Fundamentele Informatica 3

najaar 2016

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

Rudy van Vliet
kamer 143 Snellius, tel. 071-527 5777
rvvliet(at)liacs(dot)nl

college 13, 29 november 2016

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

Huiswerkopgave 3, inleverdatum vrijdag 2 december 2016, 13:45 uur

A slide from lecture 12

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 12

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 12

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...

Theorem 10.6.

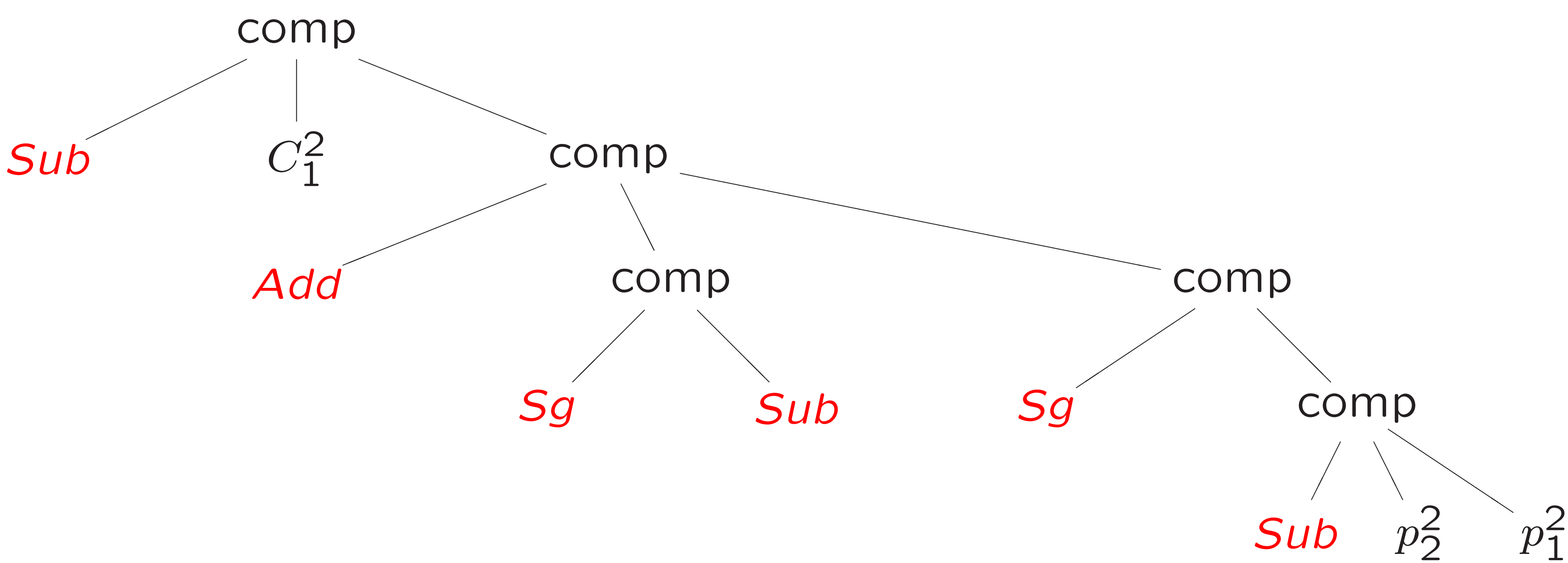
The two-place predicates *LT*, *EQ*, *GT*, *LE*, *GE*, and *NE* are primitive recursive.
(*LT* stands for "less than," and the other five have similarly intuitive abbreviations.)
If P and Q are any primitive recursive n-place predicates, then $P \wedge Q$, $P \vee Q$ and $\neg P$ are primitive recursive.

Proof. . .

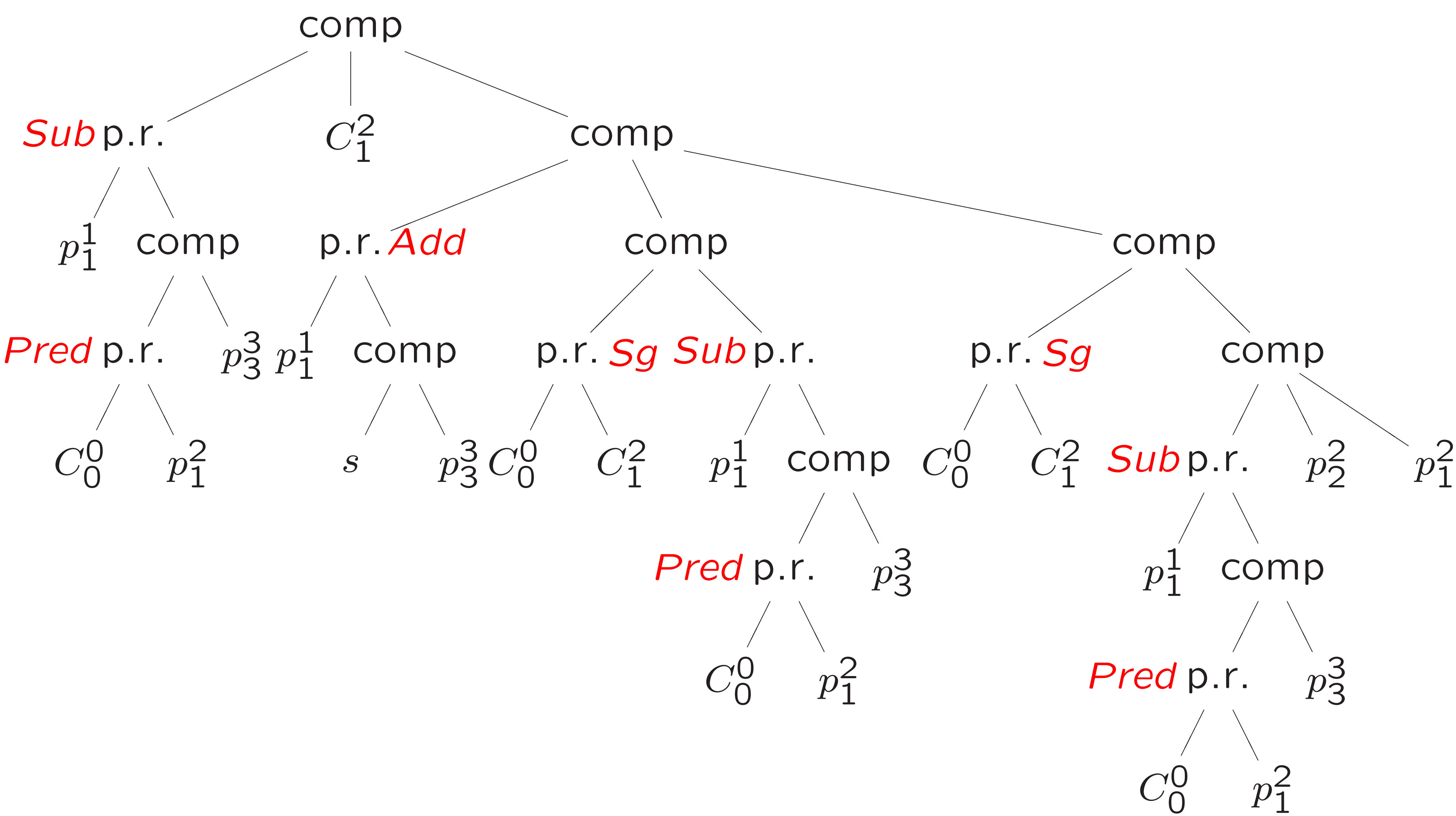
Structure tree $\chi_{EQ}\ldots$

$$\begin{tikzcd}[column sep=small]
 & \text{p.r. } \textcolor{red}{Sg} \arrow[dl, dash] \arrow[dr, dash] & \\
C_0^0 & & C_1^2
\end{tikzcd}$$

Structure tree χ_{EQ}:



Structure tree χ_{EQ}:



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.

Let P be n-place predicate,
$f_1, f_2, \ldots, f_n : \mathbb{N}^k \to \mathbb{N}$
Then $Q = P(f_1, f_2, \ldots, f_n)$ is k-place predicate, with

$$\chi_Q = \chi_P(f_1, f_2, \ldots, f_n)$$

Primitive recursiveness. . .

Let P be n-place predicate,
$f_1, f_2, \ldots, f_n : \mathbb{N}^k \to \mathbb{N}$
then $Q = P(f_1, f_2, \ldots, f_n)$ is k-place predicate,

$$\chi_Q = \chi_P(f_1, f_2, \ldots, f_n)$$

Primitive recursiveness...

Example.

$$(f_1 = (3f_2)^2 \wedge (f_3 < f_4 + f_5)) \vee \neg(P \vee Q)$$

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. . .

Example 10.8. The *Mod* and *Div* Functions

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:	Turing-computable functions:
total and computable	not necessarily total

(Un)bounded quantification

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

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

(Un)bounded quantification

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

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

$E_{Sq}(x, k) =$ 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

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

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

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

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.

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.

Theorem 10.12.

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

Proof. . .

$$h(X,y,z) = \begin{cases} z & \text{if } z \leq y \\ y+1 & \text{if } z \geq y+1 \ \wedge \ P(X,y+1) \text{ is true} \\ y+2 & \text{if } z \geq y+1 \ \wedge \ \neg P(X,y+1) \text{ is true} \end{cases}$$

$$h(X,y,z) = \begin{cases} z & \text{if } E_P(X,y) \text{ is true} \\ y+1 & \text{if } \neg E_P(X,y) \ \wedge \ P(X,y+1) \text{ is true} \\ y+2 & \text{if } \neg E_P(X,y) \ \wedge \ \neg P(X,y+1) \text{ is true} \end{cases}$$