Lambda Calculus

Recursão

Prof. Edson Alves

Faculdade UnB Gama

2020

Sumário

Recursão

- A recursão diz respeito a definição de uma função em termos de si mesma
- Ao contrário de outras notações, o cálculo λ não permite esta definição diretamente, uma vez que os termos-λ são anônimos
- Uma maneira de contornar isso é utilizar uma expressão-λ que receba a si mesma como argumento
- Além disso, é preciso lidar com os dois aspectos fundamentais de uma função recursiva: o(s) caso(s) base(s) e a chamada recursiva

Estrutura básica da recursão

$$f(x) = \begin{cases} g(x_0), & \text{se } P(x_0), \\ h(x, f(x)), & \text{caso contrário} \end{cases}$$

$P(x)$ é um predicado que retorna verdadeiro se x_0 é o valor que caracteriza o caso base. Se $P(x)$ for verdadeira, o valor de f em x_0 será dado pela função g; se $P(x)$ falsa, $f(x)$ é igual a uma função h que depende de x e de $f(x)$.

Representação da estrutura básica da recursão no cálculo-λ

$$F \equiv (\lambda fx.(Px)(gx)(hx(fx)))$$

Observe que na definição da função recursiva F é utilizado o termo-λ I_F: se o predicado (Px) retornar verdadeiro, o retorno será o primeiro parâmetro (gx), que corresponde ao valor de F para o caso base; se falso, será avaliada a função h, que tem como parâmetros x e (fx).

Não há garantias, contudo, que $F \equiv f$, pois no cálculo λ os termos são anônimos. É preciso, portanto, definir um termo que garanta esta equivalência.

Teorema do Ponto Fixo

Teorema do Ponto Fixo

Para qualquer termo-λ F existe um termo X tal que $FX \equiv X$.

Demonstração

Seja F um termo-λ qualquer. Defina $W \equiv (\lambda x.F(xx))$ e $X = WW$. Deste modo,

$$X \equiv WW \equiv (\lambda x.F(xx))W \equiv F(WW) \equiv FX$$

Combinador $\mathbf{Y}$

Proposição (Combinador $\mathbf{Y}$)

O **combinador $\mathbf{Y}$**

$$\mathbf{Y} \equiv \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

é um termo-λ tal que, para qualquer termo F,

$$\mathbf{Y}F \equiv F(\mathbf{Y}F)$$

Demonstração

Seja F um termo-λ qualquer. Daí

$$\begin{aligned}
\mathbf{Y}F &\equiv (\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)))F \\
&\equiv (\lambda x.F(xx))(\lambda x.F(xx)) \\
&\equiv F((\lambda x.F(xx))(\lambda x.F(xx))) \\
&\equiv F(\lambda f.(\lambda x.f(xx))(\lambda x.f(xx))F) \equiv F(\mathbf{Y}F)
\end{aligned}$$

Observações sobre o combinador $\mathbf{Y}$

- Veja que, para qualquer termo-λ F, $\mathbf{Y}F$ é um ponto fixo de F
- Esta propriedade é o que faltava para a definição completa da recursão, pois ao aplicar $(\mathbf{Y}F)$ ao parâmetro x da recursão, o resultado é
$$(\mathbf{Y}F)x \equiv F(\mathbf{Y}F)x,$$
ou seja, o termo F é aplicado aos parâmetros $\mathbf{Y}F$ e x, o que permite invocar F novamente quantas vezes forem necessárias
- Assim, para definir uma função recursiva $\mathbf{Y}F$ no cálculo-λ, basta determinar o predicado P e as funções g e h que compõem a função F

Exemplo de recursão: fatorial

$$F(n) = \begin{cases} 1, & \text{se } n = 0, \\ nF(n-1), & \text{caso contrário} \end{cases}$$

Neste caso, $P \equiv Z, g \equiv \lambda x.1$ e $h \equiv \lambda fx. \times n(f(Pn))$, onde $\times ab$ é a multiplicação dos naturais a e b, e Pn é o predecessor do natural n.

Deste modo,

$$F \equiv \lambda fx.(Zx)1(\times n(f(Pn)))$$

Exemplo de aplicação do fatorial

$$
\begin{aligned}
3! \equiv (\mathbf{Y}F)3 &\equiv F(\mathbf{Y}F)3 \\
&\equiv (\lambda fx.(Zx)1(\times n(f(Pn))))(\mathbf{Y}F)3 \\
&\equiv (Z3)1(\times 3((\mathbf{Y}F)(P3))) \\
&\equiv F1(\times 3((\mathbf{Y}F)(P3))) \\
&\equiv \times 3((\mathbf{Y}F)2)) \equiv \times 3(F(\mathbf{Y}F)2)) \\
&\equiv \times 3((Z2)1(\times 2((\mathbf{Y}F)(P2)))) \\
&\equiv \times 3(\times 2((\mathbf{Y}F)1)) \equiv \times 3(\times 2(F(\mathbf{Y}F)1)) \\
&\equiv \times 3(\times 2((Z1)1(\times 1((\mathbf{Y}F)(P1))) \\
&\equiv \times 3(\times 2(\times 1((\mathbf{Y}F)0))) \\
&\equiv \times 3(\times 2(\times 1((Z0)1(\times 0((\mathbf{Y}F)(P0)))))) \\
&\equiv \times 3(\times 2(\times 1(1))) \\
&\equiv \times 3(\times 2(1)) \\
&\equiv \times 3(2) \equiv 6
\end{aligned}
$$

Referências

1. **ROJAS**, Raúl. *A Tutorial Introduction to the Lambda Calculus*, FU Berlin, WS-97/98.
2. **BARENDREGT**, Henk; **BARENDSEN**, Erik. *Introduction to Lambda Calculus*, March 2000.
3. Wikipédia. Combinatory logic, acesso em 07/01/2020.
4. Wikipédia. Lambda calculus, acesso em 03/01/2020.