Advanced Topics in Theoretical Computer Science

Part 3: Recursive functions (2)

5.12.2013

Viorica Sofronie-Stokkermans

Universität Koblenz-Landau

e-mail: sofronie@uni-koblenz.de

Contents

- Recapitulation: Turing machines and Turing computability
- Register machines (LOOP, WHILE, GOTO)
- Recursive functions
- The Church-Turing Thesis
- Computability and (Un-)decidability
- Complexity
- Other computation models: e.g. Büchi Automata, λ-calculus

3. Recursive functions

- Introduction/Motivation
- Primitive recursive functions $\mapsto \mathcal{P}$
- $\mathcal{P} = \text{LOOP}$
- μ-recursive functions $\mapsto F_\mu$
- $F_\mu = \text{WHILE}$
- Summary

Until now

Primitive recursive functions

Primitive recursion. If the functions $g : \mathbb{N}^k \to \mathbb{N}$ and $h : \mathbb{N}^{k+2} \to \mathbb{N} (k \geq 0)$ are primitive recursive, then the following function is also primitive recursive:

$$f : \mathbb{N}^{k+1} \to \mathbb{N} \text{ with } \quad \begin{aligned} f(\mathbf{n}, 0) &= g(\mathbf{n}) \\ f(\mathbf{n}, m+1) &= h(\mathbf{n}, m, f(\mathbf{n}, m)) \end{aligned}$$

Notation without arguments: $f = \mathcal{PR}[g, h]$

Definition (Primitive recursive functions)

- **Atomic functions:** The functions null (0), successor (+1) and projection ($\pi_i^k \ (1 \leq i \leq k)$) are primitive recursive.
- **Composition:** The functions obtained by composition from primitive recursive functions are primitive recursive.
- **Primitive recursion:** The functions obtained by primitive recursion from primitive recursive functions are primitive recursive.

Examples of primitive recursive functions

$f(n) = n + c, \quad$ for $c \in \mathbb{N}, c \geq 0$

$(-1)(n) = n - 1$

$*(n, m) = n * m$

$+(n, m) = n + m$

$-(n, m) = n - m$

Lemma The set of primitive recursive functions is closed under re-ordering, omitting and repeating of arguments when composing functions.

Lemma. Assume $f : \mathbb{N}^k \to \mathbb{N}$ is primitive recursive.
Then, for every $l \in \mathbb{N}$, the function $f' : \mathbb{N}^k \times \mathbb{N}^l \to \mathbb{N}$ defined for every $\mathbf{n} \in \mathbb{N}^k$ and every $\mathbf{m} \in \mathbb{N}^l$ by:

$$f'(\mathbf{n}, \mathbf{m}) = f(\mathbf{n})$$

is primitive recursive.

Primitive recursive functions

Lemma (Case distinction). If g_i, h_i $(1 \leq i \leq r)$ are primitive recursive functions, and for every n there exists a unique i with $h_i(n) = 0$, then the function f defined by:

$$f(n) = \begin{cases} g_1(n) & \text{if } h_1(n) = 0 \\ \dots & \\ g_r(n) & \text{if } h_r(n) = 0 \end{cases}$$

is primitive recursive.

Theorem (Sums and products)

If $g : \mathbb{N}^k \times \mathbb{N} \to \mathbb{N}$ is a primitive recursive function then the following functions $f_1, f_2 : \mathbb{N}^k \times \mathbb{N} \to \mathbb{N}$ are also primitive recursive:

$$f_1(\mathbf{n}, m) = \begin{cases} 0 & \text{if } m = 0 \\ \sum_{i<m} g(\mathbf{n}, i) & \text{if } m > 0 \end{cases} \qquad f_2(\mathbf{n}, m) = \begin{cases} 1 & \text{if } m = 0 \\ \prod_{i<m} g(\mathbf{n}, i) & \text{if } m > 0 \end{cases}$$

Today

- More examples
- $\mathcal{P} = \mathsf{LOOP}$

Bounded μ operator

Definition.
Let $g : \mathbb{N}^{k+1} \to \mathbb{N}$ be a function.
The bounded μ operator is defined as follows:

$$\mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0) := \begin{cases} i_0 & \text{if } g(\mathbf{n}, i_0) = 0 \\ & \text{and for all } j < i_0 \;\; g(\mathbf{n}, j) \neq 0 \\ 0 & \text{if } g(\mathbf{n}, j) \neq 0 \text{ for all } 0 \leq j < m \\ & \text{or } m = 0 \end{cases}$$

$\mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0)$ is the smallest $i < m$ such that $g(\mathbf{n}, i) = 0$

Bounded μ operator

Theorem.

If $g : \mathbb{N}^{k+1} \to \mathbb{N}$ is a primitive recursive function
then the function $f : \mathbb{N}^{k+1} \to \mathbb{N}$ defined by:

$$f(\mathbf{n}, m) = \mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0)$$

is also primitive recursive

Bounded μ operator

Theorem.
If $g : \mathbb{N}^{k+1} \to \mathbb{N}$ is a primitive recursive function then the function $f : \mathbb{N}^{k+1} \to \mathbb{N}$ defined by:

$$f(\mathbf{n}, m) = \mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0)$$

is also primitive recursive

Proof: We can define f as follows:

$$\begin{aligned} f(\mathbf{n}, 0) &= 0 \\ f(\mathbf{n}, m+1) &= \begin{cases} 0 & \text{if } m = 0 \\ m & \text{if } g(\mathbf{n}, m) = f(\mathbf{n}, m) = 0 \wedge g(\mathbf{n}, 0) \neq 0 \wedge m > 0 \\ \\ f(\mathbf{n}, m) & \text{otherwise} \end{cases} \end{aligned}$$

Bounded μ operator

Theorem.
If $g : \mathbb{N}^{k+1} \to \mathbb{N}$ is a primitive recursive function then the function $f : \mathbb{N}^{k+1} \to \mathbb{N}$ defined by:

$$f(\mathbf{n}, m) = \mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0)$$

is also primitive recursive

Proof: We can define f as follows:

$$\begin{array}{rcl} f(\mathbf{n}, 0) & = & 0 \\ f(\mathbf{n}, m+1) & = & \begin{cases} 0 & \text{if } m = 0 \\ m & \text{if } g(\mathbf{n}, m) = f(\mathbf{n}, m) = 0 \wedge g(\mathbf{n}, 0) \neq 0 \wedge m > 0 \\ & \\ f(\mathbf{n}, m) & \text{otherwise} \end{cases} \end{array}$$

Bounded μ operator

Theorem.
If $g : \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ is a primitive recursive function
then the function $f : \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ defined by:

$$f(\mathbf{n}, m) = \mu_{i<m}\ i\ (g(\mathbf{n}, i) = 0)$$

is also primitive recursive

Proof: We can define f as follows:

$$\begin{aligned} f(\mathbf{n}, 0) &= 0 \\ f(\mathbf{n}, m+1) &= \begin{cases} 0 & \text{if } m = 0 \\ m & \text{if } g(\mathbf{n}, m) = f(\mathbf{n}, m) = 0 \wedge g(\mathbf{n}, 0) \neq 0 \wedge m > 0 \\ & \quad \text{i.e. if } g(\mathbf{n}, m) + f(\mathbf{n}, m) + (1 - g(\mathbf{n}, 0)) + (1 - m) = 0 \\ f(\mathbf{n}, m) & \text{otherwise} \end{cases} \end{aligned}$$

Prime number functions

Theorem: The following functions are primitive recursive:

(1) The Boolean function $| : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ defined by:

$$|(n, m) = \begin{cases} 1 & \text{if } n \text{ divides } m \\ 0 & \text{otherwise} \end{cases}$$

(2) The Boolean function $\mathrm{prime} : \mathbb{N} \to \{0, 1\}$ defined by:

$$\mathrm{prime}(n) = \begin{cases} 1 & \text{if } n \text{ prime} \\ 0 & \text{otherwise} \end{cases}$$

(3) The function $p : \mathbb{N} \to \mathbb{N}$ defined by: $p(n) = p_n$, the n-th prime number.

(4) The function $D : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ defined by: $D(n, i) = k$ iff k is the power of the i-th prime number in the prime number decomposition of n.

$$D(n, i) = \max(\{j \mid n \bmod p(i)^j = 0\})$$

Prime number functions

Proof:

(1) $\mid : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ defined by:

$$\mid(n, m) = \begin{cases} 1 & \text{if } n \text{ divides } m \\ 0 & \text{otherwise} \end{cases}$$

Prime number functions

Proof:

(1) $| : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ defined by:

$$|(n, m) = \begin{cases} 1 & \text{if } n \text{ divides } m \\ 0 & \text{otherwise} \end{cases}$$

$|(n, m) = 1$ iff $\exists z(n * z = m)$ iff $\prod_{z \leq m}(n * z - m) + (m - n * z) = 0$.

Prime number functions

Proof:

(1) $| : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ defined by:

$$|(n, m) = \begin{cases} 1 & \text{if } n \text{ divides } m \\ 0 & \text{otherwise} \end{cases}$$

$|(n, m) = 1$ iff $\exists z(n * z = m)$ iff $\prod_{z \leq m}(n * z - m) + (m - n * z) = 0$.

$|(n, m) = 1 - \prod_{z \leq m}(n * z - m) + (m - n * z)$

Prime number functions

Proof:

(2) prime : $\mathbb{N} \to \{0, 1\}$ defined by:

$$\text{prime}(n) = \begin{cases} 1 & \text{if } n \text{ prime} \\ 0 & \text{otherwise} \end{cases}$$

Prime number functions

Proof:

(2) prime : $\mathbb{N} \to \{0, 1\}$ defined by:

$$\text{prime}(n) = \begin{cases} 1 & \text{if } n \text{ prime} \\ 0 & \text{otherwise} \end{cases}$$

$\text{prime}(n) = 1$ iff $(n \geq 2$ and $\forall y < n(y = 0 \vee y = 1 \vee |(y, n) = 0)$

$\text{prime}(n) = 1 - ((2 - n) + \sum_{y<n}(|(y, n) * y * ((y - 1) + (1 - y))))$

Prime number functions

Proof:

(3) The function $p : \mathbb{N} \to \mathbb{N}$ defined by: $p(n) = p_n$, the n-th prime number.

$p(0) = 0$ and $p(1) = 2$.

$p(n+1)$ is the smallest number i which is larger than $p(n)$ and is prime.

Prime number functions

Proof:

(3) The function $p : \mathbb{N} \rightarrow \mathbb{N}$ defined by: $p(n) = p_n$, the n-th prime number.

$p(0) = 0$ and $p(1) = 2$.

$p(n+1)$ is the smallest number i which is larger than $p(n)$ and is prime.

We also have an upper bound for the number i.

Recall the proof of the fact that the set of prime numbers is infinite.

$$i \leq p(n)! + 1$$

$$p(n+1) = \mu_{i \leq p(n)!+1}\ i\ [((1 - \text{prime}(i)) + ((p(n)+1) - i)) = 0]$$

Prime number functions

Proof:

(4) $D : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ defined by: $D(n, i) = k$ iff k is the power of the i-th prime number in the prime number decomposition of n.

$$D(n, i) = \max(\{j \mid n \bmod p(i)^j = 0\})$$

$D(0, i) := 0;$

$D(n, i) = \min(\{j \leq n \mid \quad |(p(i)^{j+1}, n) = 0\})$

$D(n, i) = \mu_{j \leq n}\, j \; (|(p(i)^{j+1}, n) = 0)$

3. Recursive functions

- Introduction/Motivation
- Primitive recursive functions $\mapsto \mathcal{P}$
- $\mathcal{P} =$ LOOP
- μ-recursive functions $\mapsto F_\mu$
- $F_\mu =$ WHILE
- Summary

3. Recursive functions

- Introduction/Motivation
- Primitive recursive functions $\mapsto \mathcal{P}$
- $\mathcal{P} =$ LOOP
- μ-recursive functions $\mapsto F_\mu$
- $F_\mu =$ WHILE
- Summary

Goal

Show that $\mathcal{P} = \text{LOOP}$

Idea:

To show that $\mathcal{P} \supseteq \text{LOOP}$ we have to show that every LOOP computable function can be expressed as a primitive recursive function.

For this, we will encode the contents of arbitrarily many registers in one natural number (used as input for this primitive recursive function).

For this encoding we will use Gödelisation. We will need to show that Gödelisation is primitive recursive.

To show that $\mathcal{P} \subseteq \text{LOOP}$ we have to show that:
– all atomic primitive recursive functions are LOOP computable, and
– LOOP is closed under composition of functions and primitive recursion.

Gödelisation

To show: Gödelisation is primitive recursive

Informally:

- Coding number sequences as a number
- Corresponding decoding function (projection)

are primitive recursive

Gödelisation

To show: Gödelisierung is primitive recursive

Informally:

- Coding number sequences as a number
- Corresponding decoding function (projection)

are primitiv recursive

More precise formulation:

There exist primitive recursive functions

$$K^r : \mathbb{N}^r \to \mathbb{N} \qquad (r \geq 1)$$

$$D_i : \mathbb{N} \to \mathbb{N} \qquad (1 \leq i \leq r)$$

with:

$$D_i(K^r(n_1, \ldots, n_r)) = n_i$$

Gödelisation

To show: Gödelisation is primitive recursive

Informally:

- Coding number sequences as a number
- Corresponding decoding function (projection)

are primitive recursive

Recall:

Gödelisation: Coding number sequences as a number

Bijection between $\mathbb{N}^r$ and $\mathbb{N}$: $K^r : \mathbb{N}^r \to \mathbb{N}$, defined by:

$$K^r(n_1, \dots, n_r) = \prod_{i=1}^{r} p(i)^{n_i}.$$

Decoding: The inverses $D_i : \mathbb{N} \to \mathbb{N}$ of K^r defined by $D_i(n) = D(n, i)$

Gödelisation

Bijection between $\mathbb{N}^r$ and $\mathbb{N}$: $K^r : \mathbb{N}^r \to \mathbb{N}$, defined by:

$$K^r(n_1, \ldots, n_r) = \prod_{i=1}^{r} p(i)^{n_i}.$$

$D_i : \mathbb{N} \to \mathbb{N}$, $1 \leq i \leq r$, defined by $D_i(n) = D(n, i)$

Theorem. K^r and $D_1, \ldots, D_r$ are primitive recursive.

Gödelisation

Bijection between $\mathbb{N}^r$ and $\mathbb{N}$: $K^r : \mathbb{N}^r \to \mathbb{N}$, defined by:

$$K^r(n_1, \ldots, n_r) = \prod_{i=1}^{r} p(i)^{n_i}.$$

$D_i : \mathbb{N} \to \mathbb{N}$, $1 \leq i \leq r$, defined by $D_i(n) = D(n, i)$

Theorem. K^r and $D_1, \ldots, D_r$ are primitive recursive.

Lemma.
(1) $D_i(K^r(n_1, \ldots, n_r)) = n_i \quad$ for all $1 \leq i \leq r$.
(2) $K^r(n_1, \ldots, n_r) = K^{r+1}(n_1, \ldots, n_r, 0)$

In general, $D_i(K^r(n_1, \ldots, n_r)) = 0$ if $i > r$.

Gödelisation

Notation:

$$K^r(n_1, \ldots, n_r) = \langle n_1, \ldots, n_r \rangle$$
$$D_i(n) = (n)_i$$

For $r = 0$:

$\langle\rangle = 1$

$(\langle\rangle)_i = 0$

Gödelisation: Applications

Theorem (Simultaneous Recursion)

If

$$
\begin{aligned}
f_1(\mathbf{n}, 0) &= g_1(\mathbf{n}) \\
&\cdots \\
f_r(\mathbf{n}, 0) &= g_r(\mathbf{n}) \\
f_1(\mathbf{n}, m+1) &= h_1(\mathbf{n}, m, f_1(\mathbf{n}, m), \ldots, f_r(\mathbf{n}, m)) \\
&\cdots \\
f_r(\mathbf{n}, m+1) &= h_r(\mathbf{n}, m, f_1(\mathbf{n}, m), \ldots, f_r(\mathbf{n}, m))
\end{aligned}
$$

and if $g_1, \ldots, g_r, h_1, \ldots, h_r$ are primitive recursive

then $f_1, \ldots, f_r$ are primitive recursive.

Example

Let f_1 and f_2 be defined by simultaneous recursion as follows:

$$
\begin{aligned}
f_1(0) &= 0 \\
f_2(0) &= 1 \\[1em]
f_1(n+1) &= f_2(n) \\
f_2(n+1) &= f_1(n) + f_2(n)
\end{aligned}
$$

Example

Let f_1 and f_2 be defined by simultaneous recursion as follows:

$$
\begin{aligned}
f_1(0) &= 0 & g_1 &= 0 \\
f_2(0) &= 1 & g_2 &= 1
\end{aligned}
$$

$$
\begin{aligned}
f_1(n+1) &= f_2(n) & h_1(n, f_1(n), f_2(n)) &= f_2(n) & h_1 &= \pi^3_3 \\
f_2(n+1) &= f_1(n) + f_2(n) & h_2(n, f_1(n), f_2(n)) &= f_1(n) + f_2(n) & h_2 &= + \circ (\pi^3_2, \pi^3_3)
\end{aligned}
$$