universität innsbruck

WS 2023 lecture 1

Computability Theory

Aart Middeldorp

Initial Remarks

- Computability Theory is part of WM 9 in master program Computer Science

Initial Remarks

- Computability Theory is part of WM 9 in master program Computer Science
- WM 9 is part of Logic and Learning specialization

Initial Remarks

- Computability Theory is part of WM 9 in master program Computer Science
- WM 9 is part of Logic and Learning specialization
- other courses in Logic and Learning specialization:
  - Machine Learning for Theorem Proving LVA 703819
  - Program and Resource Analysis LVA 703316

Outline

Organisation

- LVA 703317

Organisation

- LVA 703317
- VU 3 – 5 ECTS

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15 – 18:00 in HS 10

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15 – 18:00 in HS 10
- http://cl-informatik.uibk.ac.at/teaching/ws23/ct

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15–18:00 in HS 10
- `http://cl-informatik.uibk.ac.at/teaching/ws23/ct`
- OLAT

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15 – 18:00 in HS 10
- `http://cl-informatik.uibk.ac.at/teaching/ws23/ct`
- OLAT
- consultation hours: Thursday 9:30 – 11:00 3M07 and online

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15–18:00 in HS 10
- `http://cl-informatik.uibk.ac.at/teaching/ws23/ct`
- OLAT
- consultation hours: Thursday 9:30–11:00 3M07 and online

Schedule

lecture 1	October 2	lecture 6	November 6	lecture 11	December 11
lecture 2	October 9	lecture 7	November 13	lecture 12	January 8
lecture 3	October 16	lecture 8	November 20	lecture 13	January 15
lecture 4	October 23	lecture 9	November 27	lecture 14	January 22
lecture 5	October 30	lecture 10	December 4	lecture 15	January 29

Organisation

- LVA 703317
- VU 3 – 5 ECTS
- 15:15–18:00 in HS 10
- `http://cl-informatik.uibk.ac.at/teaching/ws23/ct`
- OLAT
- consultation hours: Thursday 9:30–11:00 3M07 and online

Schedule

lecture 1	October 2	lecture 6	November 6	lecture 11	December 11
lecture 2	October 9	lecture 7	November 13	lecture 12	January 8
lecture 3	October 16	lecture 8	November 20	lecture 13	January 15
lecture 4	October 23	lecture 9	November 27	lecture 14	January 22
lecture 5	October 30	lecture 10	December 4	lecture 15	January 29 (test)

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E + P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

Grading

$$\text{score} \;=\; \min\left(\max\left(\tfrac{2}{3}(E + P) + \tfrac{1}{3}T + B,\; T + B\right),\; 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

Grading

$$\text{score} \;=\; \min\left(\max\left(\tfrac{2}{3}(E+P)+\tfrac{1}{3}T+B,\; T+B\right),\; 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E + P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

- E: points for solved exercises (at most 91)
- P: points for presentation of solutions (at most 9)
- T: points for test (at most 100)
- B: points for bonus exercises (at most 15)

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T+B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

B: points for bonus exercises (at most 15)

$$\text{grade} = \text{score} \in (-50) \to 5 \quad [50-63) \to 4 \quad [63-75) \to 3 \quad [75-88) \to 2 \quad [88-) \to 1$$

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T+B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

B: points for bonus exercises (at most 15)

$$\text{grade} = \text{score} \in (-50) \to 5 \quad [50-63) \to 4 \quad [63-75) \to 3 \quad [75-88) \to 2 \quad [88-) \to 1$$

- solved exercises must be marked and solutions (PDF format) must be uploaded in OLAT before 10 am on Monday

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T+B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

B: points for bonus exercises (at most 15)

$$\text{grade} = \text{score} \in \ (-50) \to 5 \quad [50-63) \to 4 \quad [63-75) \to 3 \quad [75-88) \to 2 \quad [88-) \to 1$$

- solved exercises must be marked and solutions (PDF format) must be uploaded in OLAT before 10 am on Monday
- presentations are optional

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T+B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

B: points for bonus exercises (at most 15)

$$\text{grade} = \text{score} \in (-50) \to 5 \quad [50-63) \to 4 \quad [63-75) \to 3 \quad [75-88) \to 2 \quad [88-) \to 1$$

- solved exercises must be marked and solutions (PDF format) must be uploaded in OLAT before 10 am on Monday
- presentations are optional
- (optional) test on January 29

Grading

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

E: points for solved exercises (at most 91)

P: points for presentation of solutions (at most 9)

T: points for test (at most 100)

B: points for bonus exercises (at most 15)

$$\text{grade} = \text{score} \in (-50) \to 5 \quad [50-63) \to 4 \quad [63-75) \to 3 \quad [75-88) \to 2 \quad [88-) \to 1$$

- solved exercises must be marked and solutions (PDF format) must be uploaded in OLAT before 10 am on Monday
- presentations are optional
- (optional) test on January 29

evaluation SS 2022

Outline

Theorem

- Turing machines

Theorem

- lambda calculus
- Turing machines

Theorem

- combinatory logic
- lambda calculus
- Turing machines
- term rewrite systems

Theorem

- combinatory logic
- lambda calculus
- Turing machines
- C / Haskell / Java / Python / . . . programs
- term rewrite systems

Theorem

- combinatory logic
- enumeration machines
- interaction nets
- lambda calculus
- Turing machines
- C / Haskell / Java / Python / . . . programs
- recursive functions
- term rewrite systems
- two-counter automata
- quantum computers

capture the same notion of computation

Theorem

- combinatory logic
- enumeration machines
- interaction nets
- lambda calculus
- Turing machines
- C / Haskell / Java / Python / . . . programs
- recursive functions
- term rewrite systems
- two-counter automata
- quantum computers

capture the same notion of computation

Literature (Recursive Function Theory)

- Nigel Cutland
  Computability: An Introduction to Recursive Function Theory
  Cambridge University Press, 1980
- Richard Epstein and Walter Carnielli
  Computability: Computable Functions, Logic, and the Foundations of Mathematics (3rd edition)
  Advanced Reasoning Forum, 2008
- Piergiorgio Odifreddi
  Classical Recursion Theory (2nd edition)
  North Holland, 1992
- Hartley Rogers Jr.
  Theory of Recursive Functions and Effective Computability
  MIT Press, 1987
- Rózsa Péter
  Rekursive Funktionen in der Komputer-Theorie
  Akadémiai Kiadó, 1976

Literature (Combinatory Logic and Lambda Calculus)

- Katalin Bimbó
  Combinatory Logic: Pure, Applied and Typed
  CRC Press, 2011
- Henk Barendregt
  The Lambda Calculus, Its Syntax and Semantics
  North Holland, 1984
- Herman Geuvers and Rob Nederpelt
  Type Theory and Formal Proof
  Cambridge University Press, 2014
- Chris Hankin
  An Introduction to Lambda Calculi for Computer Scientists
  King's College Publications, 2000
- J. Roger Hindley and Jonathan P. Seldin
  Lambda-Calculus and Combinators, an Introduction
  Cambridge University Press, 2008

Part I: Recursive Function Theory

Ackermann function, bounded minimization, bounded recursion, course−of−values recursion, diagonalization, diophantine sets, elementary functions, fixed point theorem, Fibonacci numbers, Gödel numbering, Gödel's β function, Grzegorczyk hierarchy, loop programs, minimization, normal form theorem, partial recursive functions, primitive recursion, recursive enumerability, recursive inseparability, s−m−n theorem, total recursive functions, undecidability, while programs, ...

Part II: Combinatory Logic and Lambda Calculus

α−equivalence, abstraction, arithmetization, β−reduction, CL−representability, combinators, combinatorial completeness, Church numerals, Church−Rosser theorem, Curry−Howard isomorphism, de Bruijn notation, η−reduction, fixed point theorem, intuitionistic propositional logic, λ−definability, normalization theorem, termination, typing, undecidability, Z property, ...

Part I: Recursive Function Theory

Ackermann function, bounded minimization, bounded recursion, course−of−values recursion, diagonalization, diophantine sets, elementary functions, fixed point theorem, Fibonacci numbers, Gödel numbering, Gödel's β function, Grzegorczyk hierarchy, loop programs, minimization, normal form theorem, partial recursive functions, primitive recursion, recursive enumerability, recursive inseparability, s−m−n theorem, total recursive functions, undecidability, while programs, . . .

Part II: Combinatory Logic and Lambda Calculus

α−equivalence, abstraction, arithmetization, β−reduction, CL−representability, combinators, combinatorial completeness, Church numerals, Church−Rosser theorem, Curry−Howard isomorphism, de Bruijn notation, η−reduction, fixed point theorem, intuitionistic propositional logic, λ−definability, normalization theorem, termination, typing, undecidability, Z property, . . .

Outline

Examples

- which function is computable ?

$$f(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of exactly } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of at least } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

Examples

- which function is computable ?

$$f(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of exactly } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of at least } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

- which are (total) functions ?

$$f(x) = f(x) + 1$$

Examples

- which function is computable ?

$$f(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of exactly } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of at least } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

- which are (total) functions ?

$$f(x) = f(x) + 1$$

$$g(x) = \begin{cases} 0 & \text{if } x = 0 \\ g(g(x-1)) & \text{if } x > 0 \end{cases}$$

Examples

- which function is computable ?

$$f(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of exactly } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of at least } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

- which are (total) functions ?

$$f(x) = f(x) + 1$$

$$g(x) = \begin{cases} 0 & \text{if } x = 0 \\ g(g(x-1)) & \text{if } x > 0 \end{cases}$$

$$h(x) = 0 \times f(x)$$

Examples

- which function is computable ?

$$f(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of exactly } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} 1 & \text{if decimal expansion of } \pi \text{ contains consecutive run of at least } x \text{ fives} \\ 0 & \text{otherwise} \end{cases}$$

- which are (total) functions ?

$$f(x) = f(x) + 1$$

$$g(x) = \begin{cases} 0 & \text{if } x = 0 \\ g(g(x-1)) & \text{if } x > 0 \end{cases}$$

$$i(x) = \begin{cases} 1 & \text{if } x = 0 \text{ or } x = 1 \\ i(x/2) & \text{if } x > 1 \text{ is even} \\ i(3x+1) & \text{if } x > 1 \text{ is odd} \end{cases}$$

$$h(x) = 0 \times f(x)$$

Definition

class PR of primitive recursive functions is smallest class of total functions $f\colon \mathbb{N}^n \to \mathbb{N}$

Definition

class PR of primitive recursive functions is smallest class of total functions $f : \mathbb{N}^n \to \mathbb{N}$ that contains all initial functions

- zero $\quad z(x) = 0$

Definition

class PR of primitive recursive functions is smallest class of total functions $f\colon \mathbb{N}^n \to \mathbb{N}$ that contains all initial functions

- zero $\mathrm{z}(x) = 0$
- successor $\mathrm{s}(x) = x + 1$

Definition

class PR of primitive recursive functions is smallest class of total functions $f\colon \mathbb{N}^n \to \mathbb{N}$ that contains all initial functions

- zero $\quad \mathrm{z}(x) = 0$
- successor $\quad \mathrm{s}(x) = x + 1$
- projection $\quad \pi_i^n(x_1, \dots, x_n) = x_i \quad$ for all $n \geqslant 1$ and $1 \leqslant i \leqslant n$

Definition

class PR of primitive recursive functions is smallest class of total functions $f\colon \mathbb{N}^n \to \mathbb{N}$ that contains all initial functions

- zero $\quad \mathrm{z}(x) = 0$
- successor $\quad \mathrm{s}(x) = x + 1$
- projection $\quad \pi_i^n(x_1, \dots, x_n) = x_i \quad$ for all $n \geqslant 1$ and $1 \leqslant i \leqslant n$

and is closed under composition

- $f(\vec{x}) = g(h_1(\vec{x}), \dots, h_m(\vec{x})) \in \mathrm{PR} \quad$ for all $g\colon \mathbb{N}^m \to \mathbb{N} \in \mathrm{PR}$ and $h_1, \dots, h_m\colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$

Definition

class PR of primitive recursive functions is smallest class of total functions $f\colon \mathbb{N}^n \to \mathbb{N}$ that contains all initial functions

- zero $\mathrm{z}(x) = 0$
- successor $\mathrm{s}(x) = x + 1$
- projection $\pi_i^n(x_1, \dots, x_n) = x_i$ for all $n \geqslant 1$ and $1 \leqslant i \leqslant n$

and is closed under composition

- $f(\vec{x}) = g(h_1(\vec{x}), \dots, h_m(\vec{x})) \in \mathrm{PR}$ for all $g\colon \mathbb{N}^m \to \mathbb{N} \in \mathrm{PR}$ and $h_1, \dots, h_m\colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$

and primitive recursion

- $f(x, \vec{y})\colon \mathbb{N}^{n+1} \to \mathbb{N}$ defined by

$$f(0, \vec{y}) = g(\vec{y})$$
$$f(x+1, \vec{y}) = h(f(x, \vec{y}), x, \vec{y})$$

belongs to PR for all $g\colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $h\colon \mathbb{N}^{n+2} \to \mathbb{N} \in \mathrm{PR}$

Examples

- addition $x + y = f(x, y) \in \mathrm{PR}$

$$f(0, y) = g(y)$$
$$f(x + 1, y) = h(f(x, y), x, y)$$

Examples

- addition $\quad x + y = f(x, y) \in \mathrm{PR}$

$$f(0, y) = g(y)$$
$$f(x+1, y) = h(f(x, y), x, y)$$

with $g(x) = \pi_1^1(x)$ and $h(x, y, z) = \mathrm{s}(\pi_1^3(x, y, z))$

Examples

- addition $\quad x + y = f(x, y) \in \mathsf{PR}$

$$f(0, y) = g(y)$$
$$f(x+1, y) = h(f(x, y), x, y)$$

with $g(x) = \pi_1^1(x)$ and $h(x, y, z) = \mathsf{s}(\pi_1^3(x, y, z))$

- multiplication $\quad x \times y = f(x, y) \in \mathsf{PR}$

$$f(0, y) = g(y)$$
$$f(x+1, y) = h(f(x, y), x, y)$$

Examples

- addition $\quad x + y = f(x,y) \in \mathsf{PR}$

$$f(0,y) = g(y)$$
$$f(x+1,y) = h(f(x,y),x,y)$$

with $g(x) = \pi^1_1(x)$ and $h(x,y,z) = \mathsf{s}(\pi^3_1(x,y,z))$

- multiplication $\quad x \times y = f(x,y) \in \mathsf{PR}$

$$f(0,y) = g(y)$$
$$f(x+1,y) = h(f(x,y),x,y)$$

with $g(x) = \mathsf{z}(x)$ and $h(x,y,z) = \pi^3_1(x,y,z) + \pi^3_3(x,y,z)$

Examples

- addition $x + y = f(x, y) \in \mathrm{PR}$

$$f(0, y) = g(y)$$
$$f(x + 1, y) = h(f(x, y), x, y)$$

with $g(x) = \pi^1_1(x)$ and $h(x, y, z) = \mathrm{s}(\pi^3_1(x, y, z))$

- multiplication $x \times y = f(x, y) \in \mathrm{PR}$

$$f(0, y) = g(y)$$
$$f(x + 1, y) = h(f(x, y), x, y)$$

with $g(x) = \mathrm{z}(x)$ and $h(x, y, z) = \pi^3_1(x, y, z) + \pi^3_3(x, y, z)$

- exponentiation $x^y = f(y, x) \in \mathrm{PR}$

$$f(0, y) = \mathrm{s}(\mathrm{z}(y))$$
$$f(x + 1, y) = \pi^3_1(f(x, y), x, y) \times \pi^3_3(f(x, y), x, y)$$

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathrm{PR}$ if $f \colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \ldots, x_m\}$ for all $1 \leqslant i \leqslant n$

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathrm{PR}$ if $f \colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \ldots, x_m\}$ for all $1 \leqslant i \leqslant n$

Proof

- $y_i = x_j \quad \Longrightarrow \quad y_i = \pi^m_j(x_1, \ldots, x_m)$

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathrm{PR}$ if $f\colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \ldots, x_m\}$ for all $1 \leqslant i \leqslant n$

Proof

- $y_i = x_j \implies y_i = \pi^m_j(x_1, \ldots, x_m)$
- hence g can be defined by composing f with projection functions

Lemma

$g(x_1, \dots, x_m) = f(y_1, \dots, y_n) \in \mathrm{PR}$ if $f \colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \dots, x_m\}$ for all $1 \leqslant i \leqslant n$

Proof

- $y_i = x_j \quad \Longrightarrow \quad y_i = \pi_j^m(x_1, \dots, x_m)$
- hence g can be defined by composing f with projection functions

Example

- cut-off subtraction (monus)

$$x \dot{-} y = \begin{cases} x - y & \text{if } x \geqslant y \\ 0 & \text{otherwise} \end{cases}$$

is primitive recursive

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathrm{PR}$ if $f\colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \ldots, x_m\}$ for all $1 \leqslant i \leqslant n$

Proof

- $y_i = x_j \quad \Longrightarrow \quad y_i = \pi_j^m(x_1, \ldots, x_m)$
- hence g can be defined by composing f with projection functions

Example

- cut-off subtraction (monus)

$$x \dot{-} y = \begin{cases} x - y & \text{if } x \geqslant y \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} x \dot{-} 0 &= x \\ x \dot{-} (y+1) &= \mathrm{p}(x \dot{-} y) \end{aligned}$$

is primitive recursive

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathrm{PR}$ if $f \colon \mathbb{N}^n \to \mathbb{N} \in \mathrm{PR}$ and $y_i \in \{x_1, \ldots, x_m\}$ for all $1 \leqslant i \leqslant n$

Proof

- $y_i = x_j \implies y_i = \pi_j^m(x_1, \ldots, x_m)$
- hence g can be defined by composing f with projection functions

Example

- cut-off subtraction (monus)

$$x \mathbin{\dot{-}} y = \begin{cases} x - y & \text{if } x \geqslant y \\ 0 & \text{otherwise} \end{cases}$$

$$x \mathbin{\dot{-}} 0 = x$$
$$x \mathbin{\dot{-}} (y+1) = \mathrm{p}(x \mathbin{\dot{-}} y)$$

is primitive recursive

Examples

- predecessor $\quad \mathrm{p}(x) = f(x) \in \mathrm{PR}$

$$\begin{aligned} f(0) &= 0 \\ f(x+1) &= x = \pi^2_2(f(x), x) \end{aligned}$$

Examples

- predecessor $\quad \mathrm{p}(x) = f(x) \in \mathrm{PR}$

$$f(0) = 0$$
$$f(x+1) = x = \pi^2_2(f(x), x)$$

- factorial $\quad x! = f(x) \in \mathrm{PR}$

$$f(0) = 1$$
$$f(x+1) = \mathrm{s}(x) \times f(x)$$

Examples

- predecessor $\mathrm{p}(x) = f(x) \in \mathrm{PR}$

$$\begin{aligned} f(0) &= 0 \\ f(x+1) &= x = \pi^2_2(f(x), x) \end{aligned}$$

- factorial $x! = f(x) \in \mathrm{PR}$

$$\begin{aligned} f(0) &= 1 \\ f(x+1) &= \mathrm{s}(x) \times f(x) \end{aligned}$$

- summation $\sum_{i=1}^{x} i = f(x) \in \mathrm{PR}$

$$\begin{aligned} f(0) &= 0 \\ f(x+1) &= \mathrm{s}(x) + f(x) \end{aligned}$$

Examples

- predecessor $\quad \mathrm{p}(x) = f(x) \in \mathrm{PR}$

$$f(0) = 0 \; ?$$
$$f(x+1) = x = \pi^2_2(f(x), x)$$

- factorial $\quad x! = f(x) \in \mathrm{PR}$

$$f(0) = 1 \; ?$$
$$f(x+1) = \mathrm{s}(x) \times f(x)$$

- summation $\quad \sum_{i=1}^{x} i = f(x) \in \mathrm{PR}$

$$f(0) = 0 \; ?$$
$$f(x+1) = \mathrm{s}(x) + f(x)$$

Definition

function $f\colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g\colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x)\cdots)$$

Definition

function $f \colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g \colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x) \cdots)$$

Lemma

PR is closed under iteration

Definition

function $f\colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g\colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x)\cdots)$$

Lemma

PR is closed under iteration

Proof

- suppose $g\colon \mathbb{N} \to \mathbb{N} \in \mathrm{PR}$

Definition

function $f\colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g\colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x)\cdots)$$

Lemma

PR is closed under iteration

Proof

- suppose $g\colon \mathbb{N} \to \mathbb{N} \in \mathrm{PR}$
- $f(n,x) = g^{(n)}(x)$ can be defined by primitive recursion:

$$f(0,x) = x$$
$$f(n+1,x) = g(f(n,x))$$

Definition

function $f\colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g\colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x)\cdots)$$

Lemma

PR is closed under iteration

Proof

- suppose $g\colon \mathbb{N} \to \mathbb{N} \in$ PR
- $f(n,x) = g^{(n)}(x)$ can be defined by primitive recursion:

$$f(0,x) = x = \pi_1^1(x)$$
$$f(n+1,x) = g(f(n,x))$$

Definition

function $f\colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g\colon \mathbb{N} \to \mathbb{N}$ by iteration if

$$f(n,x) = g^{(n)}(x) = \underbrace{g(\cdots g}_{n \text{ times}}(x)\cdots)$$

Lemma

PR is closed under iteration

Proof

- suppose $g\colon \mathbb{N} \to \mathbb{N} \in \mathrm{PR}$
- $f(n,x) = g^{(n)}(x)$ can be defined by primitive recursion:

$$f(0,x) = x = \pi_1^1(x)$$
$$f(n+1,x) = g(f(n,x)) = h(f(n,x),n,x)$$

with $h(x,y,z) = g(\pi_1^3(x,y,z))$

Outline

Example

- $\max(x, y) = \begin{cases} x & \text{if } x \geqslant y \\ y & \text{otherwise} \end{cases}$ is primitive recursive

Example

- $\max(x, y) = \begin{cases} x & \text{if } x \geqslant y \\ y & \text{otherwise} \end{cases}$ is primitive recursive ?

Example

- $\max(x,y) = \begin{cases} x & \text{if } x \geqslant y \\ y & \text{otherwise} \end{cases}$ is primitive recursive ?

Definition

predicate $P\colon \mathbb{N}^n \to \mathbb{B}$ is primitive recursive if its characteristic function $\chi_P\colon \mathbb{N}^n \to \mathbb{N}$

$$\chi_P(x_1,\dots,x_n) = \begin{cases} 1 & \text{if } P(x_1,\dots,x_n) \\ 0 & \text{otherwise} \end{cases}$$

is primitive recursive

Example

- $\max(x,y) = \begin{cases} x & \text{if } x \geqslant y \\ y & \text{otherwise} \end{cases}$ is primitive recursive ?

Definition

predicate $P\colon \mathbb{N}^n \to \mathbb{B}$ is primitive recursive if its characteristic function $\chi_P\colon \mathbb{N}^n \to \mathbb{N}$

$$\chi_P(x_1,\dots,x_n) = \begin{cases} 1 & \text{if } P(x_1,\dots,x_n) \\ 0 & \text{otherwise} \end{cases}$$

is primitive recursive

Lemma

if $P, Q\colon \mathbb{N}^n \to \mathbb{B}$ are primitive recursive predicates then so are

$\neg P \qquad P \wedge Q \qquad P \vee Q \qquad P \Rightarrow Q$

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x})$$

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Examples

- sign function $\mathrm{sg}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$ is primitive recursive

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Examples

- sign function $\mathrm{sg}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$ is primitive recursive

$$\begin{aligned} \mathrm{sg}(0) &= 0 \\ \mathrm{sg}(x+1) &= 1 \end{aligned}$$

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Examples

- sign function $\mathrm{sg}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$ is primitive recursive
$$\begin{aligned} \mathrm{sg}(0) &= 0 \\ \mathrm{sg}(x+1) &= 1 \end{aligned}$$
- $>$ and $\neq$ are primitive recursive predicates

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Examples

- sign function $\mathrm{sg}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$ is primitive recursive
$$\begin{aligned} \mathrm{sg}(0) &= 0 \\ \mathrm{sg}(x+1) &= 1 \end{aligned}$$
- $>$ and $\neq$ are primitive recursive predicates

$$\chi_>(x,y) = \mathrm{sg}(x \dot{-} y) \qquad \chi_{\neq}(x,y) = \mathrm{sg}((x \dot{-} y) + (y \dot{-} x))$$

Proof

$$\chi_{\neg P}(\vec{x}) = 1 \dot{-} \chi_P(\vec{x}) \qquad \chi_{P \wedge Q}(\vec{x}) = \chi_P(\vec{x}) \times \chi_Q(\vec{x})$$

Examples

- sign function $\text{sg}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$ is primitive recursive $\quad \begin{aligned} \text{sg}(0) &= 0 \\ \text{sg}(x+1) &= 1 \end{aligned}$
- $>$ and $\neq$ are primitive recursive predicates

$$\chi_>(x,y) = \text{sg}(x \dot{-} y) \qquad \chi_{\neq}(x,y) = \text{sg}((x \dot{-} y) + (y \dot{-} x))$$

- $=$, $\geqslant$, $<$ and $\leqslant$ are primitive recursive predicates

$$\begin{aligned} x = y &\iff \neg(x \neq y) & \qquad x < y &\iff y > x \\ x \geqslant y &\iff x > y \vee x = y & \qquad x \leqslant y &\iff y \geqslant x \end{aligned}$$

Lemma (case analysis)

if $f_1, \dots, f_k \colon \mathbb{N}^n \to \mathbb{N}$ and $P_1, \dots, P_k \colon \mathbb{N}^n \to \mathbb{B}$ are primitive recursive such that for all $\vec{x} \in \mathbb{N}^n$ exactly one of $P_1(\vec{x}) \cdots P_k(\vec{x})$ holds then

$$g(\vec{x}) = \begin{cases} f_1(\vec{x}) & \text{if } P_1(\vec{x}) \\ \cdots & \cdots \\ f_k(\vec{x}) & \text{if } P_k(\vec{x}) \end{cases}$$

is primitive recursive

Lemma (case analysis)

if $f_1, \dots, f_k \colon \mathbb{N}^n \to \mathbb{N}$ and $P_1, \dots, P_k \colon \mathbb{N}^n \to \mathbb{B}$ are primitive recursive such that for all $\vec{x} \in \mathbb{N}^n$ exactly one of $P_1(\vec{x}) \cdots P_k(\vec{x})$ holds then

$$g(\vec{x}) = \begin{cases} f_1(\vec{x}) & \text{if } P_1(\vec{x}) \\ \cdots & \cdots \\ f_k(\vec{x}) & \text{if } P_k(\vec{x}) \end{cases}$$

is primitive recursive

Proof

$$g(\vec{x}) = f_1(\vec{x}) \times \chi_{P_1}(\vec{x}) + \cdots + f_k(\vec{x}) \times \chi_{P_k}(\vec{x})$$

Lemma (case analysis)

if $f_1, \ldots, f_k \colon \mathbb{N}^n \to \mathbb{N}$ and $P_1, \ldots, P_k \colon \mathbb{N}^n \to \mathbb{B}$ are primitive recursive such that for all $\vec{x} \in \mathbb{N}^n$ exactly one of $P_1(\vec{x}) \cdots P_k(\vec{x})$ holds then

$$g(\vec{x}) = \begin{cases} f_1(\vec{x}) & \text{if } P_1(\vec{x}) \\ \cdots & \cdots \\ f_k(\vec{x}) & \text{if } P_k(\vec{x}) \end{cases}$$

is primitive recursive

Proof

$$g(\vec{x}) = f_1(\vec{x}) \times \chi_{P_1}(\vec{x}) + \cdots + f_k(\vec{x}) \times \chi_{P_k}(\vec{x})$$

Example

- $\max(x, y) = \begin{cases} x & \text{if } x \geqslant y \\ y & \text{if } x < y \end{cases}$ is primitive recursive

Example

how to implement

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T+B\right),\ 100\right)$$

in OLAT ?

Example

how to implement

$$\text{score} \;=\; \min\left(\max\left(\tfrac{2}{3}(E + P) + \tfrac{1}{3}T + B,\; T + B\right),\; 100\right)$$

in OLAT ?

- `E = getScore("108480307718721")`

Example

how to implement

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E + P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

in OLAT ?

- $\mathtt{E} = \mathtt{getScore("108480307718721")}$
- $\mathtt{P} = \mathtt{getScore("108480307713191")}$
- $\mathtt{T} = \mathtt{getScore("108480307740761")}$
- $\mathtt{B} = \mathtt{getScore("108480307725070")}$

Example

how to implement

$$\text{score} = \min\left(\max\left(\tfrac{2}{3}(E+P) + \tfrac{1}{3}T + B,\ T + B\right),\ 100\right)$$

in OLAT ?

- `E = getScore("108480307718721")`
- `P = getScore("108480307713191")`
- `T = getScore("108480307740761")`
- `B = getScore("108480307725070")`

Example

how to implement

$$\text{score} = \min\left(\max\left(\frac{2}{3}(E+P) + \frac{1}{3}T + B,\ T+B\right),\ 100\right)$$

in OLAT ?

- E = getScore("108480307718721")
- P = getScore("108480307713191")
- T = getScore("108480307740761")
- B = getScore("108480307725070")

```
((((2*(E + P)/3 + T/3 + B))) >= (T + B)) * (((2*(E + P)/3 + T/3 + B))) +
((((2*(E + P)/3 + T/3 + B))) <  (T + B)) * (T + B)
```

Lemma (bounded sum)

if $f : \mathbb{N}^{n+1} \to \mathbb{N}$ is primitive recursive then

$$\sum_{i=0}^{x} f(i, \vec{y})$$

is primitive recursive

Lemma (bounded sum)

if $f\colon \mathbb{N}^{n+1} \to \mathbb{N}$ is primitive recursive then

$$\sum_{i=0}^{x} f(i,\vec{y})$$

is primitive recursive

Proof

$$g(x,\vec{y}) = \sum_{i=0}^{x} f(i,\vec{y})$$

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

Lemma (bounded sum and product)

if $f\colon \mathbb{N}^{n+1} \to \mathbb{N}$ is primitive recursive then

$$\sum_{i=0}^{x} f(i,\vec{y}) \qquad \text{and} \qquad \prod_{i=0}^{x} f(i,\vec{y})$$

are primitive recursive

Proof

$$g(x,\vec{y}) = \sum_{i=0}^{x} f(i,\vec{y}) \qquad \begin{aligned} g(0,\vec{y}) &= f(0,\vec{y}) \\ g(x+1,\vec{y}) &= f(x+1,\vec{y}) + g(x,\vec{y}) \end{aligned}$$

$$h(x,\vec{y}) = \prod_{i=0}^{x} f(i,\vec{y}) \qquad \begin{aligned} h(0,\vec{y}) &= f(0,\vec{y}) \\ h(x+1,\vec{y}) &= f(x+1,\vec{y}) \times h(x,\vec{y}) \end{aligned}$$

Lemma (bounded quantification)

if $P \colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\forall\, i \leqslant x)\, P(i,\vec{y}) \qquad \text{and} \qquad (\exists\, i \leqslant x)\, P(i,\vec{y})$$

are primitive recursive

Lemma (bounded quantification)

if $P\colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\forall\, i \leqslant x)\; P(i,\vec{y}) \qquad \text{and} \qquad (\exists\, i \leqslant x)\; P(i,\vec{y})$$

are primitive recursive

Proof

$$Q(x,\vec{y}) = (\forall\, i \leqslant x)\; P(i,\vec{y}) \qquad\qquad \chi_Q(x,\vec{y}) = \prod_{i \leqslant x} \chi_P(i,\vec{y})$$

Lemma (bounded quantification)

if $P \colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\forall\, i \leqslant x)\; P(i,\vec{y}) \qquad \text{and} \qquad (\exists\, i \leqslant x)\; P(i,\vec{y})$$

are primitive recursive

Proof

$$Q(x,\vec{y}) = (\forall\, i \leqslant x)\; P(i,\vec{y}) \qquad \chi_Q(x,\vec{y}) = \prod_{i \leqslant x} \chi_P(i,\vec{y})$$

$$R(x,\vec{y}) = (\exists\, i \leqslant x)\; P(i,\vec{y}) \qquad \chi_R(x,\vec{y}) = \mathrm{sg}\left(\sum_{i \leqslant x} \chi_P(i,\vec{y})\right)$$

Examples

- x is divisor of y $\qquad x \mid y \iff (\exists\, i \leqslant y)\,[\, i \times x = y \,]$

Examples

- x is divisor of y $\quad x \mid y \iff (\exists\, i \leqslant y)\,[\, i \times x = y \,]$
- x is prime number $\quad \mathrm{prime}(x) \iff x > 1 \wedge (\forall\, i \leqslant x)\,[\, i \mid x \implies i = 1 \vee i = x \,]$

Examples

- x is divisor of y $\quad x \mid y \iff (\exists\, i \leqslant y)\,\big[\, i \times x = y \,\big]$
- x is prime number $\quad \text{prime}(x) \iff x > 1 \wedge (\forall\, i \leqslant x)\,\big[\, i \mid x \implies i = 1 \vee i = x \,\big]$

Lemma (bounded minimization)

if $P\colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\mu\, i \leqslant x)\; P(i,\vec{y}) = \min\,\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y}) \,\}$$

is primitive recursive

Examples

- x is divisor of y $\quad x \mid y \iff (\exists\, i \leqslant y)\,\big[\, i \times x = y \,\big]$
- x is prime number $\quad \mathrm{prime}(x) \iff x > 1 \wedge (\forall\, i \leqslant x)\,\big[\, i \mid x \implies i = 1 \vee i = x \,\big]$

Lemma (bounded minimization)

if $P\colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\mu\, i \leqslant x)\; P(i,\vec{y}) = \min \{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y}) \,\} \cup \{\, x+1 \,\}$$

is primitive recursive

Examples

- x is divisor of y $\quad x \mid y \iff (\exists\, i \leqslant y)\,\big[\, i \times x = y \,\big]$
- x is prime number $\quad \mathrm{prime}(x) \iff x > 1 \land (\forall\, i \leqslant x)\,\big[\, i \mid x \implies i = 1 \lor i = x \,\big]$

Lemma (bounded minimization)

if $P\colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\,\{i \mid 0 \leqslant i \leqslant x \land P(i,\vec{y})\} \cup \{x+1\}$$

is primitive recursive

Example

- $\lfloor \frac{x}{2} \rfloor$ is primitive recursive:

Examples

- x is divisor of y $\qquad x \mid y \iff (\exists\, i \leqslant y)\left[\, i \times x = y \,\right]$
- x is prime number $\qquad \text{prime}(x) \iff x > 1 \wedge (\forall\, i \leqslant x)\left[\, i \mid x \implies i = 1 \vee i = x \,\right]$

Lemma (bounded minimization)

if $P \colon \mathbb{N}^{n+1} \to \mathbb{B}$ is primitive recursive then

$$(\mu\, i \leqslant x)\, P(i, \vec{y}) = \min \{ i \mid 0 \leqslant i \leqslant x \wedge P(i, \vec{y}) \} \cup \{ x + 1 \}$$

is primitive recursive

Example

- $\lfloor \frac{x}{2} \rfloor$ is primitive recursive:

$$\lfloor \tfrac{x}{2} \rfloor = (\mu\, i \leqslant x) \left[\, (i+1) \times 2 > x \,\right]$$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min \{ i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y}) \} \cup \{ x+1 \}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases}$$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\,\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y})\,\} \cup \{\, x+1 \,\}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases} \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i \leqslant x)\, P(i,\vec{y}) \\ x+1 & \text{if } \neg(\exists\, i \leqslant x)\, P(i,\vec{y}) \text{ and } P(x+1,\vec{y}) \\ x+2 & \text{otherwise} \end{cases}$$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y})\,\} \cup \{\, x+1 \,\}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases} \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i \leqslant x)\, P(i,\vec{y}) \\ x+1 & \text{if } \neg(\exists\, i \leqslant x)\, P(i,\vec{y}) \text{ and } P(x+1,\vec{y}) \\ x+2 & \text{otherwise} \end{cases}$$

Examples

- division $\qquad x \div y = (\mu\, i \leqslant x)\,\big[\, (i+1) \times y > x \,\big]$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\,\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y}) \,\} \cup \{\, x+1 \,\}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases} \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i \leqslant x)\, P(i,\vec{y}) \\ x+1 & \text{if } \neg(\exists\, i \leqslant x)\, P(i,\vec{y}) \text{ and } P(x+1,\vec{y}) \\ x+2 & \text{otherwise} \end{cases}$$

Examples

- division $\quad x \div y = (\mu\, i \leqslant x)\,\big[\,(i+1) \times y > x\,\big]$
- exponent $\quad \exp(x,y) = (\mu\, i \leqslant x)\,\big[\,y^i \mid x \wedge \neg(y^{i+1} \mid x)\,\big]$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\,\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y}) \,\} \cup \{\, x+1 \,\}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases} \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i \leqslant x)\, P(i,\vec{y}) \\ x+1 & \text{if } \neg(\exists\, i \leqslant x)\, P(i,\vec{y}) \text{ and } P(x+1,\vec{y}) \\ x+2 & \text{otherwise} \end{cases}$$

Examples

- division $\quad x \div y = (\mu\, i \leqslant x)\, [\, (i+1) \times y > x \,]$
- exponent $\quad \exp(x,y) = (\mu\, i \leqslant x)\, [\, y^i \mid x \wedge \neg(y^{i+1} \mid x) \,]$
- remainder $\quad x \bmod y = x \mathbin{\dot{-}} (y \times (x \div y))$

Proof

$$f(x,\vec{y}) = (\mu\, i \leqslant x)\, P(i,\vec{y}) = \min\,\{\, i \mid 0 \leqslant i \leqslant x \wedge P(i,\vec{y})\,\} \cup \{\, x+1 \,\}$$

$$f(0,\vec{y}) = \begin{cases} 0 & \text{if } P(0,\vec{y}) \\ 1 & \text{otherwise} \end{cases} \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i \leqslant x)\, P(i,\vec{y}) \\ x+1 & \text{if } \neg(\exists\, i \leqslant x)\, P(i,\vec{y}) \text{ and } P(x+1,\vec{y}) \\ x+2 & \text{otherwise} \end{cases}$$

Examples

- division $\quad x \div y = (\mu\, i \leqslant x)\, [\, (i+1) \times y > x \,]$
- exponent $\quad \exp(x,y) = (\mu\, i \leqslant x)\, [\, y^i \mid x \wedge \neg(y^{i+1} \mid x) \,]$
- remainder $\quad x \bmod y = x \dot{-} (y \times (x \div y))$
- n-th prime number p_n $\quad \mathrm{p}_0 = 2 \qquad$ with $\quad f(x) = g(x!+1, x)$

  $\mathrm{p}_{n+1} = f(\mathrm{p}_n) \qquad g(x,y) = (\mu\, i \leqslant x)\, [\, \mathrm{prime}(i) \wedge i > y \,]$

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y})$$

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y})$$

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

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y})$$

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

$$Q(x,\vec{y}) = (\exists\, i < x)\; P(i,\vec{y})$$

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y}) \qquad \begin{aligned} g(0,\vec{y}) &= 0 \\ g(x+1,\vec{y}) &= f(x,\vec{y}) + g(x,\vec{y}) \end{aligned}$$

$$Q(x,\vec{y}) = (\exists\, i < x)\; P(i,\vec{y}) \qquad \chi_Q(x,\vec{y}) = \mathrm{sg}\left(\sum_{i<x} \chi_P(i,\vec{y})\right)$$

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y}) \qquad \begin{aligned} g(0,\vec{y}) &= 0 \\ g(x+1,\vec{y}) &= f(x,\vec{y}) + g(x,\vec{y}) \end{aligned}$$

$$Q(x,\vec{y}) = (\exists\, i < x)\; P(i,\vec{y}) \qquad \chi_Q(x,\vec{y}) = \mathrm{sg}\left(\sum_{i<x} \chi_P(i,\vec{y})\right)$$

$$f(x,\vec{y}) = (\mu\, i < x)\; P(i,\vec{y}) = \min\,\{i \mid 0 \leqslant i < x \wedge P(i,\vec{y})\} \cup \{x\}$$

Remark

replacing $i \leqslant x$ by $i < x$ does not affect closure under bounded minimization

Proof

$$g(x,\vec{y}) = \sum_{i<x} f(i,\vec{y}) \qquad \begin{aligned} g(0,\vec{y}) &= 0 \\ g(x+1,\vec{y}) &= f(x,\vec{y}) + g(x,\vec{y}) \end{aligned}$$

$$Q(x,\vec{y}) = (\exists\, i < x)\; P(i,\vec{y}) \qquad \chi_Q(x,\vec{y}) = \mathrm{sg}\Big(\sum_{i<x} \chi_P(i,\vec{y})\Big)$$

$$f(x,\vec{y}) = (\mu\, i < x)\; P(i,\vec{y}) = \min\,\{i \mid 0 \leqslant i < x \,\wedge\, P(i,\vec{y})\} \cup \{x\}$$

$$f(0,\vec{y}) = 0 \qquad f(x+1,\vec{y}) = \begin{cases} f(x,\vec{y}) & \text{if } (\exists\, i < x)\; P(i,\vec{y}) \\ x & \text{if } \neg(\exists\, i < x)\; P(i,\vec{y}) \text{ and } P(x,\vec{y}) \\ x+1 & \text{otherwise} \end{cases}$$

Outline

Example

- Fibonacci function $\text{fib}(x)$

$$\begin{aligned}\text{fib}(0) &= 1\\ \text{fib}(1) &= 1\\ \text{fib}(x+2) &= \text{fib}(x+1) + \text{fib}(x)\end{aligned}$$

is primitive recursive

Example

- Fibonacci function $\text{fib}(x)$

$$\begin{aligned} \text{fib}(0) &= 1 \\ \text{fib}(1) &= 1 \\ \text{fib}(x+2) &= \text{fib}(x+1) + \text{fib}(x) \qquad \text{course–of–values recursion} \end{aligned}$$

is primitive recursive ?

Example

- Fibonacci function $\text{fib}(x)$

$$\begin{aligned} \text{fib}(0) &= 1 \\ \text{fib}(1) &= 1 \\ \text{fib}(x+2) &= \text{fib}(x+1) + \text{fib}(x) \qquad \text{course–of–values recursion} \end{aligned}$$

is primitive recursive ?

Idea

combine $\text{fib}(x+1)$ and $\text{fib}(x)$ into a single number from which $\text{fib}(x+1)$ and $\text{fib}(x)$ can be obtained by suitable primitive recursive extraction functions

Definitions

- pairing function $\pi(x, y) = 2^x(2y + 1) \dot{-} 1$

Definitions

- pairing function $\pi(x,y) = 2^x(2y+1) \dot{-} 1$
- extraction functions $\pi_1(z) = (\mu\, x \leqslant z)\,(\exists\, y \leqslant z)\,\big[\, z = \pi(x,y)\,\big]$

  $\pi_2(z) = (\mu\, y \leqslant z)\,(\exists\, x \leqslant z)\,\big[\, z = \pi(x,y)\,\big]$

Definitions

- pairing function $\pi(x,y) = 2^x(2y+1) \dot{-} 1$
- extraction functions $\pi_1(z) = (\mu\, x \leqslant z)\,(\exists\, y \leqslant z)\,[\,z = \pi(x,y)\,]$

  $\pi_2(z) = (\mu\, y \leqslant z)\,(\exists\, x \leqslant z)\,[\,z = \pi(x,y)\,]$

Lemmata

1. $x, y \leqslant \pi(x,y)$

Definitions

- pairing function $\pi(x,y) = 2^x(2y+1) \dot{-} 1$
- extraction functions $\pi_1(z) = (\mu\, x \leqslant z)\,(\exists\, y \leqslant z)\,[\, z = \pi(x,y)\,]$

  $\pi_2(z) = (\mu\, y \leqslant z)\,(\exists\, x \leqslant z)\,[\, z = \pi(x,y)\,]$

Lemmata

1. $x, y \leqslant \pi(x,y)$
2. $\pi_1(\pi(x,y)) = x$ and $\pi_2(\pi(x,y)) = y$

Definitions

- pairing function $\pi(x, y) = 2^x(2y + 1) \dot{-} 1$
- extraction functions $\pi_1(z) = (\mu\, x \leqslant z)\,(\exists\, y \leqslant z)\,[\, z = \pi(x, y)\,]$

  $\pi_2(z) = (\mu\, y \leqslant z)\,(\exists\, x \leqslant z)\,[\, z = \pi(x, y)\,]$

Lemmata

1. $x, y \leqslant \pi(x, y)$
2. $\pi_1(\pi(x, y)) = x$ and $\pi_2(\pi(x, y)) = y$
3. $\pi(\pi_1(z), \pi_2(z)) = z$

Definitions

- pairing function $\pi(x,y) = 2^x(2y+1) \dot{-} 1$
- extraction functions $\pi_1(z) = (\mu\, x \leqslant z)\,(\exists\, y \leqslant z)\,[\, z = \pi(x,y)\,]$
  $\pi_2(z) = (\mu\, y \leqslant z)\,(\exists\, x \leqslant z)\,[\, z = \pi(x,y)\,]$

Lemmata

1. $x, y \leqslant \pi(x,y)$
2. $\pi_1(\pi(x,y)) = x$ and $\pi_2(\pi(x,y)) = y$
3. $\pi(\pi_1(z), \pi_2(z)) = z$
4. π, π_1, π_2 are primitive recursive

Lemmata

① $x, y \leqslant \pi(x, y)$

Proof

① $\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

Lemmata

1. $x, y \leqslant \pi(x, y)$

Proof

1. $\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

   $\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant (2y + 1) \dot{-} 1 = 2y \geqslant y$

Lemmata

① $x, y \leqslant \pi(x,y)$

② $\pi_1(\pi(x,y)) = x$ and $\pi_2(\pi(x,y)) = y$

Proof

① $\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

$\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant (2y+1) \dot{-} 1 = 2y \geqslant y$

② $\pi_1(\pi(x,y)) = (\mu\, x' \leqslant \pi(x,y))\, (\exists\, y' \leqslant \pi(x,y))\, \left[\, \pi(x,y) = \pi(x',y') \,\right]$

Lemmata

1. $x, y \leqslant \pi(x,y)$
2. $\pi_1(\pi(x,y)) = x$ and $\pi_2(\pi(x,y)) = y$

Proof

1. $\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

   $\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant (2y+1) \dot{-} 1 = 2y \geqslant y$

2. $\pi_1(\pi(x,y)) = (\mu\, x' \leqslant \pi(x,y))\, (\exists\, y' \leqslant \pi(x,y))\, \big[\, \pi(x,y) = \pi(x',y')\, \big]$

   π is injective $\implies$ $\pi(x,y) = \pi(x',y')$ only holds when $x' = x$ and $y' = y$

Lemmata

1. $x, y \leqslant \pi(x, y)$
2. $\pi_1(\pi(x, y)) = x$ and $\pi_2(\pi(x, y)) = y$
3. $\pi(\pi_1(z), \pi_2(z)) = z$

Proof

① $\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

$\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant (2y + 1) \dot{-} 1 = 2y \geqslant y$

② $\pi_1(\pi(x, y)) = (\mu\, x' \leqslant \pi(x, y))\, (\exists\, y' \leqslant \pi(x, y))\, \big[\, \pi(x, y) = \pi(x', y')\, \big]$

π is injective $\implies$ $\pi(x, y) = \pi(x', y')$ only holds when $x' = x$ and $y' = y$

③ π is surjective $\implies$ $z = \pi(x, y)$ for some x and y

Lemmata

1. $x, y \leqslant \pi(x,y)$
2. $\pi_1(\pi(x,y)) = x$ and $\pi_2(\pi(x,y)) = y$
3. $\pi(\pi_1(z), \pi_2(z)) = z$

Proof

① $\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

$\pi(x,y) = 2^x(2y+1) \dot{-} 1 \geqslant (2y+1) \dot{-} 1 = 2y \geqslant y$

② $\pi_1(\pi(x,y)) = (\mu\, x' \leqslant \pi(x,y))\, (\exists\, y' \leqslant \pi(x,y))\, \left[\, \pi(x,y) = \pi(x',y') \,\right]$

π is injective $\implies$ $\pi(x,y) = \pi(x',y')$ only holds when $x' = x$ and $y' = y$

③ π is surjective $\implies$ $z = \pi(x,y)$ for some x and y

$\pi_1(z) = x$ and $\pi_2(z) = y$

Lemmata

① $x, y \leqslant \pi(x, y)$

② $\pi_1(\pi(x, y)) = x$ and $\pi_2(\pi(x, y)) = y$

③ $\pi(\pi_1(z), \pi_2(z)) = z$

Proof

① $\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant 2^x \dot{-} 1 \geqslant x$

$\pi(x, y) = 2^x(2y + 1) \dot{-} 1 \geqslant (2y + 1) \dot{-} 1 = 2y \geqslant y$

② $\pi_1(\pi(x, y)) = (\mu\, x' \leqslant \pi(x, y))\, (\exists\, y' \leqslant \pi(x, y))\, \big[\, \pi(x, y) = \pi(x', y')\, \big]$

π is injective $\implies$ $\pi(x, y) = \pi(x', y')$ only holds when $x' = x$ and $y' = y$

③ π is surjective $\implies$ $z = \pi(x, y)$ for some x and y

$\pi_1(z) = x$ and $\pi_2(z) = y$ $\implies$ $z = \pi(\pi_1(z), \pi_2(z))$

Lemma

fib is primitive recursive

Proof

- $g(x) = \pi(\text{fib}(x), \text{fib}(x+1))$ is primitive recursive

Lemma

fib is primitive recursive

Proof

- $g(x) = \pi(\mathrm{fib}(x), \mathrm{fib}(x+1))$ is primitive recursive:

$$\begin{aligned} g(0) &= \pi(1,1) \\ g(x+1) &= \pi(\pi_2(g(x)), \pi_1(g(x)) + \pi_2(g(x))) \end{aligned}$$

Lemma

fib is primitive recursive

Proof

- $g(x) = \pi(\text{fib}(x), \text{fib}(x+1))$ is primitive recursive:

$$\begin{aligned} g(0) &= \pi(1,1) \\ g(x+1) &= \pi(\pi_2(g(x)), \pi_1(g(x)) + \pi_2(g(x))) \end{aligned}$$

- $\text{fib}(x) = \pi_1(g(x))$

Outline

Important Concepts

- bounded minimization
- bounded quantification
- case analysis
- characteristic function
- composition
- initial function
- iteration
- pairing
- PR
- primitive recursion

Important Concepts

- bounded minimization
- bounded quantification
- case analysis
- characteristic function
- composition
- initial function
- iteration
- pairing
- PR
- primitive recursion

homework for October 9