1. Recursive and Recursively Enumerable Sets

2. The Parameter Theorem
Recall that by the Enumeration Theorem, the collection of r.e. sets can be written as

\[ W_1, W_2, \ldots, W_n, \ldots \]

where \( W_n = \{ x \in \mathbb{N} \mid \Phi(x, n) \downarrow \} \).
Let $K$ be the set of all numbers $n$ such that the program number $n$ eventually halts on number $n$.

**Definition**

The set $K$ is defined as:

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

Note that $n \in W_n$ if and only if $\Phi(n, n) \downarrow$ if and only if $\text{HALT}(n, n) = \text{TRUE}$.
Theorem

The set $K$ is r.e. but not recursive.
Proof.

Note that $K = \{ n \in \mathbb{N} \mid \Phi(n, n) \downarrow \}$, $\Phi$ is partially computable, hence $K$ is r.e.

If $\overline{K}$ were r.e, by the Enumeration Theorem we would have $\overline{K} = W_i$ for some $i$. Then,

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

which is a contradiction. Therefore, $\overline{K}$ is not r.e., which implies that $K$ is not recursive.
An alternative characterization of r.e. sets is provided next.

**Theorem**

**Projection Theorem** Let $B$ be an r.e. set. There exists a *primitive recursive* predicate $R(x, t)$ such that:

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

Note that this is *unbounded* existential quantification!
Proof.

Let $B = W_n$. Then,

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

and STP is primitive recursive. The role of $R(x, t)$ is played by $\text{STP}^{(1)}(x, n, t)$. □
**Theorem**

Let $S$ be a non-empty r.e. set. Then, there is a primitive recursive function $f(u)$ such that

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

That is, $S$ is the range of $f$. 

---

**THEORY OF COMPUTATION** Recursively Enumerable Sets - 11 part 2

Recursive and Recursively Enumerable Sets
Proof.

By a previous theorem, \( S = \{ x \in \mathbb{N} \mid (\exists t)R(x, t) \} \), where \( R \) is a primitive recursive predicate. Let \( x_0 \) be some fixed member of \( S \) (for example the smallest). Let \( f \) the primitive recursive function:

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

Each value of \( f(u) \) is in \( S \), since \( x_0 \in S \), while if \( R(\ell(u), r(u)) \) is TRUE, then \((\exists t)R(\ell(u), t)\) is TRUE, which implies that \( f(u) = \ell(u) \in S \).

Conversely, if \( x \in S \), then \( R(x, t_0) \) is TRUE for some \( t_0 \). Then, \( f(\langle x, t_0 \rangle) = \ell(\langle x, t_0 \rangle) = x \), so that \( x = f(u) \) for \( u = \langle x, t_0 \rangle \). \(\square\)
Theorem

Let $f(x)$ be a partially computable function and let
$S = \{ f(x) \mid f(x) \downarrow \}$. In other words, $S$ is the range of $f$. Then, $S$ is r.e.
Proof.

Let

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

Since \( S = \{x \mid g(x) \downarrow\} \) it suffices to show that the fact that partially computability of \( f \) implies that \( g \) is partially computable.
Proof cont’d

Proof.

Let $\mathcal{P}$ be a program that computes $f$ and $\#(\mathcal{P}) = p$. Then, $g$ is computed by

<table>
<thead>
<tr>
<th>A</th>
<th>IF $STP^{(1)}(Z, p, T)$ GOTO B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$V \leftarrow f(Z)$</td>
</tr>
<tr>
<td></td>
<td>IF $V = X$ GOTO E</td>
</tr>
<tr>
<td>B</td>
<td>$Z \leftarrow Z + 1$</td>
</tr>
<tr>
<td></td>
<td>IF $Z \leq T$ GOTO A</td>
</tr>
<tr>
<td></td>
<td>$T \leftarrow T + 1$</td>
</tr>
<tr>
<td></td>
<td>$Z \leftarrow 0$</td>
</tr>
<tr>
<td></td>
<td>GOTO A</td>
</tr>
</tbody>
</table>

Note that $V \leftarrow f(Z)$ is entered only when the step-counter test has already determined that $f$ is defined.
Theorem

Suppose that $S \neq \emptyset$. The following statements are 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 partial recursive function;

This follows from previous theorems: the implication (1) $\Rightarrow$ (2) follows from the theorem on Slide 10. The implications (2) $\Rightarrow$ (3) and (3) $\Rightarrow$ (4) are obvious. The implication (4) $\Rightarrow$ (1) follows from the theorem on Slide 12.
The parameter theorem is also known as the **smn-theorem**.

**Theorem**

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

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

The proof is by induction on $n$, the number of arguments $u_1, \ldots, u_n$ packed into $S$.

For $n = 1$, the base case, we need to show that there is a function $S^1_m$ such that

$$
\Phi^{(m+n)}(x_1, \ldots, x_m, u_1, y) = \Phi^{(m)}(x_1, \ldots, x_m, S^1_m(u_1, y)).
$$

Here $S^1_m(u_1, y)$ must be the number of a program which, given $m$ inputs $x_1, \ldots, x_m$ computes the same value as the program $y$ does when given the $m + 1$ inputs $x_1, \ldots, x_m, u_1$.

Let $\mathcal{P}$ be the program with $\#(\mathcal{P}) = y$. Then, $S^1_m(u_1, y)$ can be taken as the number of a program which first gives $X_{m+1}$ the value $u_1$ and then proceeds to execute $\mathcal{P}$.
Proof.

So the new program begins with

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

\[u_1\]

Note that the code of \(X_{m+1} \leftarrow X_{m+1} + 1\) is

\[\langle 0, \langle 1, 2m + 1 \rangle \rangle = 16m + 10.\]
Proof cont’d

Proof.

So, we may take

\[ S_m^1(u_1, y) = \left( \prod_{i=1}^{u_1} p_i \right)^{16m+10 \operatorname{Lt}(y+1)} \prod_{j=1}^{\operatorname{Lt}(y+1)} p_{u_1+j}^{(y+1)_j} = 1, \]

which is a primitive recursive function.

Note that the numbers of the instructions of \( \mathcal{P} \) which appear as exponents in the prime power factorization of \( y + 1 \) have been shifted to the primes \( p_{u_1+1}, p_{u_1+2}, \ldots, p_{u_1+\operatorname{Lt}(y+1)}. \)
Proof cont’d

Proof.

Suppose now that the result holds for \( n = k \). Then, we have

\[
\begin{aligned}
\Phi^{(m+k+1)}(x_1, \ldots, x_m, u_1, \ldots, u_k, u_{k+1}, y) \\
= \Phi^{(m+k)}(x_1, \ldots, x_m, u_1, \ldots, u_k, S_{m+k}^{1}(u_{k+1}, y)) \\
= \Phi^m(x_1, \ldots, x_m, S_k^m(u_1, \ldots, u_k, S_{m+k}^{1}(u_{k+1}, y))),
\end{aligned}
\]

using the result for \( n = 1 \) and the induction hypothesis. Now, we can define

\[
S_{m+1}^{k+1}(u_1, \ldots, u_k, u_{k+1}) = S_m^k(u_1, \ldots, u_k, S_{m+k}^{1}(u_{k+1}, y)).
\]
Example

Using the smn theorem we can show the existence of a primitive recursive function \( g \) such that \( \Phi_u(\Phi_v(x)) = \Phi_{g(u,v)}(x) \). This means that

\[
\Phi_u(\Phi_v(x)) = \Phi(\Phi(x, v), u),
\]

so \( \Phi_u(\Phi_v(x)) = \Phi(x, u, v, z_0) \) is a partially computable function of \( x, u, v \). Hence

\[
\Phi_u(\Phi_v(x)) = \Phi(x, S_1^3(u, v, z_0)),
\]

and \( g(u, v) = S_1^3(u, v, z_0) \).