

Homework Questions

Question 1:
Write a program \mathcal{P} that computes $\Psi_{\mathcal{P}}^{(1)}(r) = r-2$ using no macros.

Notice that we only consider natural numbers! This means that, for example, the expression 1-2 is undefined. Any program computing such an expression **must never halt**.

Sample solution:

<p>[A] $Z \leftarrow Z+1$ IF $Z \neq 0$ GOTO A</p> <p>[B] $X \leftarrow X-1$ IF $X \neq 0$ GOTO D</p>	<p>[C] $Z \leftarrow Z+1$ IF $Z \neq 0$ GOTO C</p> <p>[D] $X \leftarrow X-1$ IF $X \neq 0$ GOTO D</p>
---	---

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 1

Homework Questions

Question 2:
a) Write a program \mathcal{S} that computes $\Psi_{\mathcal{S}}^{(1)}(r) = r-4$. Use a macro based on program \mathcal{P} of the form $W \leftarrow p(V)$ (which has the effect $W \leftarrow V-2$).

Sample solution:

$Y \leftarrow p(X)$
 $Y \leftarrow p(Y)$

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 2

Homework Questions

b) Now comes the best part: Expand all macros in \mathcal{S} using the method we discussed in the lecture.

<p>$Z_2 \leftarrow 0$ // expansion of $Y \leftarrow p(X)$ with $m = 2$</p> <p>$Z_3 \leftarrow X$ $Z_4 \leftarrow 0$ IF $Z_3 \neq 0$ GOTO A_4</p> <p>[A₃] $Z_4 \leftarrow Z_4+1$ IF $Z_4 \neq 0$ GOTO A_3</p> <p>[A₄] $Z_3 \leftarrow Z_3-1$ IF $Z_3 \neq 0$ GOTO A_6</p> <p>[A₅] $Z_4 \leftarrow Z_4+1$ IF $Z_4 \neq 0$ GOTO A_5</p> <p>[A₆] $Z_3 \leftarrow Z_3-1$ $Z_2 \leftarrow Z_2+1$ IF $Z_3 \neq 0$ GOTO A_6 $Z_2 \leftarrow Z_2-1$ $Y \leftarrow Z_2$</p>	<p>$Z_7 \leftarrow 0$ // exp. of $Y \leftarrow p(Y)$ with $m = 7$</p> <p>$Z_8 \leftarrow Y$ $Z_9 \leftarrow 0$ IF $Z_8 \neq 0$ GOTO A_9</p> <p>[A₈] $Z_9 \leftarrow Z_9+1$ IF $Z_9 \neq 0$ GOTO A_8</p> <p>[A₉] $Z_8 \leftarrow Z_8-1$ IF $Z_8 \neq 0$ GOTO A_{11}</p> <p>[A₁₀] $Z_9 \leftarrow Z_9+1$ IF $Z_9 \neq 0$ GOTO A_{10}</p> <p>[A₁₁] $Z_8 \leftarrow Z_8-1$ $Z_7 \leftarrow Z_7+1$ IF $Z_8 \neq 0$ GOTO A_{11} $Z_7 \leftarrow Z_7-1$ $Y \leftarrow Z_7$</p>
---	---

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 3

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**.

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 4

Some Primitive Recursive Functions

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

$h(x_1, \dots, x_n, 0) = f(x_1, \dots, x_n)$
 $h(x_1, \dots, x_n, t + 1) = g(t, h(x_1, \dots, x_n, t), x_1, \dots, x_n)$.

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

$h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_k(x_1, \dots, x_n))$.

Then h is said to be obtained from f and g_1, \dots, g_k by **composition**.

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 5

Some Primitive Recursive Functions

We defined the following **initial functions**:

$s(x) = x + 1$
 $n(x) = 0$
 $u_i^n(x_1, \dots, x_n) = x_i, 1 \leq i \leq n$.

September 24, 2009 Theory of Computation Lecture 6: Primitive Recursive Functions II 6

Some Primitive Recursive Functions

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:

$$f(x, 0) = u_1^1(x)$$

$$f(x, y + 1) = g(y, f(x, y), x)$$

where $g(x_1, x_2, x_3) = s(u_2^3(x_1, x_2, x_3))$.

Obviously, $u_1^1(x)$, $u_2^3(x_1, x_2, x_3)$, and $s(x)$ 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.

September 24, 2009

Theory of Computation
Lecture 6: Primitive Recursive Functions II

7

Some Primitive Recursive Functions

Example 2: $h(x, y) = x \cdot y$

We can obtain the following recursive definition:

$$h(x, 0) = 0$$

$$h(x, y + 1) = h(x, y) + x$$

This can be rewritten as:

$$h(x, 0) = n(x)$$

$$h(x, y + 1) = g(y, h(x, y), x)$$

where

$$f(x_1, x_2) = x_1 + x_2 \text{ and}$$

$$g(x_1, x_2, x_3) = f(u_2^3(x_1, x_2, x_3), u_3^3(x_1, x_2, x_3)).$$

Obviously, these are all primitive recursive functions.

Therefore, $h(x, y) = x \cdot y$ is primitive recursive.

September 24, 2009

Theory of Computation
Lecture 6: Primitive Recursive Functions II

8