ON LOG-TAPE ISOMORPHISMS

OF COMPLETE SETS

J. Hartmanis

TR 77-318

Department of Computer Science
Cornell University
Ithaca, NY 14853

ON LOG-TAPE ISOMORPHISMS OF COMPLETE SETS

J. Hartmanis

Department of Computer Science
Cornell University
Ithaca, N.Y. 14853

ABSTRACT

In this paper we study logn-tape computable reductions between sets and investigate conditions under which logn-tape reductions between sets can be extended to logn-tape computable isomorphisms of these sets. As an application of these results we obtain easy to check necessary and sufficient conditions that sets complete under logn-tape reductions in NL, CSL, P, NP, PTAPE, etc. are logn-tape isomorphic to the previously known complete sets in the respective classes. As a matter of fact, all the "known" complete sets for NL, CSL, P, NP, PTAPE, etc. are now easily seen to be, respectively, logn-tape isomorphic. These results strengthen and extend substantially the previously known results about polynomial time computable reductions and isomorphisms of NP and PTAPE complete sets. Furthermore, we show that any set complete in CSL, PTAPE, etc. must be dense and therefore, for example, cannot be over a single letter alphabet.

1. INTRODUCTION

The research in computational complexity and in particular the study of feasible computations has been strongly influenced and furthered by the investigation of efficient reductions between different problems and problem classes. The investigation of such classes of languages (or problems) as P, NP and PTAPE, and the discovery that the classes NP and PTAPE contain "natural" complete problems to which any other problem in the class can be efficiently reduced, has revealed deep and unsuspected relations between different problems and has indicated interesting structural relations between these and other classes of problems [1,3,4,8].

Originally, the completeness of sets in NP was defined in terms of polynomial time computations with oracle sets in NP, corresponding to Turing reducibility in recursive function theory [3]. Soon after that it was realized that these reductions could be performed by a polynomial time bounded Turing machine, corresponding to many-one reduction in recursive function theory [8]. Since then relations between these reductions have been investigated and a large number of new complete problems under polynomial time reductions have been discovered for NP and PTAPE and other classes [1,2,3,4,8,9].

On the other hand, it turned out that all previously studied reductions between problems in NP, PTAPE and such classes as EXPTIME and EXPTAPE, respectively, can all be performed by logn-tape bounded Turing machines [5,9]. It is easily seen that the logn-tape computable reductions are contained among the polynomial time reductions but it is not yet known whether this containment is proper (which we conjecture to be the case). Nevertheless, since the original work in NP and PTAPE complete problems under polynomial time reductions, it has been shown that there exist further, more easily computable classes of problems, such as NL, the class of languages acceptable in non-deterministic logn-tape, and P, which have natural complete problems under logn-tape reductions [5,6,7].

Similarly, as our results will show, logn-tape reductions seem to be appropriate for the study of languages complete for the family of context-sensitive languages since the previously studied "hardest" context-sensitive languages are complete under logn-tape reductions [2,4,9].

In this paper we study logn-tape reductions between sets and derive conditions when logn-tape reductions between sets can be extended to logn-tape computable isomorphisms of these sets. As an application of these results we get very easily verifiable necessary and sufficient conditions that a new complete set is logn-tape isomorphic to the classic complete sets of this family of languages. For example, let SAT denote the set of all satisfiable Boolean formulas in conjunctive normal form, which is known to be a complete set for NP [3,8]. Then our result states,

<u>Theorem</u>: Let B be a complete set (under logn-tape reductions) in NP. Then B is logn-tape isomorphic to SAT if and only if there exist two logn-tape computable functions D(,) S( ) such that

1. $(\forall x,y)[D(x,y) \in B \Longleftrightarrow x \in B]$

2. $(\forall x,y)[S \circ D(x,y) = y]$.

Similar results hold for complete sets for NL, CSL, P, NP, PTAPE, EXPTIME and EXPTAPE. As a matter of fact, these tests are so easy to apply that we have applied them to the best known (published) complete sets for different families of languages and verified that they all are, respectively, logn-tape isomorphic. Thus showing that complete sets of the well known families of languages are, respectively, very similar in a strong technical sense and that these results extend to such classes as NL and P and the family of context-sensitive languages, for which polynomial time reductions are not appropriate since P contains the classes NL and P, and it is not known whether any language polynomial time reducable to a csl is itself a csl.

It should be pointed out that if all complete sets in P (NP) are L-isomorphic then it would follow that P (NP) properly contains the family of logn-tape acceptable languages, which still is an open problem. Similarly, if all complete sets for non-deterministic logn-tape acceptable languages are logn-tape isomorphic then the deterministic and non-deterministic logn-tape acceptable languages are different, which would solve a classic open problem.

The previous theorem resembles very strongly the corresponding isomorphism result for polynomial time reductions of complete sets and the proof techniques are quite similar. At the same time, the intermediate technical results leading to the above theorem are different. For the logn-tape reducible sets we establish the following analogue to the Cantor-Bernstein-Myhill theorem, from which we derive the above and other applications. For the polynomial time analogue see [2].

<u>Theorem:</u> Let f_1 and f_2 be reductions A to B and B to A, respectively, such that

1. f_1 and f_2 are one-to-one maps,
2. f_1, f_2, f_1^{-1}, f_2^{-1} are logn-tape computable,
3. $(\forall x)[|f_i(x)| > |x|^2]$ for $i = 1,2$.

Then A and B are isomorphic under a logn-tape mapping.

It is interesting to note that because of an intermediate technical result the length squaring condition (3), required in our proof of the above theorem (the corresponding result for polynomial time maps required only that the maps are length increasing [2]), does not appear in the first result giving necessary and sufficient conditions that an NP complete set is logn-tape isomorphic to SAT.

Furthermore, we show that sets complete for CSL, PTAPE and classes above PTAPE cannot have sparse complete sets under logn-tape reductions. This extends and strengthens further the previously known results for polynomial time reductions [2].

2. <u>PRELIMINARIES AND ISOMORPHISM THEOREM</u>

A <u>logn-tape</u> transducer or an <u>L-transducer</u> is a three tape Turing machine (Tm) with a two-way read only input tape, a one-way output tape and, for input w, a $\log|w|$ long two-way, read-write work tape.

We say that a function is <u>logn-tape</u> or <u>L-computable</u> if there exists a logn-tape transducer which computes the function. A set A, $A \subseteq \Sigma^*$, is <u>logn-tape</u> reducible or <u>L-reducible</u> to the set B, $B \subseteq \Gamma^*$, if there exists an L-transducer M_i such that

$$x \in A \iff M_i(x) \in B.$$

We denote this by

$$A \leq_L B.$$

Two sets A, $A \subseteq \Sigma^*$, and B, $B \subseteq \Gamma^*$, are log-tape isomorphic or L-isomorphic iff there exists an L-reduction p of A to B such that

a) p is a bijection,

b) p^{-1} is a log-tape transduction.

Thus we see that p is a one-to-one onto L-reduction of A to B and p^{-1} is a corresponding L-reduction of B to A.

We denote the set accepted by the Tm M_i by $T(M_i)$ and the maximal running time and tape used for inputs of length n by $T_i(n)$ and $L_i(n)$, respectively.

Let

$L = \{T(M_i) \mid L_i(n) \leq \log n\}$,

$NL = \{T(M_i) \mid M_i$ non-deterministic and $L_i(n) \leq \log n\}$,

$P = \{T(M_i) \mid T_i(n) \leq n^k,\ k = 1,2,\ldots\}$,

$NP = \{T(M_i) \mid M_i$ non-det. and $T_i(n) \leq n^k,\ k = 1,2,\ldots\}$,

$PTAPE = \{T(M_i) \mid L_i(n) \leq n^k,\ k = 1,2,\ldots\}$.

$CSL = \{T(M_i) \mid M_i$ non-det. and $L_i(n) \leq n\}$.

A set B is logn-tape complete or L-complete for a family of languages F iff

a) $B \in F$

b) $(\forall A \in F)[A \leq_L B]$.

We now start our investigation of L-isomorphisms with the goal of developing very simple necessary and sufficient conditions that a new complete set in one of the previously listed families of languages is L-isomorphic to the previously known complete sets in that family. Thus showing that in a very strong technical sense all the "known" complete sets in these families of languages are indeed very similar. Recall that, showing that all the complete sets in NL or P are, respectively, L-isomorphic, would have some very profound consequences.

<u>Theorem 1</u>: Let p and q be L-reductions of A to B and B to A, respectively, such that

a) p and q are one-to-one,

b) p^{-1} and q^{-1} are L-computable,

c) $|p(x)| > |x|^2$ and $|q(x)| > |x|^2$.

Then A and B are L-isomorphic.

<u>Proof</u>: From p and q we will construct an isomorphism ϕ between A and B and show that ϕ and ϕ^{-1} are L-computable.

Let

$$R_1 = \{(q \circ p)^k x \mid k \geq 0 \text{ and } x \notin q(\Gamma^*)\}$$
$$R_2 = \{q \circ (p \circ q)^k x \mid k \geq 0 \text{ and } x \notin p(\Sigma^*)\}$$
$$S_1 = \{(p \circ q)^k x \mid k \geq 0 \text{ and } x \notin p(\Sigma^*)\}$$
$$S_2 = \{p \circ (q \circ p)^k x \mid k \geq 0 \text{ and } x \notin q(\Gamma^*)\}.$$

It is easily seen that

$$R_1 \cap R_2 = S_1 \cap S_2 = \phi, \text{ and } \Sigma^* = R_1 \cup R_2,\ \Gamma^* = S_1 \cup S_2.$$

Note that if $p^{-1}(x)$ or $q^{-1}(x)$ are undefined then they output a special symbol *. This can be done since we can detect cycling on logn tape.

Define

$$\phi(z) = \text{if } z \in R_1 \text{ then } p(z) \text{ else } q^{-1}(z)$$

$$\phi^{-1}(z) = \text{if } z \in S_2 \text{ then } p^{-1}(z) \text{ else } q(z).$$

We first show that ϕ and ϕ^{-1} have the desired properties and then show that ϕ and ϕ^{-1} are L-computable. Since

$$R_1 \cap R_2 = \phi \qquad \text{and} \qquad R_1 \cup R_2 = \Sigma^*$$

the function ϕ is well defined. Similarly $\phi^{-1}(z)$ is well defined. It is seen that

$$\phi(R_1) = S_2,\ \phi(R_2) = S_1,\ \phi^{-1}(S_2) = R_1,\ \phi^{-1}(S_1) = R_2,$$

and that ϕ is a one-to-one onto mapping and that ϕ and ϕ^{-1} are indeed inverses.

To compute $\phi(z)$ we have to determine whether z is in R_1 or R_2. This can be done by computing the minimal $k \geq 0$ such that

$$(q^{-1} \circ p^{-1} \circ)^k q^{-1}(x) = * \quad \text{or} \quad p^{-1} \circ (q^{-1} \circ p^{-1} \circ)^k q^{-1}(x) = *.$$

In the first case

$$\phi(x) = p(x),$$

in the second case

$$\phi(x) = q^{-1}(x).$$

Since p and q are such that

$$|p(x)| > |x|^2 \text{ and } |q(x)| > |x|^2,$$

we see that

$$|p^{-1}(x)| < |x|^{1/2} \text{ and } |q^{-1}(x)| < |x|^{1/2}.$$

Since

$$\log|x|^{1/2} = 1/2 \log|x|$$

we see that the tape requirements for computations of successive inverses are cut in half. This fact can be exploited to compute $\phi(z)$ and $\phi^{-1}(z)$ on logn-tape. The only difficulty is that during the computation there may be intermediate values for

$$q^{-1}(z),\ p^{-1} \circ q^{-1}(z),\ q^{-1} \circ p^{-1} \circ q^{-1}(z), \text{ etc.}$$

which are too large to be written down on the available working tape. This difficulty can be avoided by never writing these values down but recomputing them (recursively) when they are required in the subsequent computations.

We now outline how this computation is performed on Tm M_ϕ. We can assume that the Tm's computing p and q use exactly $\log|w|$ tape for input w and let M_ϕ have effectively $8 \log|w|$ tape available for input w (this can be achieved by choosing a sufficiently large tape alphabet). Let the tape of M_ϕ be divided in $\log|w|$ segments of length

$$4 \log|x|,\ 2 \log|w|,\ \log|w|,\ 1/2 \log|w|, \text{ etc.}$$

These segments will be used to compute, respectively,

$$q^{-1}(w),\ p^{-1} \circ q^{-1}(w),\ q^{-1} \circ p^{-1} \circ q^{-1}(w) \text{ etc.},$$

as far as needed to determine whether

$$\phi(w) = p(w) \text{ or } \phi(w) = q^{-1}(w).$$

We will refer to the computations of

$$q^{-1}(w),\ p^{-1} \circ q^{-1}(w),\ldots,$$

respectively, as the first stage, second stage, etc. On each tape segment for each stage of the computation will be recorded the state of the Tm (which computes p^{-1} or q^{-1} of the output of the previous stage), the contents of the work tape of this Tm and which tape square it is scanning, furthermore it will be recorded which tape symbol it is scanning on the (simulated) input tape and how many output symbols it has received from the previous stage.

The computation starts by the first stage computing the first output symbol of $q^{-1}(w)$.

If during the computation in a given stage an output symbol is computed the computation in this stage stops and the output symbol is passed to the next stage which becomes active. If this output symbol permits the next stage to continue its computation, it computes until it produces an output or needs a new input. In the first case the next stage is activated and the output is passed on, in the second case, the previous stage is activated and an output is requested. In this manner all stages are cycling through computing (many times over) their output which is passed on to the next stage for it to find the appropriate input symbol it is waiting for. At some point one of the computations must yield a * from which M_ϕ can determine whether to compute $p(w)$ or $q^{-1}(w)$ to get $\phi(w)$, which again can be done on $\log|w|$-tape.

This completes the proof that there exists an L-isomorphism between the sets A and B. ∎

Next we derive several technical results which will lead to a very simple result giving necessary and sufficient conditions that a complete set is L-isomorphic to a fixed, well known complete set in the class under consideration.

First, we derive condition under which a logn-tape reduction can yield a one-to-one logn-tape invertable reduction.

<u>Lemma 2</u>: Let A be a set for which two L-computable functions $S_A(\ ,\ )$ and $D_A(\ )$ exist such that

1) $(\forall x,y)[S_A(x,y) \in A \text{ iff } x \in A]$

2) $(\forall x,y)[D_A(S(x,y)) = y]$.

Then if f is any L-reduction of C to A, the map

$$f'(x) = S_A[f(x),\ x]$$

is a one-to-one L-reduction of C to A and $(f')^{-1}$ is L-computable.

<u>Proof</u>: From condition 1) we see that

$$S_A[f(x),\ x]$$

is an L-reduction of C to A since

$$x \in A \text{ iff } S_A[f(x),\ x] \in A.$$

To see that f' is one-to-one assume that

$$f'(x) = f'(y)$$

then

$$x = D_A[f'(x)] = D_A[S_A(f(x), x)] = D_A[S_A(f(y), y)] = y,$$

so f' is a one-to-one map. Furthermore,

$$q(x) = \underline{\text{if}}\ x = S_A[f(D_A(x)), D_A(x)]\ \underline{\text{then}}\ D_A(x)\ \text{else}\ *$$

is the inverse of f' and since S_A, D_A and f are all logn-tape computable so is $(f')^{-1}$, as was to be shown.

To apply Theorem 1 we still need the length squaring property, which we consider next.

Let $A \subseteq \Sigma^*$. Then $Z_A: \Sigma^* \to \Sigma^*$ is a <u>padding</u> <u>function</u> for the set A if

1. $Z_A(x) \in A$ iff $x \in A$
2. Z_A is one-to-one.

<u>Lemma 3</u>: Let f be one-to-one L-reduction of A to B and let f^{-1} be L-computable. Assume that either A or B has a padding function Z_X (X = A or B) which satisfies the conditions:

1) Z_X and Z_X^{-1} are L-computable,

2) $(\forall y)[|Z_X(y)| > |y|^2 + 1]$.

Then there exists a one-to-one L-reduction f' of A to B such that

i) $(f')^{-1}$ is L-computable,

ii) $(\forall y)[|f'(y)| > |y|^2]$.

Proof: Let x = A. Since f and f^{-1} are computable on logn-tape there exists a (non-decreasing) polynomial p such that $p(n) > n^2$ and

$$\forall(x)\,[\,|f(x)| < p(|x|) \text{ and } |f^{-1}(x)| < p(|x|)\,].$$

Then from condition 2) we know that there exists an integer r such that

$$(\forall x)\,[\,|Z_A^r(x)| > p(|x|^2)\,].$$

Therefore

$$|f \circ Z_A^r(x)| > |x|^2,$$

since

$$|f \circ Z_A^r(x)| < |x|^2$$

implies that

$$|f^{-1} \circ f \circ Z_A^r(x)| = |Z_A^r(x)| < p(|x|^2),$$

which is a contradiction. Thus

$$f'(x) = f \circ Z_A^r(x)$$

is an L-reduction of A to B with the desired properties.

Assume that the padding function exists for the set B, Z_B. Then there exists an integer r such that for all x

$$|Z_B^r(x)| > p \circ p(|x|)$$

where p is the polynomial defined in the first part of the proof. Then

$$(\forall x)[p(|f(x)|) \geq |x|],$$

since

$$p(|f(x)|) < |x|$$

implies that $f(x) = y$ and

$$|f^{-1}(y)| < p(|y|) < |x|,$$

contradicting the fact that

$$f^{-1} \circ f(x) \doteq x.$$

But then

$$f' = Z_B^r \circ f$$

is a one-to-one L-reduction of A to B with $(f')^{-1}$ L-computable, and

$$|f'(x)| = |Z_B^r \circ f(x)| > p \circ p(|f(x)|) \geq p(|x|) > |x|^2.$$

This completes the proof. ∎

We now combine these results to get our next result.

Theorem 4: Let the set A be L-reducible to B and B be L-reducible to A; furthermore let the set A have a padding function Z_A satisfying Lemma 3 and functions S_A and D_A satisfying Lemma 2. Then B is L-isomorphic to A iff B has functions S_B and S_A satisfying Lemma 2.

Proof: If A and B are L-isomorphic under the bijection ϕ then we can define

$$S_B[x, y] = \phi^{-1} \circ S_A[\phi(x), y]$$

and

$$D_B(x) = D_A[\phi(x)].$$

Clearly, S_B and D_B are L-computable since S_A, D_A, ϕ and ϕ^{-1} are. Furthermore for all x and y

$$x \in B \text{ iff } \phi(x) \in A \text{ iff } S_A[\phi(x), y] \in A$$
$$\text{iff } \phi^{-1} \circ S_A[\phi(x), y] = S_B[x,y] \in B$$

and

$$D_B[S_B(x, y)] = y.$$

Conversely, if the functions S_B and D_B exist then by using Lemmas 2 and 3 with Theorem 1 we get an L-isomorphism between A and B. This completes the proof. □

3. APPLICATIONS

To apply the previously developed result, Theorem 4, we will pick from each of the families

NL, CSL, P, NP and PTAPE,

a well known complete language and show that each of these languages has a padding function Z_A and functions D_A and S_A satisfy

Lemmas 2 and 3, respectively. Then any complete set B in the corresponding class will be L-isomorphic to these "classic" sets if and only if B possesses the two L-computable functions S_B and D_B, satisfying Lemma 2. As it will be seen from examples, this is a very simple test.

The following are our representative L-complete sets from different classes.

NL: The graph accessability problem, GAP, defined below, is complete for NL [6].

INPUT: Directed graph G with an "in" and "out" node.

PROPERTY: There is a directed path from the "in" to the "out" node.

CSL and PTAPE: The regular expression problem denoted by R_{Σ^*}, is complete for CSL and PTAPE [1,4,8].

INPUT: Regular expression R over $\Sigma, \cdot, \cup, *,), ($.

PROPERTY: $L(R) \neq \Sigma^*$.

P: The context-free language emptiness problem, CFLϕ, is complete for P [7].

INPUT: Context-free grammar G.

PROPERTY: $L(G) = \phi$.

NP: The conjunctive normal form Boolean function satisfiability problem, SAT, is complete for NP [1,3,8].

INPUT: A Boolean function $F(x_1, x_2, \ldots, x_n)$ in CNF.

PROPERTY: There exists an assignment $x_i \in \{0,1\}$, $1 \leq i \leq n$, such that $F(x_1, x_2, \ldots, x_n) = 1$.

Lemma 5: Each of the sets GAP, $R_{\Sigma*}$, CFLϕ, and SAT has a padding function Z and functions D and S, satisfying Lemmas 2 and 3, respectively.

Proof: We note that the GAP property cannot be destroyed nor introduced by adding a set of new nodes which do not connect to any of the old nodes. We now describe informally the desired functions.

1. Z_{GAP} adds to G in the description of (G, "in", "out") a new node with an edge leading back to itself, followed by enough new nodes without edges, to square the length of the description of the old problem.

2. $D_{GAP}(x, y)$ (let $x \in \Sigma^*$ and $y \in \{0,1\}^*$) adds to the graph described in x a new node with two edges and then encodes y in a sequence of new edges with one edge and no-edge, representing ones and zeros, respectively.

3. S_{GAP} is such that

$$S_{GAP} \circ D_{GAP}(x, y) = y.$$

It is easily seen that all three functions exist and are L-computable and that Z_{GAP} is one-to-one and Z^{-1}_{GAP} is L-computable.

Next we consider the set $R_{\Sigma*}$, which is complete for CSL and PTAPE. We denote the corresponding functions by Z_R, D_R and S_R.

1. $Z_R(R) = (\Lambda + 0 + 1)^{|R|^2} + (0 + 1)^{|R|^2 + 1}\ R$

2. $D_R(R, y) = y + (\Lambda + 0 + 1)^{|y|} + (0 + 1)^{|y| + 1}\ R$.

3. $S_R \circ D_R(R, y) = y.$

Again it is easily seen that the three functions have the desired properties and that Z_R, D_R, S_R and Z_R^{-1} are L-computable.

For the CFLϕ problem let the context-free grammar be given by

$$G = (N, T, \tau, Pr).$$

where N, T, τ and Pr denote, respectively, the set non-terminal symbols, terminal symbols, the starting symbol and productions. To construct the desired functions Z_C, D_C and S_C we will add three new symbols to N and sufficiently many new productions in these symbols (the productions may be repeated many times in Pr) to square the length of G. This yields

$$Z_C(G) = G'.$$

Similarly, $D_C(G,y)$, $y \in \{0,1\}^*$ encodes the sequence y with new productions added to Pr by using, say $N_1 \rightarrow N_1N_1$ as a marker and then $N_1 \rightarrow N_1N_2$ for a "zero" and $N_1 \rightarrow N_1N_3$ for a "one". Clearly Z_C, Z_C^{-1}, D_C and S_C are all L-computable and satisfy the required conditions.

Finally, we note that the desired functions Z_{SAT}, D_{SAT} and S_{SAT} exist for the satisfiability problem. Let F have the variables $x_1, x_2, \ldots, x_n$ then let

$$Z_{SAT}(F) = F \cdot (x_{n+1} + \bar{x}_{n+1})x_{n+2} \cdot x_{n+3} \cdots x_{n+k}$$

with k such that

$$|Z_{SAT}(F)| > |F|^2.$$

Let

$$D_{SAT}(F, y) = F\cdot(x_{n+1} + \bar{x}_{n+1})\bar{x}_{n+2}^{y_1} \cdot \bar{x}_{n+3}^{y_2} \cdots \bar{x}_{n+2+|y|}^{y_{|y|}}$$

where

$$\bar{x}^{y_i} = \text{if } y_i = 0 \text{ then } x \text{ else } \bar{x}.$$

Again it is easily seen that these functions are L-computable and that so are Z_{SAT}^{-1} and S_{SAT}, satisfying

$$S_{SAT} \circ D_{SAT}(F, y) = y.$$

This completes the proof. ∎

Theorem 6: Any complete set B in NL, CSL, P, NP or PTAPE is, respectively, L-isomorphic to

$$\text{GAP},\ R_{\Sigma^*},\ \text{CFL}\phi,\ \text{SAT or } R_{\Sigma^*}$$

iff there exist two L-computable functions $D_B(\ ,\ )$ and $S_B(\ )$ such that

1. $(\forall x,y)[D(x,y) \in B \Longleftrightarrow x \in B]$
2. $(\forall x,y)[S \circ D(x,y) = y]$.

Proof: Combining Lemma 5 and Theorem 4. ∎

As seen from the examples in Lemma 5 and [2] the necessary and sufficient conditions of Theorem 6 are very easy to check. We have checked them for a large number of complete sets for different families of languages without encountering any difficulties. Thus we can assert the following.

CLAIM: The "known" complete sets for NL, CSL, P, NP, PTAPE, etc. are all, respectively L-isomorphic.

At the same time it should be recalled that we do not know whether all complete sets in these classes are L-isomorphic. Furthermore, a proof of this fact for the classes NL and P, for example, would yield that

$$\{T(M_i) \mid L_i(n) \leq \log n\} \subsetneq \text{NL and P},$$

which are at the present unsolved problems and appear to be quite difficult.

4. DENSITY CONSIDERATIONS

It was shown in [2] that languages complete for EXPTIME and EXPTAPE, under polynomial time reductions, could not be over a single letter alphabet nor could they be sparse. In this section we strengthen these results for completeness defined under L-reductions and extend them to computationally less complex families of languages.

A language A, $A \subseteq \Sigma^*$, is sparse if there exists a polynomial p such that

$$|A \cap \Sigma^n| \leq p(n).$$

Theorem 7: No language L-complete for CSL, PTAPE, EXPTIME or EXPTAPE can be sparse.

Proof: We will outline a diagonalization process which constructs a non-sparse set A, acceptable on deterministic linear tape, $A \in \text{TAPE}[n]$, such that any L-reduction of A to any other set must be one-to-one almost everywhere. Therefore, a complete set to which A is L-reducible cannot be sparse, since one-to-one L-mappings map non-sparse sets into non-sparse sets.

Construction of A:

Let M_1, M_2,... be an enumeration of all L-transducers. Let I_n be a subset of this enumeration, namely those L-transducers which up to stage n in our construction of A have been found no to be L-reductions of A to any set.

STAGE 0: Let $I_o = \phi$, $A_o = \phi$, $A'_o = \phi$. Go to STAGE 1.

STAGE n: Let w be the first string not in $A_{n-1} \cup A'_{n-1}$. On $|w|$ tape enumerate as many M_{ij} not in I_{n-1} as possible, refer to this enumerated list as L_n. If L_n is empty let

$$A_n = A_{n-1} \cup \{y \bigm| |y| = |w|\}$$

$$A'_n = A_{n-1}$$

$$I_n = I_{n-1}$$

and go to STAGE n+1.

SUBSTEP: If sublist L_n is empty go STAGE n+1, else pick first M_{ij} in L_n, let $L_n \leftarrow L_n - \{M_{ij}\}$, and try to find on the available tape (a minimal pair) x and z such that

$$x \neq z, \; M_{ij}(z) = M_{ij}(x),$$

and

$$x \text{ and } z \text{ are not both in } A_{n-1} \text{ or } A'_{n-1}.$$

If such x and z are found, add them to A_{n-1} or A'_{n-1} so that (or check that)

$$x \in A_n \text{ and } z \in A'_n \text{ or vica versa,}$$

furthermore, to make A dense, let A_n contain all y such that $|y| = |w| + 1$, and let

$$I_n = I_{n-1} \cup \{i_j\}.$$

Go to STAGE n+1.
If no x and z is found in available tape, go to SUBSTEP.

It is seen from the construction that A is such that any M_i which L-reduces A to any set must be one-to-one from some point on, or else we would have placed and x in A and z in $\bar{A}$ such that

$$M_i(x) = M_i(z),$$

insuring that M_i is not an L-reduction of A. Furthermore, it is seen that A is in TAPE [n] and that A is not sparse. Thus we conclude that there cannot exist any sparse set, recursive or otherwise, to which A, A in TAPE [n], can be L-reduced. Thus no L-complete set in CSL or PTAPE can be sparse, as was to be shown.

It remains an interesting open question whether NL or P can have, respectively, L-complete sets which are sparse. We conjecture that this is not the case.

5. REFERENCES

1. Aho, V. A., J. E. Hopcroft and J. D. Ullman, "The Design and Analysis of Computer Algorithms", Addison-Wesley, Reading, Mass., 1974.

2. Berman, L. and J. Hartmanis, "On Isomorphisms and Density of NP and Other Complete Sets", SIAM J. of Computing 6(1977) 305-322.

3. Cook, S. A., "The Complexity of Theorem Proving Procedures", Proceedings 3rd Annual ACM Symposium on Theory of Computing, 1971, 151-158.

4. Hartmanis, J. and J. Simon, "On the Structure of Feasible Computations", Advances in Computers, editors Morris Rubinoff and Marshall C. Yovits, Academic Press, New York, 1976, 1-43.

5. Jones, N. D., "Space-Bounded Reducibility Among Combinatorial Problems", J. of Computer and System Sciences, 11(1975) 68-85.

6. Jones, N. D., Y. E. Lien and W. T. Laaser, "New Problems Complete for Log Space", Mathematical Systems Theory, 10(1976) 1-17.

7. Jones, N. D., and W. T. Laaser, "Problems Complete for Deterministic Polynomial Time", Theoretical Computer Science (to appear).

8. Karp, R. M., "Reducitility Among Combinatorial Problems", In Complexity of Computer Computations, edited by R. E. Miller and J. Thatcher, Plenum Press, New York, 1972, 85-104.

9. Stockmeyer, L. J., and A. R. Meyer, "Word Problems Requiring Exponential Time: Preliminary Report", Proceedings Fifth Annual ACM Symposium on Theory of Computing, 1973, 1-9.