A NOTE ON NATURAL CREATIVE SETS AND GOEDEL NUMBERINGS*

J. Hartmanis

TR80-405

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

*This research has been supported in part by National Science Foundation Grant MCS 78-00418.

A NOTE ON NATURAL CREATIVE SETS AND GOEDEL NUMBERINGS*

by

J. Hartmanis

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

Abstract

Creative sets (or the complete recursively enumerable sets) play an important role in logic and mathematics and they are known to be recursively isomorphic. Therefore, on the one hand, all the creative sets can be viewed as equivalent, on the other hand, we intuitively perceive some creative sets as more "natural and simpler" than others. In this note we try to capture this intuitive concept precisely by defining a creative set to be natural if all other recursively enumerable sets can be reduced to it by computationally simple reductions and show that these natural creative sets are all isomorphic under the same type of computationally simple mappings. The same ideas are also applied to define natural Goedel numberings.

*This research has been supported in part by National Science Foundation Grant MCS 78-00418.

1. Introduction

It is well known that the sets of provable theorems of sufficiently rich axiomatized mathematical theories form creative sets [7]. Furthermore, it is known that all the creative sets are recursively isomorphic and that they are the same class of sets as the class of complete recursively enumerable sets. Therefore, from a recursive function theory point of view all the creative sets can be viewed as equivalent since between any two of them there exists a recursive bijection. At the same time, we have an intuitive feeling that conceptually and computationally some creative sets are simpler than others. Since the creative sets are not recursive, there cannot exist recursive computational complexity bounds for their recognition and therefore we cannot use standard computational complexity techniques to classify them as we can for recursive sets. On the other hand, we observe that for every creative set the computational complexity of reducing all other recursively enumerable sets to it is recursively bounded and that this computational complexity bound on the reductions naturally classifies the complexity of creative sets. The same observation holds, for example, for complete sets in any other level of the Kleene hierarchy as well as for the classification of Goedel numberings of partial recursive functions.

In this note we exploit these ideas to classify the complexity of complete sets (for some family of sets) by the computational complexity required to reduce all other sets (from that family) to them. In particular we define a creative set A to be natural (or p-creative) if all other recursively enumerable sets can be reduced to A by a

one-to-one polynomial-time (Turing machine) computable mapping whose (left) inverse can also be computed in polynomial-time. This definition is justified by the fact that many creative sets that appear naturally in logic and mathematics are of this type and that, as will be seen, the family of natural creative sets has desirable mathematical properties. More explicitely, we show that above defined natural creative sets are isomorphic under polynomial-time mappings and therefore the computationally simple polynomial-time reductions play the same role for the natural creative sets as the recursive mappings for the whole family of creative sets.

We observe that any natural creative set has certain polynomial-time computable and invertable padding functions. Furthermore, we show that any creative set to which any other recursively enumerable set can be reduced to by a many-one polynomial-time mapping is a natural creative set iff it possesses the above mentioned padding functions.

We leave as an interesting open problem whether there exist any creative sets to which any other recursively enumerable set can be reduced to by a many-one polynomial-time mapping, but which are not natural creative sets, i.e. they are not polynomial-time isomorphic to the classic natural creative sets, such as

$$\{M_i \mid M_i(-) \text{ halts}\}.$$

In the last part of this paper we indicate how these results can be extended to other classes of computationally simple reductions, such as log-tape and polynomial-tape bounded computations.

Finally, we apply the same concepts to define natural Goedel numberings and derive the corresponding isomorphism results.

2. Preliminaries

For the sake of completeness we summarize the fundamental concepts used in this note.

Let $M_1, M_2, \ldots$ be a standard enumeration of the set of (one-tape) Turing machines and let $L(M_i)$ denote the set of inputs accepted by the Turing machine (Tm), M_i. Let $T_i(n)$ denote the maximal number of operations performed by M_i on inputs of length n. We say that M_i runs in polynomial-time (or is a polynomial-time Tm) iff for some $k \geq 0$,

$$T_i(n) \leq n^k + k.$$

A set B is productive if there exists a recursive function ρ such that for any Tm, M_i,

$$L(M_i) \subseteq B \text{ implies } \rho(i) \in B - L(M_i).$$

A recursively enumerable set (r.e.) is creative iff its complement is productive. It is well known from logic and recursive function theory [7] that the provable theorems of sufficiently powerful axiomatized mathematical systems form creative sets and that the true theorems of the corresponding mathematical areas form productive sets.

The creative sets are exactly the complete r.e. sets, as define next [7].

A set A is recursivly reducible to a set B iff there exists a recursive function f such that

$$x \in A \text{ iff } f(x) \in B.$$

A set A is a complete recursively enumerable set iff any other r.e. set can be recursively reduced to A.

The sets A and B are recursively isomorphic iff there exists a recursive bijection f which reduces A to B. Thus f is a recursive one-to-one, onto function from Σ^* to Γ^*, $A \subseteq \Sigma^*$, $B \subseteq \Gamma^*$, such

$$x \in A \text{ iff } f(x) \in B.$$

Similarly we define polynomial-time reductions and polynomial-time isomorphisms to capture the concepts of computationally simple reductions of sets and computationally simple isomorphisms.

A set A is polynomial-time reducible (or p-reducible) to a set B iff there exists a polynomial-time computable mapping, f, which reduces A to B, i.e.

$$x \in A \text{ iff } f(x) \in B.$$

The sets A and B are polynomial-time isomorphic (or p-isomorphic) iff there exists a one-to-one onto function f such that f and f^{-1} can be computed in polynomial-time and

$$x \in A \text{ iff } f(x) \in B.$$

It is well known [7] that all the creative (or r.e. complete) sets are recursively isomorphic. Therefore from a recursive func-

tion theory point of view all creative sets can be considered to be similar. On the other hand, intuitively we feel that some creative sets are conceptually and computationally simpler than others and we believe that our next definition captures this intuitive idea.

Definition: A creative set B is natural (or p-natural) iff any other r.e. set can be reduced by B by a one-to-one mapping f such that f and f^{-1} are polynomial-time computable.*

For example, the following creative sets are easily seen to be natural:

$$U_1 = \{(M_i,x) \mid M_i \text{ halts on } x\};$$
$$U_2 = \{M_i \mid M_i \text{ halts on blank tape}\};$$
$$U_3 = \{M_i \mid M_i \text{ halts on } i\}.$$

The reduction of $L(M_{i_0})$ to U_1 is given by the polynomial-time mapping

$$x \longmapsto (M_{i_0},x).$$

The reduction of $L(M_{i_0})$ to U_2 is given by

$$x \longmapsto M_{\sigma(i_0,x)}$$

where $M_{\sigma(i_0,x)}$ is a machine which for all inputs first writes x on its tape and then simulates M_{i_0} on x. Therefore

$$x \in L(M_{i_0}) \text{ iff } M_{\sigma(i_0,x)} \in U_2,$$

*All through this paper let f^{-1} denote the left inverse of f. Note that since f^{-1} is polynomial-time computable we can determine in polynomial-time for all x whether $f^{-1}(x)$ is defined.

and it is easily seen that the construction of $M_{\sigma(i_0,x)}$ can be carried out in polynomial-time in the length of x. It is also seen that

$$x \in L(M_{i_0}) \text{ iff } M_{\sigma(i_0,x)} \in U_3.$$

Many other creative sets can be shown to be natural and it can easily be proven that not all creative sets are natural, as indicated below.

Theorem 1: For any given recursive function T(n) there exists a creative set A_T such that not all r.e. sets can be reduced to A_T by Tm's running in time T(n).

Proof: By a reasonably straightforward diagonalization argument. ∎

As a matter of fact, there is no natural creative set over a single letter alphabet. To see this, we just have to observe that under a one-to-one polynomial-time reduction G, the image of U_1 cannot be polynomially sparse, i.e., there is no polynomial p such that

$$|\{G(x) \mid x \in U_1 \text{ and } |G(x) \mid \leq n\}| \leq p(n).$$

Therefore, no polynomially sparse set can be a natural complete set and since any subset of a* is polynomially sparse no such subset can form a natural creative set.

3. Isomorphism Results

In this section we prove that all natural creative sets are p-isomorphic and discuss some other properties of these sets. The p-isomorphism theorem shows that natural creative sets are very similar indeed and that the p-reductions and p-isomorphisms play the same role for natural creative sets as recursive reductions and isomorphisms for the family of all creative sets.

To prove the p-isomorphism result we recall some results about polynomial-time reductions [1]. Note that the following lemma can be viewed as a feasibly computable analogue of the Cantor-Bernstein theorem.

Lemma 2: Let f and g be p-reductions of A to B and B to A, respectively, such that

a) f and g are one-to-one,

b) f^{-1} and g^{-1} are p-computable,

c) $|f(z)| > |z|$ and $|g(z)| > |z|$.

Then A and B are p-isomorphic.

Proof: For details of the proof see [1]. To see the basic idea of the proof let $A \subseteq \Sigma^*$ and $B \subseteq \Gamma^*$. We say that $x \in \Sigma^*$ (Γ^*) is a parentless element if x is not in the range of g (f). We say that x in Σ^* is a descendent of an element $z \in \Sigma^*$ (Γ^*) if

$$(g \circ f)^k(z) = x \quad ((g \circ f)^k g(z) = x),\ k = 0,1,2,\ldots$$

Conditions a), b) and c) imply that we can compute in polynomial time for x in Σ^* (or Γ^*) whether it has a parentless ancestor in

Σ^* or Γ^* (since f^{-1} and g^{-1} are polynomial-time computable we can detect in polynomial-time that $f^{-1}(x)$ is not defined, i.e. that there is no y such that $f(y) = x$, the same holds for $g^{-1}(x)$). The desired p-isomorphism is given by

$$\phi(x) = \underline{\text{if}}\ x \text{ has a parentless ancestor in } \Sigma^* \ \underline{\text{then}}\ p(x) \ \underline{\text{else}}\ q^{-1}(x).$$

With little effort it can be seen that ϕ and ϕ^{-1} are polynomial-time computable, one-to-one, onto reductions of A to B and B to A respectively. ∎

To show that any two natural creative sets are p-isomorphic it suffices to show that any given natural creative set B is p-isomorphic to universal set U_1. Furthermore, it is easily seen that the reduction of any r.e. set, $A = L(M_{i_0})$, to U_1 (given by $x \longmapsto (M_{i_0}, x)$) satisfies the conditions of the above lemma. Therefore, to show that any two natural creative sets are p-isomorphic we only have to show that U_1 can be reduced to any natural creative set by a p-computable function f satisfying the conditions of the above lemma. By definition U_1 can be reduced to any natural creative set by a p-reduction satisfying conditions a) and b). To show that this reduction can be chosen to be length increasing (condition c), we exploit a special property of the set U_1 (which we will show later to hold for all natural creative sets).

Lemma 3: For the set U_1 there exists a one-to-one function $Z: \Sigma^* \to \Sigma^*$ such that

a) Z and Z^{-1} are p-computable;

b) $(\forall y)\ [|Z(y)| > |y|^2 + 1]$;

c) $y \in U_1$ iff $Z(y) \in U_1$.

Proof: If y is not of the format (M_i,x) (which we assume can be detected in polynomial-time) then let $Z(y) = y^k$, $k = |y|^2 + 1$ (if $y = \epsilon$ then let $Z(y) = a$). For $y = (M_i,x)$ let $Z[(M_i,x)] = (M_{\sigma(i)},x)$ where $M_{\sigma(i)}$ is obtained from M_i by adding new states, which are not reachable from any other states, to guarantee that

$$|(M_{\sigma(i)},x)| > |(M_i,x)|^2 + 1.$$

Clearly, $y \in U_1$ iff $Z(y) \in U_1$. ∎

We now show that the existence of the padding function Z for U_1 allows us to construct a desired reduction of U_1 to any natural creative sets.

Lemma 4: Any natural creative set A can be reduced to U_1 (and therefore to any other natural creative set) by a one-to-one mapping f such that f and f^{-1} are computable in polynomial-time and for all y

$$|f(y)| > |y|.$$

Proof: By definition of a natural creative set there exists a one-to-one reduction h of U_1 to A such that h and h^{-1} are p-computable.

Because h and h^{-1} are p-computable there exists a nondecreasing polynomial p such that

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

From the previous lemma we know that there exists an integer r such that

$$(\forall x)[|z^r(x)| > p(|x|)]$$

Therefore

$$|h \circ z^r(x)| > |x|,$$

since

$$|h \circ z^r(x)| \leq |x|$$

implies that

$$|h^{-1} \circ h \circ z^r(x)| = |z^r(x)| \leq p(|h \circ z^r(x)|) \leq p(|x|)$$

which is a contradiction. Therefore we conclude that

$$f(x) = h \circ z^r(x)$$

is a one-to-one reduction of U_1 to A for which f and f^{-1} are polynomial-time computable and f is length increasing. ∎

Theorem 5: Any two natural creative sets are polynomial-time isomorphic.

Proof: From our lemmas we know that there exist polynomial-time reductions of any creative natural set to U_1 and vica versa which satis the conditions of Lemma 2. Therefore A and U_1 are isomorphic and since the composition of p-isomorphisms yields p-isomorphisms we conclude that all natural creative sets are p-isomorphic. ∎

4. Second Characterization

For the set U_1 we can construct two polynomial-time computable functions S and D such that

$$(\forall x,y)[S(x,y) \in U_1 \text{ iff } x \in U_1]$$
$$(\forall x,y)[D \circ S(x,y) = y].$$

In essence the function S padds any given x by y in polynomial-time and $S(x,y)$ is in U_1 iff x is in U_1, the function D retrieves in polynomial-time the string y from the padded string.

It can be seen that for many creative sets involving Tm's in their definition, as for U_1, U_2 and U_3, we can construct the padding functions S and D by just padding the Turing machine descriptions with patterns of new states without changing their behaviour.

Similarly, for many creative sets which are formed by provable theorems of an axiomatized formal mathematical theory, we can constru polynomial-time computable padding functions. This can be done by taking any string (theorem) and adding to it patterns of simply provable statements by the logical "and" operation. Clearly, the resulting string is a provable theorem if and only if the original

string is a provable theorem.

From the above observations we see that for many creative sets arising in mathematics and logic there exist easily computable padding functions. As a matter of fact we know that they exist for all natural creative sets.

Corollary 6: If A is a natural creative set then there exist two p-time computable functions S and D such that

$$(\forall x,y)[S(x,y) \in A \text{ iff } x \in A]$$
$$(\forall x,y)[D \circ S(x,y) = y].$$

Proof: Let f be a p-isomorphism between U_1 and A. Let S and D be the p-time computable padding function for U_1. Then the corresponding p-time computable functions for A, S_A and D_A, are given by

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

and

$$D_A(x) = D \circ f(x).$$

■

As our next result shows we can also characterize natural creative sets in terms of padding functions.

Theorem 7: Let A be an r.e. set to which any other set can be reduced by a (many-one) polynomial-time mapping. Then A is a natural creative set if and only if there exist two polynomial-time computable

functions S and D such that

$$(\forall x,y)[S(x,y) \in A \text{ iff } x \in A]$$
$$(\forall x,y)[D \circ S(x,y) = y].$$

Proof: By Corollary 6 we know that if A is a natural creative set then two p-time computable functions S and D exist.

On the other hand, if A is an r.e. set to which any other r.e. set, B, can be reduced by a p-time mapping f then

$$f'(x) = S[f(x),x]$$

is a p-time computable reduction of B to A because

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

Furthermore, f' is one-to-one, since if

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

then

$$x = D[f'(x)] = D[S(f(x),x)] = D[S(f(y),y)] = D[f'(y)] = y.$$

Finally, $(f')^{-1}$ can be computed in polynomial-time, since it is given by

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

where * indicates that x is not in the range of f'.

From the above we see that A is a natural creative set since

any r.e. set can be reduced to A by a one-to-one mapping f' such that f' and $(f')^{-1}$ are computable in polynomial-time. ∎

The above result gives an easy test whether an r.e. set is a natural creative set.

Test: An r.e. set A is a natural creative set iff the set U_1 can be reduced to it by a p-time computable mapping and there exist two p-time computable functions S and D such that

$$D(x,y) \in A \text{ iff } x \in A$$

and

$$S \circ D(x,y) = y.$$

5. Other Reductions

The previous results were derived for polynomial-time computations, which are currently accepted as a good mathematical model for the feasible computations. At the same time, these results can be extended to several other classes of computationally simple reductions

The first class is the class of LOGTAPE computations. These are computations which can be performed by a three-tape Turing machine which has--a read only input tape, a one-way write only output tape and for input x a $\lceil \log |x| \rceil$ long two-way, read-write work tape [2]. Clearly the LOGTAPE computations are contained among the polynomial-time computations but it is not known whether this containment is pro

The second class of functions is the PTAPE class and consists of all the functions which can be computed on polynomially bounded tape. For sake of simplicity we assume that the input and output will be written on the polynomial long work tape and therefore the length of the output is also polynomial bounded in the length of the input. (This assumption is not essential and the results can be extended to polynomial tape computations with an unbounded one-way, write-only output tape.) Again, it is not known whether the above defined PTAPE reduction properly contain tha polynomial-time reductions, but it is easily seen that LOGTAPE $\subseteq$ PTAPE.

The results for these reductions look quite similar to the previous results for polynomial-time computations, with the appropriate changes in the three assumptions.

Lemma 8: Let f and g be LOGTAPE reductions of A to B and B to A, respectively, such that

a) f and g are one-to-one;

b) f^{-1} and g^{-1} are LOGTAPE-computable;

c) $|f(z)| > |z|^2$ any $|g(z)| > |z|^2$ (it actually suffices to require $|f(z)| > |z|^{1+\varepsilon}$, $|g(z)| > |z|^{1+\varepsilon}$ for some $\varepsilon > 0$).

Then A and B are LOGTAPE-isomorphic.

In the next result we assume that $A,B \subseteq \Sigma^*$ and that Σ^* is ordered lexiographically and that $x > y$ denotes this ordering.

Lemma 9: Let f and g be PTAPE-reductions of A to B and B to A, respectively, such that

a) f and g are one-to-one

c) $f(z) > z$ and $g(z) > z$.

Then A and B are PTAPE-isomorphic.

The proofs of these results follow the same pattern as the proof of Lemma 2, with the difference that for the LOGTAPE-reductions we strengthened condition c) to length squaring, to be able to trace the mappings backwards on the small amount of tape available for this computation. For complete proof of this result see [2].

For the PTAPE computations the ability to reuse the tape permits us to drop condition b) (since for any one-to-one PTAPE function f with $f(z) > z$, f^{-1} is PTAPE computable) and again because of the ability to reuse the tape we need to require for condition c) only that the value of elements is increased by the mappings f and g, to guarantee that on polynomial tape we can trace the computaions back to parentless ancestors to compute the desired isomorphism.

The special conditions about how fast the mapping must grow for the three isomorphism results, Lemmas 2, 8 and 9, reflect the nature of the three different types of resource bounds used in these reductions. At the present we do not know whether any of the growth conditions in these results can be weakened. We conjecture that they cannot be eliminated.

Quite surprisingly, when we apply the previous results to the creative sets the different conditions imposed on the growth of the

reductions disappear and all three theorems have the same form.

A creative set is LOGTAPE-creative iff any other r.e. set can be reduced to it by a one-to-one LOGTAPE-computable and reversable reduction.

A creative set is PTAPE-creative iff any other r.e. set can be reduced to it by a one-to-one PTAPE-reduction.

Clearly, the sets U_1, U_2 and U_3 are seen to be LOGTAPE-creative as well as PTAPE-creative.

The LOGTAPE-isomorphisms and PTAPE-isomorphisms are defined similarly to p-isomorphisms.

By methods similar to the ones used in the treatment of p-creative sets we can derive the following result.

Theorem 10: Any two p-creative, LOGTAPE-creative and PTAPE-creative sets are p-isomorphic, LOGTAPE-isomorphic and PTAPE-isomorphic, respectively.

Furthermore, we can characterize the LOGTAPE-creative, p-creative and PTAPE-creative sets by their respective padding functions.

Theorem 11: An r.e. set A to which any other set can be reduced to by a LOGTAPE-reduction, p-reduction or PTAPE-reduction is LOGTAPE-creative, p-creative or PTAPE-creative iff there exist two LOGTAPE-computable, p-computable and PTAPE-computable functions S and D, respectively such that

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

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

6. Natural Goedel Numberings

It is well known that acceptable Goedel numberings (GN), as defined and characterized in [6], form the class of sets of names of the partial recursive functions which can be recursively translated into each other. Furthermore, it is known that the acceptable Goedel numberings are all recursively isomorphic to each other.

We can view the accepetable Goedel numberings as models for programming systems [3,4]. Unfortunately, the above mentioned definition permits arbitrarily complicated Goedel numberings and the translations between them can be arbitrarily difficult to compute. To avoid these difficulties one can consider subclasses of Goedel numberings for which it is required that all other Goedel numberings can be translated into by restricted classes of mappings; such investigations have been initiated in [3,4,8].

Among the classes of Goedel numberings studied so far, the most natural appears to be the class of GN's to which any other GN can be translated by a polynomial-time computation. In [3] it was conjectured that any two such GN's are p-isomorphic and thus they would form a intuitively satisfying class of GN's with nice mathematical closure properties forming the feasibly computable analogue to the family of acceptable GN's. Unfortunately, in [4] this p-isomorphism problem was linked to the difficult problem of determining whether deterministic and nondeterministic polynomial-time computations are the same, namely the classic P = NP? problem. In view of this result, it was furthermore strongly conjectured in [4] that this family con-

tains GN's which are not p-isomorphic.

In this section we point out that the methods discussed in this paper permit us to define natural Goedel numberings in the same manner as natural creative sets.

Let $\phi_1,\phi_2,\ldots$ and $\sigma_1,\sigma_2,\ldots$ be any two acceptable Goedel numberings, i.e. a listing of algorithms to compute all partial recursive functions and which have a universal algorithm and satisfy the S-m-n Theorem [6].

The GN $\sigma_1,\sigma_2,\ldots$ is p-translated to the GN $\phi_i,\phi_2,\ldots$ if there exists a polynomial-time computable function f such that

$$f(\sigma_i) = \phi_j \text{ and } \sigma_i \equiv \phi_j.$$

(We deviate here from the conventional way of writing $\sigma_i \equiv \phi_{f(i)}$, since we think of σ_i and ϕ_j as programs and the translation is a mapping of programs into programs.)

Definition: A GN, $\phi_1,\phi_2\ldots$, is natural (or p-natural) iff any other GN can be translated to it by a one-to-one polynomial-time translation whose (left) inverses are also polynomial-time computable.

We observe that there exist many natural Goedel numberings [3,4,8]. As a matter of fact, we can easily see that the standard enumeration of Turing machines, $M_1,M_2,M_3,\ldots$, [say by listing the sets of quintuples, (state, input, next state, symbol printed, head motion), for each machine]yields a natural GN. To see this we just have to observe that by definition any GN $\phi_1,\phi_2,\ldots$ can be trans-

lated by a recursive mapping σ in to the GN $M_1, M_2, \ldots$,

$$\sigma(\phi_i) = M_j \text{ and } \phi_i \equiv M_j.$$

To see that a one-to-one polynomial-time computable and invertable translation exists we construct for each ϕ_i a Tm $p(\phi_i)$ which:

> for input x saves the input, prints ϕ_i on its tape, computes $\sigma(\phi_i) = M_j$ and then simulates M_j on the input x.

Clearly, $p(\phi_i)$ is equivalent to ϕ_i and the construction of $p(\phi_i)$, using ϕ_i and a fixed Tm which computes $\sigma(\phi_i)$, can be carried out in polynomial-time (in the length of ϕ_i). Furthermore, if we use a simple construction for $p(\phi_i)$ in which ϕ_i is represented in a form we can compute the left inverse of p also in polynomial-time.

Finally, we observe that $M_1, M_2, \ldots$ has polynomial-time computable padding functions Z, S and D such that:

$$Z(M_i) \equiv M_i \text{ and } |Z(M_i)| > |M_i|^2 + 1$$
$$S(M_i, x) \equiv M_i \text{ and } D \circ S(M_i, x) = x.$$

Using the above observations, methods similar to those used in the first part of this paper, we can derive the following results.

Theorem 12: Any two natural Goedel numberings are p-isomorphic.

Theorem 13: A Goedel numbering, $\phi_1, \phi_2, \ldots$, to which any other GN can be reduced by a polynomial-time mapping is natural iff there exist two polynomial-time computable functions S and D such that

$$(\forall_i, x)[S(\phi_i, x) \equiv \phi_i]$$
$$(\forall_i, x)[D \circ S(\phi_i, x) = x].$$

The above results can also be extended to LOGTAPE and PTAPE translations.

From the above results we see that the p-computable padding functions play an important role and that they exist for all natural GN's. Furthermore, if there exist GN's to which any other GN can be reduced to by a many-one polynomial-time mapping but not by a one-to-one p-time computable and invertable mapping, then this GN cannot have the p-computable padding functions. We leave as an open problem whether such GN's exist (see also [4]).

Furthermore, it would be interesting to derive other properties of natural Goedel Numberings and, in particular, to see whether any of the specially constructed GN's in recursive function theory to prove different properties of GN's, fail to be natural GN's (without having tried to make them not to be natural, which is easily done by diogonalization).

In conclusion, we observe that, though all natural GN's are p-isomorphic, this does not guarantee that the sets of minimal length machines of these GN's are isomorphic under recursive mappings. By methods similar to the ones used in [5] we can show

Corollary 14: There exist pairs of natural GN's whose sets of minimal machines are not recursively isomorphic.

REFERENCES

1. Berman, L. and J. Hartmanis, "On Isomorphisms and Density of NP and other Complete Sets," SIAM J. Comput., vol. 6, (1977) 305-322.

2. Hartmanis, J., "On Log-Tape Isomorphisms of Complete Sets," Theoretical Computer Science, vol. 8, (1978) 273-286.

3. Hartmanis, J. and T.P. Baker, "On Simple Goedel Numberings and Translations," SIAM J. Comput., vol. 4, (1975) 1-11.

4. Machtey, M., K. Winklmann and P. Young, "Simple Goedel Numberings, Isomorphisms, and Programming Properties," SIAM J. Comput., vol. 7, (1978) 39-60.

5. Meyer, A.R., "Program Size in Restricted Programming Languages," Information and Control, vol. 21, (1972) 382-394.

6. Rogers, . Jr., "Goedel Numberings and Partial Recursive Functions," J. Symbolic Logic, vol. 23, (1971) 444-475.

7. Rogers, H. Jr., "Theory of Recursive Functions and Effective Computability," McGraw-Hill, New York, 1967.

8. Schorr, C.P., "Optimal Enumerations and Optimal Goedel Numberings," Math. Systems Theory, vol. 8, (1975) 182-191.