Recursive Function Theory

Abhishek Kr Singh

TIFR Mumbai.

21 August 2014
Primitive Recursive Functions

- Initial Functions.
  - \( s(x) = x + 1 \)
  - \( n(x) = 0 \)
  - \( u^n_i(x_1, \ldots, x_n) = x_i \), where \( 1 \leq i \leq n \).

- Composition:
  - Let \( h(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)) \).
  - Then \( h \) is said to be obtained from \( f \) and \( g_1, \ldots, g_k \) by composition.

- Primitive Recursion:
  - Let \( h(x_1, \ldots, x_n, 0) = f(x_1, \ldots, x_n) \), and
    \[ h(x_1, \ldots, x_n, t + 1) = g(t, h(x_1, \ldots, x_n, t), x_1, \ldots, x_n) \.
  - Then \( h \) is said to be obtained from \( f \) and \( g \) by primitive recursion, or simply recursion.

**Definition:** A function is called **primitive recursive** if it can be obtained from the initial functions by a finite number of applications of composition and recursion.
Bounded Quantifiers:

- If the predicate $P(t, x_1, \ldots, x_n)$ is primitive recursive then so are the predicates $(\forall t)_{\leq y} P(t, x_1, \ldots, x_n)$ and $(\exists t)_{\leq y} P(t, x_1, \ldots, x_n)$.

Bounded Minimalization:

- $\min_{t \leq y} P(t, x_1, \ldots, x_n)$ is the least value of $t \leq y$ for which $P(t, x_1, \ldots, x_n)$ is true, if such exists; otherwise it assumes the default value 0.
- $\min_t P(t, x_1, \ldots, x_n)$ is the unbounded version. However in this case if there is no value of $t$ for which $P$ is true, then $\min_t P(t, x_1, \ldots, x_n)$ is undefined.
- If the predicate $P(t, x_1, \ldots, x_n)$ is primitive recursive then so is the predicate $\min_{t \leq y} P(t, x_1, \ldots, x_n)$.
Some Primitive Recursive Functions

1. $x + y$
2. $x \cdot y$
3. $x!$
4. $x^y$
5. $p(x)$ the predecessor function
6. $x ÷ y$
7. $|x - y|$
8. $\alpha(x)$ the IsZero predicate
9. $x = y$
10. $x \leq y$
11. $x < y$
12. $y \mid x$ y divides x
13. $\text{Prime}(x)$
14. $\lfloor x / y \rfloor$
15. $R(x, y)$
16. $p_n$ the nth prime number
17. $< x, y >$ the pairing function
18. $[a_1, ..., a_n]$ the Godel number
19. $Lt(x)$ where $x = [a_1, ..., a_n]$
20. $([a_1, ..., a_n])_i$
Programs and Computable Functions

- Programming language \(S\).
  - Our concept of computable function will be based on programming language \(S\) which has following instruction types.
    1. \(V \leftarrow V\)
    2. \(V \leftarrow V + 1\)
    3. \(V \leftarrow V - 1\)
    4. \(IF \ V \neq 0 \ GOTO \ L\)

- A program in \(S\) is a sequence of labeled or unlabeled instructions of above type.

- Syntax of the language \(S\)
  - Conventions:
    - Input variables \(X_1, X_2, X_3, \ldots\)
    - Output variable \(Y\) and
    - Local Variables \(Z_1, Z_2, Z_3, \ldots\)
  - State \(\sigma\) and snapshot \(s = (i, \sigma)\) of program \(P\).
  - A Computation of a program \(P\) is defined to be a sequence \(s_1, s_2, s_3, \ldots, s_k\) of snapshots of \(P\) such that \(s_{i+1}\) is the successor of \(s_i\) for each \(i\) and \(s_k\) is the terminal snapshot.
Computable Functions

- For any program $P$ and any positive integer $m$, $\psi^m_P(x_1, \ldots, x_m)$ represents the value of function computed by program $P$ on input $x_1, \ldots, x_m$.
- A given partial function $g$ is said to be *partially computable* if it is computed by some program.
- A function $g$ is called *computable* if it is both total and partially computable.

Primitive recursive Vs computable functions.

- Every primitive recursive function is computable.
- Coding program by numbers
  - $\#(l) = <a, <b, c>>$
  - $\#(P) = [\#(l_1), \#(l_2), \ldots, \#(l_k)] - 1.$
Theorem-1 Universality Theorem:

Let \( \phi^n(x_1, \ldots, x_n, y) = \psi^n_P(x_1, \ldots, x_n) \), where \( \#(P) = y \).

Then for each \( n > 0 \), the function \( \phi^n(x_1, \ldots, x_n, y) \) is partially computable.

Proof. Existence of a program \( U_n \), called the universal program, such that

\[ \psi_{U_n}^{n+1}(x_1, \ldots, x_n, x_{n+1}) = \phi^n(x_1, \ldots, x_n, x_{n+1}) = \psi^n_P(x_1, \ldots, x_n) \), where \( x_{n+1} = \#(P) \). \]
**Theorem-2 Step-Counter Theorem:**

- Let $STP^n(x_1, \ldots, x_n, y, t) \iff$ Program number $y$ halts after $t$ or fewer steps on inputs $x_1, \ldots, x_n$.

Then for each $n > 0$, the predicate $STP^n(x_1, \ldots, x_n, y, t)$ is primitive recursive.

**Proof** $STP^n(x_1, \ldots, x_n, y, t) \iff \text{Term}(\text{Snap}^n(x_1, \ldots, x_n, y, t), y)$

where $\text{Snap}^n(x_1, \ldots, x_n, y, 0) = \text{Init}^n(x_1, \ldots, x_n)$ and

$\text{Snap}^n(x_1, \ldots, x_n, y, i + 1) = \text{Succ}(\text{Snap}^n(x_1, \ldots, x_n, y, i), y)$. 
**Theorem-3** Normal Form Theorem: Let \( f(x_1, \ldots, x_n) \) be a partially computable function. Then there is a primitive recursive predicate \( R(x_1, \ldots, x_n, y) \) such that \( f(x_1, \ldots, x_n) = l(\min_z R(x_1, \ldots, x_n, z)) \).

**Proof.** Let \( y_0 = \#(P) \) where \( P \) is the program that computes \( f \).
Consider the following predicate, call it \( R(x_1, \ldots, x_n, z) \),

\[
\begin{align*}
STP^n(x_1, \ldots, x_n, y_0, r(z)) \\
\& (r(Snap^n(x_1, \ldots, x_n, y_0, r(z))))_1 = l(z)
\end{align*}
\]

Then we have

\( f(x_1, \ldots, x_n) = l(\min_z R(x_1, \ldots, x_n, z)) \)

*Note that* if there is no value of \( z \) for which \( \min_z R(x_1, \ldots, x_n, z) \) is true, then according to the definition of minimalization \( \min_z R(x_1, \ldots, x_n, z) \) is undefined.
Corollary-3.1 A function is partially computable if and only if it can be obtained from the initial functions by a finite number of applications of composition, recursion, and minimalization.

Corollary-3.2 A function is computable if and only if it can be obtained from the initial functions by a finite number of applications of composition, recursion, and proper minimalization.

- When $\text{min}_z R(x_1, \ldots, x_n, z)$ is a total function, we say that we are applying proper minimalization to $R$. 
Recursively Enumerable Sets

Definition The set $B \subset N$ is called recursively enumerable if there is a partially computable function $g(x)$ such that $B = \{ x \in N \mid g(x) \downarrow \}$.

Definition We write $W_n = \{ x \in N \mid \phi(x, n) \downarrow \}$. We define $K = \{ n \in N \mid n \in W_n \}$. Therefore, $n \in W_n \iff \phi(n, n) \downarrow \iff \text{HALT}(n, n)$.

Since we have Gödel numbering functions $[x_1, \ldots, x_m]$ and $(x)_i$, we can restrict to the subsets of $N$ in our discussion of Computability theory. Therefore, we have,

Theorem-4 Let $C$ be a PRC class, and let $B$ be a subset of $N^m$, $m \geq 1$. Then $B$ belongs to $C$ if and only if $B' = \{ [x_1, \ldots, x_m] \in N \mid (x_1, \ldots, x_m) \in B \}$ belongs to $C$.

Proof. $P_{B'}(x) \iff (Lt(x) = m) \& P_B((x)_1, \ldots, (x)_m)$

$P_B(x_1, \ldots, x_m) \iff P_{B'}([x_1, \ldots, x_m])$
Theorem-5  **Enumeration Theorem:** A set \( B \) is r.e if and only if there is an \( n \) for which \( B = W_n \).

Theorem-6  The set \( B \) is recursive if and only if \( B \) and \( \bar{B} \) are both r.e.

**Proof.** Let \( P \) and \( Q \) be the programs corresponding to \( B \) and \( \bar{B} \). Consider the following program where \( p = \#(P) \) and \( q = \#(Q) \)

\[
\begin{align*}
[A] & \quad \text{If } STP(X, p, T) \text{ Goto } C \\
& \quad \text{If } STP(X, q, T) \text{ Goto } E \\
& \quad T \leftarrow T + 1 \\
& \quad \text{Goto } A \\
[C] & \quad Y \leftarrow 1
\end{align*}
\]

Theorem-7  Let \( B \) be an r.e. set. Then there is a primitive recursive predicate \( R(x, t) \) such that 
\[ B = \{ x \in N | (\exists t) R(x, t) \} . \]

**Proof.** Let \( B = W_n \). Then 
\[ B = \{ x \in N | (\exists t) STP(x, n, t) \} . \]

Theorem-8  If \( B \) and \( C \) are r.e sets so are \( B \cup C \) and \( B \cap C \).
Theorem-9  Let $S$ be a nonempty r.e. set. Then there is a primitive recursive function $f(u)$ such that $S = \{f(n) | n \in \mathbb{N}\} = \{f(0), f(1), \ldots\}$. That is, $S$ is the range of $f$.

Proof.  By Theorem-7 we can write, $S = \{x | (\exists t) R(x, t)\}$.

Let $x_0 \in S$ and $u = <x, t>$. Consider the following function

$$f(u) = \begin{cases} x & \text{if } R(x, t) \\ x_0 & \text{otherwise} \end{cases}$$

That is,

$$f(<x, t>) = x.R(x, t) + x_0 \sim R(x, t)$$

Which is same as

$$f(u) = l(u).R(l(u), r(u)) + x_0.\alpha(R(l(u), r(u))).$$
Theorem-10  Let \( f(x) \) be a partially computable function and let \( S = \{ f(x) | f(x) \downarrow \} \). (That is, \( S \) is the range of \( f \).) Then \( S \) is r.e.

Proof  Let program \( P \) computes \( f \) and \( p = \#(P) \). Then we need to demonstrate a program, say \( Q \), which behaves as follow,

\[ Q \text{ stops at } x \iff \exists u \exists t, (STP(u, p, t) \& f(u) = x) \]

\[ \iff \exists < u, t >, (STP(u, p, t) \& f(u) = x) \]

\[ \iff \exists n, (STP(l(n), p, r(n)) \& f(l(n)) = x) \]

Thus \( Q \) can be the following program,

\([A] \quad IF \sim STP(l(Z), p, r(Z)) \quad Goto B\]

\[ IF \ f(l(Z)) = X \quad Goto E\]

\([B] \quad Z \leftarrow Z + 1 \quad Goto A\]
Theorem-11  Suppose that $S \neq \emptyset$. Then the following statements are all equivalent:

1. $S$ is r.e.