Computational Logic

Logic (master program)

Georg Moser

Institute of Computer Science @ UIBK

Winter 2008

Summary of Last Lecture

Definition superposition (right, left)

$$\frac{C \vee s = t \quad D \vee A[s']}{C\sigma \vee D\sigma \vee A[t]\sigma} \qquad \frac{C \vee s = t \quad D \vee \neg A[s']}{C\sigma \vee D\sigma \vee \neg A[t]\sigma}$$

- σ is mgu of s and s'
- s' is not a variable

Definition factoring (ordered, equality)

$$\frac{C \vee A \vee B}{C\sigma \vee A\sigma} \qquad \frac{C \vee s = t \vee s' = t'}{C\sigma \vee t\sigma \neq t'\sigma \vee s'\sigma = t'\sigma}$$

- σ is mgu of A and B, or mgu of s and s', respectively

Definition resolution (equality, standard)

$$\frac{C \vee s \neq t}{C\sigma} \qquad \frac{C \vee P(s_1, \dots, s_n) \quad D \vee \neg P(t_1, \dots, t_n)}{C\sigma \vee D\sigma}$$

- σ is mgu of s and t or of $P(s_1, \dots, s_n)$, $P(t_1, \dots, t_n)$ respectively

Observation
factoring is only necessary for positive atoms

Theorem

- paramodulation is sound and complete
- superposition is sound, complete, and can be efficiently implemented

Content

introduction, propositional logic, semantics, formal proofs, resolution (propositional)

first-order logic, semantics, structures, theories and models, formal proofs, Herbrand theory, completeness of first-order logic, properties of first-order logic, resolution (first-order)

introduction to computability, introduction to complexity, finite model theory

beyond first order: modal logics in a general setting, higher-order logics, introduction to Isabelle

Computability Theory

We refer to problems as decidable or undecidable according to whether or not there exists an algorithm that solves the problem. Computability theory considers undecidable problems and the brink between the undecidable and the decidable.

Characterisation of Computable Functions

Example

consider

- the zero function $\mathsf{Z}(x) = 0$
- the successor function $\mathsf{s}(x) = x + 1$
- the projection functions $\mathsf{p}^n_i(x_1, x_2, \ldots, x_n) = x_i$

these functions are certainly computable

Definition — basic functions

the functions Z, s, p^n_i are called basic functions

Example

consider

- a computable function f
- a computable function g

then the composition $h(x) = f(g(x))$ is certainly computable

Definition — closed under composition

let $\mathcal{S}$ be a set of functions on $\mathbb{N}$ and suppose

- $\forall\ h \colon \mathbb{N}^m \to \mathbb{N}$ in $\mathcal{S}$
- $\forall\ 1 \leqslant i \leqslant m\ g_i \colon \mathbb{N}^n \to \mathbb{N}$ in $\mathcal{S}$

the function defined as:

$$f(x_1, \ldots, x_n) = h(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))$$

is contained in $\mathcal{S}$, then $\mathcal{S}$ is closed under composition

Example

consider a function f defined by induction

- $f(0) = 1$
- $f(x + 1) = f(x) \cdot (x + 1)$

then f is certainly computable

Definition — closed under primitive recursion

let $\mathcal{S}$ be a set of functions on $\mathbb{N}$ and suppose

- $\forall\ h \colon \mathbb{N}^{n-1} \to \mathbb{N}$ in $\mathcal{S}$ $\qquad n > 0$
- $\forall\ g \colon \mathbb{N}^{n+1} \to \mathbb{N}$ in $\mathcal{S}$

the function defined as:

$$f(0, x_2, \ldots, x_n) = h(x_2, \ldots, x_n)$$
$$f(x_1 + 1, \ldots, x_n) = g(x_1, \ldots, x_n, f(x_1, \ldots, x_n))$$

is contained in $\mathcal{S}$, then $\mathcal{S}$ is closed under primitive recursion

Primitive Recursive Functions

Definition

the primitive recursive functions are the smallest set containing the basic functions that is closed under composition and primitive recursion

Example

the following function are primitive recursive

- the addition function $\mathsf{a}(x, y) = x + y$
- the predecessor function $\mathsf{p}(x) = x \dot{-} 1$
- the (modified) subtraction function $\mathsf{sub}(x, y) = x \dot{-} y$
- the multiplication function $\mathsf{m}(x, y) = x \cdot y$
- the exponentiation function $\exp(x, y) = x^y$

Proposition

given a polynomial $p(x)$ with natural numbers as coefficients, then $p(x)$ is primitive recursive

Definition

closed under bounded sums

$\mathcal{S}$ is closed under bounded sums if

- $\forall\, f : \mathbb{N}^n \to \mathbb{N}$

the function $\mathsf{sum}_f(y, x_2, \ldots, x_n) = \sum_{z<y}(f(z, x_2, \ldots, x_n))$ is in $\mathcal{S}$

Proposition

the set of primitive recursive functions is closed under bounded sums

Proof

let $f(x_1, \ldots, x_n)$ be primitive recursive

- $h_1(x_2, \ldots, x_n) = 0$
- $h_2(x_1, \ldots, x_n, x_{n+1}) = f(x_1, \ldots, x_n) + x_{n+1}$
- h_1, h_2 are primitive recursive; so is the function g:

$$g(0, x_2, \ldots, x_n) = h_1(x_2, \ldots, x_n) = 0$$
$$g(x_1 + 1, x_2, \ldots, x_n) = h_2(x_1, x_2, \ldots, g(x_1, x_2, \ldots, x_n))$$

- clearly $g(y, x_2, \ldots, x_n) = \mathsf{sum}_f(y, x_2, \ldots, x_n)$ ∎

Recursive Functions

Definition

closed under unbounded search

let $\mathcal{S}$ be a set of functions on $\mathbb{N}$ and suppose

- $\forall\, f : \mathbb{N}^{n+1} \to \mathbb{N}$ in $\mathcal{S}$

the function defined as:

$$\mu_f(x_1, \ldots, x_n, y) = \begin{cases} z & \forall\, y \leqslant z\ f(\vec{x}, y) \text{ is defined and } \\ & z = \min\{v \mid f(\vec{x}, v) = 0\} \\ \text{undefined} & \text{otherwise} \end{cases}$$

is contained in $\mathcal{S}$, then $\mathcal{S}$ is closed under unbounded search

Definition

the set of recursive functions is the smallest set containing the primitive recursive functions that is closed under unbounded search

Example

the Ackermann function

$$\mathsf{ack}(0, n) = n + 1 \qquad \mathsf{ack}(n + 1, m + 1) = \mathsf{ack}(n, \mathsf{ack}(n + 1, m))$$
$$\mathsf{ack}(n + 1, 0) = \mathsf{ack}(n, 1)$$

is a total non-primitive recursive function that is recursive

Theorem

Kleene

- every (total) recursive function f is computable by a (total) TM and vice versa
- the n-ary recursive functions are recursively enumerable:

$$\varphi_0^n, \varphi_1^n, \varphi_2^n, \varphi_3^n, \ldots$$

Church-Turing Thesis

f is computable $=$ f TM computable $=$ f is recursive

Computable Sets and Relations

Definition

characteristic function

the characteristic function χ_A of $A \subseteq \mathbb{N}^n$:

$$\chi_A(x_1,\dots,x_n) = \begin{cases} 1 & (x_1,\dots,x_n) \in A \\ 0 & (x_1,\dots,x_n) \notin A \end{cases}$$

Example

consider the relation $x < y$

$$\chi_<(x,y) = \begin{cases} 1 & x < y \\ 0 & \text{otherwise} \end{cases}$$

$\chi_<$ is primitive recursive: $\chi_<(x,y) = 1 \dot{-} (1 \dot{-} (y \dot{-} x))$

Definition

set $A \subseteq \mathbb{N}^n$ is called

- primitive recursive if χ_A is primitive recursive
- recursive if χ_A is recursive

let $\mathbf{N} = (\mathbb{N}, +, \cdot, 0, 1)$ denote the structure with domain $\mathbb{N}$ and vocabulary $\mathcal{V}_{ar} = \{+, \cdot, 0, 1\}$

Proposition

if A is definable by a quantifier-free $\mathcal{V}_{ar}$-formula, then A is primitive recursive

Proof

let $\varphi_A(x_1,\dots,x_n)$ be a $\mathcal{V}_{ar}$-formula

- assume $\theta(\vec{x})$ and $\psi(\vec{x})$ are formulas and define primitive recursive subsets B and C
- χ_B, χ_C are primitive recursive
- if $\varphi_A(\vec{x}) \equiv \neg\theta(\vec{x})$ then
$$\chi_A(\vec{x}) = 1 \dot{-} \chi_B$$
holds
- hence χ_A is primitive recursive, thus A is

- if $\varphi_A(\vec{x}) \equiv \theta(\vec{x}) \wedge \psi(\vec{x})$ then
$$\chi_A(\vec{x}) = \chi_B(\vec{x}) \cdot \chi_C(\vec{x})$$
holds
- hence A is primitive recursive

this concludes the step case, now we consider the base case

- each $\mathcal{V}_{ar}$-term defines a polynomial with naturals as coefficients
- the relation $p(\vec{x}) = q(\vec{x})$ is primitive recursive for polynomials p, q
- as $x < y$ is primitive recursive
$\neg(x < y) \equiv y \leqslant x$ is primitive recursive
- $x = y :\Leftrightarrow x \leqslant y \wedge y \leqslant x$ is primitive recursive
- let $\chi_{eq}(p(\vec{x}), q(\vec{x}))$ denote the induced characteristic function
- if $\varphi_A(\vec{x}) \equiv s = t$ then
$$\chi_A(\vec{x}) = \chi_{eq}(p(\vec{x}), q(\vec{x}))$$
holds if $p(\vec{x})$, $q(\vec{x})$ are defined by s, t ∎