Some Primitive Recursive Functions

So we have learned that every primitive recursive function is computable.
We will now look at some examples of primitive recursive functions.
In order to prove that a function is primitive recursive, we have to show how that function can be derived from the initial functions by using composition and recursion.

Some Primitive Recursive Functions

Remember? The recursive definition of a function looks like this:

\[ h(x_1, \ldots, x_n, 0) = f(x_1, \ldots, x_n) \]
\[ h(x_1, \ldots, x_n, t + 1) = g(t, h(x_1, \ldots, x_n, t), x_1, \ldots, x_n) \]

If \( f \) is a function of \( k \) variables, \( g_1, \ldots, g_k \) are functions of \( n \) variables, and

\[ h(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)). \]

Then \( h \) is said to be obtained from \( f \) and \( g_1, \ldots, g_k \) by composition.

Some Primitive Recursive Functions

We defined the following initial functions:

\[ s(x) = x + 1 \]
\[ n(x) = 0 \]
\[ u_i^n(x_1, \ldots, x_n) = x_i, \ 1 \leq i \leq n. \]

Example 1: \( f(x, y) = x + y \)

We can transform this into a recursive definition:

\[ f(x, 0) = x \]
\[ f(x, y + 1) = f(x, y) + 1 \]

This can be rewritten as:

\[ f(x, 0) = u_1^n(x) \]
\[ f(x, y + 1) = g(y, f(x, y), x) \]

where \( g(x_1, x_2, x_3) = s(u_2^n(x_1, x_2, x_3)) \).

Obviously, \( u_1^n(x) \) and \( u_2^n(x_1, x_2, x_3) \) are primitive recursive functions – they are initial functions.

\( g(x_1, x_2, x_3) \) is obtained by composition of primitive recursive functions, so it is primitive recursive itself.

Therefore, \( f(x, y) = x + y \) is primitive recursive.

Some Primitive Recursive Functions

Example 2: \( h(x) = x! \)

Here are the recursion equations:

\[ h(0) = 1 \]
\[ h(t + 1) = h(t) \cdot s(t) \]

This can be rewritten as:

\[ h(0) = u_1^n(x) \]
\[ h(t + 1) = g(t, h(t)) \]

where \( g(x_1, x_2) = s(u_2^n(x_1, x_2)) u_2^n(x_1, x_2) \).

Multiplication is already known to be primitive recursive. Therefore, \( h(x) = x! \) is primitive recursive.

Some Primitive Recursive Functions

Example 3: \( h(x) = x! \)

We can obtain the following recursive definition:

\[ h(x, 0) = 0 \]
\[ h(x, y + 1) = h(x, y) + s(h(x, y)) \]

This can be rewritten as:

\[ h(x, 0) = u_1^n(x) \]
\[ h(x, y + 1) = g(y, h(x, y), x) \]

where \( f(x_1, x_2, x_3) = x_1 + x_2 + x_3 \) and \( g(x_1, x_2, x_3) = f(u_2^n(x_1, x_2, x_3), u_2^n(x_1, x_2, x_3)) \).

Obviously, these are all primitive recursive functions. Therefore, \( h(x, y) = x \cdot y \) is primitive recursive.
Some Primitive Recursive Functions

In the following examples, we just show the recursive mechanism without developing the precise form of the recursion equations.

Example 4: $x^y$

The recursion equations are:

\[
x^0 = 1 \\
x^{y+1} = x^y \cdot x
\]

Some Primitive Recursive Functions

Example 5: The predecessor function $p(x)$

It is defined as follows:

\[
p(x) = \begin{cases}  x - 1 & \text{if } x \neq 0 \\ 0 & \text{if } x = 0 \end{cases}
\]

The recursion equations are:

\[
p(0) = 0 \\
p(t + 1) = t
\]

Some Primitive Recursive Functions

Example 6: $x - y$ (monus)

It is defined as follows:

\[
x - y = \begin{cases}  x - y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}
\]

The recursion equations are:

\[
x - 0 = x \\
x - (t + 1) = p(x - t)
\]

Some Primitive Recursive Functions

Example 7: $|x - y|$

The function $|x - y|$ is defined as the absolute value of the difference between $x$ and $y$.

It can be written as follows:

\[
|x - y| = (x - y) + (y - x)
\]

Therefore, $|x - y|$ is primitive recursive.

Some Primitive Recursive Functions

Example 8: $\alpha(x)$ (“negation”)

The function $\alpha(x)$ is defined as follows:

\[
\alpha(x) = \begin{cases}  1 & \text{if } x = 0 \\ 0 & \text{if } x \neq 0 \end{cases}
\]

$\alpha(x)$ is primitive recursive, because:

\[
\alpha(x) = 1 - x
\]

If we prefer, we can just write down the recursion equations:

\[
\alpha(0) = 1 \\
\alpha(t + 1) = 0
\]

Some Primitive Recursive Functions

Example 9: $x = y$

We can describe this predicate by the function $d(x, y)$:

\[
d(x, y) = \begin{cases}  1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}
\]

d(x, y) is primitive recursive because of the following equation:

\[
d(x, y) = \alpha((x - y))
\]
Primitive Recursive Predicates

Example 10: $x \leq y$
Again, we can describe this predicate by the function $d(x, y)$:

$$d(x, y) = \begin{cases} 1 & \text{if } x \leq y \\ 0 & \text{if } x > y \end{cases}$$

$d(x, y)$ is primitive recursive because of the following equation:

$$d(x, y) = \alpha(x - y)$$

Theorem 5.1: Let $C$ be a PRC class.
If $P$, $Q$ are predicates that belong to $C$, then so are $\neg P$, $P \lor Q$, and $P \land Q$.

Proof:
Since $\neg P = \alpha(P)$, it follows that $\neg P$ belongs to $C$.
Since $P \land Q = P \cdot Q$, it follows that $P \land Q$ belongs to $C$.
Since $P \lor Q = \neg(\neg P \land \neg Q)$, it follows that $P \lor Q$ belongs to $C$.

Corollary 5.2: If $P$, $Q$ are primitive recursive predicates, then so are $\neg P$, $P \lor Q$, and $P \land Q$.

Corollary 5.3: If $P$, $Q$ are computable predicates, then so are $\neg P$, $P \lor Q$, and $P \land Q$.

Example 11: $x < y$
So far, we only know that $x \leq y$ and $x = y$ are primitive recursive predicates (Examples 9 and 10).
Since we have the tautology $x < y \iff x \leq y \land \neg(x = y)$
or even simpler:
$x < y \iff \neg(y \leq x)$,
according to Corollary 5.2, $x < y$ is also a primitive recursive predicate.

Theorem 5.4: Let $C$ be a PRC class.
Let the functions $g$, $h$ and the predicate $P$ belong to $C$.
Let

$$f(x_1, \ldots, x_n) = \begin{cases} g(x_1, \ldots, x_n) & \text{if } P(x_1, \ldots, x_n) \\ h(x_1, \ldots, x_n) & \text{otherwise} \end{cases}$$

Then $f$ belongs to $C$.

Proof: $f$ obviously belongs to $C$, because it can be written as:

$$f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n) \cdot P(x_1, \ldots, x_n) + h(x_1, \ldots, x_n) \cdot \alpha(P(x_1, \ldots, x_n))$$

Corollary 5.5: Let $C$ be a PRC class.
Let $n$-ary functions $g_1, \ldots, g_m$, $h$ and predicates $P_1, \ldots, P_m$ belong to $C$ and let

$$P_1(x_1, \ldots, x_n) \land P_2(x_1, \ldots, x_n) = 0$$

for all $1 \leq i < j \leq m$ and all $x_1, \ldots, x_n$. If

$$f(x_1, \ldots, x_n) = \begin{cases} g_1(x_1, \ldots, x_n) & \text{if } P_1(x_1, \ldots, x_n) \\ g_2(x_1, \ldots, x_n) & \text{if } P_2(x_1, \ldots, x_n) \\ \vdots & \text{otherwise} \\ g_m(x_1, \ldots, x_n) & \text{if } P_m(x_1, \ldots, x_n) \\ h(x_1, \ldots, x_n) & \end{cases}$$

Then $f$ belongs to $C$.
Primitive Recursive Predicates

**Proof:** The idea is to use induction on the variable $m$. Theorem 5.4 provides the case for $m = 1$, so we just have to do the inductive step. Let

\[
f(x_1, \ldots, x_n) = g_1(x_1, \ldots, x_n) \quad \text{if} \quad P_1(x_1, \ldots, x_n)
\]

\[
\vdots
\]

\[
= g_m(x_1, \ldots, x_n) \quad \text{if} \quad P_m(x_1, \ldots, x_n)
\]

\[
= h(x_1, \ldots, x_n) \quad \text{otherwise, and let}
\]

\[
h'(x_1, \ldots, x_n) = g_{m+1}(x_1, \ldots, x_n) \quad \text{if} \quad P_{m+1}(x_1, \ldots, x_n)
\]

\[
= h'(x_1, \ldots, x_n) \quad \text{otherwise. Then}
\]

\[
f(x_1, \ldots, x_n) = g_1(x_1, \ldots, x_n) \quad \text{if} \quad P_1(x_1, \ldots, x_n)
\]

\[
\vdots
\]

\[
= g_{m+1}(x_1, \ldots, x_n) \quad \text{if} \quad P_{m+1}(x_1, \ldots, x_n)
\]

\[
= h'(x_1, \ldots, x_n) \quad \text{otherwise.}
\]

The previous steps showed that

- $f$ belongs to $C$ for $m = 1$
- Whenever $f$ belongs to $C$ for a particular $m$, then $f$ also belongs to $C$ for $m + 1$.

**Conclusion:** $f$ belongs to $C$ for any natural number $m$. 