# UPPER BOUNDS ON THE COMPLEXITY OF SOME PROBLEMS CONCERNING L SYSTEMS

by

Neil D. Jones
and
Sven Skyum

DAIMI PB-69
February 1977

Institute of Mathematics University of Aarhus
DEPARTMENT OF COMPUTER SCIENCE
Ny Munkegade - 8000 Aarhus C - Denmark
Phone 06-12 83 55

# Upper Bounds
# on the complexity of some problems
# concerning L systems

Neil D. Jones *

and

Sven Skyum

Department of Computer Science

University of Aarhus, Aarhus

Denmark

*) On leave from Computer Science Department, University of Kansas, Lawrence, Kansas, USA

Mailing Address:

Department of Computer Science
University of Aarhus
Ny Munkegade
DK-8000 Aarhus C
Denmark

## ABSTRACT

We determine the computational complexity of some decidable problems concerning several types of Lindenmayer systems. The problems are membership, emptiness and finiteness; the L systems are the EDOL, EOL, EDTOL, and ETOL systems. For each problem and type of system we establish upper bounds on the time or memory required for solution by Turing machines. This paper contains algorithms achieving the upper bounds, and a companion paper (PB-70) contains proofs of lower bounds.

# INTRODUCTION AND TERMINOLOGY

Recently, considerable interest has been shown in questions concerning the complexity of the membership problem for various types of L systems. Van Leeuwen showed in [12] that there are ET0L systems G such that L(G) is complete for $\mathcal{NP}$ (nondeterministic polynomial time). Opatrný and Culik showed in [8] that E0L membership (for fixed grammars) may be decided deterministically in time $n^4$, and Sudborough gave a $(\log n)^2$ space algorithm for the same problem in [10], based on a construction by van Leeuwen [13]. Sudborough also gave a deterministic log n space algorithm for ED0L membership in [11], and showed in [10] that some linear languages (and hence some E0L and deterministic ET0L languages) are complete for nondeterministic log space. In [5], we have shown that each deterministic ET0L language can be recognized nondeterministically in log n space, and therefore deterministically in polynomial time. Vitányi showed in [14] that general membership for PD0L systems and the infiniteness for D0L systems can be decided deterministically in polynomial time.

In this paper we establish upper bounds on the complexity of the emptiness and finiteness questions of each of the classes ET0L, E0L, and their deterministic and propagating counterparts, as well as the <u>general</u> membership problem (that is, to determine whether $x \in L(G)$, if given both x <u>and</u> G as data). In each case the upper bound is established by exhibiting an algorithm to solve the problem, and analyzing its time or space requirements. The lower bounds will be established in a successor to this paper.

The problems we discuss may all be represented in terms of membership in the following sets. C denotes any of the system classes just mentioned, and $\overline{G}$ is a description of an L system, specified later.

1. $\text{NONEMPTY}^C = \{\overline{G} \mid G \text{ is in } C \text{ and } L(G) \neq \emptyset\}$
2. $\text{INFINITE}^C = \{\overline{G} \mid G \text{ is in } C \text{ and } L(G) \text{ is infinite}\}$
3. $\text{MEMBER}^C = \{<\overline{G},\overline{x}> \mid G \text{ is in } C \text{ and } x \text{ is in } L(G)\}$
4. $L(G) =$ for a fixed grammar $G$ in $C$

The following table summarizes the results of both papers, and also contains the complexity of the corresponding problems for context-free and context-sensitive grammars for the sake of comparison.

| | PROBLEM | | | | |
|---|---|---|---|---|---|
| GRAMMAR CLASS | MEMBER (FIXED G) | MEMBER (GENERAL) | NONEMPTY | INFINITE | BOUNDS |
| CONTEXT - SENSITIVE | NSPACE(n) | NSPACE (n log n) | UNDECIDABLE | UNDECIDABLE | UPPER |
| | | NSPACE(n) | | | LOWER |
| ET0L, EPT0L | NP | NSPACE (n log n) | NSPACE(n) | NSPACE(n) | UPPER |
| | | NSPACE ($n^{1-\epsilon}$) | NSPACE ($n^{1-\epsilon}$) | NSPACE ($n^{1-\epsilon}$) | LOWER |
| EDT0L, EPDT0L | NL | NSPACE (n log n) | NSPACE(n) | NSPACE(n) | UPPER |
| | | NSPACE ($n^{1-\epsilon}$) | NSPACE ($n^{1-\epsilon}$) | NSPACE ($n^{1-\epsilon}$) | LOWER |
| E0L, EP0L | DSPACE($\log^2 n$) DTIME($n^4$) | NP | DSPACE(n) | NSPACE(n) | UPPER |
| | NL | | NP | NP | LOWER |
| ED0L, EPD0L | L | P | NP | NP | UPPER |
| | L | L | | | LOWER |
| CONTEXT FREE | DSPACE($\log^2 n$) DTIME($n^3$) | P | P | P | UPPER |
| | NL | | | | LOWER |

## Terminology used in the table

We assume the reader is familiar with the basic concepts concerning L systems and time- or tape-bounded Turing machines (e.g. see [2, 3]).

1. In the system class names, D indicates "deterministic", P indicates "propagating" (i.e. the absence of productions with the empty string on the right side), and T indicates "tables".

2. In this paper n will always be the size of the problem under consideration (that is, the number of symbols in $\overline{G}$ or $<\overline{G},\overline{x}>$).

3. DSPACE(S(n)) = $\{$L $|$ L is accepted by some <u>deterministic</u> offline Turing machine which operates within <u>space</u> S(n) on all inputs of length n$\}$

   NSPACE(S(n)) is defined analogously for nondeterministic machines, and DTIME(S(n)), NTIME(S(n)) are defined similarly for the time measure.

4. $\mathcal{L}$ = DSPACE(log n), $\quad n\mathcal{L}$ = NSPACE(log n)

$$\mathcal{P} = \bigcup_{k=1}^{\infty} \mathrm{DTIME}(n^k), \qquad n\mathcal{P} = \bigcup_{k=1}^{\infty} \mathrm{NTIME}(n^k)$$

$$\mathrm{PSPACE} = \bigcup_{k=1}^{\infty} \mathrm{DSPACE}(n^k) = \bigcup_{k=1}^{\infty} \mathrm{NSPACE}(n^k)$$

5. A table entry of the form $\boxed{\frac{U}{L}}$ for problem P indicates that

   a) P is in class U.

   b) If L is $n\mathcal{L}$, $\mathcal{P}$, $n\mathcal{P}$ or NSPACE(n), then some complete problem (and so <u>any</u> problem) in class L is reducible to P.

   c) If L is NSPACE(S(n,$\epsilon$)), then for any $\epsilon > 0$, P is <u>not</u> in NSPACE(S(n,$\epsilon$)).

   d) If L is $\mathcal{L}$, then any algorithm which solves P in DSPACE(S(n)) must satisfy $\sup_{n\to\infty} \frac{S(n)}{\log n} > 0$.

6. A table entry | LU | for problem P indicates that P is complete for class LU (i.e. both a) and b) hold).

Note. In this paper we shall not be concerned with completeness or reducibility.

Note that an upper bound for a problem is automatically an upper bound for a subproblem. Thus for example the upper bound on E0L membership also applies to EP0L and ED0L systems.

Grammar representation

Represent an alphabet $V = \{v_1, v_2, \ldots, v_p\}$ in the form $\overline{V} = [V\overline{1}; V\overline{2}; \ldots; V\overline{p}]$, where $\overline{i}$ is the binary representation of i, $1 \le i \le p$. This can be naturally extended to strings and productions. Finally, we encode an ET0L system $G = (V, \mathcal{P}, w, \Sigma)$ as the string $\overline{G} = [\overline{V}; \overline{\mathcal{P}}; \overline{w}; \Sigma]$ over the alphabet $\{V, 0, 1, [, ;, ], \rightarrow\}$. Note that we always have $p \log p = 0(|\overline{G}|)$.

The works referenced in the introduction, except for [14], establish the complexity of the problem of membership in L(G) for fixed G. The general membership problem can be significantly more complex. The most extreme case is the EDT0L systems – each L(G) may be recognized in log n space, but deciding whether $x \in L(G)$ if both x and G are given requires essentially linear space (both by nondeterministic algorithms).

In general it appears that problems about propagating systems are no more complex than for non-propagating systems, although some upper bound constructions are complicated by the presence of $\lambda$-productions, and lower bound constructions are complicated by their absence.

## DETERMINISTIC E0L SYSTEMS

Let $G = (V, P, w, \Sigma)$ be an ED0L system. Throughout this paper we will let $p = \#V$ denote the number of symbols in the alphabet $V$. We call a symbol $b$ dying if $b \overset{*}{\Rightarrow} \lambda$, and let $V_d$ denote the set of all dying symbols.

In [14] Vitányi gave "feasible" algorithms for the general membership and finiteness problems for D0L systems. His algorithms use the fact that if $w \overset{r}{\Rightarrow} x$ for some $r > p|x|$, all derivation steps after the first $p|x|$ can only use productions of the form $a \to \alpha$ where $\alpha$ contains at most one non-dying symbol. His algorithm yields

Theorem 1 $\quad$ $\text{MEMBER}^{\text{EPD0L}} \in \mathcal{P}$.

The algorithms of [14] do not yield polynomial time algorithms for non-propagating systems, since they involve directly simulating G's derivation for $p(|x| - |w| + 1)$ steps. This derivation can produce intermediate strings whose length is exponential in p if G has many dying letters. Our algorithm for $\text{MEMBER}^{\text{ED0L}}$ involves a more efficient way to simulate short derivations, and the construction of an auxiliary propagating system.

Lemma 2 $\quad$ Let G be an ED0L system and $x \in \Sigma^*$.

The relation "$\alpha \overset{*}{\Rightarrow} x$ in k or fewer steps" can be decided in time bounded by a polynomial function of $|\langle\bar{G}, \bar{x}\rangle|$ and k.

Proof $\quad$ It is sufficient to show that the following functions a(i) (where $0 \le i \le k$ and $a \in V$) can be computed in polynomial time:

$$a(i) = \begin{cases} \beta & \text{if } a \overset{i}{\Rightarrow} \beta \text{ and } \beta \text{ is a subword of } x \\ \# & \text{otherwise} \end{cases}$$

Let $a \in V$ and $0 \leq i \leq k$, and let the unique a-production in P be $a \to b_1 b_2 \ldots b_r$. It is immediate that

$$a(i) = \begin{cases} a & \text{if } i = 0 \text{ and } a \text{ is a subword of } x; \\ b_1(i-1)b_2(i-1)\ldots b_r(i-1) & \text{if } i \neq 0 \text{ and } b_1(i-1)\ldots b_r(i-1) \text{ is} \\ & \text{a subword of } x; \\ \# & \text{otherwise} \end{cases}$$

Thus the a(i)'s may be computed in the order $i = 0, 1, \ldots, k$; the time bound is immediate, since only subwords of x are stored. A similar technique was used in [5]. □

Define the homomorphism $h : V^* \to V^*$ by

$$h(a) = \begin{cases} \lambda & \text{if } a \in V_d \\ a & \text{otherwise} \end{cases}$$

From G we may construct the propagating system $H = (V, P', h(w), \Sigma)$, where

$$P' = \{a \to h(\alpha) \mid a \to \alpha \text{ is in P and } a \notin V_d\}$$

<u>Lemma 3</u> For any $k \geq 0$, $\alpha, \beta \in V^*$

1. $\alpha \overset{k}{\underset{G}{\Rightarrow}} \beta$ implies $h(\alpha) \overset{k}{\underset{H}{\Rightarrow}} h(\beta)$
2. $h(\alpha) \overset{k}{\underset{H}{\Rightarrow}} \gamma$ implies $\alpha \overset{k}{\underset{G}{\Rightarrow}} \beta$ for some $\beta$ such that $h(\beta) = \gamma$
3. $\alpha \overset{p}{\underset{G}{\Rightarrow}} \beta$ if and only if $h(\alpha) \overset{p}{\underset{G}{\Rightarrow}} \beta$.

<u>Proof</u> Parts 1 and 2 are immediate from the fact that only dying letters may be derived from a dying letter. For 3, any letter a in y which is

not in h(y) must be dying; and $a \overset{*}{\Rightarrow} \lambda$ implies $a \overset{p}{\Rightarrow} \lambda$. □

Theorem 4 $\quad MEMBER^{ED0L} \in \mathcal{P}$.

Proof Consider the following algorithm:

1. Accept $<\bar{G}, \bar{x}>$ if $w \underset{G}{\overset{k}{\Rightarrow}} x$ for some $k \leq p|x| + p$.
2. Construct H as above.
3. Determine whether $h(w) \underset{H}{\Rightarrow^*} h(x)$; reject if not.
4. Find z such that $h(w) \underset{H}{\overset{r}{\Rightarrow}} z \underset{H}{\overset{p}{\Rightarrow}} h(x)$ and $r > p|x|$.
5. Accept if and only if $z \underset{G}{\overset{p}{\Rightarrow}} x$.

Correctness of the algorithm follows from Lemma 3.
Note in step 4 that z is uniquely determined by h(x) because $r > p|x|$. We have $|z| = |h(x)|$ and both z and h(x) consist of monorecursive symbols only (a is monorecursive iff $a \Rightarrow^+ a$, see [14]). For monorecursive symbols we have that $a \overset{k}{\Rightarrow} b$ and $c \overset{k}{\Rightarrow} b$ implies that $a = c$. z can therefore be found from h(x) in polynomial time.
Steps 1 and 5 can be done in polynomial time by Lemma 1, and step 3 is polynomially bounded by Theorem 1. □

For any word $w \in V^*$, we define Alph(w) to be the smallest $A \subseteq V$ such that $w \in A^*$.

Theorem 5 $\quad NONEMPTY^{ED0L} \in \mathcal{NP}$.

Proof Let $G = (V, P, w, \Sigma)$ be an ED0L system. Construct a non-deterministic finite automaton $M = (V, \{0\}, \delta, S_0, V-\Sigma)$ where $S_0 = Alph(w)$

and $\delta(a,0) = \{a_1, a_2, \ldots, a_m\}$ just in case $a \rightarrow a_1 a_2 \ldots a_m$ is a production in P. It is easily seen that $L(G) \neq \emptyset$ iff $L(M) \neq 0^*$. By Stockmeyer and Meyer [9], this test can be carried out nondeterministically in polynomial time. Since the construction of M can easily be done in polynomial time, the theorem follows. □

<u>Theorem 6</u> $\quad \text{INFINITE}^{\text{EDOL}} \in \mathcal{NP}$.

<u>Proof</u> Let $G = (V, P, w, \Sigma)$ be an EDOL system, and let $w = w_0 \Rightarrow w_1 \Rightarrow \ldots$ be its derivation. Consider the sequence $\text{Alph}(w_0), \text{Alph}(w_1), \ldots$ . Now L(G) will be infinite if and only if $|w_0|, |w_1|, |w_2|, \ldots$ grows infinitely and there is a j such that $2^p < j \leq 2^{p+1}$ and $\text{Alph}(w_j) \subseteq \Sigma$.

In [14] it is shown that we can test the infinity of $|w_1|, |w_2|, \ldots$ in polynomial time.

To check the j condition, form a connection matrix M, where for each $a, b \in V$:

$$M(a,b) = \begin{cases} 1 & \text{if } a \rightarrow xby \text{ is in P for some } x, y \in V^* \\ 0 & \text{otherwise} \end{cases}$$

Then $M^i(a,b)$ will be 1 if $a \overset{i}{\Rightarrow} xby$ for some $x, y \in V^*$ (using and-or matrix multiplication). We can calculate $M^1, M^2, M^4, \ldots, M^{2^{p+1}}$ in polynomial time by repeated squaring. We may now nondeterministically guess j between $2^p$ and $2^{p+1}$, and obtain $M^j$ by multiplying the elements of $M^1, M^2, \ldots$ corresponding to the ones in the binary representation of j. The condition that $\text{Alph}(w_j) \subseteq \Sigma$ is easily determined from $M^j$. □

In the following paper we shall see that $\mathrm{NONEMPTY}^{\mathrm{ED0L}}$ and $\mathrm{INFINITE}^{\mathrm{ED0L}}$ are $\mathcal{NP}$-complete, so this is a best possible upper bound. Also, note that $\mathrm{INFINITE}^{\mathrm{PD0L}}$ is in $\mathcal{P}$ by [14], so one effect of terminal letters is to increase complexity.

## E0L SYSTEMS

<u>Theorem 7</u> $\text{MEMBER}^{E0L} \in \mathcal{NP}$.

<u>Proof</u> Let $G = (V, P, w, \Sigma)$ be an E0L system and let $D: y_1a_1y_2a_2 \dots a_ky_{k+1} \overset{*}{\Rightarrow} x_1 \dots x_k$ be a derivation. We say that $a_1a_2 \dots a_k$ is the <u>productive part</u> of $y_1a_1 \dots a_ky_{k+1}$ in D if $a_i \overset{*}{\Rightarrow} x_i \neq \lambda$ and $y_i \overset{*}{\Rightarrow} \lambda$ are subderivations of D, for $1 \leq i \leq k+1$. Note that if $\alpha \Rightarrow \beta \overset{*}{\Rightarrow} x$, the productive part of $\alpha$ is no longer than the productive part of $\beta$ in $\beta \overset{*}{\Rightarrow} x$. Thus any derivation $D: w \overset{r}{\Rightarrow} x$ with $r \geq p$ can be decomposed into

$$D: w = u_1 \overset{*}{\Rightarrow} v_1 \Rightarrow u_2 \overset{*}{\Rightarrow} v_2 \Rightarrow \dots \Rightarrow u_m \overset{*}{\Rightarrow} v_m \overset{p}{\Rightarrow} x$$

where $|\alpha_1| = |\beta_1| < |\alpha_2| = |\beta_2| < \dots < |\alpha_m| = |\beta_m|$, and $\alpha_i, \beta_i$ are the productive parts of $u_i, v_i$ respectively for $1 \leq i \leq m$. Further, $\alpha_i \overset{*}{\Rightarrow} x$ by the same productions as those in D, for each i. We will write $\gamma < \delta$ if $\gamma = a_1a_2 \dots a_q$ and $\delta = y_1a_1y_2a_2 \dots a_qy_{q+1}$ where $y_i \in V_d^*$ for $1 \leq i \leq q+1$. Clearly $\alpha_i < u_i$ and $\beta_i < v_i$.

Conversely, suppose we are given a sequence $\alpha_1, \beta_1, \alpha_2, \beta_2, \dots, \alpha_m, \beta_m \overset{p}{\Rightarrow} x$ such that $\alpha_1 < w$ and for each i, $1 \leq i \leq m$ we have $|\alpha_i| = |\beta_i|$, $\alpha_i \overset{*}{\Rightarrow} \delta_i$ for some $\delta_i$ with $\beta_i < \delta_i$, and $\beta_i \Rightarrow \gamma_{i+1}$ for some $\gamma_{i+1}$ with $\alpha_{i+1} < \gamma_{i+1}$. We can extend the derivation $\alpha_1 \overset{*}{\Rightarrow} \delta_1$ to get $w = u_1 \overset{*}{\Rightarrow} v_1$ where $\beta_1 < \delta_1 < v_1$, so $\beta_1 < v_1$. Similarly $\beta_1 \Rightarrow \gamma_2$ can be extended to $v_1 \Rightarrow u_2$ where $\alpha_2 < \gamma_2 < u_2$, so $\alpha_2 < u_2$. We may continue to yield $v_2, \dots, u_m, v_m$ so $w = y_1 \overset{*}{\Rightarrow} v_1 \Rightarrow u_2 \overset{*}{\Rightarrow} \dots \dots \overset{*}{\Rightarrow} v_m$ with $\beta_m < v_m$. Now $a \overset{*}{\Rightarrow} \lambda$ implies $a \overset{p}{\Rightarrow} \lambda$; thus $\beta_m \overset{p}{\Rightarrow} x$ implies $v_m \overset{p}{\Rightarrow} x$, so $w \overset{*}{\Rightarrow} x$ and $x \in L(G)$.

Following is our algorithm based on these remarks. Note that $m \leq |x|$.

<u>if</u> $w \overset{q}{\Rightarrow} x$ for some $q \leq p$ <u>then</u> accept;

choose $\alpha < w$;

<u>for</u> i := 1 <u>to</u> m <u>do</u>

<u>begin</u> choose $\beta$ so $|\alpha| = |\beta|$ and $\alpha \overset{*}{\Rightarrow} \delta$ for some $\delta$ with $\beta < \delta$;

<u>if</u> $\beta \overset{p}{\Rightarrow} x$ <u>then</u> accept;

choose $\alpha$ so $\beta \Rightarrow \gamma$ for some $\gamma$ with $\alpha < \gamma$

<u>end</u>

This procedure will give a nondeterministic polynomial time membership algorithm provided (1) determination of $\alpha \overset{q}{\Rightarrow} \beta$ for $q \leq p$ and (2) the step "choose $\beta$ ...." can be done in polynomial time. If $\alpha = a_1 a_2 \ldots a_k$ and $\beta = b_1 b_2 \ldots b_k$ then $\alpha \overset{u}{\Rightarrow} \delta$ for some $\beta < \delta$ if and only if $a_i \overset{u}{\Rightarrow} \delta_i$ for some $b_i < \delta_i$, $1 \leq i \leq k$. If we form the connection matrix M so that $M(a,b) = 1$ just in case there is a production $a \to y$ with $b < y$, we see that $\alpha \overset{u}{\Rightarrow} \delta$, $\beta < \delta$ just in case $M^u(a_1, b_1) = \ldots = M^u(a_k, b_k) = 1$. Further if u exists then it can be chosen less than $p^k = 2^{k \cdot \log p}$. Thus $M^u$ may be found (if it exists), as in the proof of $\text{INFINITE}^{\text{ED0L}} \in \mathcal{NP}$, by a nondeterministic polynomial time algorithm. So (2) can be tested in polynomial time.

$\alpha \overset{q}{\Rightarrow} \beta$ for some $q \leq p$ can be determined in polynomial time by guessing $\alpha_0, \alpha_1, \ldots, \alpha_q \in V^*$ such that $|\alpha_i| \leq |\beta|$, $\alpha = \alpha_0$, $\alpha_q = \beta$, and for $1 \leq i < q$ there is a word $\delta_i$ where $\alpha_i \Rightarrow \delta_i$ and $\alpha_{i+1} < \delta_i$. □

<u>Theorem 8</u> $\text{NONEMPTY}^{\text{E0L}} \in \text{DSPACE}(n)$.

<u>Proof</u> Let $G = (V, P, w, \Sigma)$ be an E0L system. Define $A_0 = \Sigma, A_{i+1} = \{a \mid a \to \alpha \text{ is a production in } P \text{ and } \alpha \in A_i^*\}$.

If $\beta \in A_i^*$ and $\beta \Rightarrow \delta$ then $\delta \in A_{i-1}^*$, for $i > 1$. Therefore $\beta \overset{i}{\Rightarrow} x$ for some $x \in \Sigma^*$ iff $\beta \in A_i^*$. Since $A_i \subseteq V$ there are no more than $2^n$ such sets. Thus $L(G) \neq \emptyset$ iff $w \in A_i^*$ for some $i \leq 2^n$.

The DSPACE(n) algorithm will just calculate $A_0, A_1, \ldots,$ storing only the most recent one (as a bit vector) and comparing the letters in w against it. We can stop if $i > 2^n$; and i can be stored within n bits. □

$INFINITE^{E0L}$ will be shown to be in NSPACE(n) in the next section.

## SYSTEMS WITH TABLES

Theorem 9 $\quad \text{MEMBER}^{\text{ET0L}} \in \text{NSPACE}(n \log n)$.

Proof $\quad$ Let $G = (V, \{P_1, P_2, \ldots, P_k\}, w, \Sigma)$ be an ET0L system. If G is propagating, we can simply simulate a derivation from w stepwise, accepting if x is found and halting if a word longer than x is obtained. This uses space n log n, due to the fact that $0(\log |\overline{G}|)$ bits are required to store one symbol of V.

If G has $\lambda$-productions, we simulate the derivation, nondeterministically guessing at each step of derivations which symbols will yield $\lambda$, keeping track of our guesses by marks on $\overline{G}$. Thus the state of a derivation can be represented as a pair $(\alpha, A)$ where $\alpha$ is the productive part of the word and A is the set of letters guessed to yield $\lambda$.

Define $\alpha \overset{A}{<} \beta$ to be true if $\alpha = a_1 a_2 \ldots a_q$ and $\beta = y_1 a_1 y_2 \ldots a_q y_{q+1}$ where $y_i \in A^*$ for $1 \leq i \leq q+1$. We write $(\alpha, A) \Rightarrow (\beta, B)$ just in case $|\beta| \leq |x|$ and there is a table $P_i$ and strings $\gamma, \delta \in V^*$ such that $\alpha \underset{P_i}{\Rightarrow} \gamma$, $\beta \overset{B}{<} \gamma$, and $\text{Alph}(\delta) \subseteq B$, where $A = \{a_1, a_2, \ldots, a_m\}$ and $a_1 a_2 \ldots a_m \underset{P_i}{\Rightarrow} \delta$. Let $\overset{*}{\Rightarrow}$ be the transitive, reflexive closure of $\Rightarrow$.

It is easily seen that $w \overset{*}{\Rightarrow} x$ if and only if there is an $\alpha, A$ such that $\alpha \overset{A}{<} w$ and $(\alpha, A) \overset{*}{\Rightarrow} (x, \emptyset)$. Construction of an NSPACE(n log n) algorithm based on this is straightforward. □

Corollary 10 $\quad \text{MEMBER}^{\text{EDT0L}} \in \text{NSPACE}(n \log n)$.

Theorem 11 $\quad \text{NONEMPTY}^{\text{ET0L}} \in \text{NSPACE}(n)$.

Let $G = (V, \{P_1, P_2, \ldots, P_k\}, w, \Sigma)$ be an ET0L system. If we change the algorithm for $\text{NONEMPTY}^{\text{E0L}}$ to an algorithm working in the same way except that we now calculate $A_{i+1}$ from $A_i$ nondeterministically by choosing a table $P_j$ and then define $A_{i+1}$ as $\{a \mid a \to \alpha$ is a production in $P_j$ and $\alpha \in A_i^*\}$ we obtain an algorithm for solving NONEMPTY$^{ET0L}$ in NSPACE(n).

<u>Corollary 12</u> NONEMPTY$^{EDT0L}$ $\in$ NSPACE(n).

<u>Theorem 13</u> INFINITE$^{EDT0L}$ $\in$ NSPACE(n).

<u>Proof</u> Let $G = (V, \mathcal{P}, w, \Sigma)$ be an EDT0L system, and let m be the length of the longest right hand side of any production in $\mathcal{P}$. Suppose G has a derivation

$$D : w = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_q = x$$

with $|x| \geq m^{2^{2p}} \cdot |w|$. Then $q > 2^{2p}$, so there must be $i < j$ such that (1) $|z_i| < |z_j|$, (2) $\text{Alph}(z_i) = \text{Alph}(z_j)$, (3) $\text{Alph}(w_i) = \text{Alph}(w_j)$, and (4) $\text{Alph}(w_q) = \text{Alph}(z_q) \subseteq \Sigma$ where $z_i$ is the productive part of $w_i$ (as defined in the proof of MEMBER$^{E0L}$ $\in \mathcal{NP}$).

Let $\tau_1, \tau, \tau_2$ be the table sequences such that $w \underset{\tau_1}{\Rightarrow} w_i \underset{\tau}{\Rightarrow} w_j \underset{\tau_2}{\Rightarrow} x$. Conditions (3) and (4) imply that the table sequences $\tau_1 \tau^k \tau_2 (k = 0, 1, 2, \ldots)$ will yield strings in $\Sigma^*$, and conditions (1) and (2) imply that $\tau_1 \tau^k \tau_2$ will yield longer and longer strings as k grows, so L(G) must be infinite.

Conversely if L(G) is infinite, arbitrarily long words may be derived, so there must be a derivation satisfying $q \leq 2 \cdot 2^{2p}$, and (1) through (4) as well.

The algorithm proceeds by simulating a derivation D as above, storing only enough information to determine whether (1) through (4) are true. D can be simulated closely enough to recognize (2), (3) and (4) by storing only $\text{Alph}(w_k)$ and $\text{Alph}(z_k)$ at the k-1st and k'th simulation step, as well as

Alph($w_i$) and Alph($z_i$). At the k'th step we nondeterministically select a table $P \in \mathcal{P}$ to apply; it is easy to build Alph($w_k$) from Alph($w_{k-1}$) and P. We then nondeterministically select a set Alph($z_k$); such a set is legal just in case every symbol b in Alph($z_{k-1}$) has at least one occurrence of a letter from Alph($z_k$) in the right hand side of the b-production in P.

To verify condition (1) note that $|z_0| \leq |z_1| \leq \ldots$, so $|z_i| < |z_j|$ if and only if some symbol a in $z_i$ derives two or more symbols in $z_j$ in $z_i \underset{T}{\Rightarrow} z_j$. This condition may be recognized during the simulation by remembering whether any a in some $z_{k-1}$ $(i < k \leq j)$ has two (or more) descendants in $z_k$. This is easy to check when Alph($z_k$) is constructed from Alph($z_{k-1}$). The storage required for this is $0(p)$ bits and so no greater than $|\overline{G}|$. □

<u>Theorem 14</u> $\text{INFINITE}^{\text{ET0L}} \in \text{NSPACE}(n)$.

Let $G = (V, \mathcal{P}, w, \Sigma)$ be an ET0L system. Let $G_c = (V, \mathcal{P}', w, \Sigma)$ be the EDT0L system where $\mathcal{P}'$ is the set of all deterministic tables P with the property that $\{a \to \alpha \text{ in } P\} \subseteq \{a \to \alpha \text{ in } T\}$ for some T in $\mathcal{P}$. $G_c$ is often called the combinatorial complete version of G. Now L(G) is infinite iff L($G_c$) is infinite. Infiniteness of L($G_c$) can be decided in NSPACE(n) by the method above. Since we do not have to store $G_c$ explicitly but just make sure that we use the same production for a symbol in one step of the derivation, we can test infiniteness of any ET0L system in NSPACE(n) as well.

<u>Corollary 15</u> $\text{INFINITE}^{\text{E0L}} \in \text{NSPACE}(n)$,

## REFERENCES

[1] Alt, H. Mehlhorn, K. Lower bounds for the space complexity of context-free recognition, Automata, Languages and Programming, 338-354, July 1976, University Press, Edinburgh, Scotland.

[2] Herman, G.T., Rozenberg, G. Developmental Systems and Languages, North-Holland/American Elsevier, 1975.

[3] Hopcroft, J.E., Ullman, J.D. Formal Languages and their Relation to Automata, Addison-Wesley, 1969, 242 pp.

[4] Jones, N.D., Laaser, W.T. Complete problems for deterministic polynomial time, J. Theoretical Computer Science, 1976.

[5] Jones, N.D., Skyum, S. Recognition of deterministic E0L languages in logarithmic space, to appear in Information and Control, 1977.

[6] Lewis, P.M., Stearns, R.E., Hartmanis, J. Memory bounds for the recognition of context-free and context-sensitive languages, IEEE Conference Record on Switching Circuit Theory and Logical Design, Ann Arbor, Michigan, pp. 191-202.

[7] Meyer, A.R., Stockmeyer, L.J. The equivalence problem for regular expressions with squaring requires exponential space, 13th IEEE Symposium on Switching and Automata Theory, Oct. 1972, pp. 125-129.

[8] Opatrný, J., Culik, K. II, Time complexity of L languages, Abstracts of papers, Conference of formal languages, automata and development, University of Utrecht, Netherlands, 1975.

[9] Stockmeyer, L.J., Meyer, A.R. Word problems requiring exponential time, 5th ACM Symposium on Theory of Computing, May 1973, pp. 1-9.

[10] Sudborough, I.H. A note on tape-bounded complexity classes and linear context-free languages, J. ACM 22, 1975, pp. 499-500.

[11] Sudborough, I.H. The Complexity of the membership problem for some extensions of context-free languages, Technical Report, Northwestern University Computer Science Dept., Evanston, Ill., 1976.

[12] van Leeuwen, J. The Membership question for ET0L languages is polynomially complete, Information Processing Letters 3, 1975, pp. 138-143.

[13] van Leeuwen, J. The Tape complexity of context-independent developmental languages, J. Computer and Systems Sciences 15, 1975, pp. 203-211.

[14] Vitányi, P.M.B. On the size of D0L languages, in Rozenberg, G. and Salomaa, A. (ed.) L systems, Springer Verlag 1974.

[15] Younger, D.H. Recognition and parsing of context-free languages in time $n^3$, Information and Control 10:2, pp. 189-208.