Computability Theory

Aart Middeldorp

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

1. **Organisation**

2. **Contents**

3. **Primitive Recursive Functions**

4. **Primitive Recursive Predicates**

5. **Pairing**

6. **Summary**

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

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

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

- 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, ...

Outline

► which function is computable ?

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

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

► which are (total) functions ?

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

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

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

Definition

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

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

and is closed under composition

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

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

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

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

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

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

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

with $g(x) = \mathsf{z}(x)$ and $h(x, y, z) = \pi_1^3(x, y, z) + \pi_3^3(x, y, z)$

▶ exponentiation $\qquad x^y = f(y, x) \in \mathsf{PR}$

$$f(0, y) = \mathsf{s}(\mathsf{z}(y))$$
$$f(x + 1, y) = \pi_1^3(f(x, y), x, y) \times \pi_3^3(f(x, y), x, y)$$

Lemma

$g(x_1, \ldots, x_m) = f(y_1, \ldots, y_n) \in \mathsf{PR}$ if $f \colon \mathbb{N}^n \to \mathbb{N} \in \mathsf{PR}$ 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)

$$x \mathbin{\dot-} y = \begin{cases} x - y & \text{if } x \geqslant y \\ 0 & \text{otherwise} \end{cases} \qquad\qquad \begin{aligned} x \mathbin{\dot-} 0 &= x \\ x \mathbin{\dot-} (y + 1) &= \mathsf{p}(x \mathbin{\dot-} y) \end{aligned}$$

is primitive recursive

Examples

- **predecessor** $p(x) = f(x) \in PR$

$$f(0) = 0 \;?$$
$$f(x+1) = x = \pi_2^2(f(x), x)$$

- **factorial** $x! = f(x) \in PR$

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

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

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

Definition

function $f \colon \mathbb{N}^2 \to \mathbb{N}$ is obtained from function $g \colon \mathbb{N} \to \mathbb{N}$ by iteration if
$$f(n, x) = g^{(n)}(x) = \underbrace{g(\cdots g(x) \cdots)}_{n \text{ times}}$$

Lemma

PR is closed under iteration

Proof

- suppose $g \colon \mathbb{N} \to \mathbb{N} \in \mathsf{PR}$
- $f(n, x) = g^{(n)}(x)$ 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 ?

Definition

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

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

is primitive recursive

Lemma

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

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

Proof

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

Examples

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

- $>$ and $\neq$ are primitive recursive predicates

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

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

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

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

how to implement

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

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 and product)

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

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

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 $P: \mathbb{N}^{n+1} \to \mathbb{B}$ 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

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

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

Examples

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

Example

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

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

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

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

- remainder $\qquad x \bmod y = x \mathbin{\dot{-}} (y \times (x \div y))$

- n-th prime number $\mathsf{p}_n \qquad \mathsf{p}_0 = 2 \qquad$ with $\qquad f(x) = g(x! + 1, x)$

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

Remark

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

Proof

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

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

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

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

Outline

Example

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

$$\mathsf{fib}(0) = 1$$
$$\mathsf{fib}(1) = 1$$
$$\mathsf{fib}(x + 2) = \mathsf{fib}(x + 1) + \mathsf{fib}(x) \qquad \mathsf{course-of-values} \text{ recursion}$$

is primitive recursive ?

Idea

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

Definitions

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

- **extraction** functions $\pi_1(z) = (\mu\, x \leqslant z)\, (\exists\, y \leqslant z)\, \big[\, z = \pi(x, y)\, \big]$
  $$\pi_2(z) = (\mu\, y \leqslant z)\, (\exists\, x \leqslant z)\, \big[\, z = \pi(x, y)\, \big]$$

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. $\pi,\ \pi_1,\ \pi_2$ are primitive recursive

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

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

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:

$$g(0) = \pi(1, 1)$$
$$g(x+1) = \pi(\pi_2(g(x)), \pi_1(g(x)) + \pi_2(g(x)))$$

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

Outline

Important Concepts

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

homework for October 9