Fundamentele Informatica 3

voorjaar 2015

`http://www.liacs.leidenuniv.nl/~vlietrvan1/fi3/`

Rudy van Vliet
kamer 124 Snellius, tel. 071-527 5777
rvvliet(at)liacs(dot)nl

college 13, 11 mei 2015

10. Computable Functions
10.2. Quantification, Minimalization, and μ-Recursive Functions

Huiswerkopgave 3,
inleverdatum 12 mei 2015, 13:45 uur

A slide from lecture 11:

Definition 10.1. Initial Functions

The initial functions are the following:

1. *Constant* functions: For each $k \geq 0$ and each $a \geq 0$, the constant function $C_a^k : \mathbb{N}^k \to \mathbb{N}$ is defined by the formula

$$C_a^k(X) = a \quad \text{for every } X \in \mathbb{N}^k$$

2. The *successor* function $s : \mathbb{N} \to \mathbb{N}$ is defined by the formula

$$s(x) = x + 1$$

3. *Projection* functions: For each $k \geq 1$ and each i with $1 \leq i \leq k$, the projection function $p_i^k : \mathbb{N}^k \to \mathbb{N}$ is defined by the formula

$$p_i^k(x_1, x_2, \ldots, x_k) = x_i$$

A slide from lecture 11:

Definition 10.2. The Operations of Composition and Primitive Recursion

1. Suppose f is a partial function from $\mathbb{N}^k$ to $\mathbb{N}$, and for each i with $1 \leq i \leq k$, g_i is a partial function from $\mathbb{N}^m$ to $\mathbb{N}$.
   The partial function obtained from f and $g_1, g_2, \ldots, g_k$ by composition is the partial function h from $\mathbb{N}^m$ to $\mathbb{N}$ defined by the formula

$$h(X) = f(g_1(X), g_2(X), \ldots, g_k(X)) \text{ for every } X \in \mathbb{N}^m$$

A slide from lecture 11:

Definition 10.2. The Operations of Composition and Primitive Recursion (continued)

2. Suppose $n \geq 0$ and g and h are functions of n and $n+2$ variables, respectively. (By "a function of 0 variables," we mean simply a constant.)
   The function obtained from g and h by the operation of *primitive recursion* is the function $f : \mathbb{N}^{n+1} \to \mathbb{N}$ defined by the formulas
   $$\begin{aligned} f(X, 0) &= g(X) \\ f(X, k+1) &= h(X, k, f(X, k)) \end{aligned}$$
   for every $X \in \mathbb{N}^n$ and every $k \geq 0$.

A slide from lecture 12:

n*-place predicate* P is function from $\mathbb{N}^n$ to $\{\text{true}, \text{false}\}$

characteristic function χ_P defined by

$$\chi_P(X) = \begin{cases} 1 & \text{if } P(X) \text{ is true} \\ 0 & \text{if } P(X) \text{ is false} \end{cases}$$

We say P is primitive recursive...

A slide from lecture 12:

Theorem 10.6.

The two-place predicates *LT*, *EQ*, *GT*, *LE*, *GE*, and *NE* are primitive recursive.
(*LT* stands for "less than," and the other five have similarly intuitive abbreviations.)
If P and Q are any primitive recursive n-place predicates, then $P \wedge Q$, $P \vee Q$ and $\neg P$ are primitive recursive.

Proof. . .

A slide from lecture 12:

Exercise.

Let $f : \mathbb{N}^{n+1} \to \mathbb{N}$ be a primitive recursive function.

Show that the predicate $P : \mathbb{N}^{n+1} \to \{\text{true}, \text{false}\}$ defined by

$$P(X, y) = \ (f(X, y) = 0)$$

is primitive recursive.

A slide from lecture 12:

Theorem 10.7.

Suppose $f_1, f_2, \ldots, f_k$ are primitive recursive functions from $\mathbb{N}^n$ to $\mathbb{N}$,
$P_1, P_2, \ldots, P_k$ are primitive recursive n-place predicates,
and for every $X \in \mathbb{N}^n$,
exactly one of the conditions $P_1(X), P_2(X), \ldots, P_k(X)$ is true.
Then the function $f : \mathbb{N}^n \to \mathbb{N}$ defined by

$$f(X) = \begin{cases} f_1(X) & \text{if } P_1(X) \text{ is true} \\ f_2(X) & \text{if } P_2(X) \text{ is true} \\ \ldots & \\ f_k(X) & \text{if } P_k(X) \text{ is true} \end{cases}$$

is primitive recursive.

Proof. . .

10.2. Quantification, Minimalization, and μ-Recursive Functions

A slide from lecture 12:

Theorem 10.4.

Every primitive recursive function is total and computable.

PR:
total and computable

Turing-computable functions:
not necessarily total

(Un)bounded quantification

Sq$(x, y) =$ $(y^2 = x)$

PerfectSquare$(x) =$ there exists y such that $y^2 = x$

(Un)bounded quantification

$\mathit{Sq}(x, y) = \quad (y^2 = x)$

$\mathit{PerfectSquare}(x) =$ there exists y such that $y^2 = x$

$E_{\mathit{Sq}}(x, k) =$ there exists $y \leq k$ such that $y^2 = x$

(Un)bounded quantification

$H(x, y) =$ T_u halts after exactly y moves on input s_x

(Un)bounded quantification

$H(x, y) =$ T_u halts after exactly y moves on input s_x

Halts$(x) =$ there exists y such that
T_u halts after exactly y moves on input s_x

(Un)bounded quantification

$H(x, y) =$ T_u halts after exactly y moves on input s_x

Halts$(x) =$ there exists y such that
T_u halts after exactly y moves on input s_x

$E_H(x, k) =$ there exists $y \leq k$ such that
T_u halts after exactly y moves on input s_x

Definition 10.9. Bounded Quantifications

Let P be an $(n+1)$-place predicate. The *bounded existential quantification* of P is the $(n+1)$-place predicate E_P defined by

$E_P(X,k) =$ (there exists y with $0 \leq y \leq k$ such that $P(X,y)$ is true)

The *bounded universal quantification* of P is the $(n+1)$-place predicate A_P defined by

$$A_P(X,k) = (\text{for every } y \text{ satifying } 0 \leq y \leq k,\ P(X,y) \text{ is true})$$

Theorem 10.10.

If P is a primitive recursive $(n+1)$-place predicate,
both the predicates E_P and A_P are also primitive recursive.

Proof. . .

A slide from lecture 12:

Theorem 10.4.

Every primitive recursive function is total and computable.

PR:
total and computable

Turing-computable functions:
not necessarily total

Definition 10.11. Bounded Minimalization

For an $(n+1)$-place predicate P, the *bounded minimalization* of P is the function $m_p : \mathbb{N}^{n+1} \to \mathbb{N}$ defined by

$$m_p(X,k) = \begin{cases} \text{min}\{y \mid \ 0 \leq y \leq k \text{ and } P(X,y)\} & \text{if this set is not empty} \\ k+1 & \text{otherwise} \end{cases}$$

Definition 10.11. Bounded Minimalization

For an $(n+1)$-place predicate P, the *bounded minimalization* of P is the function $m_P : \mathbb{N}^{n+1} \to \mathbb{N}$ defined by

$$m_P(X,k) = \begin{cases} \text{min}\{y \mid \ 0 \le y \le k \text{ and } P(X,y)\} & \text{if this set is not empty} \\ k+1 & \text{otherwise} \end{cases}$$

The symbol μ is often used for the minimalization operator, and we sometimes write

$$m_P(X,k) = \overset{k}{\mu}\, y[P(X,y)]$$

An important special case is that in which $P(X,y)$ is $(f(X,y) = 0)$, for some $f : \mathbb{N}^{n+1} \to \mathbb{N}$. In this case m_P is written m_f and referred to as the bounded minimalization of f.

Theorem 10.12.

If P is a primitive recursive $(n+1)$-place predicate,
its bounded minimalization m_P is a primitive recursive function.

Proof. . .

Example 10.13. The nth Prime Number

$$PrNo(0) = 2$$
$$PrNo(1) = 3$$
$$PrNo(2) = 5$$

Example 10.13. The nth Prime Number

$PrNo(0) = 2$
$PrNo(1) = 3$
$PrNo(2) = 5$

$$\begin{aligned} Prime(n) \;=\; & (n \geq 2) \wedge \neg(\text{there exists } y \text{ such that} \\ & \qquad y \geq 2 \wedge y \leq n-1 \wedge Mod(n,y) = 0) \end{aligned}$$

Example 10.13. The nth Prime Number

Let

$$P(x, y) = \ (y > x \wedge \mathit{Prime}(y))$$

Then $m_P(x, k)$ . . .
and

$$\begin{aligned} \mathit{PrNo}(0) &= 2 \\ \mathit{PrNo}(k+1) &= m_P(\mathit{PrNo}(k), (\mathit{PrNo}(k))! + 1) \end{aligned}$$

is primitive recursive, with $h(x_1, x_2) = \ldots$

A slide from lecture 12:

Theorem 10.4.

Every primitive recursive function is total and computable.

PR:
total and computable

Turing-computable functions:
not necessarily total

Unbounded minimalization

Total?

Unbounded minimalization

Total?

A possible definition:

$$M(X) = \begin{cases} (\min\{y \mid \ P(X,y) \text{ is true}\}) + 1 & \text{if this set is not empty} \\ 0 & \text{otherwise} \end{cases}$$

Computable?

(Un)bounded quantification

$H(x, y) =$ T_u halts after exactly y moves on input s_x

Halts$(x) =$ there exists y such that
T_u halts after exactly y moves on input s_x

Definition 10.14. Unbounded Minimalization

If P is an $(n+1)$-place predicate, the *unbounded minimalization* of P is the partial function $M_P : \mathbb{N}^n \to \mathbb{N}$ defined by

$$M_P(X) = \text{min}\{y \mid \ P(X, y) \text{ is true}\}$$

$M_P(X)$ is undefined at any $X \in \mathbb{N}^n$ for which there is no y satisfying $P(X, y)$.

Definition 10.14. Unbounded Minimalization

If P is an $(n+1)$-place predicate, the *unbounded minimalization* of P is the partial function $M_P : \mathbb{N}^n \to \mathbb{N}$ defined by

$$M_P(X) = \min\{y \mid \ P(X,y) \text{ is true}\}$$

$M_P(X)$ is undefined at any $X \in \mathbb{N}^n$ for which there is no y satisfying $P(X,y)$.

The notation $\mu\, y[P(X,y)]$ is also used for $M_P(X)$.
In the special case in which $P(X,y) = (f(X,y) = 0)$, we write $M_P = M_f$ and refer to this function as the unbounded minimalization of f.

Definition 10.15. μ-Recursive Functions

The set $\mathcal{M}$ of μ-recursive, or simply *recursive*, partial functions is defined as follows.

1. Every initial function is an element of $\mathcal{M}$.

2. Every function obtained from elements of $\mathcal{M}$ by composition or primitive recursion is an element of $\mathcal{M}$.

3. For every $n \geq 0$ and every total function $f : \mathbb{N}^{n+1} \to \mathbb{N}$ in $\mathcal{M}$, the function $M_f : \mathbb{N}^n \to \mathbb{N}$ defined by
$$M_f(X) = \mu\, y[f(X, y) = 0]$$
is an element of $\mathcal{M}$.

Example.

Let

$$f(x,k) = p_1^2(x,k) \dot{-} C_1^2(x,k)$$

$M_f(x)$ . . .

Exercise.

a. Give an example of a non-total function f and another function g, such that the composition of f and g is total.

b. Can you also find an example of a non-total function f and another function g, such that the composition of g and f is total?

Theorem 10.16.

All μ-recursive partial functions are computable.

Proof. . .

10.3. Gödel Numbering

Definition 10.17.
The Gödel Number of a Sequence of Natural Numbers

For every $n \geq 1$ and every finite sequence $x_0, x_1, \ldots, x_{n-1}$ of n natural numbers, the *Gödel number* of the sequence is the number

$$\mathit{gn}(x_0, x_1, \ldots, x_{n-1}) = 2^{x_0} 3^{x_1} 5^{x_2} \ldots (\mathit{PrNo}(n-1))^{x_{n-1}}$$

where $\mathit{PrNo}(i)$ is the ith prime (Example 10.13).

An exercise from exercise class 11:

Exercise 10.16.

Show that for any $n \geq 1$, the functions Add_n and Mult_n from $\mathbb{N}^n$ to $\mathbb{N}$, defined by

$$\begin{aligned} \mathit{Add}_n(x_1, \ldots, x_n) &= x_1 + x_2 + \cdots + x_n \\ \mathit{Mult}_n(x_1, \ldots, x_n) &= x_1 * x_2 * \cdots * x_n \end{aligned}$$

respectively, are both primitive recursive.