CS 260
Design and Analysis of Algorithms
9. Partial Recursive Functions

Mikhail Moshkov

Computer, Electrical and Mathematical Sciences & Engineering Division
King Abdullah University of Science and Technology

Partial Recursive Functions

In this section, we consider the notion of *partial recursive function* which is the recognized formalization of the notion of computable function on nonnegative integers.

Let us denote $\omega = \{0, 1, 2, \ldots\}$. We will consider *partial* functions $f : \omega^n \to \omega$.

Partial means that f is, possibly, undefined on some n-tuples from ω^n.

Elementary Functions

Let $n = 1, 2, \ldots$, and $1 \leq m \leq n$. The following functions will be called *elementary* functions:

1. 0,
2. $s(x) = x + 1$,
3. $I^n_m(x_1, \ldots, x_n) = x_m$.

Substitution Operation C

Let $f(x_1, \ldots, x_m), f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n)$ be partial functions from ω^m or ω^n to ω. Then

$$\Phi(x_1, \ldots, x_n) = f(f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n))$$

is a result of the use of *substitution operation*.

Substitution Operation C

Let $\bar{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \omega^n$. Then the value $\Phi(\bar{\alpha})$ is defined if and only if the values $f_1(\bar{\alpha}), \ldots, f_m(\bar{\alpha})$ and $f(f_1(\bar{\alpha}), \ldots, f_m(\bar{\alpha}))$ are defined.

If $\Phi(\bar{\alpha})$ is defined then

$$\Phi(\bar{\alpha}) = f(f_1(\bar{\alpha}), \ldots, f_m(\bar{\alpha})).$$

Substitution Operation C

Example 9.1. $s(0) = 1$, $s(s(0)) = 2, \ldots$

Operation of Primitive Recursion Pr

Let $\varphi(x_1, \ldots, x_n)$ and $\psi(x_1, \ldots, x_n, x_{n+1}, x_{n+2})$ be partial functions from ω^n and from ω^{n+2} to ω.

We construct a partial function $f(x_1, \ldots, x_n, x_{n+1})$ using the *scheme of primitive recursion*:

$$\begin{cases} f(x_1, \ldots, x_n, 0) = \varphi(x_1, \ldots, x_n), \\ f(x_1, \ldots, x_n, y+1) = \psi(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)). \end{cases}$$

Operation of Primitive Recursion Pr

Let $\bar{\alpha} = (\alpha_1, \ldots, \alpha_{n+1}) \in \omega^{n+1}$. Then $f(\bar{\alpha})$ is defined if and only if $\varphi(\alpha_1, \ldots, \alpha_n)$ is defined, and if $\alpha_{n+1} > 0$ then, for $y = 1, \ldots, \alpha_{n+1} - 1$, the value

$$\psi(\alpha_1, \ldots, \alpha_n, y, f(\alpha_1, \ldots, \alpha_n, y))$$

is defined.

Note that some variables in φ and ψ can be omitted.

Operation of Primitive Recursion Pr

Example 9.2. The following scheme of primitive recursion allows us to obtain the function $f(x_1, x_2) = x_1 + x_2$:

$$\begin{cases} f(x_1, 0) = I_1^1(x_1), \\ f(x_1, y+1) = s(f(x_1, y)). \end{cases}$$

Operation of Primitive Recursion Pr

Example 9.3. Function $f(x_1, x_2) = x_1 \cdot x_2$. It is clear that $x_1 \cdot 0 = 0$ and $x_1 \cdot (x_2 + 1) = x_1 \cdot x_2 + x_1$.

The following scheme of primitive recursion allows us to obtain $x_1 \cdot x_2$:

$$\begin{cases} f(x_1, 0) = 0, \\ f(x_1, y + 1) = f(x, y) + I_1^1(x_1). \end{cases}$$

Operation of Primitive Recursion Pr

Example 9.4. Function $f(x) = 2^x$. It is clear that $2^0 = 1$ and $2^{x+1} = 2 \cdot 2^x$.

The following scheme of primitive recursion allows us to obtain 2^x:

$$\begin{cases} f(0) = s(0), \\ f(y+1) = f(y) + f(y). \end{cases}$$

Operation of Primitive Recursion Pr

Example 9.5. Function $h(n) = 2^{\left.2^{\cdots^2}\right\}n}$. It is clear that $h(0) = 2$ and $h(x+1) = 2^{h(x)}$.

The following scheme of primitive recursion allows us to obtain $h(x)$:

$$\begin{cases} h(0) = 2, \\ h(y+1) = 2^{h(y)}. \end{cases}$$

Operation of Minimization μ

Let $\varphi(x_1, \ldots, x_{n-1}, x_n)$ be a partial function from ω^n to ω. Now we define the function

$$f(x_1, \ldots, x_n) = \mu_y(\varphi(x_1, \ldots, x_{n-1}, y) = x_n).$$

Operation of Minimization μ

Let $\bar{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \omega^n$. We consider the equation

$$\varphi(\alpha_1, \ldots, \alpha_{n-1}, y) = \alpha_n.$$

If this equation has no solutions from ω then $f(\bar{\alpha})$ is undefined.
Let this equation have a solution, and y_0 be the minimum solution.

If all values $\varphi(\alpha_1, \ldots, \alpha_{n-1}, j)$, where $0 \leq j < y_0$, are defined then $f(\bar{\alpha}) = y_0$. Otherwise, the value of $f(\bar{\alpha})$ is undefined.

Operation of Minimization μ

Example 9.6. Let $\varphi(x_1, x_2) = x_1 + x_2$ and

$$f(x_1, x_2) = \mu_y(\varphi(x_1, y) = x_2).$$

Then $x_1 + y = x_2$, $y = x_2 - x_1$, and

$$f(x_1, x_2) = \begin{cases} \text{undefined}, x_2 < x_1, \\ x_2 - x_1, x_2 \geq x_1. \end{cases}$$

Operation of Minimization μ

Example 9.7. Let $\varphi(x_1, x_2) = x_1 \cdot x_2$ and

$$f(x_1, x_2) = \mu_y(\varphi(x_1, y) = x_2).$$

Then $x_1 \cdot y = x_2$, $y = \frac{x_2}{x_1}$, and $f(x_1, x_2)$ is undefined if x_1 is not a divisor of x_2 and

$$f(x_1, x_2) = \frac{x_2}{x_1}$$

otherwise.

Partial Recursive Functions

The set of functions which can be obtained from elementary functions 0, $s(x)$, I_m^n, $n = 1, 2, \ldots$, $1 \leq m \leq n$ with the use of operations C, Pr and μ is called the class of *partial recursive functions*.

Total functions from this class are called *recursive functions*. The set of partial recursive functions is a denumerable set.

Partial Recursive Functions

Almost all mathematicians believe that the set of partial recursive functions coincides with the class of all computable partial functions on nonnegative integers.

This statement is called *Church's thesis*. This is a result of long investigations of various types of computations.

Recursive Sets

Let D be a subset of ω. *Characteristic* function of the set D is the function

$$f_D(n) = \begin{cases} 1, n \in D, \\ 0, n \notin D. \end{cases}$$

The set D is called *recursive* if f_D is a recursive function.

Recursive Sets

We believe that the notion of a recursive set coincides with the notion of a set for which there exists an algorithm that for a given $n \in \omega$ can recognize if n belongs to D.

Recursively Enumerable Set

The set D is called a *recursively enumerable* set if $D = \emptyset$ or there exists a partial recursive function $f(x)$ such that $D = \{f(n) : n \in \omega\}$.

Instead of partial recursive functions we can use total recursive functions in this definition. The result will be the same.

Recursively Enumerable Set

We believe that the notion of recursively enumerable set coincides with the notion of a set for which there exists an algorithm that enumerates D, i.e., generates all numbers from D and only numbers from D.

Recursively Enumerable Set

Theorem 9.8 *(Theorem of Post)* Let $D \subseteq \omega$. Then the set D is a recursive set if and only if the sets D and $\omega \setminus D$ are recursively enumerable sets.

Recursively Enumerable Set

Proof. The proof will be based on Church's thesis. Let D be a recursive set. We will show that D is a recursively enumerable set.

Let us describe a function f: for any $n \in \omega$, the value of f is equal to n if $n \in D$ and undefined otherwise. Since D is a recursive set, f is a computable function.

Recursively Enumerable Set

By Church's thesis, f is a partial recursive function. It is clear that $D = \{f(n) : n \in \omega\}$. In the same way we can prove that $\omega \setminus D$ is a recursively enumerable set.

Let D and $\omega \setminus D$ be recursively enumerable sets. Let us show that D is a recursive set.

Recursively Enumerable Set

We use algorithms that enumerate D and $\omega \setminus D$ and, for a given $n \in \omega$, we wait for appearance of n either in the sequence generated by the first algorithm (in this case $n \in D$), or in the sequence generated by the second one (in this case $n \in \omega \setminus D$).

Therefore there exists an algorithm which, for a given $n \in \omega$, recognizes if n belongs to D. Using Church's thesis we obtain that f_D is a recursive function. $\square$

Recursively Enumerable Set

To use the considered notions for analysis of a decision problem – a language L over an alphabet A – we should encode words from A^* by numbers from ω such that

- different numbers correspond to different words,
- there exists an algorithm that for a given word constructs corresponding number, and
- there exists an algorithm which for a given number n from ω either constructs corresponding word or recognizes that there is no word with number n.

Recursively Enumerable Set

Let $\omega(L)$ be the set of numbers corresponding to words from L. An algorithm which solves the membership problem for L exists if and only if the set $\omega(L)$ is a recursive set.

The set $\omega(L)$ is recursive if and only if the set $\omega(L)$ and the set $\omega \setminus \omega(L)$ are recursively enumerable sets.

Diophantine Equations

To illustrate the notions under study, let us consider so called Diophantine equations. A *Diophantine equation* is an equation of the form $F(x_1, \ldots, x_n) = 0$, where F is a polynomial with integer coefficients.

Hilbert's tenth problem is the following: to find an algorithm which for a given Diophantine equation can recognize if this equation is solvable in integers.

Diophantine Equations

We know that such an algorithm does not exists (Martin Davis, Yuri Matiyasevich, Hilary Putnam and Julia Robinson).

We can encode Diophantine equations by numbers from ω. Let D be the set of numbers corresponding to equations which are solvable in integers. Then D is not a recursive set.

Diophantine Equations

One can show that D is a recursively enumerable set: we can enumerate all pairs (equation, tuple of integer values of variables) and recognize if the considered integers form a solution for the equation. In such a way, we can enumerate all numbers from D.

Therefore (by Theorem of Post) the set $\omega \setminus D$ is not recursively enumerable set. From here it follows that there is no algorithm which can enumerate all Diophantine equations that are not solvable in integers.