

Some Primitive Recursive Functions

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

Here are the recursion equations:

$$h(0) = 1$$

$$h(t + 1) = h(t) \cdot s(t)$$

This can be rewritten as:

$$h(0) = s(n(0))$$

$$h(t + 1) = g(t, h(t))$$

where

$$g(x_1, x_2) = s(x_1) \cdot x_2, \text{ which can be written as:}$$

$$g(x_1, x_2) = s(u_1^2(x_1, x_2)) \cdot u_2^2(x_1, x_2).$$

Multiplication is already known to be primitive recursive.

Therefore, $h(x) = x!$ is primitive recursive.

September 29, 2009

Theory of Computation Lecture 7: Primitive
Recursive Functions III

1

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:

$$x^0 = 1$$

$$x^{y+1} = x^y \cdot x$$

September 21, 2006

Theory of Computation
Lecture 6: Primitive Recursive Functions II

2

Some Primitive Recursive Functions

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

It is defined as follows:

$$p(x) = x - 1 \quad \text{if } x \neq 0$$

$$= 0 \quad \text{if } x = 0$$

The recursion equations are:

$$p(0) = 0$$

$$p(t + 1) = t$$

September 21, 2006

Theory of Computation
Lecture 6: Primitive Recursive Functions II

3

Some Primitive Recursive Functions

Example 6: $x \text{ } ^\circ \text{ } y$ (monus)

It is defined as follows:

$$x \text{ } ^\circ \text{ } y = x - y \quad \text{if } x \geq y$$

$$= 0 \quad \text{if } x < y$$

The recursion equations are:

$$x \text{ } ^\circ \text{ } 0 = x$$

$$x \text{ } ^\circ \text{ } (t + 1) = p(x \text{ } ^\circ \text{ } t)$$

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

4

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:

$$|x - y| = (x \text{ } ^\circ \text{ } y) + (y \text{ } ^\circ \text{ } x)$$

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

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

5

Some Primitive Recursive Functions

Example 8: $\alpha(x)$ ("negation")

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

$$\alpha(x) = 1 \quad \text{if } x = 0$$

$$= 0 \quad \text{if } x \neq 0$$

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

$$\alpha(x) = 1 \text{ } ^\circ \text{ } x$$

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

$$\alpha(0) = 1$$

$$\alpha(t + 1) = 0$$

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

6

Primitive Recursive Predicates

Example 9: $x = y$

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

$$\begin{aligned} d(x, y) &= 1 && \text{if } x = y \\ &= 0 && \text{if } x \neq y \end{aligned}$$

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

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

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

7

Primitive Recursive Predicates

Example 10: $x \leq y$

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

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

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

$$d(x, y) = \alpha(x -^{\circ} y)$$

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

8

Primitive Recursive Predicates

Theorem 5.1: Let \mathcal{C} be a PRC class.

If P, Q are predicates that belong to \mathcal{C} , then so are $\sim P, P \vee Q$, and $P \& Q$.

Proof:

Since $\sim P = \alpha(P)$, it follows that $\sim P$ belongs to \mathcal{C} .

Since $P \& Q = P \cdot Q$, it follows that $P \& Q$ belongs to \mathcal{C} .

Since $P \vee Q = \sim(\sim P \& \sim Q)$, it follows that $P \vee Q$ belongs to \mathcal{C} .

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

9

Primitive Recursive Predicates

We already know two different PRC classes, namely the class of **all primitive recursive functions**, and the class of **all computable functions**.

Correspondingly, we have the following corollaries:

Corollary 5.2: If P, Q are primitive recursive predicates, then so are $\sim P, P \vee Q$, and $P \& Q$.

Corollary 5.3: If P, Q are computable predicates, then so are $\sim P, P \vee Q$, and $P \& Q$.

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

10

Primitive Recursive Predicates

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

$$x < y \Leftrightarrow x \leq y \& \sim(x = y)$$

or even simpler:

$$x < y \Leftrightarrow \sim(y \leq x),$$

according to Corollary 5.2, $x < y$ is also a primitive recursive predicate.

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

11

Primitive Recursive Predicates

Theorem 5.4: Let \mathcal{C} be a PRC class.

Let the functions g, h and the predicate P belong to \mathcal{C} .

Let

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

Then f belongs to \mathcal{C} .

Proof: f obviously belongs to \mathcal{C} , because it can be written as:

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

September 29, 2009

Theory of Computation
Lecture 7: Primitive Recursive Functions III

12

Primitive Recursive Predicates

Corollary 5.5: Let C be a PRC class.

Let n -ary functions g_1, \dots, g_m, h and predicates P_1, \dots, P_m belong to C , and let

$P_i(x_1, \dots, x_n) \ \& \ P_j(x_1, \dots, x_n) = 0$ for all $1 \leq i < j \leq m$ and all x_1, \dots, x_n . If

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

Then f belongs to C .

September 29, 2009 Theory of Computation 13
Lecture 7: Primitive Recursive Functions III

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

$$f(x_1, \dots, x_n) = \begin{cases} g_1(x_1, \dots, x_n) & \text{if } P_1(x_1, \dots, x_n) \\ \vdots & \vdots \\ g_{m+1}(x_1, \dots, x_n) & \text{if } P_{m+1}(x_1, \dots, x_n) \\ h(x_1, \dots, x_n) & \text{otherwise, and let} \end{cases}$$

$$h'(x_1, \dots, x_n) = \begin{cases} g_{m+1}(x_1, \dots, x_n) & \text{if } P_{m+1}(x_1, \dots, x_n) \\ h(x_1, \dots, x_n) & \text{otherwise. Then} \end{cases}$$

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

September 29, 2009 Theory of Computation 14
Lecture 7: Primitive Recursive Functions III

Primitive Recursive Predicates

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 .

September 29, 2009 Theory of Computation 15
Lecture 7: Primitive Recursive Functions III

Primitive Recursive Predicates

Theorem 6.1: Let C be a PRC class. If $f(t, x_1, \dots, x_n)$ belongs to C , then so do the functions

$$g(y, x_1, \dots, x_n) = \sum_{t=0}^y f(t, x_1, \dots, x_n)$$

$$h(y, x_1, \dots, x_n) = \prod_{t=0}^y f(t, x_1, \dots, x_n)$$

To prove this theorem, we will show that g and h can be obtained from f by primitive recursion.

September 29, 2009 Theory of Computation 16
Lecture 7: Primitive Recursive Functions III