1 A Universal Program (4)

1.1 Recursively Enumerable Sets (4.4)

Review: Sets and Characteristic Functions

Given a predicate $P$ on a set $S$, there is a corresponding subset $R$ of $S$ consisting of all elements $a \in S$ for which $P(a) = 1$. We write

$$R = \{a \in S \mid P(a)\}.$$
Conversely, given a subset $R$ of a given set $S$, the expression $x \in R$ defines a predicate $P$ on $S$:

$$P(x) = \begin{cases} 1 & \text{if } x \in R \\ 0 & \text{if } x \notin R. \end{cases}$$

The predicate $P$ is called the characteristic function of the set $R$. Note the easy translations between the two notations:

$$\{x \in S \mid P(x) \& Q(x)\} = \{x \in S \mid P(x)\} \cap \{x \in S \mid Q(x)\},$$
$$\{x \in S \mid P(x) \lor Q(x)\} = \{x \in S \mid P(x)\} \cup \{x \in S \mid Q(x)\},$$
$$\{x \in S \mid \sim P(x)\} = S - \{x \in S \mid P(x)\}.$$

Sets and Classes of Functions

- The predicate HALT$(x, y)$ is the characteristic function of the set
  $$\{(x, y) \in N^2 \mid \text{HALT}(x, y)\}.$$

- A set $B \subseteq N^m$ is said to belong to some class of functions means that the characteristic function $P(x_1, \ldots, x_n)$ of $B$ belongs to the class in question.

- $B$ is computable or recursive is just to say that $P(x_1, \ldots, x_n)$ is a computable function.

- $B$ is a primitive recursive set if $P(x_1, \ldots, x_n)$ is primitive recursive.

**Theorem 4.1.** Let the sets $B, C$ belong to some PRC class $\mathcal{C}$. The so do the sets $B \cup C$, $B \cap C$, and $\bar{B}$. □

**Need Only Consider Subsets of $N$**

**Theorem 4.2.** Let $\mathcal{C}$ be a PRC class, and let $B$ be a subset of $N^m, m \geq 1$. Then $B$ belongs to $\mathcal{C}$ if and only if

$$B' = \{[x_1, \ldots, x_m] \in N \mid (x_1, \ldots, x_m) \in B\}$$

belongs to $\mathcal{C}$. □. Proof. If $P_B(x_1, \ldots, x_m)$ is the characteristic function of $B$, then

$$P_{B'} \Leftrightarrow P_B((x_1), \ldots, (x)_m) \& Lt(x) \leq m \& x > 0$$

is the characteristic function of $B'$. Clearly, $P_{B'}$ belongs to $\mathcal{C}$ if $P_B$ belongs to $\mathcal{C}$. On the other hand, if $P_{B'}(x)$ is the characteristic function of $B'$, then

$$P_B(x_1, \ldots, x_m) \Leftrightarrow P_{B'}([x_1, \ldots, x_m])$$

is the characteristic function of $B$. Clearly, $P_B$ belongs to $\mathcal{C}$ if $P_{B'}$ belongs to $\mathcal{C}$. □
**Recursively Enumerable**

**Definition.** The set \( B \subseteq \mathbb{N} \) is called **recursively enumerable** if there is a partially computable function \( g(x) \) such that

\[
B = \{ x \in \mathbb{N} \mid g(x) \downarrow \}.
\]

- A set is recursively enumerable just when it is the domain of a partially computable function.
- If \( P \) is a program that computes function \( g \) above, then \( B \) is the set of all input to \( P \) for which \( P \) eventually halts.
- \( B \) can be thought of intuitively as a set for which there exists a semi-decision procedure to solve the membership problem of \( B \). This algorithm answers “yes” for number \( n \in B \), but never terminates for \( n \notin B \).
- The term *recursively enumerable* is usually abbreviated *r.e.*

**Recursive Sets**

**Theorem 4.3.** If \( B \) is a recursive set, then \( B \) is r.e.  

**Proof.** Consider the following program \( P \)

\[
[A] \text{ IF } \neg (X \in B) \text{ GOTO } A
\]

Since \( B \) is recursive, the predicate \( x \in B \) is computable and \( P \) can be expanded to a program of \( T \). Let \( P \) computes the function \( h(x) \). Then, clearly,

\[
B = \{ x \in \mathbb{N} \mid h(x) \downarrow \}.
\]

**What If Both \( B \) and \( \overline{B} \) Are r.e.?**

**Theorem 4.4.** The set \( B \) is recursive if and only if \( B \) and \( \overline{B} \) are both r.e.  

**Proof.** (\( \Rightarrow \)) If \( B \) is recursive, then by Theorem 4.1 so is \( B \). By Theorem 4.3, they are both r.e.  

(\( \Leftarrow \)) If both \( B \) and \( \overline{B} \) are r.e., then there are programs \( P \) and \( Q \) such that

\[
B = \{ x \in \mathbb{N} \mid \psi_{1}^{p}(x) \downarrow \}
\]

\[
\overline{B} = \{ x \in \mathbb{N} \mid \psi_{1}^{q}(x) \downarrow \}
\]

Then \( B \) is recursive as it is computed by the following program:

\[
[A] \text{ IF STP}^{(1)}(X,\#(P),T) \text{ GOTO } C
\]

IF STP\(^{(1)}\)(X,\(\#(P),T\)) GOTO E

\[
T \leftarrow T + 1
\]

GOTO A

\[
[C] Y \leftarrow 1
\]

\[
\]
The Union of Two r.e. Sets

Theorem 4.5. If \( B \) and \( C \) are r.e. sets so are \( B \cup C \) and \( B \cap C \). Proof. Let
\[
B = \{ x \in N \mid g(x) \downarrow \}
\]
\[
C = \{ x \in N \mid h(x) \downarrow \}
\]
where \( g \) and \( h \) are partially computable. Let \( f(x) \) be the function computed by the program
\[
Y \leftarrow g(X)
\]
\[
Y \leftarrow h(X)
\]
Hence
\[
B \cap C = \{ x \in N \mid f(x) \downarrow \}
\]
hence \( B \cap C \) is r.e.

The Intersection of Two r.e. Sets

Proof. (Continued) Let \( g \) and \( h \) be computed by programs \( P \) and \( Q \), respectively. Let \( k(x) \) be the function computed by the program:

\[
[A] \quad \text{IF STP}^{(1)}(X, #(P), T) \quad \text{GOTO E}
\]
\[
\text{IF STP}^{(1)}(X, #(Q), T) \quad \text{GOTO E}
\]
\[
T \leftarrow T + 1
\]
\[
\text{GOTO A}
\]
Then \( k(x) \) is defined just in case \( \text{either } g(x) \text{ or } h(x) \) is defined. That is,
\[
B \cup C = \{ x \in N \mid k(x) \downarrow \}
\]
so that \( B \cup C \) is also r.e. \( \square \)

Enumeration Theorem

Definition. We write
\[
W_n = \{ x \in N \mid \Phi(x, n) \downarrow \}.
\]
Then we have Theorem 4.6. A set \( B \) is r.e. if and only if there is an \( n \) for which \( B = W_n \). Proof. This is simply by the definition of \( \Phi(x, n) \). \( \square \) Note that
\[
W_0, W_1, W_2, \ldots
\]
is an enumeration of all r.e. sets.
The Set $K$

Let

$$K = \{ n \in N \mid n \in W_n \}.$$  

Now

$$n \in K \iff \Phi(n, n) \downarrow \iff \text{HALT}(n, n)$$  

This, $K$ is the set of all numbers $n$ such that program number $n$ eventually halts on input $n$.

$K$ Is r.e. but Not Recursive

**Theorem 4.7.** $K$ is r.e. but not recursive.  _Proof._ By the universality theorem, $\Phi(n, n)$ is partially computable, hence $K$ is r.e. If $K$ were recursive, then by Theorem 4.4, $\overline{K}$ must be r.e. Therefore, by the enumeration theorem,

$$\overline{K} = W_i$$

for some $i$. We then arrive at

$$i \in K \iff i \in W_i \iff i \in \overline{K}$$

which is a contradiction. We conclude that $K$ is not recursive.  \(\square\)

r.e. Sets and Primitive Recursive Predicates

**Theorem 4.8.** Let $B$ be an r.e. set. Then there is a primitive recursive predicate $R(x, t)$ such that

$$B = \{ x \in N \mid (\exists t) R(x, t) \}.$$  

_Proof._ Let $B = W_n$. Then

$$B = \{ x \in N \mid (\exists t) \text{STP}^{(1)}(x, n, t) \}.$$  

By Theorem 3.2, $\text{STP}^{(1)}$ is primitive recursive.  \(\square\)

A r.e. Set Is the Range of A Primitive Recursive Function

**Theorem 4.9.** Let $S$ be a nonempty r.e. set. Then there is a primitive recursive function $f(u)$ such that

$$S = \{ f(n) \mid x \in N \} = \{ f(0), f(1), f(2), \ldots \}$$  

That is, $S$ is the range of $f$.  _Proof._ By Theorem 4.8

$$S = \{ x \mid (\exists t) R(x, t) \}$$

where $R$ is primitive recursive. Let $x_0$ be some fixed member of $S$ (say, the smallest), and let

$$f(u) = \begin{cases} l(u) & \text{if } R(l(u), r(u)) \\ x_0 & \text{otherwise.} \end{cases}$$

Clearly $f$ is primitive recursive. It follows that the range of $f$ is a subset of $S$. Conversely, if $x \in S$, then $R(x, t_0)$ is true for some $t_0$. Then $f(\langle x, t_0 \rangle) = l(\langle x, t_0 \rangle) = x$. That is, $S$ is a subset of the range of $f$. We conclude $S = \{ f(n) \mid x \in N \}$.  \(\square\)
The Range of A Partially Computable Function Is r.e.

**Theorem 4.10.** Let $f(x)$ be a partially computable function and let $S = \{ f(x) \mid f(x) \downarrow \}$. Then $S$ is r.e.  

**Proof.** Let 

$$g(x) = \begin{cases} 0 & \text{if } x \in S \\ \uparrow & \text{otherwise.} \end{cases}$$ 

Clearly $S = \{ x \mid g(x) \downarrow \}$. It suffices to show that $g$ is partially computable. Let $\mathcal{P}$ be a program that computes $f$ and let $\#(\mathcal{P}) = p$. Then the following program computes $g(x)$:

\[
\begin{align*}
[A] & \text{IF } \sim \text{STP}(1)(Z, p, T) \text{ GOTO } B \\
& V \leftarrow f(Z) \\
& \text{IF } V = X \text{ GOTO } E \\
[B] & Z \leftarrow Z + 1 \\
& \text{IF } Z \leq T \text{ GOTO } A \\
& T \leftarrow T + 1 \\
& Z \leftarrow 0 \\
& \text{GOTO } A
\end{align*}
\]

Recursively Enumerable Sets, Revisited

**Theorem 4.11.** Suppose that $S \neq \emptyset$. Then the following statements are all equivalent:

1. $S$ is r.e.
2. $S$ is the range of a primitive recursive function;
3. $S$ is the range of a recursive function;
4. $S$ is the range of a partially recursive function.

**Proof.** By Theorem 4.9, 1. implies 2. Obviously, 2. implies 3., and 3. implies 4. By Theorem 4.10, 4. implies 1. Hence all four statements are equivalent. □

1.2 The Parameter Theorem (4.5)

**The Parameter Theorem**

The Parameter theorem (which has also been called the $s - m - n$ theorem) relates the various functions $\Phi^{(n)}(x_1, x_2, \ldots, x_n, y)$ for different values of $n$.  

**Theorem 5.1.** For each $n, m > 0$, there is a primitive recursive function $S^n_m(u_1, u_2, \ldots, u_n, y)$ such that 

$$\Phi^{(m+n)}(x_1, \ldots, x_m, u_1, \ldots, u_n, y) = \Phi^{(m)}(x_1, \ldots, x_m, S^n_m(u_1, \ldots, u_n, y))$$
The Parameter Theorem, Continued

\[ \Phi^{(m+n)}(x_1, \ldots, x_m, u_1, \ldots, u_n, y) = \Phi^{(m)}(x_1, \ldots, x_m, S^m_m(u_1, \ldots, u_n, y)) \]

Suppose the values for variables \( u_1, \ldots, u_n \) are fixed and we have in mind some particular value of \( y \). Then left hand side of the above equation is a partially computable function \( f \) of \( m \) arguments \( x_1, \ldots, x_m \).

Let \( q \) be the number of a program that computes this function of \( m \) variables, we have

\[ \Phi^{(m+n)}(x_1, \ldots, x_m, u_1, \ldots, u_n, y) = \Phi^{(m)}(x_1, \ldots, x_m, q) \]

The parameter theorem tells us that not only does there exist such a number \( q \), but it can be obtained from \( u_1, \ldots, u_n, y \) by using a primitive recursive function \( S^m_m \).

The Parameter Theorem, Continued

\[ \Phi^{(m+n)}(x_1, \ldots, x_m, u_i, \ldots, u_n, y) = \Phi^{(m)}(x_1, \ldots, x_m, S^m_m(u_i, \ldots, u_n, y)) \]

Suppose the values for variables \( u_1, \ldots, u_n \) are fixed and we have in mind some particular value of \( y \). Then left hand side of the above equation is a partially computable function \( f \) of \( m \) arguments \( x_1, \ldots, x_m \).

Let \( q \) be the number of a program that computes this function of \( m \) variables, we have

\[ \Phi^{(m+n)}(x_1, \ldots, x_m, u_i, \ldots, u_n, y) = \Phi^{(m)}(x_1, \ldots, x_m, q) \]

The parameter theorem tells us that not only does there exist such a number \( q \), but it can be obtained from \( u_1, \ldots, u_n, y \) by using a primitive recursive function \( S^m_m \).

The Parameter Theorem, Proof

The proof is by a mathematical induction on \( n \). For \( n = 1 \), we need to show that there is a primitive recursive function \( S^1_m(u, y) \) such that

\[ \Phi^{(m+1)}(x_1, \ldots, x_m, u, y) = \Phi^{(m)}(x_1, \ldots, x_m, S^1_m(u, y)) \]

Let \( \mathcal{P} \) be the program such that \( \#(\mathcal{P}) = y \). Then \( S^1_m(u, y) \) can be taken to the number of the program which first gives variable \( X_{m+1} \) the value \( u \) and then proceeds to carry out \( \mathcal{P} \).

The Parameter Theorem, Proof

\( X_{m+1} \) will be given the value \( u \) by the program:

\[
\begin{align*}
X_{m+1} &\leftarrow X_{m+1} + 1 \\
&\vdots \\
X_{m+1} &\leftarrow X_{m+1} + 1
\end{align*}
\]

\[ u \]
The number of the instruction \( X_{m+1} \leftarrow X_{m+1} + 1 \) is \( \langle 0, (1, 2m + 1) \rangle = 16m + 10 \). So we may take
\[
S^1_m(u, y) = [(\prod_{i=1}^u p_i)^{16m+10} \cdot (\prod_{j=1}^{Lt(y+1)} p_{u+j}^{(y+1)_j})^1 - 1
\]
as the primitive recursive function.

**The Parameter Theorem, Proof**

To complete the proof, suppose the result is known for \( n = k \). Then we have
\[
\Phi^{(m+k+1)}(x_1, \ldots, x_m, u_1, \ldots, u_k, u_{k+1}, y) \\
= \Phi^{(m+k)}(x_1, \ldots, x_m, u_1, \ldots, u_k, S^1_{m+k}(u_{k+1}, y)) \\
= \Phi^{(m)}(x_1, \ldots, x_m, S^k_m(u_1, \ldots, u_k, S^1_{m+k}(u_{k+1}, y)))
\]
using first the result for \( n = 1 \) and then the induction hypothesis.

By now, if we define
\[
S^{k+1}_m(u_1, \ldots, u_k, u_{k+1}, y) = S^k_m(u_1, \ldots, u_k, S^1_{m+k}(u_{k+1}, y))
\]
we have the desired result.

**The Parameter Theorem, Examples**

Is there a computable function \( g(u, v) \) such that
\[
\Phi_g(\Phi_v(x)) = \Phi_{g(u,v)}(x)
\]
for all \( u, v, x \)? Yes! Note that
\[
\Phi_g(\Phi_v(x)) = \Phi(\Phi(x, v), u)
\]
is a partially computable function of \( x, u, v \). Hence, we have
\[
\Phi(\Phi(x, v), u) = \Phi^{(3)}(x, u, v, z_0)
\]
for some number \( z_0 \). By the parameter theorem,
\[
\Phi^{(3)}(x, u, v, z_0) = \Phi(x, S^2_1(u, v, z_0)) = \Phi_{S^2_1(u,v,z_0)}(x).
\]