INE 5416/5636 - Paradigmas de programação

Turmas 04208/08238

Prof. Dr. João Dovicchi – dovicchi@inf.ufsc.br

http://www.inf.ufsc.br/˜dovicchi

Alonso Church e Stephen Kleene

- Problemas da computabilidade (início do séc. XX)
- Teoria da recursividade (Stephen Kleene)
- Base para linguagem funcional (Alonso Church)

Church, Alonzo ”An unsolvable problem of elementary number theory”, American Journal of Mathematics, 58 (1936) pp 345-363.

Church, Alonzo ”The Calculi of Lambda-Conversion” 1941.

Definição

O cálculo-λ consiste em regras simples de transformação (substituição de argumentos ou variáveis) que pode representar qualquer função computável por meio deste formalismo.

Definição

O cálculo-λ consiste em regras simples de transformação (substituição de argumentos ou variáveis) que pode representar qualquer função computável por meio deste formalismo.

Equivale a uma máquina de Turing onde a transformação é mais importante do que a máquina em si (mais *software* do que *hardware*)

Ex: se encontrar `x` na expressão, troque pelo valor da aplicação; prox. substituição.

Definição

O cálculo-λ consiste em regras simples de transformação (substituição de argumentos ou variáveis) que pode representar qualquer função computável por meio deste formalismo.

Equivale a uma máquina de Turing onde a transformação é mais importante do que a máquina em si (mais *software* do que *hardware*)

Ex: se encontrar $\mathtt{x}$ na expressão, troque pelo valor da aplicação; prox. substituição.

Pode ser considerada como uma linguagem universal de programação.

Lógica combinatória

Elementos combinatórios suficientes para a declaração de predicados arbitrários de primeira ordem que podem ser expressos sem o uso de variáveis dependentes.

Abstração: λ-termo $\cdot$ Expressão
Cálculo-λ: Abstração (Aplicação)

Lógica combinatória

Elementos combinatórios suficientes para a declaração de predicados arbitrários de primeira ordem que podem ser expressos sem o uso de variáveis dependentes.

Abstração: λ-termo $\cdot$ Expressão
Cálculo-λ: Abstração (Aplicação)

Cálculo de abstração puramente funcional e aplicação de funções:

λ-termo	Expressão	Aplicação
λx	$x^2 + 2x$	3

Exemplo:

Suponha uma variável x, uma expressão E, uma aplicação A e uma abstração da variável, onde $x = A$. Então podemos escrever:

```
let x = A em E
```

ou seja, aplica-se A em toda ocorrência livre de x em E.

Exemplo:

Suponha uma variável x, uma expressão E, uma aplicação A e uma abstração da variável, onde $x = A$. Então podemos escrever:

```
let x = A em E
```

ou seja, aplica-se A em toda ocorrência livre de x em E.

P. ex.:
Se $E = x^2 + xy + z$ e $A = 3$, então a abstração será $3^2 + 3y + z$.

Na matemática tradicional podemos formular a abstração de outra forma:

$$f(x) = x^2 + xy + z$$

para qualquer aplicação em $f(x)$ resulta na substituição de x pela aplicação.

Cálculo Lambda

Na matemática tradicional podemos formular a abstração de outra forma:

$$f(x) = x^2 + xy + z$$

para qualquer aplicação em $f(x)$ resulta na substituição de x pela aplicação.

No cálculo lambda escrevemos:

$$\lambda x.x^2 + xy + z$$

Exemplo:

Suponha que, na abstração:

$$\lambda x.x^2 + xy + z$$

apliquemos y, ou seja, cada ocorrência livre de x deve ser substituída por y:

$$\lambda x.x^2 + xy + z \ (y)$$

então a expressão se reduz a:

$$2y^2 + z$$

Exemplo:

Suponha que, na abstração:

$$\lambda x.x^2 + xy + z$$

apliquemos y, ou seja, cada ocorrência livre de x deve ser substituída por y:

$$\lambda x.x^2 + xy + z\ (y)$$

então a expressão se reduz a:

$$2y^2 + z$$

Como representar isto no modelo $f(x)$ da matemática?

Redução Lambda

Se:

$$f(x) = x^2 + 4 \equiv \lambda x.x^2 + 4$$

a aplicação de $f(2)$ pode ser escrita como:

$$\lambda x.x^2 + 4\ (2)$$

que se reduz pela regra de substituição de toda ocorrência livre de x por 2:

$$2^2 + 4 = 8$$

Regras

1. Se x é um identificador, então x é uma expressão do cálculoλ.
2. Se x é um identificador e E uma expressão , $\lambda x.E$ é uma expressão do cálculo-λ chamada abstração (x é o identificador que delimita a abstração e a expressão E é o corpo da abstração).
3. Se E e A são expressões do cálculo lambda, então EA é uma expressão do cálculo lambda chamada aplicação. A é o operador e E é o operando.
4. Uma expressão lambda é formada pela aplicação múltipla (recursiva) e finita das regras 1 a 3 .

Cálculo Lambda

Um simples identificador pode ser uma expressão lambda. Por ex.:

$$\lambda x.x$$

Define a função identidade. Neste caso:

$$\lambda x.x\ (y) \Rightarrow y$$

ou seja, a aplicação de y à função identidade, retorna y. Então:

$$\lambda x.x \equiv \lambda y.y \equiv \lambda z.z \ldots$$

Variáveis dependentes e independentes

Uma variável é livre (independente), se for diferente do termo-λ. Por exemplo, em:

$$\lambda x.xy$$

y não está vinculado à abstração λx.

Variáveis dependentes e independentes

Uma variável é livre (independente), se for diferente do termo-λ. Por exemplo, em:

$$\lambda x.xy$$

y não está vinculado à abstração λx.

em:

$$\lambda x.xy \; \lambda y.y$$

O primeiro y em λx é livre, enquanto o outro y em λy é dependente.

Mudança de domínio

Seja a abstração:

$$\lambda x.2x + 3$$

Podemos aplicá-a ao domínio de z^2:

$$\lambda x.2x + 3\ (z^2) \Rightarrow 2z^2 + 3$$

Reduções

As expressões lambda são computadas por redução lambda, ou seja,

Redução λ: Combinação das reduções α, β e η.

Por exemplo:

$$\lambda y.y^2 \ (\lambda x.(x^3 + 2) \ (2)) \ \Rightarrow \ \lambda y.y^2 \ (2^3 + 2)$$

$$\lambda y.y^2 \ (10) \ \Rightarrow \ 10^2 \ \Rightarrow \ 100$$

ou, alternativamente (currying):

$$\lambda y.y^2 \ (\lambda x.(x^3 + 2) \ (2)) \ \Rightarrow \ (\lambda x.x^3 + 2 \ (2))^2$$

$$\lambda x.(x^3 + 2)^2 \ (2) \ \Rightarrow \ (2^3 + 2)^2 \ \Rightarrow \ 100$$

Exercício em sala

Calcule a abstração:

$$\lambda x.x + 2(\lambda y.y^2 + 2y + 1\ (3))$$

Exercício em sala

Calcule a abstração:

$$\lambda x.x + 2(\lambda y.y^2 + 2y + 1\ (3))$$

$$\begin{aligned}
\lambda x.(x+2)(\lambda y.(y^2+2y+1)\ (3)) &\Rightarrow \lambda x.(x+2)\ (3^2+6+1) \\
\lambda x.(x+2)\ (3^2+6+1) &\Rightarrow \lambda x.(x+2)\ (16) \\
(16+2) &= 18
\end{aligned}$$

Redução α

Uma expressão pode ser reduzida a outra pela renomeação de um identificador limitado, por qualquer outro identificador que não esteja contido na expressão. (Redução α).
Por exemplo, a abstração

$$\lambda x.(\lambda y.xy)y \Rightarrow \lambda y.yy$$

pode ser confusa, pois o segundo y pode ser independente, pode ser aplicação ou pode ser dependente. A substutuição α pode ajudar a compreensão:

$$\lambda x.(\lambda t.xt)y \Rightarrow \lambda t.yt$$

Redução α

Uma expressão pode ser reduzida a outra pela renomeação de um identificador limitado, por qualquer outro identificador que não esteja contido na expressão. (Redução α).
Por exemplo, a abstração

$$\lambda x.(\lambda y.xy)y \Rightarrow \lambda y.yy$$

pode ser confusa, pois o segundo y pode ser independente, pode ser aplicação ou pode ser dependente. A substutuição α pode ajudar a compreensão:

$$\lambda x.(\lambda t.xt)y \Rightarrow \lambda t.yt$$

Exemplos:

$$\lambda x.x \rightarrow \lambda y.y$$

$$\lambda x.2x + z \rightarrow \lambda y.2y + z$$

Redução

Exemplo numérico:

$$\lambda x.(\lambda z.(z + x)(4)) \Rightarrow \lambda x.(4 + x)$$

renomeação errada de x para z:

$$\lambda x.(\lambda z.(z + x)(4)) \rightarrow \lambda z.(\lambda z.(z + z)(4))$$

$$\lambda z.(\lambda z.(z + z)(4)) \Rightarrow \lambda z.8$$

Redução

Exemplo numérico:

$$\lambda x.(\lambda z.(z+x)(4)) \Rightarrow \lambda x.(4+x)$$

renomeação errada de x para z:

$$\lambda x.(\lambda z.(z+x)(4)) \rightarrow \lambda z.(\lambda z.(z+z)(4))$$

$$\lambda z.(\lambda z.(z+z)(4)) \Rightarrow \lambda z.8$$

Considere uma aplicação 1 e teremos no primeiro caso:

$$\lambda x.(\lambda z.z+x(4))(1) \ = \ 5$$

e, no segundo caso:

$$\lambda z.(\lambda z.z+z(4))(1) \ = \ 8$$

Redução

Uma subexpressão da forma $\lambda x.E(A)$ pode ser reduzida pela substituição de uma cópia de E na qual toda ocorrência livre de x é substituída por A, desde que isto não resulte em que qualquer identificador livre de A torne-se um delimitador. (Redução β)

Exemplo:

$$\lambda x.2x + 1(3) \Rightarrow 2 \times 3 + 1 = 7$$

$$\lambda x.(\lambda y.xy)(n) \Rightarrow \lambda y.ny$$

$$\lambda x.(\lambda y.xy(n)) \Rightarrow \lambda x.xn$$

Exemplo:

$$\begin{aligned}
\lambda y.(\lambda x.(\lambda z.z + x(3))(y))(1) &\Rightarrow \lambda y.(\lambda z.z + y(3))(1) \\
\lambda y.(\lambda z.z + y(3))(1) &\Rightarrow \lambda y.3 + y(1) \\
\lambda y.3 + y(1) &\Rightarrow 3 + 1 = 4
\end{aligned}$$

Exercício em sala

Seja `quad` uma função que toma o seu argumento e retorna o seu quadrado, definida para os números inteiros:

```
quad :: Int -> Int
quad n = n * n
```

Leia-se `quad` é do tipo que recebe um Int e retorna Int

Calcule a abstração lambda:

$$\lambda f.(\lambda x.(f(f(x)))3)(\texttt{quad})$$

Exercício em sala

$$\lambda f.(\lambda x.(f(f(x)))3)(\texttt{quad})$$

$$\lambda f.(\lambda x.(f(f(x)))3)(\texttt{quad}) \Rightarrow \lambda x.(\texttt{quad}(\texttt{quad}(x)))3 \quad (1)$$

$$\lambda x.(\texttt{quad}(\texttt{quad}(x)))3 \Rightarrow \texttt{quad}(\texttt{quad}(3)) \quad (2)$$

$$\texttt{quad}(\texttt{quad}(3)) \Rightarrow \texttt{quad}(9) = 81 \quad (3)$$

Reduções

Redução η: usa o conceito de extensionalidade (duas funções notadas de forma diferente produzem o mesmo resultado para a mesma entrada). A redução η é usada para eliminar redundâncias nas abstrações lambda.

Por exemplo:

$$\mathrm{Se} \lambda x.Mx \xrightarrow{\eta} M$$

desde que x não seja livre em M podemos usar a redução η. E podemos escrever uma abstração η usando-se o raciocínio inverso:

$$M \xrightarrow{\eta} \lambda x.Mx$$

Forma normal

Chamamos de forma normal o máximo de reduções de uma abstração lambda. Por ex.:

$$
\begin{aligned}
\lambda x.x\ (y) &\underset{\beta}{\rightarrow} y \\
\lambda y.fy\ (a) &\underset{\beta}{\rightarrow} fa \\
\lambda x.xx\ (y) &\underset{\beta}{\rightarrow} yy
\end{aligned}
$$

Exemplo

Considere a abstração lambda:

$$\lambda x.xx\ (\lambda y.y)$$

Cada ocorrência livre de x em λx será substituida por $\lambda y.y$. Então a abstração se reduz a:

$$\lambda y.y\ \lambda y.y \Rightarrow \lambda y.y$$

Exemplo

Considere a abstração lambda:

$$\lambda x.xx\ (\lambda y.y)$$

Cada ocorrência livre de x em λx será substituida por $\lambda y.y$. Então a abstração se reduz a:

$$\lambda y.y\ \lambda y.y \Rightarrow \lambda y.y$$

Faça a redução à forma normal de: $\lambda x.xx\ \lambda x.xx$

Exemplo

Considere a abstração lambda:

$$\lambda x.xx\ (\lambda y.y)$$

Cada ocorrência livre de x em λx será substituida por $\lambda y.y$. Então a abstração se reduz a:

$$\lambda y.y\ \lambda y.y \Rightarrow \lambda y.y$$

Faça a redução à forma normal de: $\lambda x.xx\ \lambda x.xx$

$$\lambda x.xx\ \lambda x.xx \Rightarrow \lambda x.xx\ \lambda x.xx$$

Roteiros da aula prática 04