i , and therefore satisfies f. Therefore, $L(\beta) = \{0,1\}^n$ iff f is a tautology.

To prove (1) => (4), it suffices to show that $\{(\alpha,\beta)$ | α and β are regular expressions over $\{0,1\}$ with no occurrence of * and $L(\alpha) \neq L(\beta)\}$ is in NPTIME. Intuitively, this follows since $y \in L(\beta)$, for β a regular expression not containing *, implies that $|y|$ is less than or equal to the number of 0's and 1's appearing in β. Hence, $|y| \leq |\beta|$.

Let α and β be two regular expressions over $\{0,1\}$ not using *. Then α is not equivalent to β iff there exists an x in $\{0,1\}^*$ such that $x \in [L(\alpha) \cap \overline{L(\beta)}] \cup [\overline{L(\alpha)} \cap L(\beta)]$. However, $x \in L(\alpha)$ implies $|x|$ is less than or equal to $|\alpha|$. Therefore, to verify α is not equivalent to β we need only consider those x in $\{0,1\}^*$ of length less than or equal to $\max(|\alpha|,|\beta|)$.

We convert α and β to nondeterministic finite automata M and M' such that $L(M) = L(\alpha)$ and $L(M') = L(\beta)$. There are standard polynomial time algorithms to do this. (See Salomaa [14]). <u>Note</u>: These automata have a polynomially bounded number of states. We then test the equivalence of M and M'.

Our Tm operates as follows:

<u>STEP1</u>: It constructs M and M'.

<u>STEP2</u>: It guesses a string x of length less than or equal to $\max(|\alpha|,|\beta|)$ that differentiates between M and M', i.e., it guesses that x is in $[L(M) \cap \overline{L(M')}] \cup [\overline{L(M)} \cap L(M')]$.

<u>STEP3</u>: It verifies that x does in fact differentiate between M and M'. It accepts if this is the case.

Clearly α and β are accepted iff they are not equivalent. We show that the algorithm requires only polynomial time. Our Tm can execute STEP3 by keeping track of all states that M and M' can be in, when applied to string x. The number of such states is less than or equal to $|M'| + |M|$. (Note: finite automata are described as strings and hence have length. We require $|M|$ to be greater than or equal to the number of states in M.) For each state in M (each state in M') the TM need only check at most $|M|$ ($|M'|$) move-rules. Finally, $|x|$ is less than or equal to $\max(|\alpha|,|\beta|)$. Hence, the time required is $O(\max(|\alpha|,|\beta|) \cdot [|M|^2 + |M'|^2] \cdot$ [time required to move back and forth along the tape] + [time needed to construct M and M' given α and β]).

(1) => (5): If M is a nondeterministic finite automaton and if L(M) if finite, then x is in L(M) only if $|x|$ is less than the number of states of M. Thus, given two star-free regular expressions α and β over $\{0,1\}$, to show $L(\alpha) \neq L(\beta)$ we need only guess some x with $|x| \leq$ max(number of states of α, number of states of β) that differentiates between α and β, and verify in polynomial time that x does in fact differentiate between them. But, we showed how to do this in the proof that (1) => (4).

(5) => (1): The proof is very similar to that of (4) => (1). Let f be a Boolean form in D_3 with clauses $C_1, C_2, \ldots, C_p$ and variable set $\{x_1,x_2,\ldots,x_n\}$. We show how to construct a nondeterministic finite automaton M whose accepted language L(M) is finite. Furthermore, $L(M) = \{0,1\}^n$ iff f is a tautology. Finally, the time necessary to con-

struct M is bounded by a polynomial in $|c_1| + |c_2| + \ldots + |c_p|$.

The set of literals appearing in f is contained within the set $\{x_1, x_2, \ldots, x_n, \bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$. Without loss of generality, we assume each clause c_i does not contain a pair of complementary literals. Therefore, for $1 \leq i \leq p$ $c_i = y_j\, y_k\, y_\ell$ where $1 \leq j, k, \ell \leq n$ and $y_j \in \{x_j, \bar{x}_j\}$, $y_k \in \{x_k, \bar{x}_k\}$, and $y_\ell \in \{x_\ell, x_\ell\}$. $M = (\{q_0\} \cup \{q_{ij} \mid 1 \leq i \leq p \text{ and } 1 \leq j \leq n+1\} \cup \{\text{reject}\}, \{0,1\}, \delta, \{q_{i\ n+1} \mid 1 \leq i \leq p\})$[1].

Here, $\delta(q_0, \epsilon) = \{q_{i,1} \mid 1 \leq i \leq p\}$. $\delta(q_0, 1) = \delta(q_0, 0) = \text{reject}$.

For $1 \leq i \leq p$ and

$$1 \leq j \leq n,\ \delta(q_{ij}, 0) = \begin{cases} q_{i\ j+1} \text{ if } x_j \text{ or } \bar{x}_j \text{ is not a literal in the } \underline{ith} \text{ clause of } f, \\ q_{i\ j+1} \text{ if } \bar{x}_j \text{ is a literal of the } \underline{ith} \text{ clause of } f, \\ \text{reject if } x_j \text{ is a literal of the } \underline{ith} \text{ clause of } f. \end{cases}$$

$$\text{Similarly, } \delta(q_{ij}, 1) = \begin{cases} q_{i\ j+1} \text{ if } x_j \text{ or } \bar{x}_j \text{ is not a literal in the } \underline{ith} \text{ clause of } f, \\ q_{i\ j+1} \text{ if } x_j \text{ is a literal of the } \underline{ith} \text{ clause of } f, \\ \text{reject if } \bar{x}_j \text{ is a literal of the } \underline{ith} \text{ clause of } f. \end{cases}$$

---

[1] Our definition of nondeterministic finite automaton is similar to that of Hopcroft and Ullman [8], but it allows ϵ-moves. It is clear that these could be removed.

$\delta(\text{reject},0) = \delta(\text{reject},1) = \text{reject}$, and for $1 \leq i \leq p$
$\delta(q_{i,n+1},0) = \delta(q_{i,n+1},1) = \text{reject}$.

Then $L(M) \subseteq \{0,1\}^n$. Moreover, $L(M) = \{0,1\}^n$ iff f is a tautology. Let $y \in L\ \{0,1\}^n$. Then $y \in L(M)$ iff $\delta(q_0,y) = q_{i,n+1}$ for some i. This implies y satisfies the ith clause of f. Hence, $f(y) = 1$.

Note: The construction of M in the proof that (5) => (1) is due to J. Simon.

We relate the equivalence of (1), (3), (4) and (5) to the work of Karp. To do this we introduce several definitions from Karp [10]. Let $\Sigma = \{0,1\}$. Let Π be the class of functions from Σ^* into Σ^* computable in polynomial time by one-tape deterministic Tm's. Let L and M be languages $L \propto M$ (L is reducible to M) if there is a function $f \in \Pi$ such that $f(x) \in M$ if and only if $x \in L$. Call L (polynomial) complete if L (or $\bar{L}$) is in NPTIME and every language in NPTIME is reducible to L. Karp mentions the following three problems to which every complete problem is reducible, but which are not known to be elements of NPTIME:

1) Equivalence of Regular Expressions

Input: A pair of regular expressions over the alphabet {0,1}

Property: The two expressions define the same language

2) Equivalence of Nondeterministic Finite Automata

Input: A pair of nondeterministic finite automata with input alphabet $\{0,1\}$

Property: The two automata define the same language,

and 3) Context - Sensitive Recognition

Input: A context-sensitive grammar Γ and a string x

Property: x is in the language generated by Γ.

We will give a strong reason for Karp's inability to show that (1), (2), and (3) are in NPTIME in section 4. The reader should note that in Karp's sense the sets $\{(\alpha,\beta) \mid \alpha$ and β are star-free regular expressions over $\{0,1\}$ and $L(\alpha) = L(\beta)\}$ and $\{(\alpha,\beta) \mid \alpha$ and β are nondeterministic finite state automata over $\{0,1\}$, $L(\alpha) = L(\beta)$, and $L(\alpha)$ and $L(\beta)$ are finite$\}$ are polynomial complete. The reader should also note that all languages recognized by $n\ell ogn$ time - bounded nondeterministic Tm's are Context-Sensitive. (See Hopcroft and Ullman [9].) In fact, the nondeterministic $n\ell ogn$ time bound single tape Tm's used in the proof that (3) => (1) were actually linear bounded automata. Thus, we have presented proper subproblems of Karp's three open problems that are polynomial complete. The reader should also note that any language recognizable by a single-tape nondeterministic Tm in time less than $O(n \ell og\ n)$ is regular and in PTIME. Thus, PTIME = NPTIME if the simplest (in terms of time) nonregular sets in

NPTIME are also elements of PTIME. (See Hennie [7]).

Lemma 2.1 and 2.2 and Theorem 2.3 suggest that there are polynomial complete languages of very low nondeterministic time complexity on either single or 2-tape Tms. In fact, this is so.

Corollary 2.4:

1) There are polynomial complete problems of nondeterministic time complexity $O(n \log n)$ on single tape Tms.

2) There are polynomial complete problems of linear nondeterministic time complexity on 2-tape Tms.

Proof of 2.4: We only sketch the proof of 1. Both 1 and 2 follow from the fact that we are able to deterministically pad inputs within the allowed time bounds.

Using well-known results concerning encodings, we can encode arbitrary single-tape Tms as strings over a fixed finite alphabet. We can do this in such a way that the following hold:

A. The encoding process is in PTIME;

B. there is a single-tape nondeterministic Tm M, which given the code for the ith Tm M_i together with the code of an input x for M_i, simulates the action of M_i on x; and

C. the time required for M to simulate n moves of M_i on x is bounded by a polynomial in the sum of n and the lengths of the encodings of M_i and of x.

We modify M by adding a clock to shut M off if M_i computes for too long ($> |x| \ell og\ |x|$) and by padding its inputs to cover the costs of C and the bookkeeping required for running the clock. The total time required is $O(n\ \ell og\ n)$.

Next, we investigate the relationship between the closure of PTIME under ϵ-free homomorphism and the equality of PTIME and NPTIME.

Theorem 2.5 (Book and Greibach [2]): L_0 is recognized by a linear time nondeterministic multitape Tm if and only if L_0 is the ϵ-free homomorphic image of the intersection of 3 context-free languages.

Since all context-free languages are elements of PTIME (See Hopcroft and Ullman [8] pp. 156-160.) and PTIME is closed under intersection, Corollary 2.4 and Theorem 2.5 imply

Corollary 2.6: There is an ϵ-free homomorphism h_0 and 3 context-free languages L_0, L_1, L_2 such that $h_0(L_0 \cap L_1 \cap L_2) \in$ PTIME if and only if PTIME = NPTIME.

We next present an ϵ-free homomorphism h_0 and a language $L_0 \in$ PTIME such that $h_0(L_0) \in$ PTIME if and only if PTIME = NPTIME.

<u>Theorem 2.7</u>: Let $L = \{f$ ç $x \mid f$ is a D_3 Boolean form expressed as a string in some finite alphabet Σ.

$0,1 \not\in \Sigma$. $x = x_1 \cdot x_2 \cdot \ldots \cdot x_n \in \{0,1\}^n$ for some $n > 0$.

f is a function of n variables $t_1, \ldots, t_n$ and $f(x_1, \ldots, x_n)$ is false$\}$. Let h be defined as follows:

1) h is an ϵ homomorphism from $[\Sigma \cup \{¢,0,1\}]^*$ into $[\Sigma \cup \{¢,0,1\}]^*$,
2) for all $\sigma \in \Sigma \cup \{c\}$, $h(\sigma) = \sigma$,
3) $h(0) = 0$, and
4) $h(1) = 0$.

Then $L \in$ PTIME. $h(L) \in$ PTIME if and only if PTIME = NPTIME.

<u>Proof of 2.7</u>: Let f be a D_3-Boolean function of n variables. Then $f.c\ 0^n \in h(L)$ if and only if f is not a tautology.

1) $f\ c\ 0^n \in h(L)$ implies that $\exists\ x = x_1 \cdot \ldots \cdot x_n \in \{0,1\}^n$ such that $f\ c\ x \in L$. But $f\ c\ x \in L$ implies $f(x_1, \ldots, x_n)$ is false. Hence, f is not a tautology.

2) f. is not a tautology implies $\exists$ an assignment of values $(t_1, \ldots, t_n)$ to the n variables of f such that $f(t_1, \ldots, t_n)$ is false. Then $f\ c\ t_1 \ldots t_n \in L$ and $f\ c\ 0^n \in h(L)$.

<u>Corollary 2.8</u>: PTIME = NPTIME if and only if PTIME is closed under ϵ-free homomorphism.

Actually more is true. We know that the language $h(L)$ of Theorem 2.7 generates NPTIME by applications of deterministic polynomial time bounded algorithms. That is given a language $L_i \in$ NPTIME, there is a polynomial time bound algorithm S_i such that $x \in L_i$ if and only if $S_i(x) \in h(L)$. Thus, NPTIME has a generator L which is an element of PTIME.

Thus, we have a new technique for investigating the relationship between PTIME and NPTIME. Let there be a set of operations H such that the following hold:

1) NPTIME is closed under application of elements of H and
2) there is a set of languages $L_0 \subset$ PTIME such that $H(L_0)$ generates NPTIME by applications of deterministic polynomial-time bounded algorithms.

Then PTIME = NPTIME if and only if PTIME is closed under application of elements of H.

Finally, we note that the equivalence of (1) and (2) of Theorem 2.3 and Corollary 2.8 are due independently to R. Book and appear in [1].

Section 3: Greibach Undecidability Theorems

We state and prove new extensions of Greibach's undecidability theorems (Greibach [6]). We show that these extensions lead, naturally, to analoguous theorems, which give nontrivial lower bounds on the minimal deterministic time complexity needed to decide many properties of the regular sets.

We state without proof Greibach's First Undecidability Theorem. We then state and prove our extension of this theorem. We give two examples where our theorem applies and that of Greibach does not. The reader can easily verify that any predicate that satisfies the conditions of Greibach's theorem also satisfies the conditions of our theorem. Thus, our theorem is strictly stronger than that of Greibach.

We state several definitions:

Definition 3.0: $L/c = \{x \mid c\,x \in L\}$. We call the set L/c the quotient of L with the set $\{c\}$.

Definition 3.1 (Greibach [6]): An effective family of languages is a quintuple $(\Sigma, F, f_1, f_2, \mu)$ where

1) Σ is a countable vocabulary and μ a total recursive function such that for any finite subset Σ_1 of Σ, $\mu(\Sigma_1)$ is in $\Sigma - \Sigma_1$. F is a family of languages.

2) f_1 is a function from N onto F such that the mapping g defined on $N \times \Sigma^*$ by

$$g(n,w) = \begin{cases} 1 \text{ if } w \text{ is in } f_1(n) \\ \text{undefined otherwise} \end{cases}$$

is partial recursive.

3) f_2 is a total recursive function from N into the finite subsets of Σ such that for all n in N,

$$f_1(n) \subseteq [f_2(n)]^* .$$

Definition 3.2: (Σ,F,f_1,f_2,μ) is effectively closed under a binary operation α on F, if there exists a total recursive function $\bar{\alpha} : N \times N \to N$ such that

$$f_1(\bar{\alpha}(n_1,n_2)) = \alpha(f_1(n_1),f_1(n_2)).$$

(Σ,F,f_1,f_2,μ) is effectively closed under a binary operation β on F and the regular sets over Σ (denoted by R_2), if there exists a total recursive function $\bar{\beta} : N \times R_\Sigma \to N$ such that $f_1(\bar{\beta}(n,R)) = \beta(f_1(n),R)$.

Theorem 3.3 (Greibach [6]): Let (Σ,F,f_1,f_2,μ) be an effective family of languages. Let (Σ,F,f_1,f_2,μ) be effectively closed under union and under concatenation by regular set and let "$f_1(n) = [f_2(n)]^*$" be undecidable for $L_1 = f_1(n)$ in F. If P is any property that is defined on F and

a) is false for at least one L_2 in F;

b) is true for all regular sets,

c) is preserved by inverse gsm, union with $\{\epsilon\}$, and intersection with regular sets,

then P is undecidable for F.

Theorem 3.4: Let $(\Sigma, F, f_1, f_2, \mu)$ be an effective family of languages. Let $(\Sigma, F, f_1, f_2, \mu)$ be effectively closed under union and under concatenation by regular set and let "$f_1(n) = [f_2(n)]^*$" be undecidable for $L_n = f_1(n)$ in F. If P is any property that is defined on F and

a) is false for at least one L_0 in F,

b) is true for all regular sets of the form $f_2(n)^* \, c \, f_2(n_0)^*$ where $f_1(n_0) = L_0$ and $c = \mu(f_2(n) \cup f_2(n_0))$,

c) is preserved by quotient (/) with single symbol, then P is undecidable for F.

Proof of 3.4: For $i \neq 0$, denote $f_1(i)$ by L_i and denote $f_2(i)$ by Σ_i. Let $L_0 = f_1(n_0)$ and $\Sigma_0 = f_2(n_0)$. Given any i, we can effectively find an index i_0 such that

$L_{i_0} = f_1(i_0) = L_i \, c \, \Sigma_0^* \cup \Sigma_i^* \, c \, L_0$ where $c = \mu(\Sigma_i \cup \Sigma_0)$.

Case 1. $L_i = \Sigma_i^*$.

If $L_i = \Sigma_i^*$, then by (b) $L_{i_0} = \Sigma_i^* \, c \, \Sigma_0^*$ and $P(L_{i_0})$ is true.

Case 2. $L_i \subsetneqq \Sigma_i{}^*$.

If $L_i \subsetneqq \Sigma_i{}^*$, there is an $x \in \Sigma_i{}^* - L_i$.
But $p(L_{i_0}/xc) = p(L_0)$. By (c) this implies that $p(L_i)$ = false.

Hence, combining cases 1 and 2 $L_i = \Sigma_i{}^*$ if and only if $p(L_{i_0})$ is true.

Corollary 3.5: Let F be the context-free languages over Σ. Let $\mathscr{L}$ denote the regular sets over $\Sigma \cup \{a^n b^n | n \geq 1\}$ where a and b are two fixed distinct elements of Σ. Let $\hat{\mathscr{L}}$ be the closure of $\mathscr{L}$ under / with single symbol. Let P be the property "L_i is an element of $\hat{\mathscr{L}}$." Then P is undecidable.

Proof of 3.5: It is obvious that P satisfies the conditions of Theorem 3.4. P does not satisfy the conditions of Theorem 3.3, since it is not closed under inverse gsm. Let $c, d \in \Sigma$ where c and d are distinct and are not equal to a or to b . Let $h(c) = a$ and $h(d) = b$, then $h^{-1}(\{a^n b^n | n \geq 1\}) = \{c^n d^n | n \geq 1\}$.

Definition 3.6 (Zalcstein [6]): Let k be a positive integer. For $w \in \Sigma^+$ of length $\geq k$, let $L_k(w)$, $R_k(w)$ and $I_k(w)$ be, respectively, the prefix of w of length L, the suffix of w of length k, and the set of all interior proper subwords of w of length k. If $|w| = k$, then $L_k(w) = R_k(w) = w$. If $|w| = k$ or $k+1$, then $I_k(w)$ is empty.

We say that two words, $w, w' \in \Sigma^+$ of length $\geq k$ <u>have the same k-test vectors</u> if and only if $L_k(w) = L_k(w')$, $R_k(w) = R_k(w')$ and $I_k(w) = I_k(w')$. Then $L \subseteq \Sigma^*$ is <u>k-testable</u> if and only if for all $w, w' \in \Sigma$ of length $\geq k$, if w and w' have the same k-test vectors, then $w \in L$ if and only if $w' \in L$. A language L is <u>locally testable</u> (LT) if and only if it is k-testable for some $k > 0$.

<u>Note</u>: Readers familiar with McNaughton and Papert [11] will notice that the above definitions from Zalestein are very similar to several definitions in Chapter 2 of <u>Counter-free Automata</u>.

<u>Lemma 3.7</u>: The locally testable regular sets (LT) over a countable (finite) alphabet Σ are closed under $/$ with single symbol.

<u>Proof of Lemma 3.7</u>: Let L be a k-locally testable with $k > 0$. Let $|w|$ and $|w'| \geq k$. Let $L_k(w) = L_k(w')$, $R_k(w) = R_k(w')$, and $I_k(w) = I_k(w')$.

1) $L_k(w) = L_k(w')$ implies $L_k(cw) = L_k(cw')$,

2) $R_k(w) = R_k(w')$ implies $R_k(cw) = R_k(cw')$,

and 3) $I_k(w) = I_k(w')$ and $L_k(w) = L_k(w')$ implies $I_k(cw) = I_k(cw')$. (1) follows since $L_k(cw)$ $(L_k(cw'))$ equals c concatenated with the first $k-1$ letters of $L_k(w)(L_k(w'))$. (2) follows since $R_k(cw) = R_k(w)$ and $R_k(cw') = R_k(w')$. (3) follows since all interior segments of cw and cw' of length k (except for the first such segment in cw and cw') are interior segments of w and

and w'. But the first interior segments of length k in cw and cw' (respectively) are $L_k(w)$ and $L_k(w')$ (respectively), which are equal by assumption. A picture may help here.

$$w = \rlap{$\overbrace{a_1\, a_2 \cdots\quad a_k}^{L_k(w)}$}a_1\, \underbrace{a_2 \cdots\quad a_k\, a_{k+1}}_{I_k(w)} \qquad cw = \rlap{$\overbrace{c\, a_1\, a_2 \cdots a_{k-1}\, a_k}^{L_k\ w)}$}c\, \underbrace{a_1\, a_2 \cdots a_{k-1}\, a_k\, a_{k-1}}_{I_k(cw)} \cdots$$

$$w' = \rlap{$\overbrace{a_1\, a_2 \cdots a_k}^{L_k(w')}$}a_1\, \underbrace{a_2 \cdots a_k\, a_{k+1} \cdots}_{I_k(w')} \qquad cw' = \rlap{$\overbrace{c\, a_1\, a_2 \cdots a_{k-1}\, a_k}^{L_k(cw')}$}c\, \underbrace{a_1\, a_2 \cdots a_{k-1}\, a_k\, a_{k+1}}_{I_k(cw')} \cdots$$

w is an element of L/c if and only if $c\,w \in L$. Since L is locally testable, if w and w' have the same k-test vectors then $w \in L/c$ if and only if $w' \in L/c$. Hence L/c is locally testable.

<u>Definition 3.8</u>: [11]: Let <u>LTO</u> be the smallest class of regular events that contains all the locally testable events and is closed under the Boolean operations and concatenation. Let <u>SF</u> be the class of all regular sets that can be represented by a star-free extended regular expression (i.e., an extrended regular expression using

only $\cup$, $\cdot$, $\cap$, and $\neg$). In [11] McNaughton and Papert show that LTO = SF.

<u>Corollary 3.9</u>: Let F be the context-free language over Σ. Let P be "$L_i \in$ LTO", then P is undecidable.

<u>Proof of 3.9</u>: It is not difficult to verify that all regular sets of the form $\Sigma_1^* c \Sigma_2^*$ with $c \notin \Sigma_1$ and $c \notin \Sigma_2$ are elements of LTO. Thus, the lemma follows from Theorem 3.4 provided LTO is closed under quotient with single symbol.

Let LTO(k) = $\{R | R \in$ LTO and $\exists\, x_1, \ldots, x_n \in$ LT such that x results from $x_1, \ldots, x_n$ by at most k applications of $\cup$, $\cap$, $\cdot$, and $\neg\}$. For k = 0 , LTO(0) = LT, which is closed under quotient with single symbol by Lemma 3.7. Let k = n+1 , then $Z \in$ LTO(n+1) if and only if

a) $Z = x_1 \cup x_2$ in which case $Z/c = {}^{x_1}/c \cup {}^{x_2}/c$,

b) $Z = x_1 \cap x_2$ in which case $Z/c = {}^{x_1}/c \cap {}^{x_2}/c$,

c) $Z = \neg x_1$ in which case $Z/c = {}^{(\neg x_1)}/c$, or

d) $Z = x_1 \cdot x_2$ in which case $Z/c =$ if $\Lambda \in x_1$ then $\frac{x_1}{c} \cdot x_2 \cup \frac{x_2}{c}$ else $\frac{x_1}{c} \cdot x_2$, where x_1 , $x_2 \in$ LTO(n). But by induction $Z \in$ LTO.

Corollary 3.9 suggests that many predicates of the form "L_i is in P", where P is a subset of the regular sets over a countable alphabet Σ and L_i is a context-free language over Σ , are undecidable.

Proposition 3.10: Let G be the set of context-free grammars over a countably infinite alphabet Σ. Let $P \subseteq$ regular sets over Σ. Let P contain all regular sets of the form $\bar{\Sigma}^*$, where $\bar{\Sigma}$ is a finite subset of Σ. Let G_i be an arbitrary context-free grammar. Then the predicate "$L(G_i) \in P$" is undecidable.

Proof of 3.10: The proof is exactly the same as that of Theorem 14.6 in Hopcroft and Ullman [8]. Therefore, we only sketch it. Let $A = \{w_1,\ldots,w_k\}$ and $B = \{x_1,\ldots,x_k\}$ be two lists of strings in $\bar{\Sigma}^+$. Let $k = \{a_1,a_2,\ldots,a_k\}$ be a set of k distinct symbols not in $\bar{\Sigma}$. Then given A, B, and K, we can effectively find a context-free grammar G_i such that

1) $L(G_i) \subseteq (\bar{\Sigma} \cup K \cup \{c\})^*$,

2) $\overline{L(G_i)}$ is a context-free language if and only if $\overline{L(G_i)}$ is empty, and

3) $\overline{L(G_i)} = \{w_j,w_{j_2},\ldots,w_{j_m} a_{j_m} a_{j_{m-1}},\ldots,a_{j_1} \subset a_{j_1} a_{j_2},\ldots,a_{j_m} x_{j_m}^R x_{j_{m-1}}^R,\ldots,x_{j_1}^R | w_{j_1} w_{j_2},\ldots,w_{j_m} = x_{j_1} x_{j_2},\ldots,x_{j_m}\}$. Now, $L(G_i) \in P$ implies $\overline{L(G_i)}$ is regular and, therefore, context-free. Hence $L(G_i) \in P$ implies $L(G_i) = (\Sigma \cup K \cup \{c\})^*$. Since $(\Sigma \cup K \cup \{c\})^* \in P$, $L(G_i) \in P$ if and only if $L(G_i) = (\bar{\Sigma} \cup K \cup \{c\})^*$ or equivalently $\overline{L(G_i)} = \phi$. But $\overline{L(G_i)} = \phi$ if and only if the Post Correspondence Problem with lists A and B has no solution.

We next state the Second Greibach Undecidability Theorem (Theorem 2 in [6]). We state and prove our new extension of this theorem. We give several examples where our theorem holds and that of Greibach does not. Finally, we state a subrecursive analogue of our theorem.

Theorem 3.11: Let (Σ,F,f_1,f_2,μ) be an effective family of languages that is effectively closed under concatenation. Let "$L_1 = \phi$" be undecidable for F. If P is any property which (a) is false for some ϵ-free L_2 in F, (b) is true of ϕ, and (c) is preserved by inverse gsm and intersection with regular sets, then P is undecidable for F.

Theorem 3.12: Let (Σ,F,f_1,f_2,μ) be an effective family of languages that is effectively closed under concatenation. Let "$= \phi$" be undecidable in F. If P is any property which is

(a) false for some L_2 in F

(b) true of ϕ , and

(c) preserved by quotient with single symbol,

then P is undecidable for F.

Proof of 3.12: Let $L_i \subset \Sigma_i^* = f_1(i)^*$. Let $L_0 \subseteq \Sigma_0^*$ be a language such that $P(L_0)$ is false. Let $c = \mu(f_2(i) \cup f_2(0))$, i.e., let c be a character not in $\Sigma_i \cup \Sigma_0$. Let $L_j = L_i c L_0$. Let x be a string of minimal length in L_i. Then $P(L_j/cx) = p(L_0) =$ false. Thus, $P(L_j)$ is true if and only if $L_i = \phi$.

<u>Corollary 3.13</u>: Let F be an effective family of languages that is effectively closed under concatenation. Let "$=\phi$" be undecidable in F. If there are both finite and infinite languages in F, then the predicates "L_i if finite" and "L_i is infinite" are undecidable.

<u>Proof of 3.13</u>: The finite sets are closed under quotient with single symbol. <u>Note</u>: Neither finiteness nor infiniteness is preserved under inverse gsm.

Finally, if the recently announced result of L. Stockmeyer that the emptiness problem for extended regular expressions is not elementary recursive is correct, we get the following subrecursive analogue of Theorem 3.11.

<u>Theorem 3.14</u>: Let P be a nontrivial property of the regular sets such that (1) P is true of $\emptyset$ and (2) P is preserved by quotient with single symbol, then the set $P' = \{\alpha | \alpha$ is an extended regular expression and $P(L(\alpha))$ is true$\}$ is not elementary recursive.

<u>Lemma 3.15</u>: The set $P' = \{\alpha | \alpha$ is an extended regular expression and $L(\alpha) \in$ S F, i.e., $L(\alpha)$ is Star-free$\}$ is not elementary recursive.

<u>Proof of 3.15</u>: The lemma follows from Theorem 3.14 provided SF is closed under quotient with single symbol. But in the proof of Corollary 3.9, we showed that LTO, which equals SF, is closed under quotient with single symbol.

Section 4: The Classes PTAPE and PTIME

In this section we investigate the interrelationship between PTIME and PTAPE. We find many languages $L_i \in$ PTAPE such that $L_i \in$ PTIME only if PTAPE = PTIME. We also classify the deterministic time complexity of many problems of the regular sets.

We begin with several definitions, which are essentially the same as those in Meyer [12]. Those readers familiar with Meyer [12] may wish to proceed directly to the statement and proof of Theorem 4.6.

Definition 4.1: Let M be any Tm with tape symbols T and states S. Assume $0,1,b\!\!/$ are elements of T, where $b\!\!/$ denotes a blank tape square. An instantaneous description (i.d.) of M is a word in $T^* \cdot (S \times T) \cdot T^*$.

Definition 4.2: An initial instantaneous description is a word in $(\{q_0\} \times (T - b\!\!/)) \cdot T^*$.

Definition 4.3: Given any i.d. $x = y \cdot (s \times t) \cdot z$ for y,z in T^*, the next i.d., $\text{next}_m(x)$ is defined as follows: if when M is in state S with its read-write head scanning symbol t, M enters state S' and writes symbol t' then $\text{Next}_m(x)$ is

1) $y \cdot (s' \times t') \cdot z$ if M does not shift its head,
2) $y \cdot t' \cdot (s' \times u)\ w$ if M shifts its head right and $z = uw$ for $u \in T$ and $w \in T^*$,
3) $w \cdot (s' \times u) \cdot t' \cdot z$ if M' shifts its head left and $y = w\ u$ for $u \in T$ and $w \in T^*$,

4) undefined if (sxt) is a halting condition, or if (sxt) is the right most symbol of x and M shifts right, or if (sxt) is the leftmost symbol of x and M shifts right.

<u>Note</u>: The Tm may move onto the blank $\not{b}$.

<u>Definition 4.4</u>: $\text{Next}_m(x,0) = x$ if x is an i.d and is undefined otherwise.

$$\text{Next}_m(x,n+1) = \text{Next}_m(\text{Next}_m(x,n)).$$

<u>Definition 4.5:</u> Let $\#$ be a symbol not in $T \cup (S\text{x}T)$.
The <u>computation</u> $C_m(x)$ of M from x is the following word in $(\{\#\} \cup T \cup (S \times T))^*$: $C_m(x) = \#\cdot\text{Next}_m(x,0)\cdot\#\cdot\text{Next}_m(x,1)\cdot\#\cdot \ldots \cdot$
$\#\cdot\text{Next}_m(x,n)\cdot\#$.

Here, n is the least positive integer such that $(q_f x t)$ occurs in $\text{Next}_\mu(x,u)$ for some $t \in T$ and designated halting state q_f. The computation is undefined if there is no such n.

Given M as in the preceding definitions, let $\Sigma = \{\#\} \cup T \cup (S \times T)$. For any i.d. x, let $C_M(i,x)$ be the <u>ith</u> symbol of $C_M(x)$ for $1 \leq i \leq |C_M(x)|$. There is a function $f_M : \Sigma^3 \to \Sigma$ such that for any i.d. x and any integer i, with

$$|x| + 2 \leq i \leq |C_M(x)| ,$$

$C_M(i,x) = f_m(C_m(i - (|x| + 2), x),$
$C_M(i-(|x|+1)x), C_m(i-(|x|, x))$. This follows since the <u>ith</u> symbol of next(y) is determined uniquely by the (i-1)<u>th</u>, <u>ith</u> and (i+1)<u>th</u> symbols of y.

Theorem 4.6: Let M be a deterministic Tm with states S, tape alphabet T, and designated halting state q_f in S. Let $\Sigma = \{\#\} \cup T \cup (S \times T)$. Let $\Sigma_1 = \Sigma \cup \{A,B,\$\}$. Let L be a fixed regular expression over $\{A,B\}$. Let $P(n) = j \cdot n^k$ for some fixed positive integers j and k. Let y be an initial i.d. of M. Then, there is a Tm M' which, started with any word $y \cdot \# \cdot b\!\!\!/^{P(|y|)}$ on its tape, halts with a regular expression β over Σ_1 such that

$$L(\beta) = \begin{cases} \Sigma_1{}^* - \{C_M(b\!\!\!/^{P(|y|)} y\ b\!\!\!/^{P(|y|)}\} \cdot \$ \cdot L & \text{if } \{C_M(b\!\!\!/^{P(|y|)} y\ b\!\!\!/^{P(|y|)})\} \\ \Sigma_1{}^* & \text{is defined, and otherwise.} \end{cases}$$

Moreover M' uses only linear space and polynomial time in $|y|$.

Note: The reader familiar with Meyer [12] should notice that Theorem 4.6 differs from the Simulation Lemma, since we have embedded the fixed language L into β. We exploit this difference repeatedly in the remainder of this section.

Proof of Theorem 4.6:

β is characterized as follows:

1) words that do not begin with $\#\ b\!\!\!/^{P(|y|)} y\ b\!\!\!/^{P(|y|)} \#$, or
2) words that do not contain q_f to the left of the \$-sign, or
3) words that are not of the form $\# \cdot \Sigma^* \cdot \# \cdot \$ \cdot L$, or
4) words that violate the functional condition (explained above) to the left of the \$-sign.

Let $y = y_1, \ldots, y_i$ for some $i \geq 1$. Then

$$\beta = [((\Sigma_1 - \#) \cup \# \cdot ((\Sigma_1 - \not{b}) \cup \not{b} \cdot (\ldots((\Sigma_1 - \not{b}) \cup \not{b}\ ((\Sigma_1 - \#))\ldots))] \cdot$$
$$\Sigma_1^* \cup [\Sigma_1 - [(\{q_f\} \times T) \cup \$]]^* \cdot \$ \cdot \Sigma_1^* \cup$$

some regular expression for $\overline{(\# \cdot \Sigma^* \cdot \# \cdot \$ \cdot L)} \cup$

$$\bigcup_{\sigma_1, \sigma_2, \sigma_3 \in \Sigma} [\Sigma^* \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdot \Sigma^{P(|y|)} \cdot \Sigma^{|y|-1} \cdot \Sigma^{P(|y|)} \cdot$$
$$(\Sigma - f_m(\sigma_1, \sigma_2, \sigma_3)) \cdot \Sigma^* \cdot \$ \cdot \Sigma_1^*].$$

Since $\# \cdot \Sigma^* \cdot \# \cdot \$ \cdot L$ is regular, so is its complement. Therefore, there is a regular expression δ such that $L(\delta) = \overline{\# \cdot \Sigma^* \cdot \# \cdot \$ \cdot L}$. L is fixed for all inputs to M'. Therefore, the length of δ contributes only a constant amount of tape and a polynomial amount of time to the cost of the computation.

<u>Claim</u>: $L(\beta) = \Sigma_1^* - \{C_m(\not{b}^{P(|y|)} y\ \not{b}^{P(|y|)})\} \cdot \$ \cdot L$ if $\{C_M(\not{b}^{P(|y|)} y\ \not{b}^{P(|y|)})\}$ is defined, and $L(\beta) = \Sigma_1^*$ otherwise.

<u>Proof of Claim</u>: Suppose $\{C_M(\not{b}^{P(|y|)} y\ \not{b}^{P(|y|)})\} \cdot \$ \cdot L$ is defined. Then there is a unique element of Σ^*, Z, such that $Z = C_M(\not{b}^{P(|y|)} y\ \not{b}^{P(|y|)})$. But

a) Z begins with $\#\ \not{b}= {}^{P(|y|)} y\ \not{b}^{P(|y|)} \#$, and

b) Z contains a symbol in $\{q_f\} \times T$, and

c) Z is of the form $\# \cdot \Sigma^* \cdot \#$, and

d) Z does not violate the functional condition f_M.

Hence, any string in $\{C_M(\not b^{P(y)}y\not b^{P(y)})$ $\$ L$ does not satisfy (1), (2), (3) or (4). Therefore, $L(\beta) \subset$

$\overline{\{C_M(\not b^{P(|y|)}y\not b^{P(|y|)})\cdot\$\cdot L}$.

Suppose there is a string Z which is not an element of $L(\beta)$. Then from (3) $Z = \#\cdot x\cdot\#\cdot\$\cdot y$, where $x \in \Sigma^*$ and $y \in L$. But $\#\cdot x\cdot\#$ must satisfy the following in order for $Z = \#\cdot x\cdot\#\cdot\$\cdot y$ not to be an element of $L(\beta)$:

a) $\#\cdot x\cdot\#$ begins with $\#\not b^{P(|y|)}y\not b^{P(|y|)}\#$, and

b) $\#\cdot x\cdot\#$ contains a symbol in $\{q_f\} \times T$, and

c) $\#\cdot x\cdot\#$ ends in $\#$, and

d) $\#\cdot x\cdot\#$ does not violate the functional condition f_M. Hence, $\#\cdot x\cdot\# = C_M(\not b^{P(|y|)}\cdot y\cdot\not b^{P(|y|)})$.
Hence, $Z \in \{C_m(\not b^{P(|y|)}\cdot y\cdot\not b^{P(|y|)})\}\cdot\$\cdot L$.

$\therefore \overline{L(\beta)} \subset \{C_m(\not b^{P(|y|)}\cdot y\cdot\not b^{P(|y|)})\}\cdot\$\cdot L$.

$\therefore L(\beta) \supset \overline{\{C_m(\not b^{P(|y|)}\cdot y\cdot\not b^{P(|y|)})\}\cdot\$\cdot L}$.

The proof that the space required by M' is linear and that the time required by M' is bounded by a polynomial is standard.

In Theorem 4.6 the Tm M has an arbitrarily large tape alphabet T and state set S. Consequently, the regular expression β is over the arbitrarily large alphabet Σ_1, where $\Sigma_1 = \{\#,\$,A,B\} \cup T \cup (S \times T)$. To relate our work to that of Karp [10], we encode Σ_1 into $\{0,1\}^*$.

Let $|\Sigma_1| = n$. We enumerate the elements of Σ_1 in such a way that A is the first, and B is the second element in the enumeration. Let the $i^{\underline{th}}$ element in the enumeration be denoted by σ_i. We define $i: \Sigma_1 \rightarrow \{0,1\}^+$ as follows:

$$i(A) = 0,$$
$$i(B) = 1,$$
$$i(\sigma_j) = 1^j 0^{n+1-j} \text{ for } 2 \leq j \leq n.$$

We extend the domain of i to Σ_j* in the usual manner, i.e., $i(\epsilon) = \epsilon$ and for $y = y_1 \ldots y_m$ with each $y_j \in \Sigma$, $i(y) = i(y_1), \ldots, i(y_m)$. Note: The encoding i is not 1-1.

Corollary 4.7: Let M, Σ, Σ_1 and β be defined as in Theorem 4.6. Then there is a regular expression Γ over $\{0,1\}$ such that

$$L(r) = \{0,1\}^* \text{ if } L(p) = \Sigma_1^* \text{ and}$$
$$L(\Gamma) = \{0,1\}^* - i(z)\cdot i(\$)\cdot i(L)\cdot \text{if } L(\beta) =$$
$$\Sigma_1^* - Z\cdot\$\cdot L$$

Proof of Corollary 4.7: Define Γ as follows: Γ is the union of:

(1) words that are not of the form i(#w#\$w') where $w \in \Sigma^*$ and $w' \in L$,

(2) words x with x equal to i(#w#\$w'), $w \in \Sigma^*$, $w' \in L$, such that #w# does not begin with $\#\cancel{b}^{p(|y|)} y \cancel{b}^{p(|y|)}\#$

(3) words x with x equal to i(#w#\$w'), $w \in \Sigma^*$, $w' \in L$,

such that $\{q_f\}xt$ is not a substring of $\#w\#$, where q_f is the accepting state of M and $t \in T$, and

(4) words x with $x = i(\#w\#\$w')$, such that $\#w\#\$w' \notin$

$$\bigcup_{\sigma_1,\sigma_2,\sigma_3 \in \Sigma} \Sigma^* \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdot \Sigma^{p(|y|)} \cdot \Sigma^{|y|-1} \cdot \Sigma^{p(|y|)} \cdot (\Sigma - f_m(\sigma_1,\sigma_2,\sigma_3)) \cdot \Sigma^* \cdot \$ \cdot \{0,1\}^*.$$

<u>Note</u>: If $x = i\ (\#w_1\#\$w_1')$ and $x = i\ (\#w_2\#\$w_2')$ with $w_1, w_2 \in \Sigma^*$ and $w_1', w_2' \in L$, then $w_1 = w_2$ and $w_1' = w_2'$. This follows since i restricted to $\Sigma \cup \{\#,\$\}$ is 1-1, and i restricted to $\{A,B\}$ is 1-1.

Then $x \in L(\Gamma)$ if and only if x is not the encoding of a word in $\#\Sigma^*\#\$L$ or x is not of the form $i(z \cdot \$ \cdot L)$, where z is the computation of M from y.

We use Theorem 4.6 together with Corollary 4.7 to prove several results about the minimal time needed to deterministically decide properties of the regular expressions over $\{0,1\}$. We relate the time needed to decide these properties to the time needed to recognize arbitrary context-sensitive languages.

We need the following definitions, which are analogous to Cook's and Karp's p-complete and p-hard problems.

<u>Definition 4.8</u>: A language L is <u>tape-hard</u> if PTAPE $\propto$ L, i.e., if $\forall$ L' $\in$ PTAPE, L' $\propto$ L (See page 14.)

<u>Definition 4.9</u>: A language L is <u>tape-complete</u> if L $\in$ PTAPE and L is tape-hard.

<u>Proposition 4.10</u>: If any tape-complete problem is in PTIME, then

1) all tape-complete problems are in PTIME and
2) PTIME = NPTIME.

As an immediate consequence of Theorem 4.6 and Corollary 4.7 we have the following general theorem:

<u>Theorem 4.11</u>: Let P be a nontrivial property of the regular sets such that $P(\{0,1\}^*)$ is true and P is preserved under quotient with single symbol. Then the set $R = \{\alpha|\ \alpha$ is a regular expression and $\rho(L(\alpha))$ is true) is tape-hard.

<u>Proof of 4.11</u>: Let L in Corollary 4.7 be a regular set over $\{0,1\}$ such that $P(\neg L)$ is false. Let γ be the regular expression of invalid computations defined in Corollary 4.7. $L(\Gamma) = \{0,1\}^*$ if there is no valid computation. $L(\Gamma) = \{0,1\}^* - i(z)\cdot i(\$)\cdot L$ if there is a valid computation. If $L(\gamma) = \Sigma_1{}^*$, then $P(L(\gamma))$ is true. If $L(\gamma) = \{0,1\}^* - i(z)\cdot i(\$)\cdot L$, then $L(\gamma)/i(z)\cdot i(\$) = \neg L$, which implies $P(L(\gamma))$ is false. Hence $\gamma \in R$ if and only if $L(\gamma) = \{0,1\}^*$, if and only if $L(\beta)$ of Theorem 4.6 equals $\Sigma_1{}^*$.

<u>Definition 4.12</u> (Meyer and Stockmeyer [13]): A <u>regular expression with squaring</u> (r.e.s.) over an alphabet Σ is defined recursively as follows:

1) for all $\sigma \in \Sigma$, σ is an r.e.s.,
2) λ and ϕ are r.e.s.',

and 3) if α and β are r.e.s.', then so are $(\alpha) \cup (\beta)$, $(\alpha)\cdot(\beta)$, $(\alpha)^*$, and $(\alpha)^2$.

Using the idea of Theorem 4.6 and the proof of Lemma 2.1 in [ ] we have

<u>Corollary 4.13</u>: Let P be a nontrivial property of the regular sets such that $P(\{0,1\}^*)$ is true and P is preserved under quotient with single symbol. Then the set $R = \{\beta | \beta$ is a regular expression with squaring and $P(L(\beta))$ is true$\}$ requires exponential space and hence exponential time.

<u>Proof of 4.13</u>: We only sketch the proof. Let M,Q,T and $\Sigma = \{\#\} \cup T \cup Q \times T$ be defined as above (Definitions 4.1 through 4.5). Given $y = y_1,\dots,y_n$ we construct a r.e.s. β of length $\leq C_m$ for some constant c such that if $x \notin L(M)$ then $L(\beta) = (\Sigma \cup \$)^*$ else $L(\beta) = (\Sigma \cup \$)^* - C_m(x)\cdot\$\cdot L$.
Let $\Sigma_1 = (\Sigma \cup \$)$.

$$\begin{aligned}
\beta = ((\Sigma_1-\#) \cup \#\ ((\Sigma_1-q_0 \times y_1)) \cup x_1\cdot((\Sigma_1-y_2) \cup \\
x_2\cdot((\Sigma_1-y_,) \cup \dots \cdot (\Sigma_1-y_n))\dots)\cdot \Sigma_1^* \quad & \cup \\
\Sigma_1^{n+1}\cdot b^* \cdot (\Sigma_1-\{b,\#\}) \cdot \Sigma_1^* \quad & \cup \\
\# \cdot (\Sigma_1 \cup \lambda)^{2^n-1} \cdot \# \cdot \Sigma_1^* \quad & \cup \\
\# \cdot \Sigma_1^{2^n} \cdot (\Sigma_1-\#) \cdot \Sigma_1^* \quad & \cup \\
(\Sigma_2 - (\{q_f\} \times T))^* \quad & \cup \\
\neg\ (\# \cdot \Sigma^* \cdot \# \cdot \$ \cdot L)) \quad & \cup
\end{aligned}$$

$$\bigcup_{\sigma_1,\sigma_2,\sigma_3 \in \Sigma} (\Sigma^*\cdot\sigma_1\cdot\sigma_2\cdot\sigma_3 \cdot \Sigma^{2^n-1}\cdot(\Sigma-f\ (\sigma_1,\sigma_2,\sigma_3))\cdot\Sigma^*\cdot\$\cdot\Sigma_1^*)$$

The remainder of the proof follows from the proofs of Corollary 4.7 and Theorem 4.11 together with the following well-known fact (See Hopcroft and Ullman [8] pp. 150 and 151): If $L_1(n)$ and $L_2(n)$ are constructible tape functions with $\inf_{n\to\infty} \frac{L_1(n)}{L_2(n)} = 0$ and $L_2(n) \geq \log n$, then there is a set accepted by an $L_2(n)$ tape-bounded Tm, but not accepted by any $L_1(n)$ tape-bounded Tm.

We next state several definitions and prove several theorems which show that Theorems 4.11 and 4.13 are not vacuous.

<u>Definition 4.14</u> (Brzozowski [3]): A regular set R is a <u>star-event</u> if there is a regular set S such that $R = S^*$. It is easy to verify that a regular set R is a star-event if and only if $R = R^*$.

<u>Theorem 4.15</u>: The following are equivalent:

1) $R = \{(\alpha,\beta) | \alpha$ and β are equivalent regular expressions over $\{0,1\}\} \in$ PTIME;

2) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) = \{0,1\}^+\} \in$ PTIME;

3) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) = \{0,1\}^*\} \in$ PTIME;

4) $R = \{(\alpha,\beta) | \alpha$ and β are a pair of equivalent nondeterminstic finite automata with input alphabet $\{0,1\}\} \in$ PTIME;

5) $R = \{\alpha | \alpha$ is a cofinite regular expression over $\{0,1\}\} \in$ PTIME;

6) $R = \{\alpha | \alpha$ is a coinfinite regular expression over $\{0,1\}\} \in$ PTIME;

7) $R = \{(\Gamma',x) | \Gamma'$ is the code $i(\Gamma)$ of a context-sensitive grammar Γ and x is a string over $\{0,1,\#\}$ such that $x \in L(i(\Gamma))\} \in$ PTIME;[1]

8) CSL $\subsetneq$ PTIME, DCSL $\subsetneq$ PTIME;

9) $R = \{(\alpha,\beta) |$ α and β are regular expressions over $\{0,1\}$ and α is a regular expression of minimal length such that $L(\alpha) = L(\beta)\} \in$ PTIME;

10) $R = \{\alpha | \alpha$ is a Star-event over $\{0,1\}\} \in$ PTIME;

and 11) PTAPE = PTIME.

<u>Proof of 4.15</u>: Proof of (1), (2), (3) $\equiv$ (11) and (4) $\equiv$ (11). The sets R are tape-hard. This follows immediately from Theorem 4.6 and Corollary 4.7. There are deterministic polynomial time algorithms to convert a regular expression to an equivalent nondeterministic finite state automaton. Therefore,

---

[1]For R in 4 to be a language, it must be over a finite alphabet. Thus we must encode the productions of the context-sensitive grammar Γ. $\Gamma' = i(\Gamma)$ is not, in general, a CSG, since it has rules of the form $i(\psi)\cdot i(A)\cdot i(\phi) \to i(\psi)\cdot i(\beta)\cdot i(\psi)$, where $|i(A)| \geq 2$.

we need only show that R of (2) is in PTAPE. In fact, we show that $\bar{R}$ is a context-sensitive language.

We construct a nondeterministic Tm whose accepted language is the set of pairs (α,β) of inequivalent nondeterministic finite automata over $\{0,1\}$. Since PTAPE is closed under complementation, this is sufficient. The Tm will nondeterministically guess, one character at a time, a string x which differentiates between α and β, i.e., x is accepted by one but not the other. We show below that the amount of tape needed to do this is linear.

Let α and β be nondeterministic finite automata. Let Q_1, Q_2 be the state set of α and β respectively. As is well-known, there are deterministic finite automata α' and β' such that $\alpha' = (K_1, \{0,1\}, \Delta_1, S_1, F_1)$ and $\beta' = (K_2, \{0,1\}, \Delta_2, S_2, F_2)$, $L(\alpha) = L(\alpha')$, $L(\beta) = L(\beta')$, $|K_1| \leq 2^{|Q_1|}$ and $|K_2| \leq 2^{|Q_2|}$. (We follow Hopcroft and Ullman [8] in the definition of finite-state automata. Here K_i is a finite set of states, Δ_i is the next state function, S_i is the start state, and F_i the set of accepting states.) Let γ be as follows:

$\gamma = (K_1 \times K_2, \{0,1\}, \Delta, (S_1,S_2), (F_1 \times F_2) \cup (K_1-F_1)\times(K_2-F_2))$, where $\Delta((p,q),\sigma) = (\Delta_1(p,\sigma),\Delta_2(p,\sigma))$ for all $(p,q) \in K_1 \times K_2$. $|K_1 \times K_2| = |K_1|\cdot|K_2| \leq 2^{|Q_1|+|Q_2|}$

$L(\gamma) = \{x \mid x \in \{0,1\}^*$ and $x \in (L(\alpha') \cap L(\beta')) \cup (\overline{L(\alpha')} \cap \overline{L(\beta')})\}$.

Hence, (1) $L(\gamma) = \{0,1\}^*$ if and only if $\alpha' \equiv \beta'$ if and only if $\alpha \equiv \beta$/

(2) $L(\gamma) = \{0,1\}^*$ if and only if $\overline{L(\gamma)} = \phi$, which is true if and only if, for all x in $\{0,1\}^*$ with $x \leq |K_1 x K_2| \leq 2^{|Q_1|+|Q_2|}$ $x \in L(\gamma)$. This follows since $\bar{\gamma} = (K_1 x K_2, \{0,1\}, \Delta, (S_1, S_2), K_1 x K_2 - [(F_1 x F_2) \cup (K_1 - F_1) x (K_2 - F_2)])$ is a deterministic finite automaton whose accepted language is $\overline{L(\gamma)}$. (See Hopcroft and Ullman [8] page 40.)

But a nondeterministic Tm M can do this in tape $O(|\alpha|+|\beta|)$. M behaves as follows.

STEP1: M marks off a counter C of length $|Q_1| + |Q_2|$.

STEP2: M guesses, 1 character at a time, all strings x of length $\leq 2^{|Q_1|+|Q_2|}$ (It uses C to keep track how long a string is.)

STEP3: M verifies that a string $x \in L(\alpha) \cap L(\beta)$ or $x \in \overline{L(\alpha)} \cap \overline{L(\beta)}$ by keeping track of all possible states of α and β they can be in when applied to x, one character at a time. It accepts if and only if there is some $x \notin (L(\alpha) \cap L(\beta)) \cap (\overline{L(\alpha)} \cap \overline{L(\beta)})$. But steps 1, 2, and 3 only require $O(|Q_1|+|Q_2|)$ tape.

Proof of (5),(6) $\equiv$ (11). Cofiniteness (Coinfiniteness) of Regular Expressions are tape-hard. This follows from the proofs of Theorem 5.6 and Corollary 4.7 with $L = 0^*$. Here γ, as defined in Corollary 4.7, is cofinite if and only if

$$\{C_m(\not{b}^{\,P(|y|)}y\,\not{b}^{\,P(|y|)})\} = \phi$$

The construction to show that Cofiniteness (Coinfiniteness) of Regular expressions is in TAPE is essentially the same as that for the proof of (4) ≡ (11). It is based upon the following fact: "The set of sentences accepted by a deterministic finite automaton with n states is infinite if and only if the automaton accepts a sentence of length ℓ, $n \le \ell < 2n$." (See Hopcroft and Ullman [8] page 40.) Again, we convert the regular expression to an equivalent nondeterministic finite automaton f with state set Q in deterministic polynomial time. The equivalent deterministic finite automaton $f' = (K, \{0,1\}, \Delta, S, F)$ has $|K| \le 2^{|Q|}$. $f'' = (K_1, \{0,1\}, \Delta, S, K-F)$ accepts the complement of $L(\alpha)$. Hence, we need only check all strings of length $\le 2 \cdot 2^{|Q|} = 2^{|Q|+1}$. This shows that thesset $\{\alpha | \alpha$ is a regular expression and $\overline{L(\alpha)}$ is infinite is context-sensitive.

Proof of (7) ≡ (11) and (8) ≡ (11): Context-Sensitive Recognition is Tape-hard since CSL $\subset$ PTIME implies PTAPE = PTIME. This follows since

1) if Context-Sensitive Recognition $\in$ PTIME, then for all context-sensitive grammar Γ, the problem

   Γ-recognition

   Input: A string $x \in \{0,1,\#\}^+$

   Property: x is in the language generated by $i(\Gamma)$

is in PTIME and

2) CSL $\subset$ PTIME implies PTAPE = PTIME. Intuitively, we show how to pad a string x so that a Tm can use the padding for work space. The proof is essentially the same as that of Theorem 2.3.

It is easily seen that a Tm can check given an input y that $y = w \cdot \#^{C_1 |w|^{C_2}}$ with $w \in \{0,1\}^+$ and C_1, C_2 positive integers in linear tape. Let $L \in$ PTAPE, then for some positive integer k_0 there is an n^{k_0} tape bounded Tm T such that L(T) = L. Encode the tape alphabet of T into $\{0,1\}^+$ as shown in Lemma 4.7. Construct M to work as follows:

STEP1: M checks that its input y is of the form $y = w\ \#^{C_1 |w|^{k_0}}$. If not M rejects y.

STEP2: If y is of the proper form, then M simulates Tm w using the remainder of y as work tape. M accepts y if and only if T accepts w. Clearly, M operates within linear space. The remaining details are similar to those in the proof of Theorem 2.3 DCSL $\subset$ PTIME implies PTAPE = PTIME follows from Savitch's Theorem (See [15]) and the padding technique. (Savitch's Theorem implies CSL $\subseteq$ dtape (n^2).)

Finally, Context-Sensitive Recognition $\in$ PTAPE. Since the length of any sentential form appearing in the derivation of a string x in $L(\Gamma)$ has length $\leq |i(\Gamma)|\ |x|$, a Tm need only nondeterministically guess all derivations which use no more than $|i(\Gamma)|\cdot|x|$ symbols. This requires only $O(|i(\Gamma)|\cdot i|x|)$ storage, since the Tm need only keep at most 2 sentential forms together with x on its tape at one time.

Proof of (9) $\equiv$ (11): It is clear that (9) in Tape-hard. That (9) can be done in tape $O[(|\alpha|+|\beta|)^2]$ follows by a construction similar to that of the proof of the equivalence of (4) and (11) together with Savitch's Theorem (i.e., CSL $\subseteq$ dtape (n^2)).

Proof of (10) $\equiv$ (11): Let L in Theorem 5.6 be $\{\Lambda\}$. Then $L(\beta) = (\Sigma_1 \cup \$)^*$ if there is a computation of M from y. Otherwise, $L(\beta) = (\Sigma_1 \cup \$)^* - Z\cdot\$$ where $|Z| \geq 2$. Hence, $L(\beta)$ is a star-event if and only if $L(\beta) = (\Sigma_1 \cup \$)^*$. Hence by Corollary 4.7 the set $\{\alpha|\alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a Star-event$\}$ is tape-hard.

Since a regular set R is a Star-event if and only if $R = R^*$, a regular expression α defines a Star-event if and only if $L(\alpha) = L(\alpha^*)$. But, the set of pairs of regular expressions (α,α^*) such that $L(\alpha) \neq L(\alpha^*)$ is context-sensitive.

Note: In [1] Book independently shows that (8) $\equiv$ (11). In [13] Meyer and Stockmeyer show that (1), (2), (3), (4) and (8) are equivalent.

Corollary 4.16: The languages L in (1), (2), (3), (4) (5), (6), (7), and (10) of Theorem 4.15 are tape-complete.

Corollary 4.17: If any of the following languages is an element of NPTIME, then NPTIME = PTAPE:

1) $R = \{(\alpha,\beta) | \alpha$ and β are regular expressions over $\{0,1\}$ and $L(\alpha) \neq L(\beta)\}$;

2) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) \neq L(\beta)\}$;

3) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) \neq \{0,1\}^+\}$;

4) $R = \{(\alpha,\beta) | \alpha$ and β are nondeterministic finite automata with input alphabet $\{0,1\}$ and $L(\alpha) \neq L(\beta)\}$;

5) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is coinfinite$\}$;

6) $R = \{(\Gamma',\lambda) | \Gamma'$ is the code $i(\Gamma)$ of a context-sensitive grammar Γ and x is a string over $\{0,1,\#\}$ with $x \in L(i(\Gamma))\}$; and

7) $R = \{\alpha | \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is not a Star-event$\}$.

Corollary 4.17 is interesting since it explains Karp's inability to show that his 3 p-hard problems (See Section 2 or Karp [10]) are p-complete. Such a proof would imply that NPTIME = PTAPE.

We now give several definitions which we will need in the remainder of this section. The reader is advised to proceed directly to Theorem 4.25 and to return to read the following definitions as needed. Let K be a positive integer. For $w \in \Sigma^+$ of length $\geq k$, let $L_k(w)$, $R_k(w)$, and $I_k(w)$ be, respectively, the prefix of w of length k, the suffix of w of length k, and the set of all interior proper subwords of w of length k. If $|w| = k$ or $k+1$, then $I_k(w)$ is empty.

<u>Definition 4.18</u> (Zalcstein [16]): A regular set R is <u>k-definite</u> if and only if for all $w \in \Sigma^*$ with $|w| \geq k$ $w \in R$ implies $R_k(w) \in A$, where A is some finite subset of Σ^*. A regular set R is <u>definite</u> if it is k-definite for some k. R is <u>reserve-definite</u> if there is a positive integer k s.t. for all $w \in \Sigma^*$ with $|w| \geq k$ $w \in R$ implies $L_k(w) \in A$, where A is some finite subset of Σ^+. R is <u>generalized</u> <u>definite</u> if

$$R = F \cup (\bigcup_{i=1}^{n} A_i \cdot \Sigma^* \cdot B_i) ,$$

where F, A_i, and B_i are finite subsets of Σ^*.

<u>Definition 4.19</u>: Let k be a positive integer. R is <u>stictly</u> <u>k-testable</u> if and only if there are finite test-sets X, Y, Z of words over Σ such that for all $w \in \Sigma^+$, $|w| \geq k$ $w \in R$ if and only if $L_k(w) \in X$, $R_k(w) \in Y$, and $I_k(w) \subseteq Z$. R is <u>stictly</u> <u>locally</u> <u>testable</u> if and only if it is strictly k-testable for some $k > 0$.

<u>Definition 4.20</u> (McNaughton and Papert [11]): We say two words $w, w' \in \Sigma^+$ of length $\geq k$ have the same k-test vectors if and only if $L_k(w) = L_k(w')$, $R_k(w) = R_k(w')$, and $I_k(w) = I_k(w')$. Then $R \subseteq \Sigma^*$ is <u>k-testable</u> if and only if for all $w, w' \in \Sigma^+$ of length $\geq k$, if w and w' have the same

k-test vectors, then $w \in R$ if and only if $w' \in R$. R is <u>locally testable</u> if and only if it is <u>k-testable</u> for some $k > 0$.

<u>Definition 4.21</u>: Let <u>LTO</u> be the smallest class of events that contains all the locally testable events and is closed under the Boolean operations and concatenation. A <u>noncounting event</u> is a regular set R such that, for some n and for all words $\bar{U}$, $\bar{V}$, and $\bar{W}$ over its alphabet, $U,V^{n+x}W \in R$ if a d only if $U V^n W \in R$, for all positive integers x.

<u>Definition 4.22</u>: The extended regular expressions (e.r.e) over a finite alphabet Σ are defined recursively as follows:

1) Any $\sigma \in \Sigma$ is an e.r.e.
2) λ and ϕ are e.r.e.'s.
3) If A and B are e.r.e.'s then so are $(A) \cup (B)$, $(A) \cap (B)$, $(A) \cdot (B)$, $(A)^*$, and $\neg(A)$.
4) A is an e.r.e. if and only if it satisfies (1), (2), or (3).

<u>Definition 4.23</u>: A <u>star-free e.r.e.</u> is an e.r.e. with no occurrence of "*". Let <u>SF</u> be the class of events that can be represented by star-free e.r.e's. In [11] McNaughton and Papert show NC = LTO = SF.

<u>Definition 4.24</u>: An event of the form $(w_1 \cup \ldots \cup w_n)^*$ with each W_i a word, is called a <u>code event</u>.

Theorem 4.25: The following are tape-hard:

1) For all $k > 0$ $L_k = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is k-definite$\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a definite event$\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a reverse definite event$\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a generalized definite event$\}$;

2) for all $k > 0$ $L_k = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a strictly k-testable event$\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a strictly locally testable event$\}$;

3) For all $k > 0$ $L_k = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is k-testable$\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a locally testable event$\}$;

4) $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) \in LTO\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha) \in SF\}$, $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a noncounting event$\}$;

5) $L = \{(P,R_0) \mid P$ is a deterministic push-down automaton, R_0 is a regular expression over $\{0,1\}$,

and $L(P) = L(R_0)\}$, $L = \{(P,R_0) \mid P$ is a deterministic pushdown automaton, R_0 is a regular expression over $\{0,1\}$, and $L(P) \supseteq R_0\}$;

6) $L = \{(T_1,T_2) \mid T_1$ and T_2 are nondeterministic true automata such that $L(T_1) = L(T_2)\}$; and

7) $L = \{\alpha \mid \alpha$ is a regular expression over $\{0,1\}$ and $L(\alpha)$ is a code event$\}$.

Proof of 4.25: (1), (3), and (4) follow from Thereom 4.11. To apply Theorem 4.11 we must show (1) that $\{0,1\}^*$ is k-definite (for all $k > 0$), definite, reverse definite, generalized definite, k-testable (for all $k > 0$), locally testable, and noncounting, and (2) that the sets of k-testable events (for all $k > 0$), definite events, reverse definite events, generalized definite events, k-testable events (for all $k > 0$), locally testable events, and noncounting events are preserved under quotient with single symbol. We showed that the k-testable events, locally testable events, and noncounting events are preserved under quotient with single symbol in Lemma 3.7 and Corollary 3.9 of Section 3.

We only show that the set of generalized events is closed under quotient with single symbol. R is generalized definite if and only if $R = F \cup (\bigcup_{i=1}^{n} A_i \cdot \{0,1\}^* \cdot B_i)$ where $F, A_1, B_1, \ldots, A_n, B_n$ are finite sets over $\{0,1\}$. Then

$$R/c = F/c \cup \Big(\bigcup_{\substack{i=1 \\ \Lambda\in A_i}}^{n} A_i/c \cdot \{0,1\}^* \cdot B_i \Big) \cup \bigcup_{\substack{i=1 \\ \Lambda\in A_i}}^{n} \{0,1\}^* \cdot B_i \cup \bigcup_{i=1}^{n} B_i/c.$$

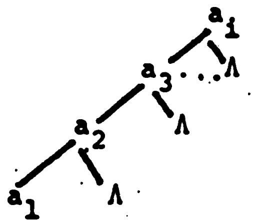
Thus R/c is generalized definite.

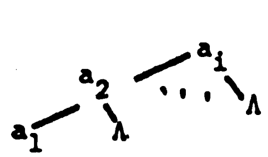
<u>Note</u>: We allow ϕ to be a definite event.

(2) follows from Theorem 4.6 and Corollary 4.7 with L in 4.7 any event over $\{0,1\}$ which has star height greater than or equal to 1, i.e. L is a counting event.

(5) follows from Theorem 4.6 and Corollary 4.7 if we fix P to be some deterministic pushdown automaton such that $L(P) = \{0,1\}^*$ and allow R_0 to vary. Then $L(P) = L(\Gamma)$, where Γ is the regular expression in Corollary 4.7 if and only if $L(\Gamma) = \{0,1\}^*$.

(6) follows from Theorem 4.6 and Corollary 4.7, since strings over $\{0,1\}$ may be viewed as very thin trees. We give an example.

$a_1a_2\ldots a_i \to$ 

It is not hard to convert a nondeterministic finite automaton M to a generalized nondeterministic finite automaton M', which accept only trees of form 

, where $a_1\, a_2\, \ldots\, a_i \in L(M)$.

7) : Γ of Corollary 4.7 is a code event if and only if $L(\Gamma) = \{0,1\}^*$.

Corollary 4.26: The following require exponential tape:

1) $L = \{\alpha|\alpha$ is a regular expression with squaring and $L(\alpha| = \{0,1\}^*\}$, $L = \{\alpha|\alpha$ is a regular expression with squaring and $L(\alpha| = \{0,1\}^+\}$, $L = \{(\alpha,\beta) \mid \alpha$ and β are regular expressions with squaring and $l(\alpha| = L(\beta)\}$;

2) $L = \{\alpha| \alpha$ is a coinfinite regular expression with squaring$\}$, $L = \{\alpha| \alpha$ is a cofinite regular expression with squaring$\}$;

3) $L = \{\alpha| \alpha$ is a regular expression with squaring and $L(\alpha)$ is a star-event, $L = \{\alpha| \alpha$ is a regular expression with squaring and $L(\alpha)$ is a code event$\}$;

4) $L = \{\alpha| \alpha$ is a regular expression with squaring and $L(\alpha|$ is a definite event$\}$, $L = \{\alpha| \alpha$ is a regular expression with squaring and $L(\alpha)$ is a generalized definite event$\}$;

5) $L = \{\alpha|\alpha$ is a regular expression with squaring and $L(\alpha)$ is strictly locally testable$\}$, $L = \{\alpha|\alpha$ is a regular expression with squaring and $L(\alpha)$ is locally testable$\}$; and

6) $L = \{\alpha \mid \alpha$ is a regular expression with squaring and $L(\alpha)$ is noncounting (star-free)$\}$.

Proof: This follows from the proof of Corollary 4.13.

Definition 4.27: A set of sentences is elementary-recursive if and only if it is recognizable in spaced bounded by $t_k(n) = \left. 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}} \right\} k$ for some fixed k and all inputs of length $\mu \geq 0$.

If the recently announced result of L. Stockmeyer, that the emptiness problem for the set of extended regular expressions over $\{0,1\}$ is not elementary recursive, is correct, then the following is true:

Theorem 4.28: Let P be a nontrivial property of the regular sets such that (1) $P(\phi)$ is true and (2) P is preserved by quotient with single symbol. Then the set $P' = \{B \mid \beta$ is an e.r.e. over $\{0,1,c\}$ and $P(L(\beta))$ is true is not elementary recursive.

Proof of 4.28: Let $L_0 \subseteq \{0,1\}^*$ be a regular set such that $P(L_0)$ is false. Let L_i be any extended regular expression over $\{0,1\}$. Let $\hat{L}_0$ be an e.r.e. s.t. $L(\hat{L}_0) = L_0$. Let $L_{\sigma(i)} = L_i \cdot c \cdot \hat{L}_0$. Then $P(L_{\sigma(i)}) = $ true if and only if $L(L_i) = \phi$.

Let $\mathcal{L}$ be any algorithm for deciding the emptiness problem for extended regular expressions over $\{0,1\}$. Let

$$C_{\mathcal{L}}(n) = \max_{|x|=n} \{\text{tape taken by } \mathcal{L} \text{ to decide whether } L(x)=\phi\}.$$

Let $\mathcal{P}$ be any algorithm for deciding the predicate P. Let

$$C_{\mathcal{P}}(n) = \max_{|x|=n} \{\text{tape taken by } \mathcal{P} \text{ to decide } P(x)\}.$$ Then, $\exists$

a positive k and $\exists$ an algorithm $\mathcal{L}_0$ for deciding the emptiness problem for e.r.e.'s such that for all positive integers μ $C_{\mathcal{L}_0}(n) \leq O(n) + C_{\mathcal{P}}(n+k)$. Clearly, if is elementary recursive, then $_0$ is.

<u>Corollary 4.29</u>: The language $L = \{\beta |\ \beta$ is an e.r.e. and $L(\beta)$ is Star-free$\}$ is not elementary recursive.

<u>Proof 4.29</u>: See the proof of 3.13.

<u>Definition 4.30</u>: A <u>dot-free</u> e.r.e. is an e.r.e. with no occurrence of "·". Let <u>DF</u> be the class of events that can be represented by dot-free e.r.e.'s.

<u>Lemma 4.31</u>: A regular set $R \in DF$ only if $R = R^r$, where $R^r = \{x_1,\ldots,x_n | n \geq 0$ and $x_n ,\ldots, x_1 \in R\}$ (i.e., R^r is the reversal of R).

<u>Proof of 4.31</u>: Any $R \in DF$ over $\Sigma = \{\sigma_1,\ldots,\sigma_n\}$ can be built up recursively from the finite sets ϕ, $\{\epsilon\}$, $\{\sigma_1\}$ $\{\sigma_1\},\ldots, \{\sigma_n\}$ by finitely many applications of $\vee$, $\neg$, and $*$. If $R \in \{\phi, \{\epsilon\}, \{\sigma_1\},\ldots,\{\sigma_n\}\}$, then $R = R^r$.

Let $A = A^r$ and $B = B^r$. If $C = A \cup B, A \cap B, A^*$, or $\neg A$, then $C = C^r$.

$x \in (A^*)^r$ if and only if $x = (x_1 \cdot \ldots \cdot x_n)^r$, where $x_i \in A$. But $(x_1 \cdot \ldots \cdot x_n)^r = x_n{}^r \cdot \ldots \cdot x_1{}^r \in (A^r)^*$. Therefore, $(A^*)^r = (A^r)^*$. $x = x_1 \ldots x_n \in (\neg A)^r$ if and only if $x_n \ldots x_1 \notin A$, if and only if $x_1 \ldots x_n \notin A^r$. Therefore, $(\neg A)^r = \neg (A^r)$.

<u>Corollary 4.32</u>: The set $P' = \{\beta |\ \beta$ is an e.r.e. and $L(\beta) \in DF\}$ is not elementary recursive.

<u>Proof of 4.32</u>: Let L_i be an arbitrary e.r.e. over $\{0,1\}$. Let $L_{\sigma(i)} = L_i \cdot c \cdot d$. Then $L(L_{\sigma(i)}) \in DF$ if and only if $L(L_i) = \phi$. (The remainder of the proof is similiar to that of Theorem 4.28).

In this section we have seen two important phenomena. First, there is a class of languages whose elements are recognizable in deterministic polynomial time only if PTIME = NPTIME = PTAPE. These languages p-hard problems of Cook and Karp. Second, the deterministic time complexity of many problems of the regular sets depends highly upon how the regular sets are enumerated. Several predicates are

1) decidable in deterministic polynomial time if the regular sets are enumerated by enumerating the deterministic finite state automata,

2) are as hard to decide as it is to recognize any context-sensitive language if the regular sets are enumerated by enumerating either the nondeterministic finite automata or the regular expressions,

3) require exponential space if the regular sets are enumerated by enumerating the regular expressions with squaring, and

4) are not elementary recursive if the regular sets are enumerated by enumerating the extended regular expressions.

Section 5: Conclusion

We feel that there are several ideas in this paper that merit repeating.

1) The existence of p-complete and tape-complete problems is an immediate consequence of the fact that there are "padded" universal nondeterministic Turing machines. Such machines can have linear nondeterministic time or tape complexity.

2) There are tape hierarchies corresponding to Cook and Karp's p-complete and p-hard problems. Karp's three p-hard problems are elements of these hierarchies.

3) We can classify several problems about the regular sets in terms of their minimal deterministic time complexity. The complexity of these problems depends strongly upon how the regular sets are enumerated (by deterministic f.s.a., by regular expressions, by extended regular expressions, etc.)

4) Informally, PTIME is closed under arbitrary ϵ-free homomorphisms iff we can, in deterministic polynomial time, simulate arbitrary nondeterministic guesses.

BIBLIOGRAPHY

[1] Book, R.V., "On Languages Accepted In Polynomial Time", Technical Report, 5-72, Harvard University.

[2] Book, R.V. and Greibach, S., "Quasi-realtime Languages", Mathematical Systems Theory 4 (1970), 97-111.

[3] Brzozowski, J.A., "Roots of Star Events", IEEE Conference Record of 1966 Seventh Annual Symposium On Switching and Automata Theory, 88-95.

[4] Cook, S.A., "The Complexity of Theorem-proving Procedures", Proceedings Third ACM Symposium On Theory of Computing (1971), 151-158.

[5] Davis, M., "Computability and Unsolvability", McGraw-Hill, New York, 1958.

[6] Greibach, S., "A Note on Undecidable Properties of Formal Languages", Mathematical Systems Theory 2:1 (1968), 1-6.

[7] Hennie, F.C., "One-tape Off-line Turing Machine Computations", Inf. and Control 8:6 (1965), 553-578.

[8] Hopcroft, J.E. and Ullman, J.D., "Formal Languages and Their Relation to Automata", Addison-Wesley Publishing Co., Reading, Mass., 1969.

[9] Hopcroft, J.E. and Ullman, J.D., "Relations Between Time and Tape Complexities", JACM, Vol. 15, 414-427.

[10] Karp, R.M., "Reducibility Among Combinatorial Problems", Technical Report No. 3, April 1972, Dept. of Computer Science, University of California, Berkeley.

[11] McNaughton, R. and Papert, S., "Counter-Free Automata", MIT Press, Cambridge, Massachusetts, 1971.

[12] Meyer, A.R., "Weak Monadic Second Order Theory of Successor is Not Elementary Recursive - Preliminary Report", Technical Report, MIT.

[13] Meyer, A.R. and Stockmeyer, L.J., "The Equivalence Problem for Regular Expressions With Squaring Requires Exponential Space", 13th Annual Symposium on Switching and Automata Theory (1972), 125-129.

[14] Salomaa, A., "Theory of Automata", Pergamon Press (1969).

[15] Savitch, W., "Relationship Between Nondeterministic and Deterministic Tape Complexities", J. of Comput. and System Sci. 4 (1970), 177-192.

[16] Zalcstein, Y., "Locally Testable Languages", J. of Comput. and System Sci. 6:2 (1972), 151-167.