planetmath.org

Math for the people, by the people.

primitive recursive function

Canonical name	PrimitiveRecursiveFunction
Date of creation	2013-03-22 12:33:19
Last modified on	2013-03-22 12:33:19
Owner	CWoo (3771)
Last modified by	CWoo (3771)
Numerical id	19
Author	CWoo (3771)
Entry type	Definition
Classification	msc 03D20
Synonym	primitive recursive
Related topic	RecursiveFunction
Defines	primitive recursive set
Defines	primitive recursive predicate
Defines	partial primitive recursive function

To define what a primitive recursive function is, the following notations are used:

> $\mathcal{F} = \bigcup\{F_k \mid k \in \mathbb{N}\}$, where for each $k \in \mathbb{N}$, $F_k = \{f \mid f\colon \mathbb{N}^k \to \mathbb{N}\}$.

Definition. The set of *primitive recursive functions* is the smallest subset $\mathcal{PR}$ of $\mathcal{F}$ where:

1. (zero function) $z \in \mathcal{PR} \cap F_1$, given by $z(n) := 0$;
2. (successor function) $s \in \mathcal{PR} \cap F_1$, given by $s(n) := n + 1$;
3. (projection functions) $p^k_m \in \mathcal{PR} \cap F_k$, where $m \leq k$, given by $p^k_m(n_1, \ldots, n_k) := n_m$;
4. $\mathcal{PR}$ is closed under composition: If $\{g_1, \ldots, g_m\} \subseteq \mathcal{PR} \cap F_k$ and $h \in \mathcal{PR} \cap F_m$, then $f \in \mathcal{PR} \cap F_k$, where
$$f(n_1, \ldots, n_k) = h(g_1(n_1, \ldots, n_k), \ldots, g_m(n_1, \ldots, n_k));$$
5. $\mathcal{PR}$ is closed under primitive recursion: If $g \in \mathcal{PR} \cap F_k$ and $h \in \mathcal{PR} \cap F_{k+2}$, then $f \in \mathcal{PR} \cap F_{k+1}$, where
$$\begin{aligned} f(n_1, \ldots, n_k, 0) &= g(n_1, \ldots, n_k) \\ f(n_1, \ldots, n_k, s(n)) &= h(n_1, \ldots, n_k, n, f(n_1, \ldots, n_k, n)). \end{aligned}$$

Many of the arithmetic functions that we encounter in basic math are primitive recursive, including addition, multiplication, and exponentiation. More examples can be found in `http://planetmath.org/ExamplesOfPrimitiveRecursiveFuncti` entry.

Primitive recursive functions are computable by Turing machines. In fact, it can be shown that $\mathcal{PR}$ is precisely the set of functions computable by programs using FOR NEXT loops. However, not all Turing-computable functions are primitive recursive: the Ackermann function is one such example.

Since $\mathcal{F}$ is countable, so is $\mathcal{PR}$. Moreover, $\mathcal{PR}$ is recursively enumerable (can be listed by a Turing machine).

Remarks.

[1] Every primitive recursive function is total, since it is built from z, s, and p^k_m, each of which is total, and that functional composition, and primitive recursion preserve totalness. By including $\varnothing$ in $\mathcal{PR}$ above, and close it by functional composition and primitive recursion, one gets the set of *partial primitive recursive functions*.

[2] Primitive recursiveness can be defined on subsets of $\mathbb{N}^k$: a subset $S \subseteq \mathbb{N}^k$ is *primitive recursive* if its characteristic function φ_S, which is defined as

$$\varphi_S(x) := \begin{cases} 1 & \text{if } x \in S, \\ 0 & \text{otherwise.} \end{cases}$$

is primitive recursive.

[3] Likewise, primitive recursiveness can be defined for predicates over tuples of natural numbers. A predicate $\Phi(\boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{N}^k$, is said to be *primitive recursive* if the set $S(\Phi) := \{\boldsymbol{x} \mid \Phi(\boldsymbol{x})\}$ is primitive recursive.