ON SIMPLE GOEDEL NUMBERINGS AND TRANSLATIONS*

J. Hartmanis and T. P. Baker

TR 73-179

July 1973

Department of Computer Science
Cornell University
Ithaca, New York 14850

---

* This research has been supported in part by the National Science Foundation Grant GJ-33171X.

ON SIMPLE GOEDEL NUMBERINGS AND TRANSLATIONS

J. Hartmanis and T. P. Baker
Department of Computer Science
Cornell University
Ithaca, N.Y.

Abstract:

In this paper we consider Goedel numberings (viewed as simple models for programming languages) into which all other Goedel numberings can be translated very easily. Several such classes of Goedel numberings are defined and their properties are investigated. We also compare these classes of Goedel numberings to optimal Goedel numberings and show that translation into optimal Goedel numberings can be computationally arbitrarily complex.

ON SIMPLE GOEDEL NUMBERINGS AND TRANSLATIONS

J. Hartmanis and T. P. Baker

1. INTRODUCTION

It is well-known [1] that all (acceptable) Goedel numberings of the partial recursive functions are recursively isomorphic and thus, from an abstract recursive function theory point of view, they can all be considered equivalent. On the other hand, from a computational complexity point of view this is definitely not the case, since translations between Goedel numberings can be computationally arbitrarily complex. In particular, if we view Goedel numberings as simple models for programming languages, we are interested in numberings into which all other Goedel numberings can be translated easily. Similarly, we are interested in Goedel numberings with computationally simple S_n^m functions and functions satisfying the Recursion Theorem.

In this paper we define classes of Goedel numberings into which all other Goedel numberings can be translated by mappings of bounded complexity, investigate properties of these classes of Goedel numberings and compare these classes for different complexity bounds on the translations.

We observe that many programming languages and natural Goedel numberings are such that all other numberings can be translated into them by finite automata. We derive several results about such Goedel numberings, and state some interesting open problems.

In the last part of this paper we consider optimal Goedel numberings [2]. Though these numberings have some nice mathematical properties, we show that, from a computer science point of view, they have undesirable properties, since the translations into or between optimal Goedel numberings can be computationally arbitrarily complex and that similarly their S_1^1 functions and the functions satisfying the Recursion Theorem must be computationally arbitrarily complex.

2. COMPLEXITY CLASSES OF GOEDEL NUMBERINGS

Let R_k and P_k denote the recursive and partial recursive functions of k variables, respectively. For all g in P_{k+1}, let $g_i(\overline{x}) = \lambda\overline{x}[g(i,\overline{x})]$.

A Goedel Numbering, GN, of P_k is a function ϕ^k in P_{k+1} such that for all g in P_{k+1} there exists a t in R_1, satisfying

$$g(i,x_1,x_2,\ldots,x_k) = g_i(x_1,x_2,\ldots,x_k) = \phi_{t(i)}(x_1,x_2,\ldots,x_k) \quad .$$

for all i.

In this paper we are primarily concerned with GN's for P_1. Since we are interested in those GN's into which all others can be easily translated we will define complexity classes of GN's in terms of the computational complexity of into translations from other GN's.

Note that in actual translations from one programming language into another the translations are mapping sequences over finite alphabets into other sequences over a finite alphabet. Thus we will view the index i of a Goedel numbering as the binary sequence representing i and express some of our results in terms of operations on sequences. It should be noted that the whole treatment can easily be transcribed to the convention that we are indexing algorithms by the set Σ^+, and thus avoid some of the technical difficulties of mixing integers and their binary representations.

For any GN ϕ^1 ($\phi^1(i,x) = \phi_i(x)$) and every ψ in P_2 a translation of ψ into ϕ^1 is a recursive function σ such that

$$\psi(i,x) = \phi^1(\sigma(i),x) = \phi_{\sigma(i)}(x) \quad .$$

We will denote translations by writing

$$\sigma\colon \ \psi \to \phi^1 \quad .$$

Definition: Let C be any class of recursive functions. Then

$$\mathrm{GNC} = \{\phi^1 | \phi^1 \text{ is a GN and } (\forall\ \psi \text{ in } P_2)(\exists\ \sigma \text{ in } C)[\sigma\colon \ \psi \to \phi^1]\} \quad .$$

Thus GNC consists of those GN's into which all other GN's can be translated into by functions from C. Usually we will let C be some well-known class of functions of bounded computational complexity.

For example, let

GNReg

denote the class of all GN's into which all other GN's can be translated by finite automata mappings (i.e. deterministic gsm mappings [3]). We refer to these as regular GN's.

Similarly, let

$$C = \mathrm{Prfx} \text{ and } C = \mathrm{Pstfx}$$

denote the class of functions which prefix and postfix, respectively, a fixed string to the representation of i. That is, σ is in Prfx iff there exists ω such that

$$\sigma(i) = \omega i \quad .$$

We refer to

GNPrfx and GNPstfx

as prefix and postfix GN's, respectively.

The class

GNLBA

consists of all those GN's into which all other GN's can be translated by deterministic linearly bounded automata.

It should be observed that several programming languages and many natural GN's belong to GNReg and GNPrfx. Intuitively speaking, every GN or formal programming language in which we can "freely" program is in GNPrfx, since for any other GN ψ we just have to use a prefix ω with the meaning:

"This is a description of GN ψ, what follows is a description of an index i, please compute ψ_i."

Thus $\sigma(i) = \omega i$ will be the desired translation of ψ into ϕ.

Next we prove formally that postfix GN's exists. The postfix GN exhibited is the same as used by Schnorr [2] to show that there exist optimal GN's.

A GN ϕ is said to be optimal iff every other GN ψ can be translated into ϕ by a mapping σ such that

$$\sigma(i) \leq c \cdot i$$

for some positive constant c. We denote the class of optimal

GN's by GNOpt.

<u>Lemma</u>: There exist postfix GN's and

$$\text{GNPstfx} \subset \text{GNReg} \subset \text{GNLBA} \subset \text{GNOpt}$$

<u>Proof</u>: Let ϕ^2 be any GN of P_2 and let

$$g(i,n) = i2^{n+1} + 2^n - 1 \quad .$$

The pairing function g is a bijection and if we interpret it as mapping sequences into sequences, where i is the binary representation of the i-th integer and 1^n represents the sequence of n ones, we get

$$g(i,n) = i(1)^n \quad .$$

Thus

$$\lambda i[g(i,n)]$$

is a postfix function for every n and therefore

$$\phi^2[i,n,x] = \phi[g(n,i),x] = \phi_{g(n,i)}(x)$$

is a postfix GN.

Straightforward constructions show that

$$\text{GNPstfx} \subset \text{GNReg} \subset \text{GNLBA} \quad .$$

The assertion

$$\text{GNLBA} \subset \text{GNOpt}$$

follows by combining the facts that:

a) in every computational complexity measure Φ the linearly bounded automata mappings are of bounded complexity;

b) (proven later in this paper) in any complexity measure Φ the complexity of translations from GN's into optimal GN's cannot be recursively bounded.

Thus GNLBA $\neq$ GNOpt.

c) every ℓba mapping is linearly size bounded.

Thus GNLBA $\subset$ GNOpt, which completes the proof.

Lemma: For every computational complexity measure Φ there exist infinitely many different complexity classes of GN's GNC_t^{Φ}.

Proof: By a diagonal argument we can construct for every ϕ a GN ψ into which ϕ cannot be mapped by any σ in C_t^{Φ}.

In [2] it was shown that ϕ is in GNOpt iff ϕ admitted an S_1^1 function which is linearly size bounded in the second variable. A similar proof shows our next result.

Lemma: The GN ϕ is in GNPrfx, GNPstfx, GNReg and GNLBA iff ϕ admits an S_1^1 function which is in the second variable a prefix, postfix, gsm or linearly bounded automaton mapping, respectively.

Proof: We give the proof for ϕ in GNReg.

Let ϕ be a GN and S_1^1 such that for all n,i,x

$$\phi_n^2(i,x) = \phi_{S_1^1(n,i)}(x) \quad ,$$

and assume that for all n $S_1^1(n,i)$ is a gsm mapping. Then for any other GN there exists an n_0 such that the numbering is given by

$$\phi_{n_0}^2(i,x) \quad .$$

But then

$$\phi_{S_1^1(n_0,i)}(x) = \phi^2_{n_0}(i,x)$$

and we see that

$$\sigma(i) = \lambda i[S_1^1(n_0,i)]$$

is the desired gsm translation.

Conversely, if ϕ is in GNReg and $\overline{\phi}$ is the GN of the previous theorem, then there exists a gsm mapping σ such that

$$\phi_{\sigma\circ g(n,i)}(x) = \overline{\phi}_{g(n,i)}(x) = \phi^2(i,n,x) \quad .$$

Since g is a postfix translation

$$\sigma\circ g(n,i)$$

is a gsm mapping. Thus ϕ admits

$$S_1^1(n,i) = \sigma\circ g(n,i)$$

as an S_1^1 function which is a gsm mapping in the second variable. The other cases follow by an identical argument.

Next we show that those GN's into which all others can be easily translated have easily computable Recursion Theorem fixed points.

We recall that by the Recursion Theorem [1] for every GN ϕ there exists a recursive function n such that for all z

$$\phi_{n(z)} = \phi_{\phi_z[n(z)]} \quad .$$

<u>Lemma</u>: If ϕ is in GNPrfx, GNPstfx, GNReg or GNLBA, then there

exists a prefix, postfix, gsm or ℓba mapping n, respectively, such that for every z

$$\phi_{n(z)} = \phi_{\phi_z}[n(z)] \ .$$

<u>Proof</u>: The proof follows the standard proof of the Recursion Theorem [1]. Define

$$\psi(u,x) = \phi_{\phi_u(u)}(x) \ \underline{\text{if}} \ \phi_u(u) \text{ converges } \underline{\text{else}} \text{ divergent} \ .$$

Let g be a recursive function (translation) such that

$$\psi(u,x) = \phi^1(g(u),x) = \phi_{g(u)}(x) \ .$$

Thus for ϕ_u total we have

$$\phi_{\phi_u(u)} = \phi_{g(u)} \ .$$

Define

$$\mu(z,x) = \phi_z \circ g(x) \ \underline{\text{if}} \ \phi_z \circ g(x) \text{ converges } \underline{\text{else}} \text{ divergent}$$

and let h be the recursive function (translation) such that

$$\mu(z,x) = \phi^1(h(z),x) = \phi_{h(z)}(x) \ .$$

Thus for ϕ_z total we have

$$\phi_{h(z)} = \phi_z \circ g \ .$$

By combining these equalitites we get for all z such that ϕ_z total

$$\phi_{g \circ h(z)} = \phi_{\phi_{h(z)}}[h(z)] = \phi_{\phi_z}[g \circ h(z)] \ .$$

Thus by setting

$$g \circ h(z) = n(z)$$

we get that

$$\phi_{\phi_z}[n(z)] = \phi_{n(z)} \quad .$$

Furthermore, since g and h are translations we can choose them to be prefix, postfix, regular, or ℓba mappings, respectively, and therefore

$$n = g \circ h$$

will be a mapping of the same type, as was to be shown.

Schnorr has shown that the optimal GN's are all isomorphic under linearly size bounded mappings. Thus, in a mathematical sense, they form a natural class of GN's.

For the prefix and postfix GN's we know that they cannot be isomorphic under prefix and postfix mappings, since these mappings are (but for the trivial case) proper into mappings. On the other hand, our next result shows that they are closed under gsm mappings. Thus they form classes of GN's which are very similar in a computational sense.

<u>Theorem</u>: Let ϕ and $\overline{\phi}$ be in GNPrfx or GNPstfx. Then there exists a permutation π such that π and π^{-1} are gsm mappings and

$$\phi_i = \overline{\phi}_{\pi(i)} \text{ and } \overline{\phi}_i = \phi_{\pi^{-1}(i)} \quad .$$

<u>Proof</u>: We give the proof for GNPrfx. Let ω and v be the prefix sequences which translate ϕ into $\overline{\phi}$ and $\overline{\phi}$ into ϕ, respectively. Then

$$1(1+0)^* = \{(\omega v)^k z \mid k=0,1,2,\ldots \text{ and } z \notin \omega(0+1)^*\} \cup$$
$$\{\omega(v\omega)^k z \mid k=0,1,2,\ldots \text{ and } z \notin v(0+1)^*\}$$
$$= \{v(\omega v)^k z \mid k=0,1,2,\ldots \text{ and } z \notin \omega(0+1)^*\} \cup$$
$$\{(v\omega)^k z \mid k=0,1,2,\ldots \text{ and } z \notin v(0+1)^*\} .$$

Define

$$\pi[(\omega v)^k z] = v(\omega v)^k z, \text{ for } k=0,1,2,\ldots \text{ and } z \notin \omega(0+1)^*$$
$$\pi[\omega(v\,\omega)^k z] = (v\omega)^k z, \text{ for } k=0,1,2,\ldots \text{ and } z \notin v(0+1)^* \quad .$$

We see that π is a permutation of the set $1(0+1)^*$ and since

$$\phi_i = \overline{\phi}_{\omega i} \text{ and } \overline{\phi}_j = \phi_{vj},$$

we see that for $j = (\omega v)^k z$

$$\overline{\phi}_j = \phi_{vj} = \phi_{\pi(j)}$$

and for $j = \omega(v\omega)^k z$

$$\phi_{(v\omega)^k z} = \overline{\phi}_{\omega(v\omega)^k z} \quad ,$$

and therefore

$$\overline{\phi}_j = \phi_{(v\omega)^k z} = \phi_{\pi(j)} \quad .$$

Finally we note that a finite automaton can perform the permutation π. The arguments for π^{-1} can be carried out similarly, which completes the proof.

It should be noted that in the previous proof the finite automaton computing π either prefixed the sequence v or removed the sequence ω. Let us call such mappings <u>restricted regular</u>

mappings and let us call the GN's into which all other GN's can be mapped by such mappings restricted regular GN's. We conjecture that the restricted regular GN's are all isomorphic under restricted regular mappings. Unfortunately, so far we have not been able to prove this conjecture.

We also conjecture that regular GN's are not closed under finite automata isomorphisms. Again, we have not been able to prove this simple sounding conjecture.

We can prove though that the complexity of the isomorphisms between GN's in GNReg or GNLBA are of bounded computational complexity and we will see that this is not true for GNOpt.

To do this let Φ be any computational complexity measure and let

$$GNC_t^\Phi$$

denote all the GN's into which all other GN's can be mapped by mappings in complexity class C_t^Φ.

Theorem: For any computational complexity measure Φ and t in R_1 there exists a t' in R_1 such that the isomorphisms between GN's in GNC_t^Φ can be chosen from $C_{t'}^\Phi$.

Proof: Follows from a close inspection of the proof of Rogers' Isomorphism Theorem of GN's.

Furthermore, we can see that Rogers' Isomorphism Theorem yields a recursive operator which maps the two one-to-one- into mappings of this theorem into an isomorphism. But then the complexity of the isomorphism is given by a recursive operator in terms of the complexity of the two mappings used.

This observation permits us to read off another result (due partially to K. Mehlhorn).

<u>Corollary</u>: For any computational complexity measure Φ there exist arbitrarily large t in R_1 such that any two GN's in

$$GNC_t^\Phi$$

are isomorphic under a permutation in C_t^Φ.

<u>Proof</u>: Let $\mathscr{R}$ be the recursive operator yielded by the Isomorphism Theorem. Then there exists a recursive operator which bounds the complexity of the resulting isomorphism in terms of the complexity of the two into mappings, denote it by $\mathscr{R}'$. Then from the Operator Gap Theorem [4] we know that there exists arbitrarily large recursive t such that

$$C_t^\Phi = C_{\mathscr{R}'[t]}^\Phi \quad .$$

Thus GNC_t^Φ is closed under isomorphisms in C_t^Φ, as was to be shown.

It should be stated again that it would be interesting to determine some <u>natural</u> classes of into translatable Goedel numberings which are closed under isomorphism in the same complexity class as the into translations. For example, we conjecture that

$$GNLBA$$

is closed under ℓba isomorphisms.

We describe one reasonably natural class of Goedel numberings which is closed under isomorphisms of the same type.

Let GNPTIME denote the class of GN's into which all others can be translated by deterministic Turing machines whose computation times are bounded by a polynomial in the size of the input (index).

The following result is due to R. Constable.

Theorem: The class GNPTIME is closed under polynomial-time bounded Turing machine computable isomorphisms.

Proof: By a lengthy and careful estimation of runtimes in the proof of the Isomorphism Theorem [1].

A somewhat more natural (and better known) class of GN's would be the class into which all other GN's can be translated by Turing machines whose computation times are bounded by a polynomial in the length of the input (index). Unfortunately, the previous proof does not show that this class is closed under isomorphisms in the same complexity classes. We conjecture that the answer is positive.

We conclude this section by showing that for any GN ϕ^1 we can give a recursive function which bounds the complexity of translations from all other GN's into ϕ^1 .

Theorem: Let Φ be any computational complexity measure and ϕ^1 fixed GN. Then we can recursively obtain indices for two recursive functions s and ℓ such that for any GN ϕ there exists a translation σ: $\phi \to \phi^1$ such that

$$\sigma \text{ is in } C_s^{\Phi} \quad \text{and} \quad |\sigma(i)| \leq |\ell(i)| \text{ a.e.}$$

Proof: Let ψ be a prefix GN and let σ_0 be a translation mapping ψ into ϕ^1. Then for any GN ϕ there exists a sequence ω such that

$$\sigma(i) = \omega i$$

is a translation of ϕ into ψ. But then

$$\sigma_0 \circ \sigma(i) = \sigma_0(\omega i)$$

is a translation of ψ into ϕ^1. To obtain the functions $\mathfrak{s}$ and ℓ let $\Sigma_0(\omega i)$ be the complexity of the computation $\sigma_0(\omega i)$ and define

$$\mathfrak{s}(i) = \max_{|\omega| \le |i|} \{ \Sigma_0(\omega i) \} \quad ,$$

similarly,

$$\ell(i) = \max_{|\omega| \le |i|} \{ |\sigma_0(\omega i)| \} \quad .$$

Clearly, for every ϕ there exists a translation $\sigma\colon\ \phi \to \phi^1$ such that

$$\sigma \text{ is in } C_{\mathfrak{s}}^{\Phi}$$

and

$$|\sigma(i)| \le |\ell(i)| \ , \quad \text{a.e.},$$

as was to be shown.

3. COMPLEXITY OF OPTIMAL GOEDEL NUMBERINGS

In this section we show that the computational complexity of translations into optimal Goedel numberings cannot be recursively bounded. Actually we will show that for any GN ϕ there exist optimal GN's whose translations into ϕ are arbitrarily complex. Similarly, we will show that for optimal GN's the computational complexity of the S_1^1 function and the function n of the Recursion Theorem cannot be recursively bounded.

Theorem: For any computational complexity measure (ϕ,Φ) and recursive function t there exists an optimal GN ψ into which ϕ cannot be translated by any

$$\sigma \text{ in } C_t^\Phi \quad .$$

Proof: Let χ be an optimal GN. We will obtain ψ from χ by a permutation which will not move any index upward by more than one place. Therefore ψ will also be an optimal GN. The construction of ψ is obtained by diagonalizing over all possible t-bounded σ.

We know that for sufficiently large recursive T the complexity class C_T^Φ can be recursively enumerated. Thus we can assume that

$$\phi_{k_1}, \phi_{k_2}, \phi_{k_3}, \ldots$$

is an enumeration of the functions in C_t^Φ, or choose a larger complexity bound T with an enumerable complexity class.

Let

$$\phi_{j_1}, \phi_{j_2}, \phi_{j_3}, \ldots$$

be a recursive sequence of constant functions such that

$$\phi_{j_k}(x) = k \quad .$$

We now define the stages in the computation of ψ.

Stage 1: let $\psi_1 = \chi_1$.

Stage i: By the i-th stage let

$$\psi_1, \psi_2, \ldots, \psi_{N_i}$$

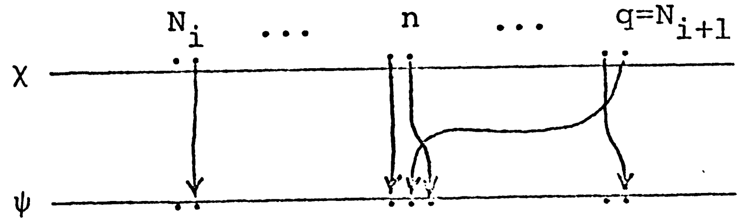
be defined. We define ψ_j, $N_i < j \leq N_{i+1}$ as follows: compute $\phi_{k_i}(j)$ dovetailed for $j = N_i+1, N_i+2, \ldots,$ until

a) $\phi_{k_i}(j) > N_i$ and $\phi_j(1)$ converges, or

b) $\phi_{k_i}(j_\ell) \leq N_i$ for more than N_i distinct values of ℓ (where j_ℓ is from the enumeration of $\phi_{j_1}, \phi_{j_2}, \ldots$) .

This computation is eventually halted by (a) or (b). In case (a) let $n = \phi_{k_i}(j)$. Let q be the first p greater than n for which $\chi_p(1) \neq \phi_j(1)$ gotten by dovetailing the computation of $\chi_p(1)$ for $p = j+1, j+2, \ldots$. Let $N_{i+1} = q$. We want $\psi_n \neq \phi_j$ so we define $\psi_p = \chi_p$ for $p < n$, $\psi_n = \chi_q$, and $\psi_p = \chi_{p-1}$ for $n < p \leq q$.

e.g.



In case (b) let

$$N_{i+1} = N_i+1 \text{ and } \psi_{N_i+1} = \chi_{N_i+1}$$

and proceed to stage i+1.

If ϕk_i translates ϕ into ψ then we would have

$$\phi_p = \psi_{\phi_{k_i}(p)}$$

for all p. This is not the case if the computation was halted by (a), because of the definition of ψ. In case (b) at least two different j_ℓ, j_m exist such that

$$\phi_{k_i}(j_\ell) = \phi_{k_i}(j_m) \quad ,$$

but then

$$\psi_{\phi_{k_i}(j_\ell)} = \psi_{\phi_{k_i}(j_m)} \text{ and } \phi_{j_\ell} \neq \phi_{j_m} \quad ,$$

an inconsistency. Thus we see that no ϕ_k from C_t^Φ can translate ϕ into the optimal GN ψ, as was to be shown.

Next we show that optimal GN's require computationally arbitrarily complex S_1^1 functions.

<u>Theorem</u>: For every computational complexity measure Φ and recursive function t there exists optimal GN ϕ such that for no S_1^1 function is

$$\lambda i[S_1^1(n,i)] \text{ in } C_t^\Phi$$

for all n.

<u>Proof</u>: Using the previous result we construct two optimal GN's ϕ and $\overline{\phi}$ such that $\overline{\phi}$ cannot be translated into ϕ by a mapping in C_t^Φ. Let S_1^1 be defined for ϕ. Then for some n_0

$$\phi^2(n_0,i,x) = \overline{\phi}(i,x)$$

and therefore

$$\phi^2(n_0,i,x) = \phi_{S_1^1(n_0,i)}(x) = \overline{\Phi}_i(x) \quad .$$

But then

$$\sigma(i) = \lambda i[S_1^1(n_0,i)]$$

is a translation of $\overline{\phi}$ into ϕ and therefore

$$\lambda i[S_1^1(n_0,i)] \text{ not in } C_t^{\Phi},$$

as was to be shown.

Similarly, we show that optimal GN's require arbitrarily complex functions satisfying the Recursion Theorem.

Theorem: For any recursive function t and complexity measure (ϕ,Φ) there is an optimal Goedel numbering ψ such that every function ϕ_k satisfying the recursion theorem condition: for all j

$$\psi_{\phi_k(j)} = \psi_{\psi_j(\phi_k(j))}$$

is of complexity greater than t.

Proof: We define the desired ψ inductively, diagonalizing over all the possible t-bounded ϕ_k.

Without loss of generality we may assume that ϕ is an optimal Goedel numbering. If it is not, we may construct another complexity measure (ϕ',Φ) with the same complexity classes such that ϕ' is an optimal Goedel numbering: start with an optimal Goedel numbering χ, we know there are infinitely many of these; let $\phi_j = \chi_{\sigma(j)}$ for every j, where σ is a recursive isomorphism from ϕ to χ.

Let $\phi_{j1}, \phi_{j2}, \ldots$ be a recursive subsequence of ϕ consisting of the constant functions

$$\phi_{j_i}(x) = \lambda(x)(i), \quad j_i > i + 3 \quad .$$

ϕ must have such a recursive subsequence by the S_n^m theorem.

Let

$$p(n) = \min \{ j_{j_c} \mid c > n \} \text{ and}$$
$$q(n) = \min \{ j_{j_d} \mid d > p(n) \}.$$

Note that these are both recursive functions and that they are indices in ϕ of constant functions which compute indices in ϕ of other constant functions. For any n

$$n < \phi_{\phi_{p(n)}} < \phi_{p(n)} < p(n) < \phi_{\phi_{q(n)}} < \phi_{q(n)} < q(n) \quad .$$

Since for sufficiently large t the functions of complexity t can be recursively enumerated, we assume without any loss of generality that $\phi_{k_1}, \phi_{k_2}, \ldots$ is a recursive enumeration of the functions of complexity t.

ψ will be defined as a recursive permutation of ϕ in which no index is increased by more than three, so there will be no question about ψ being an optimal Goedel numbering. We assume that $\psi_1, \psi_2, \ldots, \psi_{N_i}$ are defined by the i-th stage, and proceed to extend the definition to $\psi_{N_i+1}, \ldots, \psi_{N_{i+1}}$. Let $k = k_i$ and $N = N_i$. Compute

$$\phi_k(j_\ell) \quad \text{and} \quad \phi_{j_\ell}(\phi_k(j_\ell)) \quad \text{for} \quad \ell = N+1, N+2, \ldots$$

until one of the following cases holds. In each case ψ is defined for certain critical indices so that for some j

$$\psi_{\phi_k(j)} \neq \psi_{\psi_j(\phi_k(j))} .$$

Case 1: If $\phi_k(j_m) = \phi_k(j_\ell)$, $m > N$, and $\ell > j_m + 3$, let $\psi_{j_\ell} = \phi_{j_\ell}$, $\psi_\ell = \phi_{q(j_\ell)}$, $\psi_m = \phi_{p(j_m)}$, $\psi_{j_m} = \phi_{j_m}$, and $N_{i+1} = q(j_\ell)$. Then $\psi_{\psi_{j_\ell}(\phi_k(j_\ell))} = \psi_\ell = \phi_{q(j_\ell)} \neq \phi_{p(j_\ell)} = \psi_m = \psi_{\psi_{j_m}(\phi_k(j_m))}$.

Case 2: If $\phi_k(j_\ell) = j_\ell$, let $\psi_{j_\ell} = \phi_{p(j_\ell)}$, $\psi_{\phi_{p(j_\ell)}} = \phi_{\phi_{p(j_\ell)}}$ and $N_{i+1} = q(j_\ell)$. Then $\psi_{\psi_{j_\ell}(\phi_k(j_\ell))} = \psi_{\phi_{p(j_\ell)}} = \phi_{\phi_{p(j_\ell)}} \neq \phi_{p(j_\ell)}$ $= \psi_{j_\ell} = \psi_{\phi_k(j_\ell)}$.

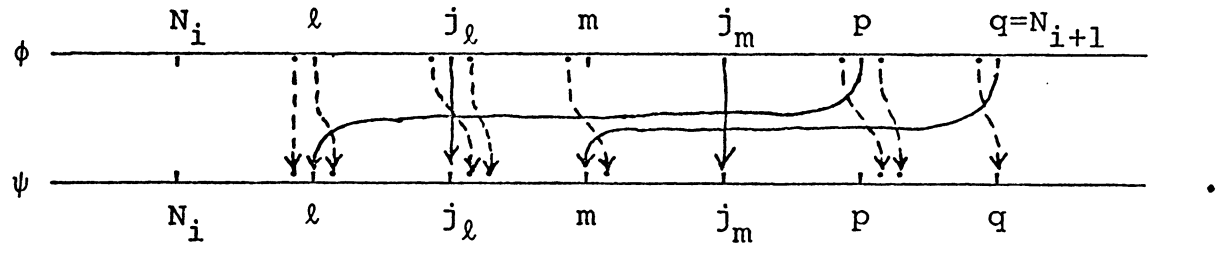
Case 3: If $\phi_k(j_\ell) = \ell$, let $\psi_\ell = \phi_{q(j_\ell)}$, $\psi_{j_\ell} = \phi_{p(j_\ell)}$, $\psi_{\phi_{p(j_\ell)}} = \phi_{\phi_{p(j_\ell)}}$, and $N_{i+1} = q(j_\ell)$. Then $\psi_{\psi_{j_\ell}(\phi_k(j_\ell))} = \psi_{\phi_{p(j_\ell)}}$ $= \phi_{\phi_{p(j_\ell)}} \neq \phi_{q(j_\ell)} = \psi_\ell = \psi_{\phi_k(j_\ell)}$.

Case 4: If $\phi_k(j_\ell) \neq \ell, \neq j_\ell$, $> N$, and $m = \max (j_\ell, \phi_k(j_\ell))$, let $\psi_{j_\ell} = \phi_{j_\ell}$, $\psi_\ell = \phi_{p(m)}$, $\psi_{\phi_k(j\ )} = \phi_{q(m)}$, $N_{i+1} = q(m)$. Then $\psi_{\psi_{j_\ell}(\phi_k(j_\ell))} = \psi_\ell = \phi_{p(m)} \neq \phi_{q(m)} = \psi_{\phi_k(j_\ell)}$.

Furthermore, ψ_j is defined for all other j $(N_i < j \leq N_{i+1})$ to be ϕ_j, ϕ_{j-1}, ϕ_{j-2}, or ϕ_{j-3}, shifting the indices as little as possible; i.e., <u>for</u> $j := N_i+1$ <u>until</u> N_{i+1} <u>do</u>

```
            if ($\psi_j$ not yet defined)
            then $\psi_j := \phi_{\min \{i \mid \phi_i \text{ not yet used to define any } \psi_k\}}$ .
```

For example, in case (1) we have:



To see that this computation must halt , suppose (1), (2), and (3) fail for every $\ell > N_i$. It follows that (4) succeeds for large enough ℓ, since:

$$\phi_k(j_\ell) \neq \ell \;, \quad \text{by } \neg\,(3)\;;$$
$$\phi_k(j_\ell) \neq j_\ell, \quad \text{by } \neg\,(2)\;;$$
$$\phi_k(j_\ell) > N_i, \quad \text{because } (\phi_k(j_\ell) \leq N \text{ for a.e.}\,\ell) \Rightarrow (1).$$

Thus for no ϕ_k in C_t^Φ can we have that

$$\psi_{\phi_k(j)} = \psi_{\psi_j}[\phi_k(j)] \quad ,$$

as was to be shown.

4. REFERENCES

1. Rogers, H. Jr., Theory of Recursive Functions and Effective Computability. McGraw-Hill Book Co., New York, 1967.

2. Schnorr, C.P., "Optimal Enumerations and Optimal Goedel Numberings." To appear in Math. Syst. Theory.

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

4. Constable, R.L., "The Operator Gap," IEEE Conference Record of 1969 Tenth Annual Symposium on Switching and Automata Theory, (1969) 20-26.