

1-AIN-625 Úvod do matematickej logiky pre  
programátorov

1-INF-450 Logika pre informatikov

Zimný semester 2012/13

1. prednáška

Ján Komara

# Obsah 1. prednášky

Cieľ a obsah predmetu

Deklaratívne programovanie

Primitívne rekurzívne funkcie

Záver

# 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 obecnými 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
  - ▶ 2-AIN-130 Teória vypočítateľnosti pre programátorov
- ▶ Softvér používaný pri výučbe týchto predmetov:
  - ▶ programovací jazyk a špecifikačno-verifikačný systém CL (Clausal Language).
- ▶ Iné deklaratívne programovacie 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

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

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

- ▶ Idea algoritmu

$$x > y \wedge z \mid y \rightarrow z \mid x \leftrightarrow z \mid x \div y.$$

# Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

## Implementácia

```
gcd(x, y) = if x ≠ 0 ∧ y ≠ 0 then  
    case  
        x > y ⇒ gcd(x ÷ y, y)  
        x = y ⇒ x  
        x < y ⇒ gcd(x, y ÷ x)  
    end  
else  
    max(x, y)
```

## Výpočet

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

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

# Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

## Implementácia

```
gcd(x, y) = if x ≠ 0 ∧ y ≠ 0 then  
    case  
        x > y ⇒ gcd(x ÷ y, y)  
        x = y ⇒ x  
        x < y ⇒ gcd(x, y ÷ x)  
    end  
else  
    max(x, y)
```

## Výpočet

Podmienky regularity zaručujú, že výpočet vždy skončí:

$$x \neq 0 \wedge y \neq 0 \wedge x > y \rightarrow \max(x \div y, y) < \max(x, y)$$

$$x \neq 0 \wedge y \neq 0 \wedge x < y \rightarrow \max(x, y \div x) < \max(x, y).$$



# Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

## Implementácia

```
gcd(x, y) = if  $x \neq 0 \wedge y \neq 0$  then  
    case  
         $x > y \Rightarrow \text{gcd}(x \div y, y)$   
         $x = y \Rightarrow x$   
         $x < y \Rightarrow \text{gcd}(x, y \div x)$   
    end  
else  
     $\max(x, y)$ 
```

## Výpočet

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

$$\text{gcd}(9, 12) = \text{gcd}(9, 3) = \text{gcd}(6, 3) = \text{gcd}(3, 3) = 3$$

# Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Alternatívna implementácia so vstupnou podmienkou

```
 $x \neq 0 \wedge y \neq 0 \rightarrow \text{gcd}(x, y) = \text{case}$   
     $x > y \Rightarrow \text{gcd}(x \div y, y)$   
     $x = y \Rightarrow x$   
     $x < y \Rightarrow \text{gcd}(x, y \div x)$   
end
```

## Výpočet

Ten nemusí skončiť:

$$\text{gcd}(1, 0) = \text{gcd}(1 \div 0, 0) = \text{gcd}(1, 0) = \dots$$

Program počíta čiastočnú funkciu!

# Deklaratívne programovanie

Euklidov algoritmus pre výpočet najväčšieho spoločného deliteľa

Alternatívna implementácia so vstupnou podmienkou

```
 $x \neq 0 \wedge y \neq 0 \rightarrow \text{gcd}(x, y) = \text{case}$   
     $x > y \Rightarrow \text{gcd}(x \div y, y)$   
     $x = y \Rightarrow x$   
     $x < y \Rightarrow \text{gcd}(x, y \div x)$   
end
```

## Výpočet

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

$$x \neq 0 \wedge y \neq 0 \wedge x > y \rightarrow \max(x \div y, y) < \max(x, y) \wedge x \div y \neq 0 \wedge y \neq 0$$

$$x \neq 0 \wedge y \neq 0 \wedge x < y \rightarrow \max(x, y \div x) < \max(x, y) \wedge x \neq 0 \wedge y \div x \neq 0.$$

Výpočet skončí pre vstupy spĺňajúce vstupnú podmienku.

# Deklaratívne programovanie

## Zhrnutie

- ▶ *Univerzum* je množina prirodzených čísel  $N = \{0, 1, 2, 3, \dots\}$ .
- ▶ *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:

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

### Rekurzia s mierou

- ▶ Fibonacciho postupnosť:

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

- ▶ 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  $Z(x) = 0$ ,
- ▶ funkcia nasledovníka  $S(x) = x + 1$ ,
- ▶ identity (projekcie):

$$I_i^n(x_1, \dots, x_n) = x_i$$

pre každé  $1 \leq i \leq n$ .

- ▶ Kompozícia (skladanie) funkcií:

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

- ▶ Primitívna rekurzia:

$$f(0, y_1, \dots, y_n) = g(y_1, \dots, y_n)$$
$$f(S(x), y_1, \dots, y_n) = h(x, f(x, y_1, \dots, y_n), y_1, \dots, y_n).$$

# Primitívne rekurzívne funkcie

## Definícia triedy primitívne rekurzívnych funkcií

- ▶ Základné funkcie:

- ▶ konštantná funkcia  $Z(x) = 0$ ,
- ▶ funkcia nasledovníka  $S(x) = x + 1$ ,
- ▶ identity (projekcie):

$$I_i^n(\vec{x}) = x_i$$

pre každé  $1 \leq i \leq n$ .

- ▶ Kompozícia (skladanie) funkcií:

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

- ▶ Primitívna rekurzia:

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

# Primitívne rekurzívne funkcie

## Definícia

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

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

## Sčítanie

$$\begin{array}{ll} 0 + y = y & h(x, z, y) = S I_2^3(x, z, y) \\ x + 1 + y = x + y + 1 & 0 + y = I(y) \\ & S(x) + y = h(x, x + y, y) \end{array}$$



# Primitívne rekurzívne funkcie

## Definícia

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

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

## Násobenie

$$\begin{array}{ll} 0 \cdot y = 0 & h_2(x, z, y) = I_2^3(x, z, y) + I_3^3(x, z, y) \\ (x + 1) \cdot y = x \cdot y + y & 0 \cdot y = Z(y) \\ & S(x) \cdot y = h_2(x, x \cdot y, y) \end{array}$$

# Primitívne rekurzívne funkcie

## Definícia

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

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

## Umocňovanie

$$C_1(x) = SZ(x)$$

$$h_3(y, z, x) = I_3^3(y, z, x) \cdot I_2^3(y, z, x)$$

$$f(0, x) = C_1(x)$$

$$f(S(y), x) = h_3(y, f(y, x), x)$$

$$x^0 = 1$$

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

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

# Primitívne rekurzívne funkcie

## Primitívne rekurzívne funkcie a deklaratívne programovanie

- ▶ Základný vývoj primitívne rekurzívnych funkcií.
- ▶ Primitívne rekurzívne predikáty a ohraničená minimalizácia.
- ▶ Párovacia funkcia a aritmetizácia (Gödelizácia).
- ▶ Vnorená jednoduchá rekurzia:

$$f(0, y) = g(y)$$

$$f(x + 1, y) = h\left(x, f(x, s_1(x, y)), f(x, s_2(x, y, f(x, s_1(x, y))))\right), y).$$

- ▶ Regulárne rekurzívne definície s mierou:

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

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

# Záver

## Organizácia kurzu

- ▶ Prednášky: utorok 13:10-14:40, m. I-9.
- ▶ Cvičenia: utorok 14:50-16:20, m. F1-248.
- ▶ Konzultácie: streda 13:00-13:30, m. I-16.
- ▶ Web: <http://ii.fmph.uniba.sk/cl/courses/>

## Hodnotenie

- ▶ Cvičenia 20% – max. 20 bodov.
- ▶ Test, prémiové úlohy 20% – max. 20 bodov.
- ▶ Požiadavka priebežného semestrálneho hodnotenia: dosiahnuť aspoň 20 bodov.
- ▶ Ústna skúška 60%.
- ▶ Znamky: E 50%, D 60%, C 70%, B 80%, A 90%.

# Záver

## 1. cvičenie

- ▶ Začína hneď po prednáške v miestnosti F1-248.
- ▶ 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.