ECE 664
LECTURE 5
PURDUE UNIV.
AVI KAK

PRIMITIVE RECURSIVE FUNCTIONS

- If for a total function $g(r_1, \ldots, r_m)$ there exists a program $\mathbb{P}$ in language $\mathcal{S}$ such that

$$g(r_1, \ldots, r_m) = \psi_{\mathbb{P}}^{(m)}(r_1, \ldots, r_m)$$

everywhere in the domain of g, then g is a computable function

- In our quest to understand the power of the language $\mathcal{S}$, we are interested in the set of all computable functions.
- It turns out that <u>all</u> Primitive Recursive Functions are computable
- Not only that, practically all computable functions are primitive recursive. (In fact, you have to work hard at coming up with an analytically definable computable function that is not primitive recursive.)
- Hence we are very interested in Primitive Recursive Functions.

The following two notions are central to the definition of a Primitive Recursive Function:

- Composition
- Recursion

Composition

Let f be a function of k variables.
Let $g_1, g_2, \ldots, g_k$ be k functions, each of n variables.
Then

$$h(x_1, \ldots, x_n) = f(g_1, g_2, \ldots, g_k)$$

is a composition obtained from f and g_1 through g_k.

If h is obtained from the computable (meaning there <u>exists</u> a program $\mathbb{P}$ that computes the function) $f, g_1, \ldots, g_k$, then h is also computable.

Program for computing h:

$$z_1 \leftarrow g_1(x_1, \ldots, x_n)$$
$$\vdots$$
$$z_k \leftarrow g_k(x_1, \ldots, x_n)$$
$$y \leftarrow f(z_1, z_2, \ldots, z_k)$$

Express $h(x) = 4x^2 - 2x$ as a composition. (There is more than one way to do it.)

Recursion

Suppose

$$h(0) = k$$
$$h(t+1) = g(t, h(t))$$

a total function of two variables

then h is said to be obtained from g by recursion. The function h is computable because we have:

```
    y ← k
[A] IF X = 0 GO TO E
    y ← g(Z, y)
    Z ← Z + 1
    X ← X − 1
    GO TO A
```

macro for

```
y ← y+1
y ← y+1
 ⋮
y ← y+1
```

k times

If $P(x)$ is a predicate, we can write a macro for IF P(X) GO TO L (chap 2)

A more general definition of recursion:

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

function of $n+1$ variables

total function of $n+2$ variables

$x_1, \ldots, x_n$

In order to build up a class of functions by recursion and composition, we need some <u>initial functions</u>:

① $s(x) = x + 1$ successor function

② $n(x) = 0$ the zero function

③ $u_i^n(x_1, \dots, x_n) = x_i$ the projection function

Note that the initial functions are all computable:

① $s(x)$ is computed by the program
$$y \leftarrow X + 1$$

② $n(x)$ is computed by the empty program. → $\mathcal{P}$ of length 0

③ $u_i^n(x_1, \dots, x_n)$ is computed by the program
$$y \leftarrow X_i$$

DEF.: A function is called primitive recursive if it can be obtained from <u>the</u> initial functions by a <u>finite</u> number of applications of composition and recursion.

- The class of primitive recursive functions is <u>closed</u> with respect to recursions and compositions.

  just as the set of integers N is closed with respect to addition

- Every primitive recursive function is computable. (total) (Because the recursions and compositions are built up from only the computable initial functions)

Some Primitive Recursive Functions

① $f(x, y) = x + y$ ← To show that f is primitive recursive, we must show that it can be obtained from <u>the</u> initial functions by composition and recursion.

Analytically, we can write it as

$$\boxed{\begin{aligned} f(x, 0) &= x \\ f(x, y+1) &= f(x, y) + 1 \end{aligned}}$$

$$\boxed{\begin{aligned} f(x, 0) &= u_1^1(x) \\ f(x, y+1) &= s\left(u_2^3(y, f(x,y), x)\right) \end{aligned}}$$

② $h(x, y) = x \cdot y$

$$\boxed{\begin{aligned} h(x, 0) &= 0 \\ h(x, y+1) &= h(x, y) + x \end{aligned}}$$

$$\begin{aligned} h(x, 0) &= n(x) \\ h(x, y+1) &= f\left(u_2^3(y, h(x,y), x), u_3^3(y, h(x,y), x)\right) \end{aligned}$$

③ $x!$

④ x^y

⑤ $p(x) = x - 1$ if $x \neq 0$
$= 0$ if $x = 0$

⑥ $x \dot{-} y = x - y$ if $x \geq y$
$= 0$ if $x < y$

⑦ $|x - y|$

⑧ $\alpha(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x \neq 0 \end{cases}$