5-1965

Decision Problems

Lowell Anderson

Utah State University

DECISION PROBLEMS

by

Lowell Anderson

Report No. 1 submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Mathematics

Plan B

UTAH STATE UNIVERSITY
Logan, Utah

1965

ACKNOWLEDGMENT

I wish to express my appreciation to Doctor Konrad Suprunowicz for serving as my major professor while preparing this report.

Lowell Anderson

TABLE OF CONTENTS

INTRODUCTION

From an intuitive point of view the notion of effective procedure consists of a set of rules or instructions that enables one, in a finite number of steps and in a purely mechanical way, to answer yes or no to any one of a given class of questions. This procedure requires no intelligence to carry out the instructions and, in fact, it is conceivable that some mechanical contrivance may be constructed to carry out these instructions. Should such an effective procedure exist, that answers either yes or no, then the group of problems in question is said to be effectively decidable; otherwise not decidable.

In this paper, some of the more important properties of the propositional and predicate calculi will be established with the thought in mind of considering the notion of effective procedure relative to these properties. In achieving this end, the propositional and predicate calculi will be considered in a purely formal context. Formal in the sense that on the outset the symbols employed within the theories will be devoid of any interpretation. Later, however, an interpretation will be placed on these symbols in order to answer certain questions concerning decidability.

In considering the propositional and predicate calculi as formal theories, a distinction must be drawn between those symbols used in the particular theory and the language used to describe this theory. The former use will be referred to as the object language and the latter as the syntax or metalanguage.

The object language, for the formal theory under consideration, will be given explicitely, whereas the metalanguage will consist of, only that portion of the English language needed to clearly describe the formal system.

In some instances, since no confusion will result, certain symbols may appear not only in the object language but also in the metalanguage. This will be evident by the two-fold use of the symbols: $\sim$, $\rightarrow$, (,), and '. Should specific reference be made to some one of the symbols of the theory, this symbol will be enclosed in single quotes.

Furthermore, the reasoning employed in establishing results about the formal systems will consist only of those notions which have great intuitive appeal. Included among these will be mathematical induction.

THE PROPOSITIONAL CALCULUS

Primitive Basis

The primitive symbols of the propositional calculus are the symbols;

$$(\quad) \quad \rightarrow \quad \sim$$

which will be called improper symbols, together with an at most denumerable list of proper symbols;

$$p \quad q \quad r \quad p_1 \quad q_1 \quad r_1 \quad p_2 \quad q_2 \quad r_2 \quad \cdot \quad \cdot \quad \cdot$$

called statement variables. The three dots are used to indicate the list continues indefinitely and are not part of the primitive symbols of the theory.

The formation rules of the propositional calculus are given by the following recursive definition:

I. If P is a statement variable then (P) is a formula.

II. If A is a formula then (∼A) is a formula.

III. If A and B are formulas then (A → B) is a formula.

IV. Only finite strings of primitive symbols which follow I-III are formulas.

Here it is to be noted that the symbols A and B are in the metalanguage and are not part of the apparatus of the propositional calculus.

The above formation rules enable formulas of the propositional calculus to be constructed from previously constructed formulas, however, the question might be asked whether a given finite string of primitive symbols constitutes a formula or not. It would be desirable if this question could be answered effectively and as will be seen, this is indeed the case. However, certain conventions and results must first be developed before this can be done.

Adopt the convention that if A and B are finite strings of primitive symbols then A and B in juxtraposition, written AB, will denote the string of primitive symbols formed by writing the primitive symbols of A, in order, followed by the primitive symbols of B in order.

D1. Let A be a finite string of primitive symbols and let f(A) denote the total number of occurrences of the symbols '(' and ')' appearing in A. Call an occurrence of '(' or ')' in A a π occurrence in A.

D2. Let A be a finite string of primitive symbols. Since A is of finite length, the π occurrences in A may be ordered, left to right, from 1 to f(A). Order them and make the following definitions:

1. $\sum_{i} A = \sum_{k=1}^{i} (-1)^{\Theta(k)}$, where $\Theta(k) = 1$ or 2, according as the k-th π occurrence in A is ')' or '(', respectively.

2. $\Sigma A = \sum_{f(A)} A$.

3. $\sum_{i} A = \Sigma A$ for all $i \geq f(A)$.

4. $\sum_{i} A = 0$ for all $i \leq 0$.

MT1. For finite strings A and B;

$$\sum_{i} A \to B = \sum_{i} A \text{ and } \sum_{i} AB = \sum_{i} A \text{ for all } i,\ i \leq f(A).$$

$$\sum_{i} A \to B = \sum_{f(A)} A + \sum_{j} B \text{ and } \sum_{i} AB = \sum_{f(A)} A + \sum_{j} B \text{ where } i = f(A) + j.$$

$$\Sigma A{\to}B = \Sigma A + \Sigma B \text{ and } \Sigma AB = \Sigma A + \Sigma B.$$

The above assertions are obvious.

MT2. Let A be a formula then $\Sigma A = 0$. Hence, the number of occurrences of '(' and ')' in A are equal, namely f(A)/2 each.

Proof: Follows directly from the formation rules I-IV.

MT3. Let A be a formula, then $\sum_i A > 0$ for all i, $1 \leq i < f(A)$. In fact, $\sum_i A \geq 1$ if $1 \leq i < f(A)$.

Proof: In view of MT2 and I-IV, the π occurrences in a formula must be even and greater than zero. Let A_{2n} be any formula such that $f(A_{2n}) = 2n$. The proof will be by induction on n.

Basis. If n = 1, then A_2 must be of the form (P) for some statement variable P and clearly $\sum_1 A_2 = 1 > 0$.

Induction step. Suppose the assertion is true for n = k and consider $A_{2(k+1)}$. From II-IV and the Basis, $A_{2(k+1)}$ must have one of two forms.

Case 1. If $A_{2(k+1)}$ is of the form (∼B) for a formula B, then $\sum_i A_{2(k+1)} = \sum_i (\sim B) = 1 + \sum_{i-1} B$ and by induction hypothesis, MT2 and D2, $\sum_{i-1} B \geq 0$ for $0 \leq i-1 \leq f(B)$. Hence $\sum_i A_{2(k+1)} > 0$ for $1 \leq i < f(A)$ since $f(B) + 2 = f(A)$.

Case 2. If $A_{2(k+1)}$ is of the form (C → D) for formulas C and D, then $\sum_i A_{2(k+1)} = \sum_i (C \rightarrow D) = 1 + \sum_{i-1} C$ if $i-1 \leq f(C)$ or, $= 1 + \sum_j D$ if $i-1 = f(C) + j$. But by induction hypothesis, MT2 and D2, $\sum_{i-1} C \geq 0$ if $0 \leq i-1 \leq f(C)$ and $\sum_j D \geq 0$ if $0 \leq j \leq f(D)$, hence $\sum_i A_{2(k+1)} > 0$ for all i, $1 \leq i < f(A)$ since $f(C) + 1 < f(A)$ and $f(C) + j + 1 \leq f(C) + f(D) + 1 = f(A) - 1 < f(A)$. Since $\sum_i A_{2(k+1)} > 0$ clearly $\sum_i A_{2(k+1)} \geq 1$.

This proves the theorem since A must be some one of the A_{2n}'s.

D3. Let A be a finite string of primitive symbols and suppose there exist nonempty strings B and C such that $f(B) \neq 0$, $f(C) \neq 0$, $\Sigma B = \Sigma C = 0$ and A can be written as $(B \rightarrow C)$. The occurrence of '$\rightarrow$' between B and C will be called a major occurrence of '$\rightarrow$' in A and this will be referred to as a μ occurrence in A. The notation, '$\underset{\mu}{\rightarrow}$', will be employed to locate that occurrence of '$\rightarrow$' in a string which is a μ occurrence. Thus, $(B \underset{\mu}{\rightarrow} C)$ will be interpreted as meaning: "The occurrence of '$\rightarrow$' between B and C is a μ occurrence in $(B \rightarrow C)$."

MT4. If A is a finite string of primitive symbols with a μ occurrence then there exists an integer i, $1 < i < f(A) - 1$ such that $\sum_i A = 1$.

Proof: By D3, there exist nonempty strings B and C such that $f(B) \neq 0$, $f(C) \neq 0$, $\Sigma B = \Sigma C = 0$ and A is $(B \rightarrow C)$, where $(B \underset{\mu}{\rightarrow} C)$. Now $f(A) = f(B) + f(C) + 2$. Choose $i = f(B) + 1$, then $\sum_{f(B)+1} A = \sum_{f(B)+1} (B \rightarrow C) = 1 + \sum_{f(B)} B = 1$. Since $f(B) \neq 0$, clearly $1 < i < f(A) - 1$.

MT5. A is a formula with no μ occurrence if and only if A is of the form (P) or $(\sim B)$; P a statement variable and B a formula.

Proof: Suppose A is a formula with no μ occurrence, then from D3 and I-IV, A must be of the form (P) or $(\sim B)$ since a formula of the form $(C \rightarrow D)$ has a μ occurrence, namely $(C \underset{\mu}{\rightarrow} D)$.

Conversely, suppose A is of the form (P) or $(\sim B)$. Clearly (P) has no μ occurrence since P is a statement variable. Hence, suppose A is $(\sim B)$. By MT3, $\sum_i A \geq 1$ for all i, $1 \leq i < f(A)$. But $\sum_i A = \sum_i (\sim B) = 1 + \sum_{i-1} B$, and when $1 \leq i-1 < f(B)$, $\sum_{i-1} B \geq 1$ hence, $\sum_i A > 1$ for all i, $1 < i < f(A) - 1$.

Since $\Sigma A = 0$, then for $\sum_i A = 1$, $i = f(A) - 1$; so by MT4, A cannot have a μ occurrence.

<u>MT6</u>. A formula A has a μ occurrence if and only if it is of the form $(B \rightarrow C)$ for formulas B and C. In fact $(B \underset{\mu}{\rightarrow} C)$.

Proof: This follows immediately from A being a formula, I-IV and MT5.

<u>MT7</u>. If A is a formula of the form $(B \rightarrow C)$, B and C formulas, then $(B \underset{\mu}{\rightarrow} C)$. Furthermore, this is the only μ occurrence in A.

Proof: The first assertion is obvious so suppose there exist nonempty strings D and E such that $f(D) \neq 0$, $f(D) \neq 0$, $\Sigma D = \Sigma E = 0$ and A can be written as $(D \rightarrow E)$. Suppose A has n occurrences of '$\rightarrow$' and let $(B \underset{\mu}{\rightarrow} C)$ and $(D \underset{\mu}{\rightarrow} E)$ be the j-th and k-th occurrences of '$\rightarrow$' in A, respectively. If $j = k$, then clearly B and D are the same as are C and E. Therefore, suppose $k < j$, then A can be written as $(D \rightarrow C_1 \rightarrow C)$ for some string C_1. Hence, B is $D \rightarrow C_1$ and $f(B) = f(D) + f(C_1) \geq f(D)$. By MT3, $\sum_i B > 0$ for all i, $1 \leq i < f(B)$ and since $\Sigma D = 0$, then by MT1 $\sum_{f(D)} B = \sum_{f(D)} D \rightarrow C_1 = \sum_{f(D)} D = 0$. Therefore, $f(B) = f(D)$ whence, $f(C_1) = 0$. But this is impossible since B is a formula and must end in ')'.

When $j < k$, a similar argument will show that the formula C cannot end with ')'. Therefore, the only alternative is for $j = k$.

Combining MT5-MT7, then if a formula A has a μ occurrence it may be referred to as <u>the μ occurrence</u> in A.

<u>MT8</u>. Let A be a formula, then A has a μ occurrence if and only if there exists an integer i, $1 < i < f(A) - 1$, such that $\sum_i A = 1$. Furthermore, the μ occurrence will be immediately preceded by the i-th π occurrence in A.

Proof: Suppose A has a μ occurrence, then by MT6, A is of the form $(B \rightarrow C)$ where B and C are formulas and $(B \underset{\mu}{\rightarrow} C)$. By MT7 this is the only μ occurrence. Choose $i = f(B) + 1$, then clearly $\sum_i A = 1$ and the μ occurrence is preceded by the i-th π occurrence.

Conversely, suppose A is a formula and suppose there exists an integer i, $1 < i < f(A) - 1$ such that $\sum_i A = 1$.

Since A is a formula, then by I-IV, A is of the form (P) for a statement variable P, of the form $(\sim B)$ for a formula B or of the form $(C \rightarrow D)$ for formulas C and D. Clearly A cannot be (P) and if A is $(\sim B)$, then by the proof of MT5, $\sum_i A = 1$ only for $i = f(A) - 1$. Hence, A must be $(C \rightarrow D)$ and by MT6-7, $(C \underset{\mu}{\rightarrow} D)$ which is the only one. Suppose $1 < j < f(A) - 1$, then $\sum_j A = \sum_j (C \rightarrow D) = 1 + \sum_{j-1} C$ if $1 \leq j - 1 \leq f(C)$, or $\sum_j A = 1 + \sum_{f(C)} C + \sum_k D = 1 + \sum_k D$ if $j-1 = f(C) + k$. Now $f(A) = f(C) + f(D) + 2$ and since $j < f(A) - 1$, then $k < f(D)$. If $1 \leq j - 1 < f(C)$ then by MT3 $\sum_{j-1} C \geq 1$ whence, $\sum_j A > 1$ and if $j - 1 = f(C) + k$, then $\sum_k D \geq 1$ and again $\sum_j A > 1$. Therefore, $j = f(C) + 1$ is the only integer with $1 < j < f(A) - 1$ such that $\sum_j A = 1$. Hence, $i = j$ and clearly the μ occurrence is preceded by the i-th π occurrence.

The preceding results will be used along with induction to establish that the notion of formula is effective. In view of I-IV, only finite strings beginning with '(' and ending in ')' need be considered.

Let A_n be a finite string of primitive symbols with n occurrences of '$\rightarrow$' or '$\sim$' and such that A_n begins with '(' and ends in ')'.

Basis. If $n = 0$ then A_0 must be of the form (P) for some statement variable P, which is clearly effective.

Induction step. Suppose the assertion is true for all occurrences less than n and consider A_n. In view of the preceding there are only two cases to consider.

Case 1. $\sum_i A_n \neq 1$ for all i, $1 < i < f(A_n) - 1$. By MT4 and MT5, A_n must be of the form (P) for a statement variable P, which is effective, or of the form ($\sim$B) for some string B. But B has n - 1 occurrence, hence, by induction hypothesis, B is effective. Thus, by II, A_n is effective.

Case 2. $\sum_i A_n = 1$ for some i, $1 < i < f(A_n)-1$. Now from the preceding results, A_n must have a μ occurrence, whence A_n is of the form (B → C). But B and C have less than n occurrences and therefore, by induction hypothesis they are effective; so by III, A_n is effective.

This establishes the following result.

Metatheorem 1.1. The notion of formula is effective.

By MT7, there is an additional result, that of

Metatheorem 1.2. A formula of the form (A → B) can be written in one and only one way.

The axioms for the propositional calculus are given by the following:

PC1. $A \to (B \to A)$

PC2. $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$

PC3. $(\sim A \to \sim B) \to (B \to A)$

where A, B and C are arbitrary formulas.

More correctly, the above should be referred to as axiom schema in as much as A, B and C are arbitrary formulas and hence, give rise to an infinite number of axioms in the propositional calculus. For example, as a special case of PC1, not only is

$$p \to (q \to p)$$

an axiom, but also

$$(p \rightarrow q) \rightarrow ((\sim r \rightarrow q_1) \rightarrow (p \rightarrow q)).$$

In the strictest sense of the word, it is evident from I-IV that PC1-PC3 are not formulas. However, since the use of '(' and ')' can become both tedious and superfluous, PC1-PC3 will be considered as formulas by the following convention.

Without further agreement, if no ambiguities result, P will be written instead of (P), $\sim A$ will be written instead of $(\sim A)$, $\sim\sim A$ will be written instead of $(\sim(\sim A))$, and $A \rightarrow B$ instead of $(A \rightarrow B)$, etc. However, in replacing A or B by strings of primitive symbols which are formulas, where ambiguities may result from abbreviations then '(' and ')' must, by necessity, be furnished. For example, while no ambiguity results in writing p or $p \rightarrow q$, if in $A \rightarrow B$, A is $p \rightarrow q$ and B is $r \rightarrow q_1$ then

$$(p \rightarrow q) \rightarrow (r \rightarrow q_1)$$

must be written instead of

$$p \rightarrow q \rightarrow r \rightarrow q_1.$$

Similarly, if in $\sim A$, A is $p \rightarrow q$, then $\sim(p \rightarrow q)$ must be written instead of $\sim p \rightarrow q$.

In the sense of the above abbreviations then, PC1-PC3 are formulas. Furthermore, they give rise to an infinite number of axioms. However, since the notion of formula is effective and axioms must have one of three forms, then

<u>Metatheorem 1.3</u>. The notion of axiom is effective.

The propositional calculus will include the following rule of inference.

RI. From the formulas A and $A \rightarrow B$, B may be inferred.

Combining Metatheorems 1.1 and 1.2, then to infer B from two formulas A and C, C must be of the form $A \to B$ or A of the form $C \to B$, hence,

<u>Metatheorem 1.4</u>. The notion of inference is effective.

In view of the preceding, the notions of formal proof and formal theorem are made precise by the following definitions.

<u>Definition 1.1</u>. A <u>formal proof</u> is a finite column of formulas, each line of which is an axiom or can be inferred from two previous lines by RI.

<u>Definition 1.2</u>. A <u>formal theorem</u> is the last line of a formal proof.

Formal proof and formal theorem will be referred to as just <u>proof</u> and <u>theorem</u> respectively.

The assertion that a formula A is a theorem will be symbolized by

$$\vdash A.$$

Since the notion of formula and of inference is effective, and since only finite columns of finite strings of primitive symbols will be considered, then

<u>Metatheorem 1.5</u>. The notion of proof is effective.

As of yet there is not sufficient means to show that the notion of theorem is effective, so this must be delayed until a few important results are established. It may be mentioned, however, that in larger logistic systems, the notion of theorem is not effective.

By way of illustration, consider the following:

$\vdash {\sim}p \to (p \to q)$

Proof:

1. $(({\sim}q \to {\sim}p) \to (p \to q)) \to ({\sim}p \to (({\sim}q \to {\sim}p) \to (p \to q)))$ PC1
2. $({\sim}q \to {\sim}p) \to (p \to q)$ PC3

3. $\sim p \to ((\sim q \to \sim p) \to (p \to q))$ 2,1,RI

4. $(\sim p \to ((\sim q \to \sim p)\to(p \to q)))\to((\sim p \to (\sim q \to \sim p))\to(\sim p \to (p \to q)))$ PC2

5. $(\sim p \to (\sim q \to \sim p)) \to (\sim p \to (p \to q))$ 3,4,RI

6. $\sim p \to (\sim q \to \sim p)$ PC1

7. $\sim p \to (p \to q)$ 6,5,RI

When a proof is given, an analysis can be given in parallel. However, since formula, proof and inference are all effective, this implies there is an effective procedure for supplying an analysis. Hence, this is usually not necessary.

By using the form of the above theorem as a guide, it is easy to prove; $\vdash \sim(p \to q) \to ((p \to q) \to r)$, $\vdash \sim r_1 \to (r_1 \to (p \to q))$ and so on. That is, if A and B are arbitrary formulas, the above proof gives rise to a proof schema of the theorem schema $\sim A \to (A \to B)$, where a theorem results for a particular choice of formulas for A and B.

Deducibility

The notion of theorem will now be extended to include a more generalized notion, that of a formula being deducible from a set of assumptions.

Definition 1.3. Let A be a formula and let Γ be a set of formulas, possibly infinite or empty. Let D[Γ;A] be the set of all finite columns X, of formulas, where the last line of X is A and each line of X is a member of Γ, an axiom or can be inferred from two previous lines by RI. If D[Γ;A] is not empty then A is said to be deducible from assumptions Γ and is symbolized by

$$\Gamma \vdash A.$$

A member of D[Γ;A] is called a formal demonstration of A from Γ.

Henceforth demonstrations will mean formal demonstrations.

<u>Metatheorem 1.6</u>.

i. If membership in Γ is effective then for each formula A there is an effective procedure for determining if a finite column of formulas is a member of $D[\Gamma;A]$.

ii. $\Gamma \mid A$ whenever A is a member of Γ.

iii. If $\Gamma \mid A$ and $\Gamma \mid A \rightarrow B$ then $\Gamma \mid B$.

iv. If $\Gamma \mid A$ then for any set of formulas Ω, $\Gamma \cup \Omega \mid A$.

v. If $\Gamma \mid A$ and Γ is empty then $\mid A$.

vi. If $\Gamma \mid A$ then there exists a finite subset Γ_1 of Γ such that $\Gamma_1 \mid A$.

<u>Proof</u>:

i. Suppose membership in Γ is effective. Any finite column X of formulas must, by necessity, end in A to be in $D[\Gamma;A]$. Since formula, membership in Γ, axiom and inference are all effective, and the fact that X is finite, then membership in $D[\Gamma;A]$ is effective.

ii. Suppose A is a member of Γ. Let X be just A. Then X is in $D[\Gamma;A]$, which is therefore not empty and hence, $\Gamma \mid A$.

iii. Suppose $\Gamma \mid A$ and $\Gamma \mid A \rightarrow B$, then neither $D[\Gamma;A]$ nor $D[\Gamma;A \rightarrow B]$ are empty so let X and Y be members of them respectively. Let Z be the column consisting of X followed by Y followed by B, that is, $Z = \langle X,Y,B \rangle$. Clearly, Z is in $D[\Gamma;B]$ which is then not empty, hence, $\Gamma \mid B$.

iv. Suppose $\Gamma \mid A$ and let Ω be a set of formulas. Since $\Gamma \mid A$, $D[\Gamma;A]$ is not empty so let X be a demonstration of A from Γ. Since lines of X, which are members of Γ, are also members of $\Gamma \cup \Omega$ then clearly X is in $D[\Gamma \cup \Omega;A]$, therefore, $\Gamma \cup \Omega \mid A$.

v. Suppose Γ is empty. Then if $\Gamma \mid A$, $D[\Gamma;A]$ is not empty so let X be a demonstration. Then each line of X is an axiom or is inferred from two previous lines by RI. Hence, X is a proof. But since X ends in A, $|A$.

vi. Suppose $\Gamma\ |A$, then $D[\Gamma;A]$ is not empty. Let X be a member of $D[\Gamma;A]$. Define Γ_1 to be one formula B from Γ together with those lines of X which are also members of Γ. Since X is finite then Γ_1 will be finite and by definition a subset of Γ. Let Y be the column $\langle B,X \rangle$. Clearly, Y is a member of $D[\Gamma_1;A]$ and therefore, $\Gamma_1\ |A$.

Within the formal system itself, formal demonstrations and formal proofs of even a simple nature tend to become long and tedious. However, having defined explicitly what constitutes a deduction from assumptions and hence formal proof, it is not always necessary to appeal directly to the definition. Instead, the above properties of deducibility may be considered to establish certain metatheorems called derived rules of inference.

<u>Metatheorem 1.7</u>. If $|A$ and $|A \rightarrow B$, then $|B$.

<u>Proof</u>: This follows immediately from Metatheorems 1.6iii and 1.6v.

<u>Metatheorem 1.8</u>. If $\Gamma\ |A$ and $|A \rightarrow B$, then $\Gamma\ |B$.

<u>Proof</u>: Let Ω be the empty set. Since $|A \rightarrow B$ there is a proof X whose last line is $A \rightarrow B$. Now each line of X is an axiom or inferred from two previous lines by RI. It is clear then that X is a member of $D[\Omega;A \rightarrow B]$, hence $\Omega\ |A \rightarrow B$. Thus in view of Metatheorem 1.6iv, $\Gamma \cup \Omega\ |A$ and $\Gamma \cup \Omega\ |A \rightarrow B$, hence $\Gamma \cup \Omega\ |B$ by Metatheorem 1.6iii. But since Ω is empty, $\Gamma \cup \Omega$ is just Γ, so $\Gamma\ |B$.

The following definition is motivated by a metatheorem which will be stated and proved following the definition.

<u>Definition 1.4</u>. Let Γ be a set of formulas and let Γ^n be the set of

all formulas deducible from Γ such that, C is a member of Γ^n if and only if there exists a demonstration X[C], in D[Γ,C], with exactly n lines, $n = 1, 2, 3, \cdots$.

To make the definition complete, $\Gamma^0 = \phi$, ϕ the empty set, since a formula deducible from Γ must have a demonstration from Γ with at least one line. Furthermore, $\Gamma^n \neq \phi$ for all n, $n = 1, 2, 3, \cdots$, and in fact, $\phi = \Gamma^0 \subset \Gamma^1 \subset \Gamma^2 \subset \Gamma^3 \cdots$. This is true because Γ^1 will consist of all the formulas of Γ together with every axiom of the system. Furthermore, any demonstration X, of a formula from Γ, with k lines, results in a demonstration, of the same formula from Γ, with k + 1 lines, by affixing an axiom or a member of Γ to it; not as the last line, however.

Metatheorem 1.9. Let B_n be a member of $[\Gamma \cup \{A\}]^n$, $n \geq 1$, then $\Gamma \mid A \rightarrow C_i$ for all i, $1 \leq i \leq n$, where $X[B_n]$ is a member of $D[\Gamma \cup \{A\}; B_n]$ and $X[B_n] = \langle C_1, C_2, \cdots, C_n \rangle$; a demonstration of B_n from $\Gamma \cup \{A\}$ with exactly n lines. (The existence of $X[B_n]$ is guaranteed by the comment following Definition 1.4.)

Proof: The proof will be by induction on n.

Basis. Suppose n = 1 and let B_1 be a member of $[\Gamma \cup \{A\}]^1$, then there exists $X(B_1)$ in $D[\Gamma \cup \{A\}; B_1]$ with exactly one line. This line must be just B_1.

Case 1. If B_1 is an axiom or a member of Γ let Y be the column

$$\langle B_1,\ B_1 \rightarrow (A \rightarrow B_1),\ A \rightarrow B_1 \rangle.$$

Clearly, Y is a member of $D[\Gamma; A \rightarrow B_1]$, hence, $\Gamma \mid A \rightarrow B_1$.

Case 2. If B_1 is A, let Y be the proof of $\mid A \rightarrow A$, that is, $\mid A \rightarrow B_1$. Y is then a member of $D[\Gamma; A \rightarrow B_1]$ so, $\Gamma \mid A \rightarrow B_1$.

Induction step. Suppose the assertion is true for n = k and let

B_{k+1} be a member of $[\Gamma \cup \{A\}]^{k+1}$. Therefore, there exists a k + 1 line demonstration $X[B_{k+1}]$, in $D[\Gamma \cup \{A\};B_{k+1}]$. Let

$$X[B_{k+1}] = \langle C_1, C_2, \cdot \cdot \cdot , C_k, C_{k+1} \rangle$$

and consider the column $X[C_k] = \langle C_1, C_2, \cdot \cdot \cdot , C_k \rangle$. Clearly, $X[C_k]$ is a member of $D[\Gamma \cup \{A\};C_k]$, and C_k a member of $[\Gamma \cup \{A\}]^k$, hence, by induction hypothesis

$$\Gamma \mid A \rightarrow C_i \qquad (1)$$

for all i, $1 \leq i \leq k$.

Now C_{k+1} is just B_{k+1}, so if C_{k+1} is an axiom or a member of Γ, then $\Gamma \mid A \rightarrow C_{k+1}$ by an argument similar to Case 1. If C_{k+1} is A, then again $\Gamma \mid A \rightarrow C_{k+1}$ by an argument similar to Case 2.

Case 3. If C_{k+1} is inferred from two previous lines C_j and C_m, then without loss of generality assume C_m is $C_j \rightarrow C_{k+1}$. Now $j,m < k$ so by (1)

$$\Gamma \mid A \rightarrow C_j \qquad (2)$$

and

$$\Gamma \mid A \rightarrow (C_j \rightarrow C_{k+1}). \qquad (3)$$

By PC2, it is clear that

$$\mid (A \rightarrow (C_j \rightarrow C_{k+1})) \rightarrow ((A \rightarrow C_j) \rightarrow (A \rightarrow C_{k+1})), \qquad (4)$$

so applying Metatheorem 1.8 to (3) and (4)

$$\Gamma \mid (A \rightarrow C_j) \rightarrow (A \rightarrow C_{k+1}). \qquad (5)$$

Finally, applying Metatheorem 1.6iii to (2) and (5)

$$\Gamma \mid A \rightarrow C_{k+1}.$$

Combining this and (1) yields, $\Gamma \mid A \rightarrow C_i$ for all i, $1 \leq i \leq k + 1$; which completes the proof.

A derived rule of inference, one which plays an important role in the material which follows, is the following metatheorem referred to as the deduction theorem for the propositional calculus.

Metatheorem 1.10. If $\Gamma \cup \{A\} \mid B$ then $\Gamma \mid A \to B$.

Proof: Let X be a member of $D[\Gamma \cup \{A\};B]$, then since X has only a finite number of lines, let $X = \langle C_1, C_2, \cdots, C_n \rangle$. By Metatheorem 1.9,

$$\Gamma \mid A \to C_i$$

for all i, $1 \leq i \leq n$ and in particular $\Gamma \mid A \to C_n$ which is just $\Gamma \mid A \to B$.

By repeated application of Metatheorem 1.10, the following result is easily established.

Metatheorem 1.11. If $A_1, A_2, A_3, \cdots, A_m \mid B$ then

$$\mid A_1 \to (A_2 \to (A_3 \to (\cdots (A_m \to B)\cdots))).$$

Another important result is the following.

Metatheorem 1.12. If $\mid A_1 \to (A_2 \to (A_3 \to (\cdots (A_m \to B)\cdots)))$ then $A_1, A_2, A_3, \cdots, A_m \mid B$.

Proof: Let Γ consist of the formulas $A_1, A_2, \cdots, A_m$ and let X be the proof of

$$\mid A_1 \to (A_2 \to (\cdots (A_m \to B)\cdots)).$$

If Y is the column

$$\langle X, A_1, A_2 \to (A_3 \to (\cdots (A_m \to B)\cdots)), A_2, A_3 \to (A_4 \to (\cdots (A_m \to B)\cdots)), \cdots$$
$$\cdots, A_i, A_{i+1} \to (A_{i+2} \to (\cdots (A_m \to B)\cdots)), \cdots$$
$$\cdots, A_m \to B, A_m, B\rangle,$$

then clearly Y is a demonstration of B from Γ, hence, $D[\Gamma;B]$ is not empty and therefore, $\Gamma \mid B$; that is, $A_1, A_2, \cdots, A_m \mid B$.

In view of Metatheorems 1.11 and 1.12, the notion of deducibility can be reduced to the notion of provability and conversely. It is this important result that aids in establishing that the notion of theorem is effective.

The following are some results which follow from the derived rules of inference so far considered.

T1.1. $\vdash \sim A \to (A \to B)$

Proof:

1.	$\sim A$	Assumption
2.	$\sim A \to (\sim B \to \sim A)$	PC1
3.	$\sim B \to \sim A$	1,2,RI
4.	$(\sim B \to \sim A) \to (A \to B)$	PC3
5.	$A \to B$	3,4,RI
6.	$\vdash \sim A \to (A \to B)$	1-5,Metatheorem 1.11.

T1.2. $\vdash A \to (\sim A \to B)$.

T1.3. $\vdash \sim\sim A \to A$

Proof:

1.	$\sim\sim A$	Assumption
2.	$\vdash \sim\sim A \to (\sim A \to \sim\sim\sim A)$	T1.1
3.	$\sim A \to \sim\sim\sim A$	1,2,Metatheorem 1.8
4.	$(\sim A \to \sim\sim\sim A) \to (\sim\sim A \to A)$	PC3
5.	$\sim\sim A \to A$	3,4,RI
6.	A	1,5,RI
7.	$\vdash \sim\sim A \to A$	1-6,Metatheorem 1.11.

T1.4. $\vdash A \to \sim\sim A$.

T1.5. $\vdash (A \to B) \to ((B \to C) \to (A \to C))$

Proof:

1.	$A \to B$	Assumption
2.	$B \to C$	Assumption
3.	A	Assumption

4. B 3,1,RI

5. C 4,2,RI

6. $\vdash (A \to B) \to ((B \to C) \to (A \to C))$ 1-5,Metatheorem 1.11.

T1.6. $\vdash \sim A \to ((B \to A) \to \sim B)$.

T1.7. $\vdash B \to (\sim C \to \sim(B \to C))$.

T1.8. $\vdash (B \to A) \to ((\sim B \to A) \to A)$.

Truth Functions

In considering the notion of truth function the following definitions are required.

Definition 1.5. Let A be a formula and suppose that the totality of distinct statement variables occurring in A are $P_1, P_2, \cdots, P_n$. Then the P_i's, $1 \leq i \leq n$, will be referred to as the prime components of A.

Let P be an arbitrary statement variable and let $V = \{0,1\}$; then associated with P is a rule f_P, from V into V, defined by $f_P(x) = x$, x in V. If $f_P(x) = x$, then P is said to have the truth value x, denoted $\upsilon(P) = x$, with assignment of truth value x to P. This is generalized by

Definition 1.6. Let A be an arbitrary formula with prime components $P_1, P_2, \cdots, P_n$. Then associated with A is a rule f_A, called a truth function, from V^n into V, where V^n is the set of all ordered n-tuples with entries from V. $\upsilon(A)$ will denote the truth value of A for an assignment of truth values to the prime components of A. That is,

$$\upsilon(A) = f_A(x_1, x_2, \cdots, x_n)$$

where $\upsilon(P_i) = f_{P_i}(x_i) = x_i$, x_i in V, $1 \leq i \leq n$. Furthermore, $\upsilon(A)$ will satisfy the following for a given truth value assignment to the P_i's of A.

1. If A is of the form $\sim B$ then

i. $\upsilon(A) = 1$ if and only if $\upsilon(B) = 0$

ii. $\upsilon(A) = 0$ if and only if $\upsilon(B) = 1$.

2. If A is of the form $B \rightarrow C$ then

i. $\upsilon(A) = 0$ if and only if $\upsilon(B) = 1$ and $\upsilon(C) = 0$

ii. $\upsilon(A) = 1$ if and only if $\upsilon(B) = 0$ or $\upsilon(B) = 1$ and $\upsilon(C) = 1$.

Definition 1.6 gives rise to the notion of truth tables. This is typified by the following example: the truth table for the formula, $p \rightarrow q$.

p	q	$p \rightarrow q$
1	1	1
1	0	0
0	1	1
0	0	1

The entries under p and q are the possible values that can be assigned to p and q, while the entries under $p \rightarrow q$ are the values taken by $p \rightarrow q$ for the given assignments to p and q.

Definition 1.7. Let A be a formula with prime components $P_1, P_2, \cdots, P_n$. Then if $f_A(x_1, x_2, \cdots, x_n) = 1$ for all possible truth value assignments to the prime components of A, A is said to be a tautology. This assertion is symbolized by

$$||A.$$

Since a formula A has only a finite number of prime components there will be only a finite number of possible truth value assignments to these prime components. In view of this and Definitions 1.6-7, then

Metatheorem 1.13. The notion of tautology is effective.

Metatheorem 1.14. If $||A$ and $||A \rightarrow B$ then $||B$.

Proof: Suppose $||A$ and $||A \rightarrow B$, then $\upsilon(A) = 1$ and $\upsilon(A \rightarrow B) = 1$ for all truth value assignments to the prime components of A and $A \rightarrow B$.

But by Definition 1.6, $\upsilon(A \rightarrow B) = 1$ if and only if $\upsilon(B) = 1$ when $\upsilon(A) = 1$. This implies $\upsilon(B) = 1$ for all truth value assignments to the prime components of B. Hence, $||B$.

To illustrate Definition 1.7, consider the formula $p \rightarrow (q \rightarrow p)$. The truth table for $p \rightarrow (q \rightarrow p)$ is given below.

p	q	$q \rightarrow p$	$p \rightarrow (q \rightarrow p)$
1	1	1	1
1	0	1	1
0	1	0	1
0	0	1	1

By the above then, it is clear that $||p \rightarrow (q \rightarrow p)$.

<u>The Decision Problem</u>

In order to establish the property that the notion of theorem is effective the following result will be needed.

<u>Metatheorem 1.15.</u> Let A be a formula with prime components $P_1, P_2, \cdots, P_n$. Define P_i' to be P_i or $\sim P_i$ according as $\upsilon(P_i) = 1$ or $\upsilon(P_i) = 0$, respectively, and define A' to be A or $\sim A$ according as $\upsilon(A) = 1$ or $\upsilon(A) = 0$, respectively. Then,

$$P_1', P_2', \cdots, P_n' \mid A'$$

for each assignment of truth values to the prime components of A.

<u>Proof:</u> The proof will be by induction on the number of occurrences of '$\sim$' and '$\rightarrow$' in A. If n = 0, then A is just some P_i and the result is obvious. Suppose the condition holds for any number of occurrences less than n and suppose A contains n occurrences.

Case 1. A is of the form $\sim B$. Since B contains n-1 occurrences and also the prime components of A, then by induction hypothesis,

$$P_1', P_2', \cdots, P_n' \mid B'.$$

i. If $\upsilon(B) = 1$, then $\upsilon(A) = 0$ and A' is $\sim\sim B$, B' is B. But $|B \rightarrow \sim\sim B$, hence, $P'_1, P'_2, \cdots, P'_n \vdash \sim\sim B$; so $P'_1, P'_2, \cdots, P'_n \vdash A'$.

ii. If $\upsilon(B) = 0$, then $\upsilon(A) = 1$ and A' is A, B' is $\sim B$, hence, B' is A; so $P'_1, P'_2, \cdots, P'_n \vdash A'$ since B' is A'.

Case 2. If A is of the form $B \to C$, then by induction hypothesis,

$$P'_1, P'_2, \cdots, P'_n \vdash B'$$
$$P'_1, P'_2, \cdots, P'_n \vdash C'$$

since both B and C contain less than n occurrences.

i. If $\upsilon(C) = 1$, then $\upsilon(A) = 1$; so C' is C and A' is A. But $\vdash C \to (B \to C)$, hence, $P'_1, P'_1, \cdots, P'_n \vdash B \to C$; so $P'_1, P'_2, \cdots, P'_n \vdash A'$.

ii. If $\upsilon(B) = 0$, then $\upsilon(A) = 1$; so B' is $\sim B$ and A' is A. But $\vdash \sim B \to (B \to C)$, hence, $P'_1, P'_2, \cdots, P'_n \vdash A'$.

iii. If $\upsilon(B) = 1$ and $\upsilon(C) = 0$, then $\upsilon(A) = 0$, hence, B' is B, C' is $\sim C$ and A' is $\sim A$, that is $\sim(B \to C)$. But $\vdash B \to (\sim C \to \sim(B \to C))$; so by repeated use of Metatheorem 1.8, $P'_1, P'_2, \cdots, P'_n \vdash A'$.

After establishing the foregoing result, there is now sufficient apparatus to prove the following important result known as the completeness theorem.

Metatheorem 1.16. If $\Vdash A$ then $\vdash A$.

Proof: Suppose $\Vdash A$ and let $P_1, P_2, \cdots, P_n$ be the prime components of A. Define $P'_1, P'_2, \cdots, P'_n$ and A' as in Metatheorem 1.15. Since $\Vdash A$, then A' is A, hence by Metatheorem 1.15, $P'_1, P'_2, \cdots, P'_n \vdash A$. In particular

$$P'_1, P'_2, \cdots, P'_{n-1}, P_n \vdash A$$
$$P'_1, P'_2, \cdots, P'_{n-1}, \sim P_n \vdash A$$

for all truth value assignments to the P_i's. By the deduction theorem,

$$P'_1, P'_2, \cdots, P'_{n-1} \vdash P_n \to A$$
$$P'_1, P'_2, \cdots, P'_{n-1} \vdash \sim P_n \to A.$$

From T1.8, $\vdash (P_n \rightarrow A) \rightarrow ((\sim P_n \rightarrow A) \rightarrow A)$, so by repeated use of Metatheorem 1.8,

$$P'_1, P'_2, \ldots, P'_{n-1} \vdash A.$$

Repeating this process of eliminating assumptions yields,

$$\vdash A.$$

<u>Metatheorem 1.17.</u> If $\vdash A$ then $\vDash A$.

<u>Proof:</u> Suppose $\vdash A$. It is easy to show that each axiom is a tautology. Using Metatheorem 1.14 and the fact that each line of the proof of A is an axiom or inferred from two previous lines by RI the result follows.

Metatheorems 1.16-17 show that $\vdash A$ if and only if $\vDash A$. By Metatheorem 1.13 the notion of tautology is effective, hence, given a formula A, there is an effective procedure for deciding if A is or is not a theorem by seeing if A is or is not a tautology.

More generally, Metatheorem 1.15 affords an effective procedure for providing a proof for a theorem which has been shown to be a theorem by showing it to be a tautology. Hence,

<u>Metatheorem 1.18.</u> The notion of theorem is effective.

<u>Metatheorem 1.19.</u> The notion of provability is effective.

THE PREDICATE CALCULUS

Primitive Basis

The propositional calculus can be extended to a more general theory, this theory being the predicate calculus. As in the case of the propositional calculus, symbols, devoid of interpretation, will be used extensively in order to put the theory in a purely formal context.

For this particular formulation the following symbols will be employed as the primitive symbols of the predicate calculus.

The improper symbols;

$$(\quad) \quad \sim \quad \rightarrow \quad ,$$

together with the three at most denumerable infinite lists of proper symbols,

$$p \quad q \quad r \quad p_1 \quad q_1 \quad r_1 \quad p_2 \quad q_2 \quad r_2 \quad \cdot \cdot \cdot$$

called statement variables;

$$x \quad y \quad z \quad x_1 \quad y_1 \quad z_1 \quad x_2 \quad y_2 \quad z_2 \quad \cdot \cdot \cdot$$

called individual variables and for each positive integer n, n-place predicate symbols,

$$\begin{array}{cccccccccccccc}
F^1 & G^1 & H^1 & F^1_1 & G^1_1 & H^1_1 & F^1_2 & G^1_2 & H^1_2 & \cdot & \cdot & \cdot \\
F^2 & G^2 & H^2 & F^2_1 & G^2_1 & H^2_1 & F^2_2 & G^2_2 & H^2_2 & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
F^n & G^n & H^n & F^n_1 & G^n_1 & H^n_1 & F^n_2 & G^n_2 & H^n_2 & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{array}$$

called predicate variables. The dots are used to indicate the lists continue indefinitely.

The formation rules are given by the recursive definition:

I. If Q is a statement variable, then (Q) is a formula.

II. If P is an n-place predicate variable, then $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$ is a formula, where $\alpha_1, \alpha_2, \cdots, \alpha_n$ are individual variables.

III. If A is a formula, then ($\sim$A) is a formula.

IV. If A and B are formulas, then $(A \rightarrow B)$ is a formula.

V. If A is a formula then, $(\alpha)A$ is a formula, where α is an individual variable.

VI. Only finite strings of primitive symbols which follow from I-V are formulas.

It is evident from II and V and Metatheorem 1.1 that

Metatheorem 2.1. The notion of formula is effective.

Definition 2.1. If A is a formula then any occurrence of the individual variable α, in the formula $(\alpha)A$, is called a bound occurrence in $(\alpha)A$. Any individual variable α, which is not a bound occurrence in a formula, is called a free occurrence.

The axioms of the predicate calculus are given by the following schema:

P1. $A \rightarrow (B \rightarrow A)$

P2. $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$

P3. $(\sim A \rightarrow \sim B) \rightarrow (B \rightarrow A)$

P4. $(\alpha)(A \rightarrow B) \rightarrow (A \rightarrow (\alpha)B)$, where α is an individual variable with no free occurrence in A.

P5. $(\alpha)A \rightarrow B$, where α is an individual variable, β an individual variable and B is obtained from A by replacing each free occurrence of α in A by β, provided that no free occurrence of α is in a part of A of the form $(\beta)C$.

It is to be noted that, as in the case of the propositional calculus, certain liberties are taken in regards to the use of the symbols '(' and ')'. However, this must be done with discretion.

Metatheorem 2.2. The notion of axiom is effective.

Proof: This follows immediately from Metatheorem 2.1 and the fact that axioms will have one of five forms.

In addition to the axioms P1-P5 the predicate calculus will have the two rules of inference:

RI. From the formulas A and A → B, B may be inferred.

UG. (Generalization) From the formula A, (α)A may be inferred where α is an individual variable.

To infer (α)A from a formula B, B must be just A and by extending Metatheorem 1.4 to the predicate calculus then

Metatheorem 2.3. The notions of RI and UG are effective.

Definition 2.2. A formal proof is a finite column of formulas, each line of which is an axiom, inferred from two previous lines by RI or inferred from a single preceding line by UG.

Definition 2.3. A formal theorem is the last line of a formal proof.

The assertion that A is a theorem will be denoted by

$$\vdash A.$$

As a result of the foregoing it can be shown that

$\vdash (x)F^1(x) \rightarrow (y)F^1(y)$.

Proof:

1.	$(x)F^1(x) \rightarrow F^1(y)$	P5
2.	$(y)((x)F^1(x) \rightarrow F^1(y))$	1,UG
3.	$(y)((x)F^1(x) \rightarrow F^1(y)) \rightarrow ((x)F^1(x) \rightarrow (y)F^1(y))$	P4

4. $(x)F^1(x) \rightarrow (y)F^1(y)$ 2,3,RI

The above proof gives rise to a proof schema for the theorem schema $(\alpha)P(\alpha) \rightarrow (\beta)P(\beta)$, where α, β are arbitrary individual variables and P an arbitrary 1-place predicate variable.

More generally $\vdash (\alpha)A \rightarrow (\beta)A$, provided no free occurrences of α in A is in a part of A of the form $(\beta)C$ and provided β is free in no part of A.

Since formula, axiom and inference are all effective and since proofs are finite columns of formulas, then for the predicate calculus

Metatheorem 2.4. The notion of proof is effective.

Deducibility

In order to extend the notion of deducibility from a set of assumptions to the predicate calculus the following definition is required.

Definition 2.4. A column Y of formulas is called a subcolumn of a finite column X of formulas provided the formulas of Y appear in X in precisely the same order as in Y.

Definition 2.5. Let Γ be a set of formulas, possibly infinite or empty, and let A be a formula. Define $D[\Gamma;A]$ to be the set of all finite columns X of formulas whose last line is A and where each line of X is an axiom, a member of Γ, inferred from two preceding lines by RI or inferred from a single previous line B, by generalization on any individual variable, provided that B is the last line of a subcolumn Y of X, which is a formal proof.

In case $D[\Gamma;A]$ is not empty, then A is said to be deducible from assumptions Γ. This assertion is symbolized by

$$\Gamma \vdash A.$$

Any member of $D[\Gamma;A]$ is called a formal demonstration of A from Γ.

By the nature of Definition 2.5, Metatheorem 1.6 can be extended to the predicate calculus and consequently the following derived rules of inference result.

Metatheorem 2.5. If $\vdash A$ and $\vdash A \rightarrow B$ then $\vdash B$.

Metatheorem 2.6. If $\Gamma \vdash A$ and $\vdash A \rightarrow B$ then $\Gamma \vdash B$.

More important, however, is that Metatheorem 1.10 can be extended to give the deduction theorem for the predicate calculus.

Metatheorem 2.7. If Γ is a set of formulas and A and B are formulas and if $\Gamma \cup \{A\} \vdash B$ then $\Gamma \vdash A \rightarrow B$.

Proof: The proof is obtained from Metatheorem 1.9, along with an additional case following Case 3.

Case 4. If C_{k+1} is inferred from a previous line C_j, $j \leq k$, by generalization on an individual variable α, where C_j is the last line of a subcolumn Z of $X[B_{k+1}]$, which is a formal proof, then C_{k+1} is just $(\alpha)C_j$.

Since Z is a formal proof whose last line is C_j, then the column, $\langle Z, (\alpha)C_j \rangle$ is also a formal proof, hence, $\vdash (\alpha)C_j$.

By P1,

$$\vdash (\alpha)C_j \rightarrow (C_j \rightarrow (\alpha)C_j); \ \vdash (C_j \rightarrow (\alpha)C_j) \rightarrow (A \rightarrow (C_j \rightarrow (\alpha)C_j))$$

and by P2,

$$\vdash (A \rightarrow (C_j \rightarrow (\alpha)C_j)) \rightarrow ((A \rightarrow C_j) \rightarrow (A \rightarrow (\alpha)C_j)).$$

Repeated use of Metatheorem 2.5 to the above yields,

$$\vdash (A \rightarrow C_j) \rightarrow (A \rightarrow (\alpha)C_j).$$

By induction hypothesis, $\Gamma \vdash A \rightarrow C_j$, hence, by Metatheorem 2.6,

$$\Gamma \vdash A \rightarrow (\alpha)C_j.$$

Now if $\Gamma \cup \{A\} \vdash B$, then $\Gamma \vdash A \rightarrow C_i$, where C_i is any line of a demonstration of B from $\Gamma \cup \{A\}$. Therefore, $\Gamma \vdash A \rightarrow B$ since B will be the last line.

The preceding result enables Metatheorems 1.11 and 1.12 to be extended to the predicate calculus, hence

<u>Metatheorem 2.8.</u> $A_1, A_2, \cdots, A_m \mid B$ if and only if $\mid A_1 \to (A_2 \to (\cdots (A_m \to B)\cdots))$.

Let A and B be formulas and abbreviate the formula, $\sim(A \to \sim B)$, by $A \,\Delta\, B$. From this abbreviation then

<u>Definition 2.6.</u> If $A_1, A_2, \cdots, A_n$ are formulas, define the conjunction, $\prod_1^n A_i$, of the formulas $A_1, A_2, \cdots, A_n$ inductively by: $\prod_1^1 A_i$ is A_1; $\prod_1^{j+1} A_i$ is $A_{j+1} \,\Delta\, (\prod_1^j A_i)$, for $j = 1, 2, \cdots, n-1$.

As a consequence of Definition 2.6 and the preceding rules of inference the following results can be established.

T2.1. $\mid A \,\Delta\, B \to A$

Proof:

1.	$\mid \sim A \to (A \to \sim B)$	
2.	$\mid (\sim A \to (A \to \sim B)) \to (\sim(A \to \sim B) \to A)$	
3.	$\mid \sim(A \to \sim B) \to A$	1,2, Metatheorem 2.5
4.	$\mid A \,\Delta\, B \to A$	3, definition of Δ.

Similarly,

T2.2. $\mid A \,\Delta\, B \to B$.

T2.3. $\mid (A \to (B \to C)) \to (A \,\Delta\, B \to C)$

Proof:

1.	$A \to (B \to C)$	Assumption
2.	$A \,\Delta\, B$	Assumption
3.	$\mid A \,\Delta\, B \to A$	T2.1
4.	A	2,3, Metatheorem 2.6
5.	$B \to C$	4,1, RI
6.	$\mid A \,\Delta\, B \to B$	T2.2

7. B — 2,6, Metatheorem 2.6

8. C — 7,5, RI

9. $|(A \to (B \to C)) \to (A \Delta B \to C)$ — 1-8, Metatheorem 2.8.

T2.4. $|(A \Delta B \to C) \to (A \to (B \to C))$

Proof:

1. $A \Delta B \to C$ — Assumption

2. A — Assumption

3. B — Assumption

4. $|A \to (B \to \sim(A \to \sim B))$

5. $B \to \sim(A \to \sim B)$ — 2,4, Metatheorem 2.6

6. $\sim(A \to \sim B)$ — 3,5, RI

7. $A \Delta B$ — 6, definition of Δ

8. C — 7,1, RI

9. $|(A \Delta B \to C) \to (A \to (B \to C))$ — 1-8, Metatheorem 2.8.

T2.5. $|(\prod_1^j A_i \to (A_{j+1} \to B)) \to (\prod_1^{j+1} A_i \to B)$

Proof:

1. $\prod_1^j A_i \to (A_{j+1} \to B)$ — Assumption

2. $\prod_1^{j+1} A_i$ — Assumption

3. $A_{j+1} \Delta \prod_1^j A_i$ — 2, Definition 2.6

4. $|A_{j+1} \Delta \prod_1^j A_i \to A_{j+1}$ — T2.1

5. A_{j+1} — 3,4, Metatheorem 2.6

6. $|A_{j+1} \Delta \prod_1^j A_i \to \prod_1^j A_i$ — T2.2

7. $\prod_1^j A_i$ — 3,6, Metatheorem 2.6

8. $A_{j+1} \to B$ — 7,1, RI

9. B — 5,8, RI

10. $|(\prod_1^j A_i \to (A_{j+1} \to B)) \to (\prod_1^{j+1} A_i \to B)$ — 1-9, Metatheorem 2.8.

T2.6. $|(\prod_1^{j+1} A_i \to B) \to (\prod_1^j A_i \to (A_{j+1} \to B))$

Proof:

1. $\prod_1^{j+1} A_i \to B$ Assumption

2. $\prod_1^{j} A_i$ Assumption

3. A_{j+1} Assumption

4. $A_{j+1} \Delta \prod_1^{j} A_i \to B$ 1, Definition 2.6

5. $\vdash (A_{j+1} \Delta \prod_1^{j} A_j \to B) \to (A_{j+1} \to (\prod_1^{j} A_i \to B))$

T2.4

6. $A_{j+1} \to (\prod_1^{j} A_i \to B)$ 4,5, Metatheorem 2.6

7. $\prod_1^{j} A_i \to B$ 6,3, RI

8. B 2,7, RI

9. $\vdash (\prod_1^{j+1} A_i \to B) \to (\prod_1^{j} A_i \to (A_{j+1} \to B))$ 1-8, Metatheorem 2.8.

Metatheorem 2.9. $A_1, A_2, \cdots, A_n \vdash B$ if and only if $\vdash \prod_1^{n} A_i \to B$.

Proof: The proof will be by induction on n. When n = 1 it is obvious that $A_1 \vdash B$ implies $\vdash \prod_1^{1} A_i \to B$.

Suppose the assertion is true for all $k < n$ and suppose $A_1, A_2, \cdots, A_n \vdash B$. By the deduction theorem $A_1, A_2, \cdots, A_{n-1} \vdash A_n \to B$, so from the induction hypothesis, $\vdash \prod_1^{n-1} A_i \to (A_n \to B)$ and by T2.5 and Metatheorem 2.5, $\vdash \prod_1^{n} A_i \to B$.

Conversely; for n = 1, $\vdash \prod_1^{1} A_i \to B$ gives $A_1 \vdash B$. Suppose the assertion is true for all $k < n$ and suppose $\vdash \prod_1^{n} A_i \to B$. From T2.6 and Metatheorem 2.5, then $\vdash \prod_1^{n-1} A_i \to (A_n \to B)$. Hence, by induction hypothesis, $A_1, A_2, \cdots, A_{n-1} \vdash A_n \to B$. But then $A_1, A_2, \cdots, A_{n-1}, A_n \vdash A_n \to B$ and $A_1, A_2, \cdots, A_n \vdash A_n$, so by Metatheorem 1.6iii extended to the predicate calculus, $A_1, A_2, \cdots, A_n \vdash B$.

Metatheorem 2.9 is equivalent to Metatheorem 2.8 but with a difference in notation.

<u>Metatheorem 2.10.</u> If $\Gamma \mid A$ and α is an individual variable not free in any formula of Γ then $\Gamma \mid (\alpha)A$.

<u>Proof:</u> Suppose $\Gamma \mid A$ and α is an individual variable not free in any formula of Γ. By Metatheorem 1.6vi, extended to the predicate calculus, there exists a finite subset, $A_1, A_2, \cdots, A_n$ of Γ, such that $A_1, A_2, \cdots, A_n \mid A$. By Metatheorem 2.9, $\mid \prod_1^n A_i \rightarrow A$, which is a formal theorem. Let X be the proof of this theorem. Since α is not free in any of the A_i's, $1 \leq i \leq n$, then α is not free in $\prod_1^n A_i$ so the column,

$$\langle X,\ (\alpha)(\prod_1^n A_i \rightarrow A),\ (\alpha)(\prod_1^n A_i \rightarrow A) \rightarrow (\prod_1^n A_i \rightarrow (\alpha)A),\ \prod_1^n A_i \rightarrow (\alpha)A \rangle$$

is a formal proof, hence $\mid \prod_1^n A_i \rightarrow (\alpha)A$ is a formal theorem. By Metatheorem 2.9 this implies that, $A_1, A_2, \cdots, A_n \mid (\alpha)A$ and by Metatheorem 1.6iv extended to the predicate calculus, then $\Gamma \mid (\alpha)A$.

In view of the preceding metatheorem, it is evident that if A is a formula with a free occurrence of an individual variable α, then in a demonstration which involves A, as an assumption formula, no generalization on α can be made. In this case, α is said to have a conditional interpretation. In contrast, if α has a free occurrence in a formula A, which is an axiom, then A in intended to mean the same as $(\alpha)A$. In this case, α is said to have a generality interpretation.

<u>Definition 2.7.</u> If A is a formula and its distinct free individual variables occur in the order of $\alpha_1, \alpha_2, \cdots, \alpha_n$ then the formula, $(\alpha_1)(\alpha_2)\cdots(\alpha_n)A$, is called the closure of A. This is symbolized by

$$\Lambda A.$$

Under the generality interpretation A and ΛA are synonymous.

<u>Metatheorem 2.11.</u> If $\Gamma \mid A$ and $\Omega \mid B$ for every formula B in Γ, then $\Omega \mid A$.

Proof: Suppose $\Gamma\ |A$ and $\Omega\ |B$ for each formula B in Γ. Since $\Gamma\ |A$ there exists a finite subset, $A_1, A_2, \cdots, A_n$ of Γ, such that $A_1, A_2, \cdots, A_n\ |A$. From Metatheorem 2.8 then, $|A_1 \to (A_2 \to (\cdots (A_n \to A)\cdots))$. Now, $\Omega\ |B$ for each formula B in Γ so in particular, $\Omega\ |A_i$ for each i, $1 \leq i \leq n$. Therefore, $\Omega\ |A_1$ and $|A_1 \to (A_2 \to (\cdots (A_n \to A)\cdots))$ so by Metatheorem 2.6, $\Omega\ |A_2 \to A_3 \to (\cdots (A_n \to A)\cdots))$. From this and the fact that $\Omega\ |A_2$, Metatheorem 1.6iii, extended to the predicate calculus, gives $\Omega\ |A_3 \to (A_4 \to (\cdots (A_n \to A)\cdots))$. Again from this and the fact that $\Omega\ |A_3$, then $\Omega\ |A_4 \to (A_5 \to (\cdots (A_n \to A)\cdots))$. Continuing this process yields $\Omega\ |A$.

Metatheorem 2.12. If $\Gamma \cup \{A\}\ |B$, then $\Gamma\ |\Lambda A \to B$.

Proof: Suppose $\Gamma \cup \{A\}\ |B$. From P5, $(\alpha)A \to A$, provided no part of A is of the form $(\alpha)C$. Let the distinct free individual variables of A be $\alpha_1, \alpha_2, \cdots \alpha_n$ in that order, than ΛA is just $(\alpha_1)(\alpha_2)\cdots(\alpha_n)A$. Since each α_i, $1 \leq i \leq n$, is free in A it will appear in no part of A of the form $(\alpha_i)C$. Hence, by repeated use of RI and P5, then $\Lambda A\ |A$. Let C be a formula of $\Gamma \cup \{A\}$. If C is a member of Γ then $\Gamma\ |C$ hence, $\Gamma \cup \{\Lambda A\}\ |C$. If C is A, then since $\Lambda A\ |A$, $\Gamma \cup \{\Lambda A\}\ |C$. Therefore, $\Gamma \cup \{A\}\ |B$ and for every formula C in $\Gamma \cup \{A\}$; $\Gamma \cup \{\Lambda A\}\ |C$ so by Metatheorem 2.11, $\Gamma \cup \{\Lambda A\}\ |B$ and by the deduction theorem $\Gamma\ |\Lambda A \to B$.

As evidenced by the preceding metatheorems, the notion of deducibility is reduced to the notion of provability and conversely.

Valuation Procedure and Validity

Suppose that associated with the predicate calculus is some nonempty set D, called a domain, such that the individual variables are associated in some way with the elements of D. Let $V = \{0,1\}$ be a set of truth values and suppose that for every n-place predicate variable P there is associated a logical function λ, where λ is a function from D^n into V. Furthermore, assume that a truth value from V can be assigned to a formula, $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$, relative to an assignment of an element of D to each distinct individual variable among $\alpha_1, \alpha_2, \cdots, \alpha_n$, in the following way. If d_i, in D, is assigned to α_i, in $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$, and if λ is assigned to P then the truth value of $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$ is $\lambda(d_1, d_2, \cdots, d_n)$.

Let C be a formula of the predicate calculus. Then from the foregoing it is assumed that a domain D is given, to each predicate variable appearing in C is assigned a logical function and to each distinct free occurrence of an individual variable in C is assigned an element from D. This constitutes an assignment to C and gives rise to a valuation procedure for assigning a truth value υ(C), to C. A truth value is assigned to C in the following way:

1. If $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$ is a part of C and if λ is assigned to P, d_i in D assigned to α_i, $1 \leq i \leq n$, then the truth value assigned to $P(\alpha_1, \alpha_2, \cdots, \alpha_n)$ is $\lambda(d_1, d_2, \cdots, d_n)$.

2. To the statement variables of C is assigned either 0 or 1.

3. For a given assignment to the predicate variables, distinct free individual variables and the statement variables of C then if C is of the form ∼A, υ(C) = 0 if and only if υ(A) = 1 and υ(C) = 1 if and only if υ(A) = 0. If C is of the form A → B then υ(C) = 0 if and only if υ(A) = 1 and υ(B) = 0; υ(C) = 1 if and only if υ(A) = 0 or υ(A) = 1 and υ(B) = 1. If C is of the form (α)A, then υ(C) = 1 if and only if υ(A) = 1 for every assignment to α; υ(C) = 0 if and only if υ(A) = 0 for at least one assignment to α.

Thus, consider the formula $(x)F^1(x) \rightarrow (y)F^1(y)$ for a domain $D = \{a,b\}$, of two individuals. The possible logical functions λ, from D into V are tabulated by:

x	$\lambda_1(x)$	$\lambda_2(x)$	$\lambda_3(x)$	$\lambda_4(x)$
a	1	1	0	0
b	1	0	1	0

The possible truth value assignments are given by:

F^1	x	y	$F^1(x)$	$F^1(y)$	$(x)F^1(x)$	$(y)F^1(y)$	$(x)F^1(x)$	$\rightarrow$	$(y)F^1(y)$
	a	a	1	1					
	a	b	1	1					
λ_1	b	a	1	1	1	1	1	1	1
	b	b	1	1					
	a	a	1	1					
	a	b	1	0					
λ_2	b	a	0	1	0	0	0	1	0
	b	b	0	0					
	a	a	0	0					
	a	b	0	1					
λ_3	b	a	1	0	0	0	0	1	0
	b	b	1	1					
	a	a	0	0					
	a	b	0	0					
λ_4	b	a	0	0	0	0	0	1	0
	b	b	0	0					

where the hotizontal blocks constitute an assignment of a logical function to F^1, together with the possible assignments to x and y.

As another example consider $p \rightarrow (x)F^2(x,y)$ for $D = \{a,b\}$.

The possible logical functions from D^2 into V are:

$\lambda(x,y)$ / (x,y)	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9,	. . . ,	λ_{15}	λ_{16}
(a,a)	1	1	1	1	1	1	1	1	0	. . .	0	0
(a,b)	1	1	1	1	0	0	0	0	1	. . .	0	0
(b,a)	1	1	0	0	1	1	0	0	1	. . .	0	0
(b,b)	1	0	1	0	1	0	1	0	1	. . .	1	0 .

The truth value assignments for an assignment of λ_1, λ_6, and λ_8 to F^2 are given below:

F^2	y	x	$F^2(x,y)$	$(x)F^2(x,y)$	p	$\rightarrow$	$(x)F^2(x,y)$
λ_1	a	a	1	1	1	1	1
	a	b	1		1	1	
	b	a	1	1	1	1	1
	b	b	1		1	1	
	a	a	1	1	0	1	1
	a	b	1		0	1	
	b	a	1	1	0	1	1
	b	b	1		0	1	
λ_6	a	a	1	0	1	0	0
	a	b	0		1	0	
	b	a	1	0	1	0	0
	b	b	0		1	0	
	a	a	1	0	0	1	0
	a	b	0		0	1	
	b	a	1	0	0	1	0
	b	b	0		0	1	
λ_8	a	a	1	0	1	0	0
	a	b	0		1	0	
	b	a	0	0	1	0	0
	b	b	0		1	0	
	a	a	1	0	0	1	0
	a	b	0		0	1	
	b	a	0	0	0	1	0
	b	b	0		0	1	

<u>Definition 2.8.</u> A formula C is said to be valid in a domain D provided $\upsilon(C) = 1$ for all assignments of logical functions to the predicate variables of C, for all assignments of elements of D to the distinct free individual variables of C and for all assignments of 0 and 1 to the statement variables of C.

<u>Definition 2.9.</u> A formula C is said to be universally valid or simply valid if and only if it is valid in every domain. This is symbolized by

$$||C.$$

As was the case for the propositional calculus, in the predicate calculus, the notion of provability reduces to the notion of validity and conversely. This important result is known as Gödel's Completeness Theorem and will be stated without proof.

<u>Metatheorem 2.13.</u> |A if and only if ||A.

<u>The Decision Problem</u>

When considering the notion of validity in the predicate calculus, for a formula C to be valid, the valuation procedure must include all domains. This means that infinite domains must be considered, but in view of the valuation procedure this suggests that in valuating C, no method exists which involves only a finite number of steps and in general this is indeed the case. However, in the predicate calculus with only 1-place predicate variables the notion of theorem is effective.

<u>Metatheorem 2.14.</u> In the predicate calculus the notion of theorem is not effective.

It might be pointed out, however, that for formulas of a certain form there exists an effective procedure for deciding whether a formula of this form is or is not valid and consequently if it is or is not a theorem.

LITERATURE CITED

Church, A. 1956. Introduction to Mathematical Logic. Princeton University Press, Princeton, New Jersey.

Stoll, R. R. 1961. Set Theory and Logic. W. H. Freeman and Company, San Francisco and London.

TURING MACHINES AND RECURSIVE FUNCTIONS

by

Lowell Anderson

Report No. 2 submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Mathematics

Plan B

UTAH STATE UNIVERSITY
Logan, Utah

1967

TABLE OF CONTENTS

INTRODUCTION

Let there be given a function, defined on some domain, then the question might be asked: "Does there exist a finite set of rules or instructions for calculating, in a finite number of steps, the functional values of the function?" If such a set of instructions exist, then the function is said to be effectively calculable and the instructions are referred to as an algorithm or effective computational procedure.

One requirement of an algorithm is that it be purely mechanical; mechanical in the sense that, at least in principle, a computing device could be constructed to carry out the instructions, with no intelligence or creativity needed to follow them.

There is, perhaps, one shortcoming to most if not all algorithms. This being, although the algorithm will furnish an answer if an answer is forthcoming, it may have one compute indefinitely should no answer be forthcoming.

With the notion of effective computational procedure in mind a class of objects, called Turing machines, is considered. A Turing machine will afford an effective procedure for computing the functional values of a certain class of functions. Such functions will be called Turing computable or merely computable.

In particular, a class of functions called recursive functions will be considered and it will be shown that these functions are Turing computable.

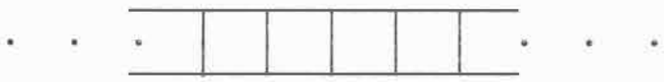
Throughout this paper some of the more fundamental concepts of sets and functions will be assumed. Moreover, only functions defined on n-tuples of non-negative integers will be considered. Also, when no confusion results, a function's name and its functional notation will be used interchangeably.

TURING MACHINES AND COMPUTABILITY

Turing machines

Intuitively, one may think of a Turing machine as a computing device which is capable of printing (or erasing) only a finite sequence of given symbols, onto a linear tape; the tape being infinite in both directions and ruled into a two-way infinite sequence of squares.

The following figure is suggestive of this infinite tape.



This machine will be "sensitive" to only one square at a time, thus, being able to print (or erase) only one symbol to a square, the square being scanned. Further, this machine will be capable of assuming only a finite number of machine states or internal configurations, where the next act or operation that the machine will perform is completely determined by the machine state together with the symbol that appears on the square being scanned. Also, the machine will be capable of only the following: a complete halt of operation, a change of the symbol on the square being scanned, a move one square to the right or left of the square being scanned; where in each case the machine will enter into a new machine state.

The symbols q_1, q_2, q_3, . . . will be used to denote possible machine states and the symbols S_0, S_1, S_2, . . . will be used to denote the symbols the machine will be capable of printing. The letters R and L will denote one move to the right or left respectively.

With the foregoing remarks as an intuitive basis, the notion of a Turing machine will be given a precise description. However, prior to defining a Turing machine, two definitions are necessary.

Definition 1.1. An expression is a finite sequence of symbols (possibly empty), from the symbols $q_1, q_2, q_3, \ldots; S_0, S_1, S_2, \ldots; R, L$.

Definition 1.2. A quadruple is an expression having one of the following four forms:

(1) $q_i S_j S_k q_m$

(2) $q_i S_j R\ q_m$

(3) $q_i S_j L\ q_m$

(4) $q_i S_j q_k q_m$.

Definition 1.3. A Turing machine is a finite, nonempty set of quadruples such that no two quadruples have their first two symbols the same. (This eliminates the possibility of a machine reaching a "confused state".)

The q_i's which appear in the quadruples of a Turing machine will be called its machine states and the S_i's its alphabet.

Definition 1.4. A Turing machine that consists entirely of quadruples of the form (1)-(3), is called simple.

Consider now, the following definitions.

Definition 1.5. An instantaneous description is an expression consisting of exactly one q_i, neither R nor L and such that q_i is not the rightmost symbol.

Definition 1.6. An expression which consists entirely of S_i's is called a tape expression.

Definition 1.7. Let Z be a Turing machine and let α be an instantaneous description. If the q_i in α is a machine state of Z and the S_i's in α belong to the alphabet of Z, then α is called an instantaneous description of Z.

Definition 1.8. Let Z be a Turing machine and let α be an instantaneous description of Z. Let q_i be the machine state of Z in α and S_j the symbol immediately to the right of q_i. Then q_i is called the machine state of Z at α, S_j the symbol scanned by Z at α and the expression obtained by deleting q_i from α is called the expression on the tape of Z at α.

From an intuitive point of view, Definition 1.8 affords a means by which an instantaneous description α may be thought of as precisely describing the status of a Turing machine at some particular time in its operation; where α gives the machine's state, the expression on its tape and the symbol being scanned.

Earlier, the tape of a Turing machine was described as being infinite in both directions. However, in view of Definitions 1.1 and 1.5, an instantaneous description is always finite. Hence, these definitions, together with Definition 1.8, dictate that a Turing machine scan only those squares on which symbols have been printed. This means that a Turing machine is not capable of scanning blank squares. However, this limitation can be overcome by adopting the following convention.

Since the expression on the tape of a Turing machine at an instantaneous description α is always finite, think of the tape as being finite where, when the Turing machine is about to run off the end of its tape it is capable of splicing on a new square on which the symbol S_0 has been printed.

The symbol S_0 then, will be reserved to stand for a blank square; B will also be written in place of S_0.

The following definition will allow an instantaneous description of a Turing machine to be replaced by a succeeding instantaneous description.

Definition 1.9. Let Z be a Turing machine and α, β instantaneous descriptions. Then α is replaced with β by Z, symbolized $\alpha \rightarrow \beta(Z)$, or when no confusion results, merely as $\alpha \rightarrow \beta$, provided there exist tape expressions P and Q (possibly empty) such that one of the following holds:

(1) α is Pq_iS_jQ, $q_iS_jS_kq_m \varepsilon Z$ and β is Pq_mS_kQ (reprint)

(2) α is $Pq_iS_jS_kQ$, $q_iS_jR\ q_m \varepsilon Z$ and β is $PS_jq_mS_kQ$ (right search)

(3) α is Pq_iS_j, $q_iS_jR\ q_m \varepsilon Z$ and β is $PS_jq_mS_0$

(4) α is $PS_kq_iS_jQ$, $q_iS_jL\ q_m \varepsilon Z$ and β is $Pq_mS_kS_jQ$ (left search)

(5) α is q_iS_jQ, $q_iS_jL\ q_m \varepsilon Z$ and β is $q_mS_0S_jQ$.

It may be noted that Definition 1.9 makes no mention of quadruples of the form $q_iS_jq_kq_m$. Turing machines having quadruples of this form will be considered later. For the present, however, only Turing machines that are simple will be dealt with.

Two results that follow from the preceding definition are the following theorems.

Theorem 1.1. If $\alpha \rightarrow \beta(Z)$ and $\alpha \rightarrow \gamma(Z)$, then β and γ are the same instantaneous descriptions.

Theorem 1.2. If Z_1 and Z_2 are Turing machines such that $Z_1 \subset Z_2$ and if $\alpha \rightarrow \beta(Z_1)$, then $\alpha \rightarrow \beta(Z_2)$.

<u>Definition 1.10.</u> An instantaneous description α is called <u>terminal with respect to Z if,</u> for all instantaneous descriptions β, it is not the case that $\alpha \rightarrow \beta(Z)$.

<u>Definition 1.11.</u> Let Z be a Turing machine, then a <u>computation of Z</u> will be a finite sequence $\alpha_1, \alpha_2, \ldots, \alpha_n$ of instantaneous descriptions such that $\alpha_i \rightarrow \alpha_{i+1}(Z)$ for $i = 1,2,\ldots,n-1$ and where α_n is terminal with respect to Z. If such be the case, then

$$\alpha_n = \mathrm{Res}_Z(\alpha_1)$$

will be written and α_n will be called the <u>resultant of α_1 with respect to Z;</u> α_1 will also be called the <u>input</u> and α_n the <u>output</u> with respect to Z.

In what follows, the symbol q_i will denote the machine state at instantaneous description α_i. Moreover, α_1 will be assumed as input.

Consider the following example, where Z is the Turing machine consisting of the quadruples:

$$q_1 S_1 L\ q_1$$
$$q_1 S_2 L\ q_1$$
$$q_1 S_3 L\ q_1$$
$$q_1 S_0 R\ q_2$$
$$q_2 S_1 S_0 q_3$$
$$q_2 S_2 S_0 q_3$$
$$q_2 S_3 S_0 q_3$$
$$q_3 S_0 R\ q_2 .$$

Let

$$\alpha_1 = S_2S_1q_1S_3,$$

then the following is a computation with respect to Z:

$$\begin{aligned}\alpha_1 &= S_2S_1q_1S_3\\ &\rightarrow S_2q_1S_1S_3\\ &\rightarrow q_1S_2S_1S_3\\ &\rightarrow q_1S_0S_2S_1S_3\\ &\rightarrow S_0q_2S_2S_1S_3\\ &\rightarrow S_0q_3S_0S_1S_3\\ &\rightarrow S_0S_0q_2S_1S_3\\ &\rightarrow S_0S_0q_3S_0S_3\\ &\rightarrow S_0S_0S_0q_2S_3\\ &\rightarrow S_0S_0S_0q_3S_0\\ &\rightarrow S_0S_0S_0S_0q_2S_0\end{aligned}$$

which is terminal, hence,

$$Res_Z(S_2S_1q_1S_3) = S_0S_0S_0S_0q_2S_0.$$

The effect of Z on α_1 is to move left until a blank is encountered, then proceed right, erasing everything until a blank is again encountered.

Should the quadruple $q_2S_0R\ q_2$ be added to Z, the machine would compute indefinitely and $Res_Z(\alpha_1)$ would not be defined. This illustrates the fact that Theorem 1.2 does not extend to computation. That is, if $Z_1 \subset Z_2$, then a computation of Z_1 need not necessarily be a computation of Z_2.

<u>Computable and partially computable functions</u>

In order for a Turing machine to perform numerical computations, a symbolic representation must be introduced so that to a given integer n, there can be associated an appropriate tape expression.

Henceforth, 1 will be written instead of S_1 and S_i^n shall denote the tape expression $S_i S_i \ldots S_i$, consisting of n occurrences of the symbol S_i, with S_i^0 being the empty expression.

For convenience, J will denote the set of all non-negative integers and J^n, n a positive integer, will denote the set of all ordered n-tuples of J.

<u>Definition 1.12.</u> To each non-negative integer n, associate the tape expression $\overline{n}$ where $\overline{n} = 1^{n+1} = 111\ldots1$ (n + 1 occurrences of 1).

<u>Definition 1.13.</u> To each k-tuple $(n_1, n_2, \ldots, n_k)$ of non-negative integers, associate the tape expression $\overline{(n_1, n_2, \ldots, n_k)}$ where

$$\overline{(n_1, n_2, \ldots, n_k)} = \overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k .$$

For example, by the above definitions, it follows that

$$\overline{4} = 1^{4+1} = 11111$$

and

$$\overline{(0,3,2)} = \overline{0}B\overline{3}B\overline{2} = 1B1111B111.$$

<u>Definition 1.14.</u> Let P be any expression. Then <P> will be the number of occurrences of the symbol 1 in the expression P.

As a consequence of this definition, it is obvious that for expressions P and Q, <PQ> = <P> + <Q>. Also, for any positive integer m, $\langle \overline{m - 1} \rangle = m$.

Definition 1.15. Let Z be a Turing machine and let n be a positive integer. Associate with Z an n-ary function ψ_Z^n in the following way.

For each n-tuple $(m_1,m_2,...,m_n)$ let

$$\alpha_1 = q_1(\overline{m_1,m_2,...,m_n}),$$

then

(1) If there exists a computation $\alpha_1, \alpha_2, \ldots, \alpha_k$ of Z, set

$$\psi_Z^n(m_1,m_2,...,m_n) = <\alpha_k> = <Res_Z(\alpha_1)>.$$

(2) If the above does not hold, that is, if $Res_Z(\alpha_1)$ is not defined, then leave ψ_Z^n at $(m_1,m_2,...,m_n)$ undefined.

When n = 1, ψ_Z will be written instead of ψ_Z^1.

Definition 1.16. Let f be an n-ary function defined on a subset D of J^n, then f is called a partial function. Should f be defined on the whole of J^n, then f is called a total function.

Taking the usual definition for equality of functions, to say two partial functions are equal implies, among other things, that their domains are identical.

Definition 1.17. Let f be an n-ary partial function and suppose there exists a Turing machine Z such that

$$f = \psi_Z^n,$$

then f is said to be partially computable and Z is said to partially compute f. Should f be total, then f is said to be computable and Z is said to compute f.

The preceding definitions show how the computability of functions can be expressed in terms of Turing machines.

Following are some examples of computable and partially computable functions.

Example 1.1. The successor function S, defined on J by

$$S(x) = x + 1,$$

is a computable function.

Let $m \varepsilon J$ and choose a Turing machine Z such that $q_1\overline{m}$ is terminal with respect to Z. Then $q_1\overline{m}$ is a computation of Z and

$$\psi_Z(m) = \langle q_1\overline{m}\rangle = \langle 1^{m+1}\rangle = m + 1 = S(m).$$

Since m was arbitrary, S is computable.

Example 1.2. The total function σ, defined on J^2 by

$$\sigma(x,y) = x + y,$$

is a computable function.

Let $(m,n) \varepsilon J^2$ and let Z be the Turing machine consisting of the following quadruples:

$$q_1 1\ B\ q_1$$
$$q_1 B\ R\ q_2$$
$$q_2 1\ R\ q_2$$
$$q_2 B\ R\ q_3$$
$$q_3 1\ B\ q_3$$

and let

$$\alpha_1 = q_1\overline{(m,n)} = q_1\overline{m}B\overline{n}.$$

Then,

$$\begin{aligned}\alpha_1 &= q_1 1^{m+1} B 1^{n+1} \\ &\rightarrow q_1 B 1^m B 1 1^n \\ &\rightarrow B q_2 1^m B 1 1^n \\ &\rightarrow \ . \ \ . \ \ . \\ &\rightarrow B 1^m q_2 B 1 1^n \\ &\rightarrow B 1^m B q_3 1 1^n \\ &\rightarrow B 1^m B q_3 B 1^n,\end{aligned}$$

which is terminal with respect to Z. Whence,

$$\psi_Z^2(m,n) = \langle B1^m Bq_3 B1^n \rangle = m + n = \sigma(m,n)$$

and so σ is computable.

Example 1.3. The n-ary function $U_i^{\ n}$, defined on J^n by

$$U_i^{\ n}(x_1, x_2, \ldots, x_n) = x_i$$

for $1 \leq i \leq n$, is a computable function.

Let $(m_1, m_2, \ldots, m_n) \in J^n$ and let

$$\alpha_1 = q_1 \overline{(m_1, \ldots, m_i, \ldots, m_n)} = 1^{m_1+1} B \ldots B 1^{m_i+1} B \ldots B 1^{m_n+1}.$$

If a Turing machine Z can be constructed to erase all blocks 1^{m_j+1}, $j \neq i$, and only the initial 1 from the i-th block, then clearly $Res_Z(\alpha_1)$ will be 1^{m_i} and

$$\psi_Z^n(m_1, m_2, \ldots, m_n) = \langle Res_Z(\alpha_1) \rangle = m_i = U_i^{\ n}(m_1, m_2, \ldots, m_n)$$

so the computability of $U_i^{\ n}$ will be established.

The required Turing machine is given by the following quadruples, where j runs through all integers not equal to i such that $1 \leq j \leq n$:

$q_j 1 B q_{2n+j}$

$q_j B R q_{j+1}$

$q_{2n+j} B R q_j$ (erase the j-th block of 1's)

$q_i 1 B q_i$

$q_i B R q_{2n+i}$ (erase the initial 1 in the i-th block)

$q_{2n+i} 1 R q_{2n+i}$

$q_{2n+i} B R q_{i+1}$ (proceed to the i+1st block of 1's).

A computation will terminate in machine state q_{n+1} since each quadruple begins with q_k, where $1 \leq k \leq n$ or $k > 2n$. Hence, U_i^n is computable.

Two more examples are given below. However, they will be discussed only briefly.

Example 1.4. The partial function f, defined by

$$f(x,y) = x - y,$$

is partially computable.

Let (m,n) be any ordered pair in J^2 and let

$$\alpha_1 = q_1\overline{(m,n)} = q_1 1^{m+1} B 1^{n+1}.$$

A Turing machine could be constructed to erase a 1 from the right-hand block of 1's each time a 1 is erased from the left-hand block of 1's, stopping or continuing indefinitely if the right-hand or left-hand block is exhausted first, respectively. Hence, f would only be partially computable since it would not be defined for those ordered pairs (m,n) of J^2 with $m < n$.

If the Turing machine had been constructed to erase everything and stop, should the left-hand block have been exhausted first, then it would compute the total function δ, defined by

$$\delta(x,y) = x \dot{-} y,$$

where

$$x \dot{-} y = x - y \text{ if } x \geq y \text{ and}$$

$$x \dot{-} y = 0 \text{ if } x < y.$$

This function is referred to as the proper subtraction function and is called the completion of the partially computable subtraction function $f(x,y) = x - y$.

Example 1.5. The function g, defined on J^2 by

$$g(x,y) = (x + 1)(y + 1),$$

is a computable function.

In as much as $0y = x0 = 0$, $(x + 1)(y + 1)$ will be easier to consider than xy. However, the computability of xy will be established later on.

Construction of the Turing machine to compute g is based on the fact that

$$(x + 1)(y + 1) = (y + 1) + (y + 1) + . . . + (y + 1)$$

$x + 1$ times.

Let $(m,n) \in J^2$ and let

$$\alpha_1 = q_1(m,n) = q_1 1^{m+1} B 1^{n+1}.$$

Then a Turing machine could be constructed to erase the leftmost 1, and for each 1 remaining in the left-hand block, erase it and at the same time copy the rightmost block on the left. This would require using special markers in order to shift the copied 1's to the left to make room for the next 1 to be copied from the rightmost block.

This would result in m + 1 blocks of n + 1 1's. Hence,

$$\psi_Z^2(m,n) = (m + 1)(n + 1) = g(m,n)$$

and the computability of g would be established.

Definition 1.9 will now be extended to include the more general form of Turing machines; namely, those incorporating quadruples of the form $q_i S_j q_k q_m$. This will result in the more general notion of computability, that of relative computability.

In effect, the quadruple $q_i S_j q_k q_m$ will allow the Turing machine to choose between alternate paths in the course of its operation. This, in the sense that, for a given set of integers A, the machine may inquire as to the membership of an integer n in A. If $n \in A$ the machine will enter into machine state q_k, hence, one path of operation. If $n \notin A$, the machine will enter into machine state q_m, whence, the alternate path of operation.

This is made precise by the following definition, where A denotes an arbitrary but fixed set of non-negative integers.

<u>Definition 1.18.</u> Let Z be a Turing machine and let α, β, be instantaneous descriptions. Then $\alpha \underset{A}{\rightarrow} \beta(Z)$ will be written, provided there exist tape expressions P and Q (possibly empty) such that α is $Pq_i S_j Q$, $q_i S_j q_k q_m \in Z$ and either

(1) $\langle\alpha\rangle \in A$, in which case β is $Pq_k S_j Q$, or

(2) $\langle\alpha\rangle \notin A$ and β is $Pq_m S_j Q$.

Definition 1.19. An instantaneous description α is said to be final with respect to Z, Z a Turing machine, provided α is of the form Pq_iS_jQ, for tape expressions P and Q (possibly empty), and Z has no quadruple whose first two symbols are q_iS_j.

Theorem 1.3. If Z is a simple Turing machine, then an instantaneous description α is terminal with respect to Z if and only if α is final with respect to Z.

Proof: Obvious

Theorem 1.4. Let Z be a Turing machine and α an instantaneous description, then α is final with respect to Z if and only if

(1) α is terminal with respect to Z and

(2) For each set of non-negative integers A, there is no instantaneous description β such that $\alpha \underset{A}{\rightarrow} \beta(Z)$.

Proof: If Z is simple the theorem follows immediately from Theorem 1.3. Therefore, suppose Z is not simple and α is of the form Pq_iS_jQ. If α is final, then Z contains no quadruple of the form q_iS_j— — and (1), (2) are obvious.

Conversely, if (1) and (2) hold and α is Pq_iS_jQ, then by (1), the only quadruple in Z beginning with q_iS_j must be $q_iS_jq_kq_m$ for some q_k and q_m. But by (2) this is impossible. Hence, α is final.

Definition 1.20. Let Z be a Turing machine and let A be a set of non-negative integers. Then an A-computation of Z shall mean a finite sequence $\alpha_1, \alpha_2, \ldots, \alpha_k$ of instantaneous descriptions such that

$$\alpha_i \rightarrow \alpha_{i+1}(Z) \text{ or } \alpha_i \underset{A}{\rightarrow} \alpha_{i+1}(z)$$

for each i, $1 \leq i < k$, with α_k being final with respect to Z.

If such be the case, α_k will be called the A-resultant of α_1 with respect to Z and this will be symbolized by

$$\alpha_k = \mathrm{Res}_Z{}^A(\alpha_1).$$

Should Z be a simple Turing machine, then the computation will be independent of A and

$$\mathrm{Res}_Z(\alpha_1) = \mathrm{Res}_Z{}^A(\alpha_1).$$

Definition 1.21. Let Z be a Turing machine and A an arbitrary set of non-negative integers. For each positive integer n, associate with Z an n-ary function $\psi^n_{Z;A}$ as follows:

For each n-tuple $(m_1,m_2,\ldots,m_n)$ set

$$\alpha_1 = q_1\overline{(m_1,m_2,\ldots,m_n)},$$

then

(1) If there exists an A-computation of α_1 with respect to Z let

$$\psi^n_{Z;A}(m_1,m_2,\ldots,m_n) = \langle \mathrm{Res}_Z{}^A(\alpha_1)\rangle.$$

(2) If the above does not hold, that is, if $\mathrm{Res}_Z{}^A(\alpha_1)$ is not defined, then leave $\psi^n_{Z;A}$ at $(m_1,m_2,\ldots,m_n)$ undefined.

In case n = 1 write $\psi_{Z;A}$ in place of $\psi^1_{Z;A}$.

If Z is a simple Turing machine, then $\psi^n_{Z;A}$ is independent of A and

$$\psi^n_{Z;A} = \psi^n_Z.$$

Definition 1.22. Let f be an n-ary function defined on a subset D of J^n. If there exists a Turing machine Z such that for some subset A of J,

$$f = \psi^n_{Z;A},$$

then f is said to be partially A-computable and Z is said to partially A-compute f. Should $D = J^n$, then f is said to be A-computable and Z is said to A-compute f.

Theorem 1.5. Let f be an n-ary function, then

(1) If f is partially computable, it is partially A-computable.

(2) If f is computable, then it is A-computable.

Proof: This follows immediately from the fact that, if f is partially computable or computable, then ψ^n_Z is independent of A, whence,

$$f = \psi^n_Z = \psi^n_{Z;A}.$$

Theorem 1.6. Let Z be a Turing machine, then there exists a simple Turing machine Z* such that for the empty set ϕ,

$$\psi^n_{Z*} = \psi^n_{Z;\phi}.$$

Proof: If Z is simple, choose Z* = Z. If Z contains quadruples of the form $q_i S_j q_k q_m$, then choose Z* to be Z with each quadruple of the form $q_i S_j q_k q_m$ in Z replaced by quadruples of the form $q_i S_j S_j q_m$. Thus, Z* is simple and since ϕ is empty, $<\alpha> \notin \phi$ for all instantaneous descriptions α, so clearly

$$\psi^n_{Z*} = \psi^n_{Z;\phi}.$$

Theorem 1.7. Let f be an n-ary function, then

(1) The function f is partially computable if and only if it is partially ϕ-computable.

(2) The function f is computable if and only if it is ϕ-computable.

Proof: This follows directly from Theorems 1.5 and 1.6.

Definition 1.23. Let S be a set and define the characteristic function of S by

$$C_S(x) = 0 \text{ if and only if } x \varepsilon S \text{ and}$$
$$C_S(x) = 1 \text{ if and only if } x \notin S.$$

Definition 1.24. Let S be a set, then S is said to be computable or A-computable, according as its characteristic function C_S is computable or A-computable.

Theorem 1.8. For every set A of non-negative integers, A is A-computable.

Proof: Let Z be the Turing machine consisting of the quadruples:

$$q_1 1\ B\ q_1$$
$$q_1 B\ q_2 q_3$$
$$q_2 B\ R\ q_4$$
$$q_4 1\ B\ q_2$$
$$q_3 B\ R\ q_5$$
$$q_5 1\ B\ q_3$$
$$q_5 B\ 1\ q_3.$$

Let $\alpha_1 = q_1\overline{n}$, then

$$\alpha_1 = q_1\overline{n} = q_1 1^{n+1} \rightarrow q_1 B 1^n.$$

Now $\langle q_1 B1^n \rangle = n$ so suppose $n \in A$, then

$$\begin{aligned} q_1 B1^n &\underset{A}{\to} q_2 B1^n \\ &\to Bq_4 1^n \\ &\to Bq_2 B1^{n-1} \\ &\to \cdots \\ &\to B^{n+1} q_4 B, \end{aligned}$$

which is final. But $\langle B^{n+1} q_4 B \rangle = 0$, whence,

$$\psi_{Z;A}(n) = 0 = C_A(n).$$

For the case when $n \notin A$, then

$$\begin{aligned} q_1 B1^n &\underset{A}{\to} q_3 B1^n \\ &\to Bq_5 1^n \\ &\to Bq_3 B1^{n-1} \\ &\to \cdots \\ &\to B^{n+2} q_5 B \\ &\to B^{n+2} q_3 1, \end{aligned}$$

which is final. Hence, $\langle B^{n+2} q_3 1 \rangle = 1$ and

$$\psi_{Z;A}(n) = 1 = C_A(n).$$

Therefore, A is A-computable.

In view of Theorem 1.7, it is evident that computability and partial computability are special cases of the more general notions, A-computability and partial A-computability, respectively. Therefore, only Turing machines involving A-computations will be considered from now on.

Additional properties of Turing machines

In as much as Turing machines can perform computations on instantaneous descriptions, which involve ordered n-tuples, it is conceivable that the output from one Turing machine may be used as input for some other Turing machine. This notion leads to the subsequent definitions. However, the following conventions will first be adopted.

Final blanks in an instantaneous description will be omitted except for the case of that blank, if any, preceded by a q_i. On the other hand, an initial blank will not be omitted. Thus, $S_3 11 S_2 q_3 1$ will be written instead of $S_3 11 S_2 q_3 1BB$, but the expression $BS_3 1 q_5 B$ must remain unchanged.

Definition 1.25. Let Z be a Turing machine and let $\theta(Z)$ denote the largest integer such that $q_{\theta(Z)}$ is a machine state of Z. Then for each positive integer n, Z is said to be n-regular, provided no quadruple of Z begins with $q_{\theta(Z)}$ and for any n-tuple $(m_1, m_2, \ldots, m_n)$, whenever $Res_Z^A[q_1 \overline{(m_1, m_2, \ldots m_n)}]$ is defined, it has the form $q_{\theta(Z)} \overline{(t_1, t_2, \ldots, t_s)}$ for some positive integer s and suitable t_i's.

Here, of course, $q_{\theta(Z)} \overline{(t_1, t_2, \ldots, t_s)}$ may contain additional occurrences of B on the right but $q_{\theta(Z)}$ must be the leftmost symbol.

Definition 1.26. Let Z be a Turing machine and for each integer $n \geq 0$, define Z^n to be the Turing machine obtained from Z by replacing each machine state q_i in Z by machine state q_{n+i}.

From this definition it follows that $Z^0 = Z$.

Theorem 1.9. Let Z be a Turing machine, then there exists a Turing machine Z* such that, for each integer $n > 0$, Z* is n-regular and in fact

$$\mathrm{Res}_{Z*}{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{\theta(Z*)}\overline{\psi^n_{Z;A}(m_1,m_2,\ldots,m_n)}.$$

<u>Proof:</u> Let λ, μ denote the first two symbols S_2, S_3, . . . which are not in the alphabet of Z and let Z_1 consist of the quadruples:

$q_1 1\ L\ q_1$

$q_1 B\ \lambda\ q_1$

$q_1 \lambda\ R\ q_2$ (print λ on the left)

$q_2 1\ R\ q_2$

$q_2 B\ R\ q_3$

$q_3 1\ R\ q_2$

$q_3 B\ L\ q_4$ (move right to a double blank)

$q_4 B\ \mu\ q_5$ (print μ on the right)

$q_5 \mu\ L\ q_5$

$q_5 1\ L\ q_5$

$q_5 B\ L\ q_5$

$q_5 \lambda\ R\ q_6$ (move left and find λ).

Then with respect to Z_1,

$$\alpha_1 = q_1(\overline{m_1,m_2,\ldots,m_n}) \to \ldots \to \lambda q_6(\overline{m_1,m_2,\ldots,m_n})\mu,$$

which is final.

Now Z^5 will be like Z except it will begin with machine state q_6 instead of machine state q_1. Let $k = \theta(Z^5)$ and let Z_2 be the quadruples of Z^5 together with the following quadruples, where q_i may be any machine state of Z^5:

$q_i \lambda\ B\ q_{k+i}$ (erase λ)

$q_{k+i} B\ L\ q_{2k+i}$

$q_{2k+i} B\ \lambda\ q_{2k+i}$ (move λ left one square)

$q_{2k+i} \lambda\ R\ q_i$ (resume main computation)

$q_i \mu\ B\ q_{3k+i}$ (erase μ)

$q_{3k+i} B\ R\ q_{4k+i}$

$q_{4k+i} B\ \mu\ q_{4k+i}$ (move μ right one square)

$q_{4k+i} \mu\ L\ q_i$ (resume main computation).

This last set of quadruples allows for a computation of Z^5 to remain within the markers λ and μ.

Now should $\mathrm{Res}_Z{}^A[q_1(\overline{m_1,m_2,\ldots,m_n})]$ be defined and if

$$\alpha = \mathrm{Res}_Z{}^A[q_1(\overline{m_1,m_2,\ldots,m_n})],$$

then with respect to Z_2

$$\lambda q_6(\overline{m_1,m_2,\ldots,m_n})\mu \rightarrow \ldots \rightarrow \lambda\alpha\mu$$

which is final.

Moreover, if $\mathrm{Res}_Z{}^A[q_1(\overline{m_1,m_2,\ldots,m_n})]$ is undefined, then so is $\mathrm{Res}_{Z_2}{}^A[\lambda q_1(\overline{m_1,m_2,\ldots,m_n})\mu]$.

Let $t = 5k + 1$ and let Z_3 consist of all quadruples of the form

$$q_i S_j S_j q_t,$$

where q_i is any machine state of Z_2, S_j is in the alphabet of Z_2 but such that no quadruple of Z_2 starts with $q_i S_j$. This is possible, elsewise, no instantaneous description would be final with respect to Z_2. Now, if $\lambda P q_i Q \mu$ is any instantaneous description which is final with respect to Z_2, then

$$\lambda P q_i Q \mu \rightarrow \lambda P q_t Q \mu \ (Z_3)$$

which is final with respect to Z_3.

Finally, let Z_4 consist of the following quadruples, where S denotes any symbol in the alphabet of Z other than 1 or B:

$q_t 1\ L\ q_t$

$q_t S\ L\ q_t$

$q_t B\ L\ q_t$

$q_t \lambda\ R\ q_{t+1}$ (find the left marker λ)

$q_{t+1} S\ B\ q_{t+1}$

$q_{t+1} B\ R\ q_{t+1}$

$q_{t+1} 1\ B\ q_{t+2}$

$q_{t+1} \mu\ B\ q_{t+4}$ (move right looking for a 1)

$q_{t+2} B\ L\ q_{t+2}$

$q_{t+2} 1\ R\ q_{t+3}$

$q_{t+2} \lambda\ R\ q_{t+3}$ (find the block of 1's)

$q_{t+3} B\ 1\ q_{t+3}$

$q_{t+3} 1\ R\ q_{t+1}$ (add 1 to the block of 1's)

$q_{t+4} B\ L\ q_{t+4}$

$q_{t+4} 1\ L\ q_{t+4}$

$q_{t+4} \lambda\ 1\ q_{t+5}$ (add 1 and terminate).

Now Z_4 will collect the 1's on the tape into a single block, add an additional 1, erase everything else and terminate. Hence, taking

$$Z^* = Z_1 \cup Z_2 \cup Z_3 \cup Z_4,$$

then

$$\mathrm{Res}_{Z^*}{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{t+5}1^{<\mathrm{Res}_Z{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})]>+1}$$

$$= q_{\theta(Z^*)}\overline{\psi^n_{Z;A}(m_1,m_2,\ldots,m_n)} \;.$$

Since Z* is clearly n-regular, the theorem follows.

Theorem 1.10. For each n-regular Turing machine Z and each integer $k > 0$, there exists a $(k + n)$-regular Turing machine Z_k such that whenever

$$\mathrm{Res}_Z{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{\theta(Z)}(\overline{t_1,t_2,\ldots,t_s}),$$

it is also true that

$$\mathrm{Res}_{Z_k}{}^{A}[q_1(\overline{r_1,r_2,\ldots,r_k,m_1,m_2,\ldots,m_n})] = q_{\theta(Z_k)}(\overline{r_1,r_2,\ldots,r_k,t_1,t_2,\ldots,t_s}).$$

Furthermore, whenever $\mathrm{Res}_Z{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})]$ is undefined, so is $\mathrm{Res}_{Z_k}{}^{A}[q_1(\overline{r_1,r_2,\ldots,r_k,m_1,m_2,\ldots,m_n})]$.

Proof: Let λ and μ denote distinct symbols not in the alphabet of Z and let Y_1 consist of the following quadruples, where i runs through all integers such that $1 < i \leq k$:

$q_1 1\ \lambda\ q_1$

$q_1\lambda\ R\ q_2$ (replace the leftmost 1 by the marker λ)

$q_i 1\ \mu\ q_i$

$q_i\mu\ R\ q_i$

$q_i B\ R\ q_{i+1}$ (replace 1 by μ for $1 < i \leq k$)

$q_{k+1} 1 \; \mu \; q_{k+1}$

$q_{k+1} \mu \; R \; q_{k+1}$

$q_{k+1} B \; \mu \; q_{k+2}$

$q_{k+2} \mu \; R \; q_{k+3}$ (replace the k-th block of 1's by μ's).

Now, with respect to Y_1

$$q_1(\overline{r_1, r_2, \ldots, r_k, m_1, m_2, \ldots, m_n}) \to \ldots \to P\mu q_{k+3}(\overline{m_1, m_2, \ldots, m_n}),$$

where P is $\lambda\mu^{r_1}B\mu^{r_2+1}B \ldots B\mu^{r_k+1}$.

Let $p = \theta(Z^{k+2})$ and let Y_2 consist of the following quadruples, where q_i may be any machine state of Z^{k+2}:

$q_i \mu \; 1 \; q_{p+i}$ (interupt main computation)

$q_{p+i} 1 \; L \; q_{p+i}$

$q_{p+i} \mu \; L \; q_{p+i}$

$q_{p+i} B \; L \; q_{p+i}$

$q_{p+i} \lambda \; B \; q_{2p+i}$ (search for the marker λ)

$q_{2p+i} B \; L \; q_{3p+i}$

$q_{3p+i} B \; \lambda \; q_{3p+i}$

$q_{3p+i} \lambda \; R \; q_{4p+i}$ (move λ left one square)

$q_{4p+i} B \; R \; q_{5p+i}$

$q_{4p+i} \mu \; R \; q_{5p+i}$

$q_{4p+i} 1 \; B \; q_i$ (resume main computation)

$q_{5p+i} \mu \; L \; q_{6p+i}$ (encountering μ, prepare to copy it)

$q_{5p+i} B \; L \; q_{7p+i}$ (encountering B, prepare to copy it)

$q_{5p+i} 1 \; L \; q_{6p+i}$ (encountering 1, prepare to resume main computation)

$q_{6p+i}\mu \ \mu \ q_{8p+i}$

$q_{6p+i}B \ \mu \ q_{8p+i}$ (copy μ)

$q_{7p+i}\mu \ B \ q_{8p+i}$

$q_{7p+i}B \ B \ q_{8p+i}$ (copy B)

$q_{8p+i}\mu \ R \ q_{4p+i}$

$q_{8p+i}B \ R \ q_{4p+i}$ (repeat until a 1 is encountered).

Thus, Y_2 will move the first k blocks of 1's one square to the left whenever Z^{k+2} tries to print over them. Hence, taking

$$Y_3 = Y_1 \cup Y_2,$$

then with respect to Y_3

$$\begin{aligned} q_1(\overline{r_1,r_2,\ldots,r_k,m_1,m_2,\ldots,m_n}) &\to \ldots \\ &\to Q\mu q_{k+3}(\overline{m_1,m_2,\ldots,m_n}) \\ &\to \ldots \\ &\to Q\mu q_p(\overline{t_1,t_2,\ldots,t_s}) \end{aligned}$$

which is final, where Q is $\lambda\mu^{r_1}B\mu^{r_2+1}B \ldots B\mu^{r_k+1}$. Moreover, it will be defined whenever $\mathrm{Res}_Z^A[q_1(\overline{m_1,m_2,\ldots,m_n})]$ is defined. Elsewise, there can be no A-computation of Y_3 beginning with the instantaneous description $q_1(\overline{r_1,r_2,\ldots,r_k,m_1,m_2,\ldots,m_n})$.

It remains to construct a Turing machine Z_k which will compute like Y_3 but in addition, replace all occurrences of λ and μ by 1.

Let $v = \theta(Y_3)$ and choose Z_k to be Y_3 together with the following quadruples:

$q_p 1\ L\ q_p$

$q_p \mu\ B\ q_{v+1}$ (restore B)

$q_{v+1} B\ L\ q_{v+1}$

$q_{v+1} \mu\ 1\ q_{v+1}$

$q_{v+1} 1\ L\ q_{v+1}$

$q_{v+1} \lambda\ 1\ q_{v+2}$ (restore each λ and μ by 1).

Since $\theta(Z_k) = v + 2$, then with respect to Z_k

$$q_1 \overline{(r_1, r_2, \ldots, r_k, m_1, m_2, \ldots, m_n)} \rightarrow \ldots \rightarrow q_{\theta(Z_k)} \overline{(r_1, r_2, \ldots, r_s, t_1, t_2, \ldots, t_s)}$$

which is final. Hence, the theorem is established.

Theorem 1.11. For each integer $n > 0$ and each integer $k \geq 0$, there exists a $(k + n)$-regular Turing machine C_k such that

$$\text{Res}_{C_k}{}^{A}[q_1 \overline{(r_1, r_2, \ldots, r_k, m_1, m_2, \ldots, m_n)}]$$
$$= q_{\theta(C_k)} \overline{(m_1, m_2, \ldots, m_n, r_1, r_2, \ldots, r_k, m_1, m_2, \ldots, m_n)}.$$

Theorem 1.12. Let n be a positive integer, then for each integer $k > 0$ there exists a $(k + n)$-regular Turing machine R_k such that

$$\text{Res}_{R_k}{}^{A}[q_1 \overline{(r_1, r_2, \ldots, r_k, m_1, m_2, \ldots, m_n)}]$$
$$= q_{\theta(R_k)} \overline{(m_1, m_2, \ldots, m_n, r_1, r_2, \ldots, r_k)}.$$

Construction of the Turing machines satisfying the conditions of Theorems 1.11 and 1.12 is straight forward but quite long. Therefore, these two theorems will be stated without proof.

Theorem 1.13. For each n-regular Turing machine Z, there exists an n-regular Turing machine Z* such that whenever

$$\text{Res}_Z{}^A[q_1\overline{(m_1,m_2,\ldots,m_n)}] = q_{\theta(Z)}\overline{(t_1,t_2,\ldots,t_s)}$$

it is also true that

$$\text{Res}_{Z*}{}^A[q_1\overline{(m_1,m_2,\ldots,m_n)}] = q_{\theta(Z*)}\overline{(t_1,t_2,\ldots,t_s,m_1,m_2,\ldots,m_n)}.$$

Furthermore, whenever $\text{Res}_Z{}^A[q_1\overline{(m_1,m_2,\ldots,m_n)}]$ is defined or undefined so is $\text{Res}_{Z*}{}^A[q_1\overline{(m_1,m_2,\ldots,m_n)}]$ defined or undefined, respectively.

Proof: By Theorem 1.10, there exists a 2n-regular Turing machine Y such that

$$\text{Res}_Y{}^A[q_1\overline{(m_1,m_2,\ldots,m_n,m_1,m_2,\ldots,m_n)}]$$
$$= q_{\theta(Y)}\overline{(m_1,m_2,\ldots,m_n,t_1,t_2,\ldots,t_s)}.$$

Using Theorem 1.11 and 1.12, taking

$$Z* = C_0 \cup Y^{\theta(C_0)-1} \cup R_n{}^{\theta(C_0)-2+\theta(Y)},$$

then with respect to C_0

$$q_1\overline{(m_1,m_2,\ldots,m_n)} \to \ldots \to q_{\theta(C_0)}\overline{(m_1,m_2,\ldots,m_n,m_1,m_2,\ldots,m_n)},$$

and with respect to $Y^{\theta(C_0)-1}$

$$q_{\theta(C_0)}\overline{(m_1,m_2,\ldots,m_n,m_1,m_2,\ldots,m_n)}$$
$$\to \ldots$$
$$\to q_{\theta(Y^{\theta(C_0)-1})}\overline{(m_1,m_2,\ldots,m_n,t_1,t_2,\ldots,t_s)}.$$

Finally, with respect to $R_n{}^{\theta(C_0)-2+\theta(Y)}$

$$q_{\theta(Y^{\theta(C_0)-1})}\overline{(m_1,m_2,\ldots,m_n,t_1,t_2,\ldots,t_s)}$$
$$\to \ldots$$
$$\to q_{\theta(Z*)}\overline{(t_1,t_2,\ldots,t_s,m_1,m_2,\ldots,m_n)}$$

which is final with respect to Z*. The second part of the theorem follows immediately.

Theorem 1.14. Let $Z_1, Z_2, \ldots, Z_k$ be Turing machines, then for each integer n > 0 there exists an n-regular Turing machine Z* such that

$$\mathrm{Res}_{Z*}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})]$$
$$= q_{\theta(Z*)}(\overline{\psi^n_{Z_1;A}(m_1,m_2,\ldots,m_n),\ldots,\psi^n_{Z_k;A}(m_1,m_2,\ldots,m_n)}).$$

Proof: The proof will be by induction on k.

Basis: Suppose k = 1, then this reduces to nothing more than Theorem 1.9.

Induction step: Suppose the assertion is true for k = j. Let the Turing machines $Z_1, Z_2, \ldots, Z_{j+1}$ be given and set

$$t_i = \psi^n_{Z_i;A}(m_1,m_2,\ldots,m_n),$$

where $1 \leq i \leq j + 1$.

By the induction hypothesis, there exists an n-regular Turing machine Y_1 such that

$$\mathrm{Res}_{Y_1}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{\theta(Y_1)}(\overline{t_1,t_2,\ldots,t_j}).$$

Hence, by Theorem 1.13, there exists an n-regular Turing machine Y_2 such that

$$\mathrm{Res}_{Y_2}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{\theta(Y_2)}(\overline{t_1,t_2,\ldots,t_j,m_1,m_2,\ldots,m_n}).$$

Moreover, by Theorem 1.9, there exists an n-regular Turing machine Y_3 such that

$$\mathrm{Res}_{Y_3}{}^{A}[q_1(\overline{m_1,m_2,\ldots,m_n})] = q_{\theta(Y_3)}\overline{t}_{k+1}.$$

Finally, by Theorem 1.10, there exists an n-regular Turing machine Y_4 such that

$$\mathrm{Res}_{Y_4}{}^{A}[q_1(\overline{t_1,t_2,\ldots,t_j,m_1,m_2,\ldots,m_n})] = q_{\theta(Y_4)}(\overline{t_1,t_2,\ldots,t_j,t_{j+1}}).$$

By taking

$$Z^* = Y_2 \cup Y_4{}^{\theta(Y_2)-1},$$

then the assertion is true for $k = j + 1$, hence, the theorem.

RECURSIVE FUNCTIONS

Composition and minimalization

Two operations, composition and minimalization, will now be considered which afford a means for constructing a large class of Turing computable functions. Moreover, by applying Theorems 1.9-1.14, it will be possible to show that functions from this class are computable or partially computable without having to appeal directly to the definition of computability.

Let f and g be unary functions, then by composition of f with g will be the function h, defined by

$$h(x) = f(g(x)),$$

where it is understood that the domain of h consists of those values of x, in the domain of g, for which g(x) is in the domain of f. This is made more general by the following definition.

Definition 2.1. Let f be an m-ary function and let $g_1, g_2, \ldots, g_m$ be m n-ary functions. Then the operation of composition gives a new function h, defined by

$$h(x_1,x_2,\ldots,x_n)$$
$$= f(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n)).$$

It is understood, of course, that the domain of h is precisely those n-tuples in the domain of each g_i such that the m-tuple $(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n))$ is in the domain of f.

Theorem 2.1. Let f be an n-ary function and $g_1, g_2, \ldots, g_m$ m n-ary functions. Suppose these functions are partially A-computable for some subset A of J. Then the function h, defined by

$$h(x_1,x_2,\ldots,x_n)$$
$$= f(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n)) \quad (1)$$

is partially A-computable.

Proof: Let Z be the Turing machine which partially A-computes f and let Z_i be the Turing machine which partially A-computes g_i, $i = 1,2,\ldots,m$. Therefore, $f = \psi_{Z;A}^{n}$ and for $i = 1,2,\ldots,m$ $g_i = \psi_{Z_i;A}^{n}$.

Now by Theorem 1.14, there exists an n-regular Turing machine Z* such that

$$\mathrm{Res}_{Z*}{}^{A}[q_1\overline{(x_1,x_2,\ldots,x_n)}]$$
$$= q_{\theta(Z*)}\overline{(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n))}.$$

Let $(x_1,x_2,\ldots,x_n)$ be an n-tuple satisfying (1) and let

$$Z' = Z* \cup Z^{\theta(Z*)-1}.$$

If $\alpha_1 = q_1\overline{(x_1,x_2,\ldots,x_n)}$, then with respect to Z'

$$\alpha_1 = q_1\overline{(x_1,x_2,\ldots,x_n)}$$
$$\to \ldots$$
$$\to q_{\theta(Z*)}\overline{(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n))}$$
$$\to \ldots$$
$$\to \alpha$$

where

$\langle\alpha\rangle = f(g_1(x_1,x_2,\ldots,x_n),g_2(x_1,x_2,\ldots,x_n),\ldots,g_m(x_1,x_2,\ldots,x_n))$.

Hence, if $(x_1,x_2,\ldots,x_n)$ satisfies (1), then $Res_{Z'}{}^{A}(\alpha_1)$ is defined, otherwise it is not defined. Thus, h is seen to be partially A-computable.

If the functions f, g_1, g_2, . . . , g_m are A-computable, then clearly h is A-computable. Hence, the following corollary.

Corollary 2.1. The class of partially A-computable functions and the class of A-computable functions are both closed under the operation of composition.

Example 2.1. The function μ, defined by

$$\mu(x,y) = xy,$$

is computable.

It has been shown in previous examples that the functions δ, ρ, S and $U_2^{\ 2}$, defined by

$$\begin{aligned} \delta(x,y) &= x \dot{-} y \\ \rho(x,y) &= (x + 1)(y + 1) \\ S(x) &= x + 1 \\ U_2^{\ 2}(x,y) &= y, \end{aligned}$$

are all computable.

Let g be the function defined by

$$\begin{aligned} g(x,y) &= S(U_2^{\ 2}(x,y)) \\ &= U_2^{\ 2}(x,y) + 1 \\ &= y + 1, \end{aligned}$$

then by Corollary 2.1, g is computable.

Let h be the function defined by

$$\begin{aligned} h(x,y) &= \delta(\rho(x,y), g(x,y)) \\ &= \rho(x,y) \dot{-} g(x,y) \\ &= (x + 1)(y + 1) \dot{-} (y + 1) \\ &= xy + x, \end{aligned}$$

then by Corollary 2.1, h is computable.

Finally, let μ be defined by

$$\begin{aligned} \mu(x,y) &= \delta(h(x,y), x) \\ &= h(x,y) \dot{-} x \\ &= (xy + x) \dot{-} x \\ &= xy. \end{aligned}$$

Hence, by Corollary 2.1, μ is computable.

Definition 2.2. Let f be an (n+1)-ary total function. Then the operation of minimalization gives a new function h, defined by

$$h(x_1,x_2,\ldots,x_n) = \min_y[f(y,x_1,x_2,\ldots,x_n) = 0].$$

That is, for a given n-tuple $(x_1,x_2,\ldots,x_n)$, h associates the least value of y for which

$$f(y,x_1,x_2,\ldots,x_n) = 0.$$

Definition 2.3. In Definition 2.2, if h is a total function, then f is called a regular function.

As in the case of composition, the operation of minimalization allows for the construction of a large class of computable and partially computable functions. This is characterized by the following theorem.

Theorem 2.2. Let f be an (n+1)-ary function that is total and A-computable. Then the function h, defined by

$$h(x_1,x_2,\ldots,x_n) = \min_y[f(y,x_1,x_2,\ldots,x_n) = 0],$$

is partially A-computable. Moreover, if f is regular, then h is A-computable.

Proof: A Turing machine will be constructed which successively computes $f(0,x_1,x_2,\ldots,x_n)$, $f(1,x_1,x_2,\ldots,x_n)$, . . . until a zero is obtained.

Let R be the Turing machine consisting of the quadruples:

$$q_1 1\ L\ q_1$$
$$q_1 B\ L\ q_2$$
$$q_2 B\ 1\ q_3.$$

Then with respect to R

$$q_1(\overline{x_1,x_2,\ldots,x_n}) \to \ldots \to q_3(\overline{0,x_1,x_2,\ldots,x_n}),$$

which is final.

By Theorems 1.9 and 1.13, there exists an (n+1)-regular Turing machine S such that

$$\mathrm{Res}_S^A[q_1(\overline{y,x_1,x_2,\ldots,x_n})] = q_{\theta(S)}(\overline{f(y,x_1,x_2,\ldots,x_n),y,x_1,x_2,\ldots,x_n}).$$

Therefore, if $N = \theta(S^2)$, then with respect to S^2

$$q_3(\overline{y,x_1,x_2,\ldots,x_n}) \to \ldots \to q_N(\overline{f(y,x_1,x_2,\ldots,x_n),y,x_1,x_2,\ldots,x_n}),$$

which is final.

Let T be the Turing machine consisting of the following quadruples:

$$q_N 1\ B\ q_N$$
$$q_N B\ R\ q_{N+1}$$
$$q_{N+1} 1\ 1\ q_{N+2}$$
$$q_{N+1} B\ R\ q_{N+4}.$$

Now if $f(y,x_1,x_2,\ldots,x_n) = k$, where $k > 0$, then with respect to T

$$q_N \overline{(f(y,x_1,x_2,\ldots,x_n),y,x_1,x_2,\ldots,x_n)} = q_N 1 1^k B \overline{(y,x_1,x_2,\ldots,x_n)}$$
$$\rightarrow \ .\ .\ .$$
$$\rightarrow q_{N+2} 1^k B \overline{(y,x_1,x_2,\ldots,x_n)},$$

which is final. However, should $f(y,x_1,x_2,\ldots,x_n) = 0$, then with respect to T

$$q_N \overline{(f(y,x_1,x_2,\ldots,x_n),y,x_1,x_2,\ldots,x_n)} = q_N 1 B \overline{(y,x_1,x_2,\ldots,x_n)}$$
$$\rightarrow \ .\ .\ .$$
$$\rightarrow q_{N+4} \overline{(y,x_1,x_2,\ldots,x_n)}.$$

Let Q be the Turing machine consisting of the quadruples:

$$q_{N+2} 1\ B\ q_{N+3}$$
$$q_{N+2} B\ 1\ q_3$$
$$q_{N+3} B\ R\ q_{N+2}.$$

Then with respect to Q

$$q_{N+2} 1^k B \overline{(y,x_1,x_2,\ldots,x_n)} \rightarrow\ .\ .\ .\ \rightarrow q_3 \overline{(y+1,x_1,x_2,\ldots,x_n)}.$$

Let U_i^m be the m-ary computable function, defined by

$$U_i^m(x_1,x_2,\ldots,x_m) = x_i,$$

where $1 \leq i \leq m$. Then by Theorem 1.9, there exists an (n+1)-reqular Turing machine Y such that

$$\begin{aligned} Res_Y^A[q_1\overline{(y,x_1,x_2,\ldots,x_n)}] &= q_{\theta(Y)}\overline{U_1^{n+1}(y,x_1,x_2,\ldots,x_n)} \\ &= q_{\theta(Y)}1^{y+1}. \end{aligned}$$

Finally, let W consist of all the quadruples of Y together with the quadruple

$$q_{\theta(Y)}1\ B\ q_{\theta(Y)}.$$

Then with respect to W^{N+3}, letting $K = \theta(W^{N+3})$,

$$q_{N+4}\overline{(y,x_1,x_2,\ldots,x_n)} \rightarrow \ldots \rightarrow q_K B 1^y.$$

Let

$$Z = R \cup S^2 \cup T \cup Q \cup W^{N+3}$$

and suppose $(x_1,x_2,\ldots,x_n)$ is arbitrary but fixed. Let

$$f(i,x_1,x_2,\ldots,x_n) = r_i,$$

for i = 1,2,... and suppose $r_0 \neq 0$, $r_1 \neq 0$, . . . , $r_{k-1} \neq 0$, $r_k = 0$. Then with respect to Z

$$\begin{aligned} &q_1\overline{(x_1,x_2,\ldots,x_n)} \\ &\rightarrow \ldots \\ &\rightarrow q_3\overline{(0,x_1,x_2,\ldots,x_n)} \qquad \text{(using R)} \end{aligned}$$

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_N(\overline{r_0,0,x_1,x_2,\ldots,x_n})$ (using S^2)

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_{N+2}(\overline{r_0-1,0,x_1,x_2,\ldots,x_n})$ (using T)

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_3(\overline{1,x_1,x_2,\ldots,x_n})$ (using Q)

$\rightarrow \quad . \quad . \quad .$

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_3(\overline{k,x_1,x_2,\ldots,x_n})$

$\rightarrow q_N(\overline{r_k,k,x_1,x_2,\ldots,x_n})$ (using S^2)

$= q_N 1B(\overline{k,x_1,x_2,\ldots,x_n})$

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_{N+4}(\overline{k,x_1,x_2,\ldots,x_n})$ (using T)

$\rightarrow \quad . \quad . \quad .$

$\rightarrow q_K B 1^k$ (using W^{N+3}).

Therefore,

$$\begin{aligned}\psi^n_{Z;A}(x_1,x_2,\ldots,x_n) &= \langle q_K B1^k \rangle \\ &= k \\ &= \min_y[f(y,x_1,x_2,\ldots,x_n) = 0] \\ &= h(x_1,x_2,\ldots,x_n).\end{aligned}$$

If $r_i \neq 0$ for all i, i = 1,2,... , then Z will never be in machine state q_{N+4} and will compute indefinitely. Thus, both $\psi^n_{Z;A}$ and h would be undefined at $(x_1,x_2,\ldots,x_n)$, hence, h is partially A-computable. If f is a regular function, the A-computablity of h is obvious.

Example 2.2. The function f, defined by

$$f(x) = [\sqrt{x}\,],$$

is computable, where [t] means the largest integer $\leq$ t.

Let x be an arbitrary element of J, then to say that y is the largest integer $\leq \sqrt{x}$ is equivalent to saying y is the largest integer such that $y^2 \leq x$. From this it follows that y is the minimum value for which $(y + 1)^2 > x$, or equivalently that y is the minimum value such that $(y + 1)^2 \dot{-} x$ is not zero. But, this is true if and only if y is the minimum value such that

$$1 \dot{-} ((y + 1)^2 \dot{-} x) = 0.$$

Therefore,

$$[\sqrt{x}\,] = \min_y[1 \dot{-} ((y + 1)^2 \dot{-} x) = 0],$$

which by Theorem 2.2 is computable since

$$1 \dot{-} ((y + 1)^2 \dot{-} x) = \delta(1,\delta(\mu(S(U_2^2(x,y)),S(U_2^2(x,y))),x))$$

is clearly total and by Corollary 2.1, computable.

Special classes of functions

Using the operations of composition and minimalization on an initial set of partially A-computable and A-computable functions, a certain class of Turing computable functions, which are of particular interest, can be obtained. This is characterized by the following definitions and theorems.

Definition 2.4. A function f is said to be A-partial recursive or partial recursive in A, provided it can be obtained from a finite number of applications of composition or minimalization on functions beginning with functions from the following list:

(1) $C_A(x)$, the characteristic function of the set A

(2) $S(x) = x + 1$

(3) $U_i^n(x_1,x_2,\ldots,x_n) = x_i$, $1 \leq i \leq n$

(4) $\sigma(x,y) = x + y$

(5) $\delta(x,y) = x \dot{-} y$

(6) $\mu(x,y) = xy$.

Theorem 2.3. The functions S, U_i^n, σ, δ and μ, in Definition 2.4, are computable; hence, partially computable, partially A-computable and A-computable.

Proof: Examples 1.1, 1.3, 1.2, 1.4 and 2.1 established the computability of these functions, respectively. The remainder of the assertion follows from Theorem 1.5.

Theorem 2.4. The characteristic function C_ϕ of the empty set ϕ, is computable; hence, partially computable, partially A-computable and A-computable.

Proof: Definition 1.24 and Theorem 1.8 imply C_ϕ is ϕ-computable. Whence, the assertion follows from Theorems 1.7 and 1.5.

Definition 2.5. A function is said to be partial recursive, provided it is ϕ-partial recursive.

Definition 2.6. A function is said to be A-recursive or recursive in A, provided it can be obtained from a finite number of applications of composition or minimalization on regular functions beginning with functions from the list of Definition 2.4.

<u>Theorem 2.5.</u> An A-recursive function is total and A-partial recursive.

<u>Proof:</u> Since all the functions listed in Definition 2.4 are total functions, this follows from definition.

Although no attempt will be made to establish the fact, the converse of Theorem 2.5 is also true. Since this is the case, the notion of A-partial recursive functions might seem artificial. However, they are considered for their relatión to computability as shown by the following theorem.

<u>Theorem 2.6.</u> Let f be a function, then

(1) If f is A-partial recursive, then it is partially A-computable.

(2) If f is partial recursive, then it is partially computable.

(3) If f is A-recursive, then it is A-computable.

(4) If f is recursive, then it is computable.

<u>Proof:</u> This follows from Theorems 2.1, 2.2, 2.3, 2.4 and definition.

Below is an example of a recursive function.

<u>Example 2.3.</u> The function f, defined by

$$f(x,y) = [x/y],$$

is recursive, where $[x/y]$ = the greatest integer $\leq x/y$ if $y \neq 0$ and $[x/y] = 0$ if $y = 0$. It is understood that x/y is a rational number.

Let

$$\lambda(x) = \delta(1,x)$$
$$= 1 \dot{-} x.$$

That is,

$$\lambda(0) = 1,$$

$$\lambda(x) = 0 \text{ if } x > 0.$$

Thus,

$$\begin{aligned} [x/y] &= \min_z[y = 0 \text{ or } y(z + 1) > x] \\ &= \min_z[y = 0 \text{ or } y(z + 1) \dot{-} x \neq 0] \\ &= \min_z[y = 0 \text{ or } \lambda(y(z + 1) \dot{-} x) = 0] \\ &= \min_z[y \cdot \lambda(\delta(\mu(y,S(z)),x)) = 0] \\ &= \min_z[\mu(y,\lambda(\delta(\mu(y,S(z)),x))) = 0]. \end{aligned}$$

Hence, f is recursive. Moreover, by Theorem 2.6, f is computable.

In view of Theorem 2.6, if f is a recursive function, then there exists an algorithm in the form of a Turing machine for computing the functional values of f. Furthermore, the converse of this also holds. Hence, the notions of computable and recursive functions are equivalent.

LITERATURE CITED

Davis, M. 1958. Computability and Unsolvability. McGraw-Hill Book Company, Inc. New York.