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 15, 19 mei 2015

10. Computable Functions
10.4. All Computable Functions are μ-Recursive
10.5. Other Approaches to Computability

A slide from lecture 13

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

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

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

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

A slide from lecture 13

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 13

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

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

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.

A slide from lecture 13

Theorem 10.12.

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

Proof. . .

A slide from lecture 13

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 $\mathcal{M}$.

$$X = (x_1, x_2, \ldots, x_n)$$

$$q_0 \ \boxed{\underline{\Delta} 1^{x_1} \Delta 1^{x_2} \Delta \cdots \Delta 1^{x_n} \Delta \cdots}$$

$$\vdash$$

$$q_{\ldots} \ \boxed{\Delta \underline{1^{x_1}} \Delta 1^{x_2} \Delta \cdots \Delta 1^{x_n} \Delta \cdots}$$

$$\vdash$$

$$\cdots$$

$$\vdash$$

$$h_a \ \boxed{\underline{\Delta} 1^{f(X)} \Delta \cdots}$$

$$f(X)$$

$$X = (x_1, x_2, \ldots, x_n)$$

$$
\begin{array}{llcl}
q_0 & \boxed{\underline{\Delta} 1^{x_1} \Delta 1^{x_2} \Delta \cdots \Delta 1^{x_n} \Delta \cdots} & \xrightarrow{\textit{confignumber}} & m_0 \\
 & \quad\vdash & & \\
q_{\ldots} & \boxed{\Delta \underline{1^{x_1}} \Delta 1^{x_2} \Delta \cdots \Delta 1^{x_n} \Delta \cdots} & \xrightarrow{\textit{confignumber}} & m_1 \\
 & \quad\vdash & & \\
 & \quad\cdots & \cdots & \cdots \\
 & \quad\vdash & & \\
h_a & \boxed{\underline{\Delta} 1^{f(X)} \Delta \cdots} & \xrightarrow{\textit{confignumber}} & m_l
\end{array}
$$

$$f(X)$$

$$
\begin{tikzcd}
 & X = (x_1, x_2, \ldots, x_n) \arrow[dr, "\mathit{InitConfig}^{(n)}"] & \\
q_0 \ \underline{\underline{\Delta}}1^{x_1}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots \arrow[rr, "\mathit{confignumber}"] \arrow[d, phantom, "\vdash"] & & m_0 \arrow[ddd, bend left, "f_T"] \\
q_{\ldots} \ \Delta\underline{1^{x_1}}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots \arrow[rr, "\mathit{confignumber}"] \arrow[d, phantom, "\vdash"] & & m_1 \\
\cdots \arrow[d, phantom, "\vdash"] & \cdots & \cdots \\
h_a \ \underline{\underline{\Delta}}1^{f(X)}\Delta \cdots \arrow[rr, "\mathit{confignumber}"] & & m_l \arrow[dl, "\mathit{Result}_T"] \\
 & f(X) &
\end{tikzcd}
$$

A slide from lecture 14

We must show that $f : \mathbb{N}^n \to \mathbb{N}$ defined by

$$f(X) = \mathit{Result}_T(f_T(\mathit{InitConfig}^{(n)}(X)))$$

is μ-recursive.

Step 2

The predicate $\mathit{IsConfig}_T$ defined by

$$\mathit{IsConfig}_T(m) = (m \text{ is configuration number for } T)$$

A slide from lecture 14

Now different numbering

Let $T = (Q, \Sigma, \Gamma, q_0, \delta)$ be Turing machine

States:

h_a	h_r	q_0	$\ldots$	.
0	1	2	$\ldots$	s_T

with $s_T = |Q| + 1$

Tape symbols:

Δ	$\ldots$	.
0	$\ldots$	ts_T

with $ts_T = |\Gamma|$

$$\begin{aligned} \mathit{tapenumber}(\Delta 1 a \Delta b 1 \Delta) &= 2^0 3^1 5^2 7^0 11^3 13^1 17^0 \ldots \\ \mathit{confignumber} &= 2^q 3^P 5^{\mathit{tapenumber}} \end{aligned}$$

Step 2 (continued)

The function $\mathit{IsAccepting}_T$ defined by

$$\mathit{IsAccepting}_T(m) = \begin{cases} 0 & \text{if } m \text{ represents accepting config. of } T \\ 1 & \text{otherwise} \end{cases}$$

Step 2 (continued)

The function $\mathit{IsAccepting}_T$ defined by

$$\mathit{IsAccepting}_T(m) = \begin{cases} 0 & \text{if } \mathit{IsConfig}_T(m) \wedge \mathit{Exponent}(0, m) = 0 \\ 1 & \text{otherwise} \end{cases}$$

Step 3

The function $\mathit{Result}_T \ldots$

Step 3

The function Result_T

$$\mathit{Result}_T(m) = \begin{cases} \mathit{HighestPrime}(\mathit{Exponent}(2, m)) & \text{if } \mathit{IsConfig}_T(m) \\ 0 & \text{otherwise} \end{cases}$$

Exercise 10.22.

Show that the function *HighestPrime* introduced in Section 10.4 is primitive recursive.

$$\mathit{HighestPrime}(k) = \begin{cases} 0 & \text{if } k \leq 1 \\ \max\{i \mid \mathit{Exponent}(i,k) > 0\} & \text{if } k \geq 2 \end{cases}$$

An exercise from exercise class 12

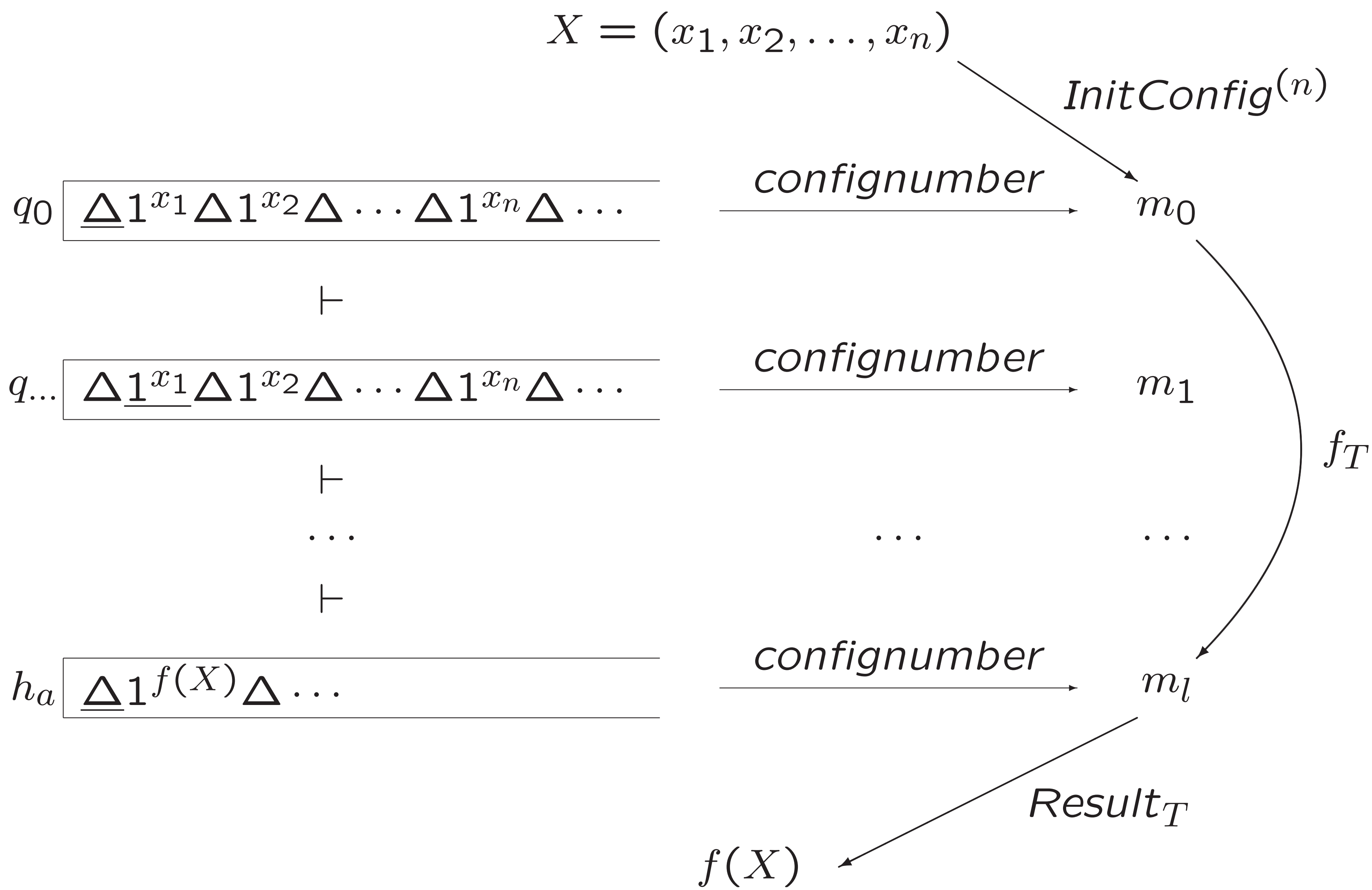
Exercise 10.23.

In addition to the bounded minimalization of a predicate, we might define the bounded maximalization of a predicate P to be the function m^P defined by

$$m^P(X,k) = \begin{cases} \text{max}\{y \leq k \mid \ P(x,y) \text{ is true}\} & \text{if this set is not empty} \\ 0 & \text{otherwise} \end{cases}$$

a. Show m^P is primitive recursive by finding two primitive recursive functions from which it can be obtained by primitive recursion.

b. Show m^P is primitive recursive by using bounded minimalization.


$X = (x_1, x_2, \ldots, x_n)$
$InitConfig^{(n)}$
q_0 $\underline{\underline{\Delta}}1^{x_1}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots$
confignumber
m_0
$\vdash$
$q_{\ldots}$ $\Delta\underline{1^{x_1}}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots$
confignumber
m_1
f_T
$\vdash$
$\cdots$ $\cdots$ $\cdots$
$\vdash$
h_a $\underline{\underline{\Delta}}1^{f(X)}\Delta \cdots$
confignumber
m_l
$Result_T$
$f(X)$

Step 4

$$\begin{aligned}
\mathit{State}(m) &= \mathit{Exponent}(0, m) \\
\mathit{Posn}(m) &= \mathit{Exponent}(1, m) \\
\mathit{TapeNumber}(m) &= \mathit{Exponent}(2, m) \\
\mathit{Symbol}(m) &= \mathit{Exponent}(\mathit{Posn}(m), \mathit{TapeNumber}(m))
\end{aligned}$$

Step 4

$$
\begin{aligned}
\mathit{NewState}(m) &= \ldots \\
\mathit{NewSymbol}(m) &= \ldots \\
\mathit{NewPosn}(m) &= \ldots \\
\mathit{NewTapeNumber}(m) &= \ldots
\end{aligned}
$$

Exercise 10.35.

Show that the function *NewTapeNumber* discussed in Section 10.4 is primitive recursive.

Suggestion: Determine the prime factor of *TapeNumber*(m) that may change by a move of the Turing machine, when the tape head is at position *Posn*(m).

Step 5

The function $\mathit{Move}_T : \mathbb{N} \to \mathbb{N}$ defined by

$$\mathit{Move}_T(m) = \begin{cases} \mathit{gn}(\mathit{NewState}(m), \mathit{NewPosn}(m), \mathit{NewTapeNumber}(m)) \\ \qquad \text{if } \mathit{IsConfig}_T(m) \\ 0 \quad \text{otherwise} \end{cases}$$

Step 6

The function $\mathit{Moves}_T : \mathbb{N}^2 \to \mathbb{N}$ defined by

$$\mathit{Moves}_T(m, 0) = \begin{cases} m & \text{if } \mathit{IsConfig}_T(m) \\ 0 & \text{otherwise} \end{cases}$$

$$\mathit{Moves}_T(m, k+1) = \begin{cases} \mathit{Move}_T(\mathit{Moves}_T(m, k)) & \text{if } \mathit{IsConfig}_T(m) \\ 0 & \text{otherwise} \end{cases}$$

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 $k \geq 0$.

Step 7

The function *NumberOfMovesToAccept*$_T : \mathbb{N} \to \mathbb{N}$ defined by

$$\textit{NumberOfMovesToAccept}_T(m) = \mu\, y[\textit{IsAccepting}_T(\textit{Moves}_T(m, y)) = 0]$$

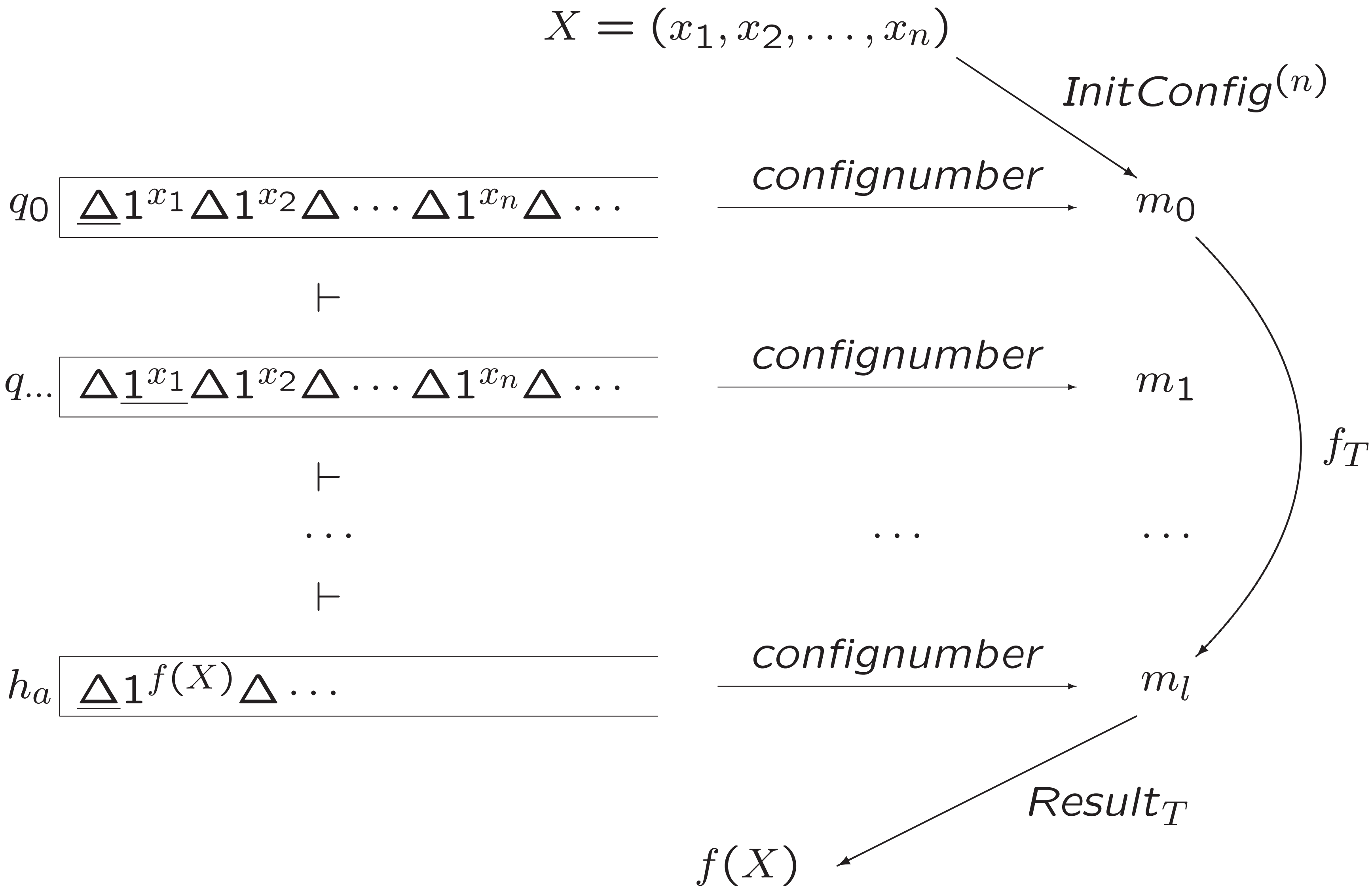
Step 7

The function $\mathit{NumberOfMovesToAccept}_T : \mathbb{N} \to \mathbb{N}$ defined by

$$\mathit{NumberOfMovesToAccept}_T(m) = \mu\, y[\mathit{IsAccepting}_T(\mathit{Moves}_T(m, y)) = 0]$$

The function $f_T : \mathbb{N} \to \mathbb{N}$ defined by

$$f_T(m) = \mathit{Moves}_T(m, \mathit{NumberOfMovesToAccept}_T(m))$$


$X = (x_1, x_2, \ldots, x_n)$
$InitConfig^{(n)}$
q_0 $\underline{\underline{\Delta}}1^{x_1}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots$
confignumber
m_0
$\vdash$
$q_{\ldots}$ $\Delta\underline{1^{x_1}}\Delta 1^{x_2}\Delta \cdots \Delta 1^{x_n}\Delta \cdots$
confignumber
m_1
f_T
$\vdash$
$\cdots$
$\cdots$
$\cdots$
$\vdash$
h_a $\underline{\underline{\Delta}}1^{f(X)}\Delta \cdots$
confignumber
m_l
$Result_T$
$f(X)$

A slide from lecture 14

We must show that $f : \mathbb{N}^n \to \mathbb{N}$ defined by

$$f(X) = \mathit{Result}_T(f_T(\mathit{InitConfig}^{(n)}(X)))$$

is μ-recursive.

Theorem 10.20.

Every Turing computable partial function from $\mathbb{N}^n$ to $\mathbb{N}$ is μ-recursive.

The Rest of the Proof...

A slide from lecture 13

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 $\mathcal{M}$.

10.5. Other Approaches to Computability

Computer programs vs. Turing machines

Computer programs vs. μ-recursive functions

Let

- $G = (V, \Sigma, S, P)$ be unrestricted grammar
- f be partial function from Σ^* to Σ^*

Then G is said to compute f, if there are $A, B, C, D \in V$, such that for every x and y in Σ^*

$$f(x) = y \quad \text{if and only if} \quad AxB \Rightarrow^* CyD$$

This definition (and simple examples of it) must be known for the exam

Exercise.

Describe an unrestricted grammar that computes the function $f : \mathbb{N} \to \mathbb{N}$ defined by $f(n) = 2^n$.

Both the input n and the answer 2^n are unary numbers.

En verder. . .

Tentamen: vrijdag 5 juni 2015, 14:00–17:00

Vragenuur. . . ?