2-INF-264 Teória deklaratívneho programovania

Zimný semester 2015/16

1. prednáška

Ján Komara

Obsah 1. prednášky

Cieľ a obsah predmetu

Cieľ predmetu

- Vybudovať matematické základy deklaratívnych programovacích jazykov.

Kde hľadáme riešenie?

- V matematickej logike – v časti, ktorá sa volá *Teória rekurzívnych funkcií*.

Prečo?

- Churchova téza: trieda intuitívne vypočítateľných funkcií je totožná s obecne rekurzívnymi funkciami (Herbrand-Gödel).
- Návrh prvého jazyka deklaratívnej paradigmy jazyka Lisp bol silne ovplyvnený formalizmom čiastočne rekurzívnych funkcií.

Deklaratívne programovanie

Výučba deklaratívneho programovania na fakulte

- Podmieňujúce predmety (doporučené):
  - 1-AIN-505 Úvod do deklaratívneho programovania,
  - 1-INF-465 Deklaratívne programovanie.
- Súvisiace predmety:
  - 1-AIN-470 Špecifikácia a verifikácia programov,
- Softvér používaný pri výučbe týchto predmetov:
  - programovací jazyk a špecifikačno-verifikačný systém CL (Clausal Language).
- Iné deklaratívne progamovacie jazyky:
  - funkcionálne programovacie jazyky – Lisp, Haskell;
  - logické programovacie jazyky – Prolog.

Deklaratívne programovanie

Paradigma deklaratívneho programovania

- Deklaratívne programy sú definície matematických objektov (funkcie, relácie).
- Zhoda medzi definičnou a výpočtovou sémantikou umožňuje analyzovať programy elementárnymi prostriedkami.
- Všetky časti tvorby programu je možné realizovať v tom istom formalizme:
  - špecifikácia,
  - implementácia,
  - verifikácia,
  - výpočet.
- Jednoduchá sémantika sa kombinuje s expresívnymi programátorskými konštrukciami.

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Špecifikácia

- Špecifikačný predikát

$$x \mid y \leftrightarrow \exists z\, y = x{\cdot}z.$$

- Špecifikácia programu

$$x \neq 0 \vee y \neq 0 \rightarrow \gcd(x,y) \mid x \wedge \gcd(x,y) \mid y \wedge \forall z\big(z \mid x \wedge z \mid y \rightarrow z \leq \gcd(x,y)\big).$$

  Tu $\gcd(x,y)$ označuje najväčšieho spoločného deliteľa prirodzených čísel x a y.

- Idea algoritmu

$$x > y \wedge z \mid y \rightarrow z \mid x \leftrightarrow z \mid x \dot{-} y.$$

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Implementácia

$$
\begin{aligned}
\gcd(x, y) = \textbf{if } & x \neq 0 \wedge y \neq 0 \textbf{ then} \\
& \textbf{case} \\
& \quad x > y \Rightarrow \gcd(x \dot{-} y, y) \\
& \quad x = y \Rightarrow x \\
& \quad x < y \Rightarrow \gcd(x, y \dot{-} x) \\
& \textbf{end} \\
\textbf{else} & \\
& \max(x, y)
\end{aligned}
$$

Výpočet

Výpočet s argumentami klesajúcimi v miere $\max(x, y)$:

$$
\begin{array}{ccccccccc}
\gcd(9,12) & = & \gcd(9,3) & = & \gcd(6,3) & = & \gcd(3,3) & = 3 \\
12 & > & 9 & > & 6 & > & 3 &
\end{array}
$$

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Implementácia

$$\begin{aligned}\gcd(x, y) = \textbf{if } & x \neq 0 \wedge y \neq 0 \textbf{ then} \\ & \textbf{case} \\ & \quad x > y \Rightarrow \gcd(x \dot{-} y, y) \\ & \quad x = y \Rightarrow x \\ & \quad x < y \Rightarrow \gcd(x, y \dot{-} x) \\ & \textbf{end} \\ \textbf{else} & \\ & \max(x, y)\end{aligned}$$

Výpočet

Podmienky regularity zaručujú, že výpočet vždy skončí:

$$x \neq 0 \wedge y \neq 0 \wedge x > y \rightarrow \max(x \dot{-} y, y) < \max(x, y)$$
$$x \neq 0 \wedge y \neq 0 \wedge x < y \rightarrow \max(x, y \dot{-} x) < \max(x, y).$$

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Implementácia

$$
\begin{aligned}
\gcd(x, y) = \textbf{if } & x \neq 0 \wedge y \neq 0 \textbf{ then} \\
& \textbf{case} \\
& \quad x > y \Rightarrow \gcd(x \dot{-} y, y) \\
& \quad x = y \Rightarrow x \\
& \quad x < y \Rightarrow \gcd(x, y \dot{-} x) \\
& \textbf{end} \\
\textbf{else} & \\
& \max(x, y)
\end{aligned}
$$

Výpočet

Podmienka $x \neq 0 \wedge y \neq 0$ sa zbytočne opakovane vyhodnocuje

$$\gcd(9, 12) = \gcd(9, 3) = \gcd(6, 3) = \gcd(3, 3) = 3$$

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Alternatívna implementácia so vstupnou podmienkou

$$x \neq 0 \wedge y \neq 0 \rightarrow \gcd(x, y) = \textbf{case}$$
$$x > y \Rightarrow \gcd(x \dot{-} y, y)$$
$$x = y \Rightarrow x$$
$$x < y \Rightarrow \gcd(x, y \dot{-} x)$$
$$\textbf{end}$$

Výpočet

Ten nemusí skončiť:

$$\gcd(1, 0) = \gcd(1 \dot{-} 0, 0) = \gcd(1, 0) = \cdots$$

Program počíta čiastočnú funkciu!

Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Alternatívna implementácia so vstupnou podmienkou

$$x \neq 0 \wedge y \neq 0 \rightarrow \gcd(x, y) = \textbf{case}$$
$$x > y \Rightarrow \gcd(x \dot{-} y, y)$$
$$x = y \Rightarrow x$$
$$x < y \Rightarrow \gcd(x, y \dot{-} x)$$
$$\textbf{end}$$

Výpočet

Rozšírené podmienky regularity pre mieru $\max(x, y)$:

$$x \neq 0 \wedge y \neq 0 \wedge x > y \rightarrow \max(x \dot{-} y, y) < \max(x, y) \wedge x \dot{-} y \neq 0 \wedge y \neq 0$$
$$x \neq 0 \wedge y \neq 0 \wedge x < y \rightarrow \max(x, y \dot{-} x) < \max(x, y) \wedge x \neq 0 \wedge y \dot{-} x \neq 0.$$

Výpočet skončí pre vstupy spĺňajúce vstupnú podmienku.

Deklaratívne programovanie

Zhrnutie

- *Univerzum* je množina prirodzených čísel $\mathrm{N} = \{0, 1, 2, 3, \dots\}$.
- *Dátové štruktúry* kódujeme do N v štýle jazyka Lisp s pomocou vhodnej párovacej funkcie.
- *Programy* sú vlastnosti totálnych funkcií nad oborom N spĺňajúce určité vstupné podmienky.
- *Formálny systém* je druhorádová formalizácia aritmetiky.

Metamatematika deklaratívneho programovania

- Primitívne rekurzívne funkcie:
  - aritmetizácia dátových štruktúr, rekurzia s mierou.
- Obecne rekurzívne funkcie.
- Čiastočne rekurzívne funkcie:
  - Churchova téza, algoritmicky nerozhodnuteľné problémy (problém zastavenia).

Primitívne rekurzívne funkcie

Typy rekurzie

Primitívna rekurzia

- Umocňovanie:

$$x^0 = 1 \qquad x^{y+1} = x \cdot x^y.$$

Rekurzia s mierou

- Fibonacciho postupnosť:

$$f_0 = 0 \qquad f_1 = 1 \qquad f_{n+2} = f_{n+1} + f_n.$$

- Euklidov algoritmus.

Obecná rekurzia

- Ackermannova funkcia (1928).
- Univerzálna funkcia pre primitívne rekurzívne funkcie.

Primitívne rekurzívne funkcie

Definícia triedy primitívne rekurzívnych funkcií

- Základné funkcie:
  - konštantná funkcia $\mathsf{Z}(x) = 0$,
  - funkcia nasledovníka $\mathsf{S}(x) = x + 1$,
  - identity (projekcie):

$$\mathsf{I}_i^n(x_1, \ldots, x_n) = x_i$$

    pre každé $1 \leq i \leq n$.
- Kompozícia (skladanie) funkcií:

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

- Primitívna rekurzia:

$$\begin{aligned} f(0, y_1, \ldots, y_n) &= g(y_1, \ldots, y_n) \\ f\big(\mathsf{S}(x), y_1, \ldots, y_n\big) &= h\big(x, f(x, y_1, \ldots, y_n), y_1, \ldots, y_n\big). \end{aligned}$$

Primitívne rekurzívne funkcie

Definícia triedy primitívne rekurzívnych funkcií

- Základné funkcie:
  - konštantná funkcia $\mathsf{Z}(x) = 0$,
  - funkcia nasledovníka $\mathsf{S}(x) = x + 1$,
  - identity (projekcie):

$$\mathsf{I}_i^n(\vec{x}) = x_i$$

    pre každé $1 \leq i \leq n$.
- Kompozícia (skladanie) funkcií:

$$f(\vec{x}) = h\big(g_1(\vec{x}), \ldots, g_m(\vec{x})\big).$$

- Primitívna rekurzia:

$$\begin{aligned} f(0, \vec{y}) &= g(\vec{y}) \\ f\big(\mathsf{S}(x), \vec{y}\big) &= h\big(x, f(x, \vec{y}), \vec{y}\big). \end{aligned}$$

Primitívne rekurzívne funkcie

Definícia

- Základné funkcie: $\mathsf{Z}(x) = 0$, $\mathsf{S}(x) = x + 1$, $\mathsf{I}_i^n(\vec{x}) = x_i$.
- Kompozícia funkcií: $f(\vec{x}) = h(g_1(\vec{x}), \ldots, g_m(\vec{x}))$.
- Primitívna rekurzia:

$$f(0, \vec{y}) = g(\vec{y}) \qquad f(\mathsf{S}(x), \vec{y}) = h(x, f(x, \vec{y}), \vec{y}).$$

Sčítanie

$$0 + y = y$$
$$x + 1 + y = x + y + 1$$

$$h(x, z, y) = \mathsf{S}\,\mathsf{I}_2^3(x, z, y)$$
$$0 + y = \mathsf{I}(y)$$
$$\mathsf{S}(x) + y = h(x, x + y, y)$$

Primitívne rekurzívne funkcie

Definícia

- Základné funkcie: $\mathsf{Z}(x) = 0$, $\mathsf{S}(x) = x + 1$, $\mathsf{I}_i^n(\vec{x}) = x_i$.
- Kompozícia funkcií: $f(\vec{x}) = h(g_1(\vec{x}), \ldots, g_m(\vec{x}))$.
- Primitívna rekurzia:

$$f(0, \vec{y}) = g(\vec{y}) \qquad f(\mathsf{S}(x), \vec{y}) = h(x, f(x, \vec{y}), \vec{y}).$$

Násobenie

$$0 \cdot y = 0$$
$$(x + 1) \cdot y = x \cdot y + y$$

$$h_2(x, z, y) = \mathsf{I}_2^3(x, z, y) + \mathsf{I}_3^3(x, z, y)$$
$$0 \cdot y = \mathsf{Z}(y)$$
$$\mathsf{S}(x) \cdot y = h_2(x, x \cdot y, y)$$

Primitívne rekurzívne funkcie

Definícia

- Základné funkcie: $\mathsf{Z}(x) = 0$, $\mathsf{S}(x) = x + 1$, $\mathsf{I}_i^n(\vec{x}) = x_i$.
- Kompozícia funkcií: $f(\vec{x}) = h(g_1(\vec{x}), \ldots, g_m(\vec{x}))$.
- Primitívna rekurzia:

$$f(0, \vec{y}) = g(\vec{y}) \qquad f(\mathsf{S}(x), \vec{y}) = h(x, f(x, \vec{y}), \vec{y}).$$

Umocňovanie

$$x^0 = 1$$

$$x^{y+1} = x \cdot x^y$$

$$\mathsf{C}_1(x) = \mathsf{S}\,\mathsf{Z}(x)$$

$$h_3(y, z, x) = \mathsf{I}_3^3(y, z, x) \cdot \mathsf{I}_2^3(y, z, x)$$

$$f(0, x) = \mathsf{C}_1(x)$$

$$f(\mathsf{S}(y), x) = h_3(y, f(y, x), x)$$

$$x^y = f(I_2^2(x, y), I_1^2(x, y))$$

Primitívne rekurzívne funkcie

Primitívne rekurzívne funkcie a deklaratívne programovanie

- Základný vývoj primitívne rekurzívnych funkcií.
- Primitívne rekurzívne predikáty a ohraničená minimalizácia.
- Párovacia funkcia a aritmetizácia dátových štruktúr.
- Vnorená jednoduchá rekurzia:

$$f(0, y) = g(y)$$
$$f(x+1, y) = h\Big(x, f\big(x, s_1(x, y)\big), f\big(x, s_2(x, y, f(x, s_1(x, y))), y\Big).$$

- Regulárne rekurzívne definície s mierou:

$$f(\vec{x}) = \tau[f; \vec{x}].$$

Podmienka regularity $\Gamma_{f(\vec{\rho})} \to \mu[\vec{\rho}] < \mu[\vec{x}]$ pre rekurzívne volanie $f(\vec{\rho})$ funkcie f v terme τ.

Záver

Organizácia kurzu

- Prednášky: pondelok 10:40-12:10, m. I-9.
- Cvičenia: streda 13:10-14:40, m. I-H6.
- Konzultácie: štvrtok 13:00-13:30, m. I-16.
- Web: http://ii.fmph.uniba.sk/cl/courses/

Hodnotenie

- Cvičenia: max. 50 bodov.
- Test, prémiové úlohy: max. 50 bodov.
- Známky: E 50, D 60, C 70, B 80, A 90.

Záver

1. cvičenie

- Začína v stredu o 13:10 v miestnosti I-H6.
- Pracujte vo dvojiciach.
- Úlohy odovzdať najneskôr do 12:00 v pondelok budúci týždeň.

2. prednáška

- Aritmetizácia dátových štruktúr (karteziánsky súčin, obecné stromy) s pomocou párovacej funkcie.
- Poza primitívnu rekurziu: univerzálna funkcia pre primitívne rekurzívne funkcie.