Paradigmata programování 2

Přednáška 10. Výpočty v λ-kalkulu

Michal Krupka

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Obsah

Výrazy λ-kalkulu

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x)` *M*`)`	`(lambda (x)` *M*`)`
aplikace	$(M\ N)$	`(`*M N*`)`	`(funcall` *M N*`)`

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x)` *M*`)`	`(lambda (x)` *M*`)`
aplikace	$(M\ N)$	`(`*M N*`)`	`(funcall` *M N*`)`

Zjednodušení:

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x)` *M*`)`	`(lambda (x)` *M*`)`
aplikace	$(M\ N)$	`(`*M* *N*`)`	`(funcall` *M* *N*`)`

Zjednodušení:

- vynecháváme závorky

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x)` *M*`)`	`(lambda (x)` *M*`)`
aplikace	$(M\ N)$	`(`*M N*`)`	`(funcall` *M N*`)`

Zjednodušení:

- vynecháváme závorky
- aplikace více výrazů:

$$((M\ N)\ L) \equiv M\ N\ L$$

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x)` *M*`)`	`(lambda (x)` *M*`)`
aplikace	$(M\ N)$	`(`*M N*`)`	`(funcall` *M N*`)`

Zjednodušení:

- vynecháváme závorky
- aplikace více výrazů:

$$((M\ N)\ L) \equiv M\ N\ L$$

- vše za tečkou je tělo:

$$\lambda x.M\lambda y.NK \equiv \lambda x.(M\lambda y.(NK))$$

Výrazy λ-kalkulu

	λ-kalkul	Scheme	Lisp
proměnná	$x, y, z, \ldots$	`x`, `y`, `z`, ...	`x`, `y`, `z`, ...
abstrakce	$(\lambda x.M)$	`(lambda (x) M)`	`(lambda (x) M)`
aplikace	$(M\ N)$	`(M N)`	`(funcall M N)`

Zjednodušení:

- vynecháváme závorky
- aplikace více výrazů:

$$((M\ N)\ L) \equiv M\ N\ L$$

- vše za tečkou je tělo:

$$\lambda x.M\lambda y.NK \equiv \lambda x.(M\lambda y.(NK))$$

- currying:

$$\lambda xy.M \equiv \lambda x.\lambda y.M$$

β-redukce

β-redukce

Podvýrazy výrazu lze redukovat:

$$(\lambda x.M)N \to M[x := N].$$

- $(\lambda x.M)N$ je redex
- $M[x := N]$ — volné výskyty proměnné x v M se nahradí N

Normální formy

Normální forma: neobsahuje redex

Normální formy

Normální forma: neobsahuje redex

Hlavová normální forma: může obsahovat redex, ale nechceme ho redukovat:

1. protože je v těle abstrakce (funkce)
2. protože je v argumentu aplikace, ale hlavu nelze redukovat na abstrakci

Model vyhodnocení (redukční strategie)

Model vyhodnocení (redukční strategie)

V jakém pořadí redukovat?

Model vyhodnocení (redukční strategie)

V jakém pořadí redukovat?

Aplikativní vyhodnocovací model.

- Redukujeme zevnitř ven.
- U aplikace nejprve hlavu, pak (pokud je to abstrakce) argument, nakonec β-redukce.

Model vyhodnocení (redukční strategie)

V jakém pořadí redukovat?

Aplikativní vyhodnocovací model.

- Redukujeme zevnitř ven.
- U aplikace nejprve hlavu, pak (pokud je to abstrakce) argument, nakonec β-redukce.

Normální vyhodnocovací model.

- Redukujeme zvenku dovnitř.
- U aplikace nejprve hlavu, pak (pokud je to abstrakce) β-redukce.

Obsah

Výpočty

Výrazy λ-kalkulu a jejich redukce zdánlivě popisují jen výpočetní proces bez konkrétního obsahu, tj. bez konkrétních výpočtů. Uvidíme, že to ale není pravda, protože v λ-kalkulu lze modelovat všechny prvky, které model výpočtu musí obsahovat: větvení, práci s čísly a rekurzi.

Všechno, co na této přednášce probíráme teoreticky, můžete vyzkoušet na interpretu Scheme s líným vyhodnocováním, který máte k dispozici.

Interpret v základu pracuje stejně, jako funkcionální programovací jazyky s líným vyhodnocováním (Haskell).

Logika: TRUE, FALSE, IF

Logika: TRUE, FALSE, IF

Označení:

$$\texttt{TRUE} := \lambda xy.x$$
$$\texttt{FALSE} := \lambda xy.y$$
$$\texttt{IF} := \lambda pca.pca$$

Logika: TRUE, FALSE, IF

Označení:

$$\begin{aligned} \texttt{TRUE} &:= \lambda xy.x \\ \texttt{FALSE} &:= \lambda xy.y \\ \texttt{IF} &:= \lambda pca.pca \end{aligned}$$

Vychází

$$\begin{aligned} \texttt{IF}\,\texttt{TRUE}\,MN &\equiv (\lambda pca.pca)\texttt{TRUE}\,MN \to \texttt{TRUE}\,MN \to (\lambda xy.x)MN \to M \\ \texttt{IF}\,\texttt{FALSE}\,MN &\equiv (\lambda pca.pca)\texttt{FALSE}\,MN \to \texttt{FALSE}\,MN \to (\lambda xy.y)MN \to N \end{aligned}$$

Logika: TRUE, FALSE, IF

Označení:

$$\begin{aligned} \texttt{TRUE} &:= \lambda xy.x \\ \texttt{FALSE} &:= \lambda xy.y \\ \texttt{IF} &:= \lambda pca.pca \end{aligned}$$

Vychází

$$\begin{aligned} \texttt{IF}\,\texttt{TRUE}\,MN &\equiv (\lambda pca.pca)\texttt{TRUE}\,MN \to \texttt{TRUE}\,MN \to (\lambda xy.x)MN \to M \\ \texttt{IF}\,\texttt{FALSE}\,MN &\equiv (\lambda pca.pca)\texttt{FALSE}\,MN \to \texttt{FALSE}\,MN \to (\lambda xy.y)MN \to N \end{aligned}$$

(Normální vyhodnocovací model)

Logika: TRUE, FALSE, IF

Označení:

$$\begin{aligned} \texttt{TRUE} &:= \lambda xy.x \\ \texttt{FALSE} &:= \lambda xy.y \\ \texttt{IF} &:= \lambda pca.pca \end{aligned}$$

Vychází

$$\begin{aligned} \texttt{IF}\,\texttt{TRUE}\,MN &\equiv (\lambda pca.pca)\texttt{TRUE}\,MN \to \texttt{TRUE}\,MN \to (\lambda xy.x)MN \to M \\ \texttt{IF}\,\texttt{FALSE}\,MN &\equiv (\lambda pca.pca)\texttt{FALSE}\,MN \to \texttt{FALSE}\,MN \to (\lambda xy.y)MN \to N \end{aligned}$$

(Normální vyhodnocovací model)
V normálním vyhodnocovacím modelu můžeme definovat IF jako funkci.

Logika: TRUE, FALSE, IF

Označení:

$$\begin{aligned} \texttt{TRUE} &:= \lambda xy.x \\ \texttt{FALSE} &:= \lambda xy.y \\ \texttt{IF} &:= \lambda pca.pca \end{aligned}$$

Vychází

$$\begin{aligned} \texttt{IF}\,\texttt{TRUE}\,MN &\equiv (\lambda pca.pca)\texttt{TRUE}\,MN \to \texttt{TRUE}\,MN \to (\lambda xy.x)MN \to M \\ \texttt{IF}\,\texttt{FALSE}\,MN &\equiv (\lambda pca.pca)\texttt{FALSE}\,MN \to \texttt{FALSE}\,MN \to (\lambda xy.y)MN \to N \end{aligned}$$

(Normální vyhodnocovací model)
V normálním vyhodnocovacím modelu můžeme definovat IF jako funkci.
Jako funkce můžeme definovat i odvozené logické operace.

Páry a seznamy: CONS, CAR, CDR

Páry a seznamy: CONS, CAR, CDR

Označení:

Páry a seznamy: CONS, CAR, CDR

Označení:

$$
\begin{aligned}
[M, N] &:= \lambda z.zMN \\
\texttt{CONS} &:= \lambda xyz.zxy \\
\texttt{CAR} &:= \lambda p.p\,\texttt{TRUE} \\
\texttt{CDR} &:= \lambda p.p\,\texttt{FALSE}
\end{aligned}
$$

Páry a seznamy: CONS, CAR, CDR

Označení:

$$
\begin{aligned}
[M, N] &:= \lambda z.zMN \\
\texttt{CONS} &:= \lambda xyz.zxy \\
\texttt{CAR} &:= \lambda p.p\,\texttt{TRUE} \\
\texttt{CDR} &:= \lambda p.p\,\texttt{FALSE}
\end{aligned}
$$

Vychází

Páry a seznamy: CONS, CAR, CDR

Označení:

$$
\begin{aligned}
[M, N] &:= \lambda z.zMN \\
\texttt{CONS} &:= \lambda xyz.zxy \\
\texttt{CAR} &:= \lambda p.p\,\texttt{TRUE} \\
\texttt{CDR} &:= \lambda p.p\,\texttt{FALSE}
\end{aligned}
$$

Vychází

$$
\begin{aligned}
\texttt{CONS}\,MN &\equiv (\lambda xyz.zxy)MN \to \lambda z.zMN \equiv [M, N] \\
\texttt{CAR}[M, N] &\equiv (\lambda p.p\,\texttt{TRUE})(\lambda z.zMN) \to (\lambda z.zMN)\,\texttt{TRUE} \to \texttt{TRUE}\,MN \to M \\
\texttt{CDR}[M, N] &\equiv (\lambda p.p\,\texttt{FALSE})(\lambda z.zMN) \to (\lambda z.zMN)\,\texttt{FALSE} \to \texttt{FALSE}\,MN \to N
\end{aligned}
$$

Páry a seznamy: CONS, CAR, CDR

Označení:

$$
\begin{aligned}
[M, N] &:= \lambda z.zMN \\
\texttt{CONS} &:= \lambda xyz.zxy \\
\texttt{CAR} &:= \lambda p.p\,\texttt{TRUE} \\
\texttt{CDR} &:= \lambda p.p\,\texttt{FALSE}
\end{aligned}
$$

Vychází

$$
\begin{aligned}
\texttt{CONS}\,MN &\equiv (\lambda xyz.zxy)MN \to \lambda z.zMN \equiv [M, N] \\
\texttt{CAR}[M, N] &\equiv (\lambda p.p\,\texttt{TRUE})(\lambda z.zMN) \to (\lambda z.zMN)\,\texttt{TRUE} \to \texttt{TRUE}\,MN \to M \\
\texttt{CDR}[M, N] &\equiv (\lambda p.p\,\texttt{FALSE})(\lambda z.zMN) \to (\lambda z.zMN)\,\texttt{FALSE} \to \texttt{FALSE}\,MN \to N
\end{aligned}
$$

V normálním vyhodnocovacím modelu jsou páry a seznamy automaticky „líné“.
(viz příklad seznamu všech přirozených čísel)

Čísla

Čísla

Označení:

Čísla

Označení:

$$
\begin{aligned}
0 &:= \lambda x.x \\
1 &:= [\texttt{FALSE}, 0] \\
2 &:= [\texttt{FALSE}, 1] \\
&\vdots \\
n+1 &:= [\texttt{FALSE}, n]
\end{aligned}
$$

Čísla

Označení:

$$
\begin{aligned}
0 &:= \lambda x.x \\
1 &:= [\texttt{FALSE}, 0] \\
2 &:= [\texttt{FALSE}, 1] \\
&\vdots \\
n+1 &:= [\texttt{FALSE}, n]
\end{aligned}
$$

Test na nulu:

Čísla

Označení:

$$
\begin{aligned}
0 &:= \lambda x.x \\
1 &:= [\texttt{FALSE}, 0] \\
2 &:= [\texttt{FALSE}, 1] \\
&\vdots \\
n+1 &:= [\texttt{FALSE}, n]
\end{aligned}
$$

Test na nulu:

$$\texttt{ZERO?} := \texttt{CAR}$$

Čísla

Označení:

$$
\begin{aligned}
0 &:= \lambda x.x \\
1 &:= [\texttt{FALSE}, 0] \\
2 &:= [\texttt{FALSE}, 1] \\
&\vdots \\
n+1 &:= [\texttt{FALSE}, n]
\end{aligned}
$$

Test na nulu:

$$\texttt{ZERO?} := \texttt{CAR}$$

Vychází

Čísla

Označení:

$$
\begin{aligned}
0 &:= \lambda x.x \\
1 &:= [\texttt{FALSE}, 0] \\
2 &:= [\texttt{FALSE}, 1] \\
&\vdots \\
n+1 &:= [\texttt{FALSE}, n]
\end{aligned}
$$

Test na nulu:

$$\texttt{ZERO?} := \texttt{CAR}$$

Vychází

$$\texttt{ZERO?}\ 0 \equiv \texttt{CAR}\ \lambda x.x \equiv (\lambda p.p\ \texttt{TRUE})\ \lambda x.x \to (\lambda x.x)\ \texttt{TRUE} \to \texttt{TRUE}$$

Pro $n > 0$:

$$\texttt{ZERO?}\ n \equiv \texttt{CAR}\ [\texttt{FALSE}, n-1] \to \texttt{FALSE}$$

Následník a předchůdce čísla

Následník a předchůdce čísla

Označení:

Následník a předchůdce čísla

Označení:

$$\texttt{SUCC} := \lambda n.\texttt{CONS}\,\texttt{FALSE}\,n$$

$$\texttt{PRED} := \texttt{CDR}$$

Následník a předchůdce čísla

Označení:

$$\texttt{SUCC} := \lambda n.\texttt{CONS}\,\texttt{FALSE}\,n$$

$$\texttt{PRED} := \texttt{CDR}$$

Vychází:

Následník a předchůdce čísla

Označení:

$$\texttt{SUCC} := \lambda n.\texttt{CONS}\,\texttt{FALSE}\,n$$
$$\texttt{PRED} := \texttt{CDR}$$

Vychází:

$$\texttt{SUCC}\,n \equiv (\lambda n.\texttt{CONS}\,\texttt{FALSE}\,n)\,n \to \texttt{CONS}\,\texttt{FALSE}\,n \to [\texttt{FALSE}, n] \equiv n+1$$
$$\texttt{PRED}\,0 \equiv (\lambda p.p\,\texttt{FALSE})\,0 \to 0\,\texttt{FALSE} \equiv (\lambda x.x)\,\texttt{FALSE} \to \texttt{FALSE}$$

Pro $n > 0$:

$$\texttt{PRED}\,n \equiv \texttt{CDR}\,[\texttt{FALSE}, n-1] \to n-1$$

Sčítání

Sčítání

Na sčítání čísel potřebujeme rekurzi:

Sčítání

Na sčítání čísel potřebujeme rekurzi:

$$\texttt{+} = \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Sčítání

Na sčítání čísel potřebujeme rekurzi:

$$+ = \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(+\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Ve skutečnosti nepotřebujeme „$=$“ (ať už to znamená cokoliv), ale stačí „$\rightarrow$“:

$$+ \rightarrow \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(+\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Sčítání

Na sčítání čísel potřebujeme rekurzi:

$$\texttt{+} = \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Ve skutečnosti nepotřebujeme „$=$“ (ať už to znamená cokoliv), ale stačí „$\rightarrow$“:

$$\texttt{+} \rightarrow \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

I tak ale je použita rekurze v tom smyslu, že výraz + se má redukovat na větší výraz, který + obsahuje jako podvýraz.

Sčítání

Na sčítání čísel potřebujeme rekurzi:

$$\texttt{+} = \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Ve skutečnosti nepotřebujeme „$=$“ (ať už to znamená cokoliv), ale stačí „$\rightarrow$“:

$$\texttt{+} \rightarrow \lambda ab.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

I tak ale je použita rekurze v tom smyslu, že výraz + se má redukovat na větší výraz, který + obsahuje jako podvýraz.

Tohoto efektu lze docílit pomocí *Y-kombinátoru*.

Y-kombinátor

(Též *Curryho paradoxní kombinátor*)

Y-kombinátor

(Též *Curryho paradoxní kombinátor*)

$$\mathtt{Y} := \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

Y-kombinátor

(Též *Curryho paradoxní kombinátor*)

$$\mathtt{Y} := \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

Y-kombinátor má zvláštní vlastnost, že jeho aplikace se redukováním zvětšuje, a to velmi užitečným způsobem:

$$\begin{aligned}\mathtt{Y}f &\equiv (\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)))\, f \to (\lambda x.f(xx))(\lambda x.f(xx)) \\ &\to f\,(\lambda x.f(xx))(\lambda x.f(xx)) \equiv f\,(\mathtt{Y}f)\end{aligned}$$

Y-kombinátor

(Též *Curryho paradoxní kombinátor*)

$$\mathtt{Y} := \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

Y-kombinátor má zvláštní vlastnost, že jeho aplikace se redukováním zvětšuje, a to velmi užitečným způsobem:

$$\begin{aligned}\mathtt{Y}f &\equiv (\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)))\, f \to (\lambda x.f(xx))(\lambda x.f(xx)) \\ &\to f\,(\lambda x.f(xx))(\lambda x.f(xx)) \equiv f\,(\mathtt{Y}f)\end{aligned}$$

Výraz $\mathtt{Y}f$ se redukuje na výraz, který obsahuje výraz $\mathtt{Y}f$ jako podvýraz. Na příkladu sčítání ukážeme, jak lze tuto vlastnost využít k modelování rekurzivního výpočtu.

Sčítání znovu

Sčítání znovu

Položíme

Sčítání znovu

Položíme

$$\texttt{\%+} := \lambda f a b.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(f\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$
$$\texttt{+} := \texttt{Y}\ \texttt{\%+}$$

Sčítání znovu

Položíme

$$\begin{aligned} \texttt{\%+} &:= \lambda f a b.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(f\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b)) \\ \texttt{+} &:= \texttt{Y}\;\texttt{\%+} \end{aligned}$$

Vychází

$$\texttt{+} \equiv \texttt{Y}\;\texttt{\%+} \to \texttt{\%+}\,(\texttt{Y}\;\texttt{\%+}) \equiv \texttt{\%+}\;\texttt{+} \to \lambda a b.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

Sčítání znovu

Položíme

$$\begin{aligned} \texttt{\%+} &:= \lambda f a b.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(f\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b)) \\ \texttt{+} &:= \texttt{Y}\;\texttt{\%+} \end{aligned}$$

Vychází

$$\texttt{+} \equiv \texttt{Y}\;\texttt{\%+} \to \texttt{\%+}\,(\texttt{Y}\;\texttt{\%+}) \equiv \texttt{\%+}\;\texttt{+} \to \lambda a b.\texttt{IF}\,(\texttt{ZERO?}\,b)\,a\,(\texttt{+}\,(\texttt{SUCC}\,a)\,(\texttt{PRED}\,b))$$

A to je to, co jsme potřebovali!

Porovnání čísel

Podobně:

$$\texttt{\%<=} \equiv \lambda f a b.\texttt{IF}\,(\texttt{ZERO?}\,a)\,\texttt{TRUE}\,(\texttt{IF}\,(\texttt{ZERO?}\,b)\,\texttt{FALSE}\,(f\,(\texttt{PRED}\,a)\,(\texttt{PRED}\,b)))$$

$$\texttt{<=} \equiv \texttt{Y}\,\texttt{\%<=}$$