Teoria da Computação

Funções Recursivas

Cristiano Lehrer, M.Sc.

ybadoo

Introdução (1/2)

- Tipos de formalismos usados para especificar algoritmos:
  - Operacional:
    - Define-se uma máquina abstrata, baseada em estados, em instruções primitivas e na especificação de como cada instrução modifica cada estado. Uma máquina abstrata deve ser suficientemente simples para não permitir dúvidas sobre a sua computação.
  - Axiomático:
    - Associam-se regras aos componentes da linguagem. As regras permitem afirmar o que será verdadeiro após a ocorrência de cada cláusula, considerando-se o que era verdadeiro antes da ocorrência.
  - Denotacional ou Funcional:
    - Trata-se de uma função construída a partir de funções elementares de forma composicional (horizontalmente) no sentido em que o algoritmo denotado pela função pode ser determinado em termos de suas funções componentes.

Introdução (2/2)

- Exemplos dos tipos de formalismos:
  - Operacional:
    - Máquina de Turing.
    - Máquina de Post.
    - Máquina com Pilhas.
  - Axiomático:
    - Gramáticas Regulares.
    - Gramáticas Livre de Contexto.
    - Gramáticas Sensíveis ao Contexto.
    - Gramáticas Irrestritas.
  - Denotacional ou Funcional:
    - Funções Recursivas Parciais de Kleene (1936).
    - Linguagem Lambda de Alonzo Church (1941).

Linguagem Lambda

- Formalismo apresentado por Alonzo Church, em 1941, também conhecida como λ-Linguagem.
- É uma função anônima (sem nome), formada por uma sequência de padrões representando os argumentos da função, e um corpo que especifica como o resultado pode ser calculado usando os argumentos.
- Abstração Lambda – abstração da definição da função:
  - $\lambda x.x^3 + 4$
- Aplicação Lambda – determinação do valor da função aplicada a um dado parâmetro:
  - $(\lambda x.x^3 + 4)(2)$

Termos da Linguagem Lambda

- Termos da Linguagem Lambda são denotados por letras maiúsculas M, N, P, …
- Exemplo:
  - $(\lambda x.x^3 + 4)(2)$
  - M denota $x^3 + 4$ $(\lambda x.M)(2)$
  - N denota $\lambda x.x^3 + 4$
  - P denota 2 $(N)(P)$

Regra de Redução Beta (1/2)

- $(\lambda x.M)(P) = [P/x]M$, ou seja, substituição de x por P em M
  - $(\lambda x.x^3 + 4)(2)$
  - $= [2/x]\ x^3 + 4$
  - $= 2^3 + 4$
  - $= 12$

Regra de Redução Beta (2/2)

- $(\lambda k.(\lambda x.x^k)(5))(2)$
  - $= [2/k](\lambda x.x^k)(5)$
  - $= (\lambda x.x^2)(5)$
  - $= [5/x](x^2)$
  - $= 5^2$
  - $= 25$
- $(\lambda k.(\lambda x.x^k))(5)(2)$
  - $= [5/k](\lambda x.x^k)(2)$
  - $= (\lambda x.x^5)(2)$
  - $= [2/x]\ x^5$
  - $= 2^5$
  - $= 32$

Termo Lambda

- A Linguagem Lamba é constituída por termos Lambda.
  - Sejam X e C conjuntos contáveis de variáveis e constantes, respectivamente.
  - Um Termo Lambda, Expressão Lambda ou Palavra Lambda sobre X e C é indutivamente definido por:
    - Toda variável $x \in X$ é um termo lambda.
    - Toda constante $c \in C$ é um termo lambda.
    - Se M e N são termos lambdas, e x é uma variável, então:
      - (M N) é um termo lambda.
      - (λx.M) é um termo lambda.
  - Uma Linguagem Lambda L sobre X e C é o conjunto de todos os termos lambda sobre esses conjuntos.

Variável Ligada/Livre

- Uma variável em um termo lambda é denominada:
  - Variável ligada, se está dentro do escopo de uma abstração lambda.
  - Variável livre, caso contrário.
- Exemplo:
  - No termo $\lambda x.x^k$, as variáveis x e k são ditas ligada e livre, respectivamente.
  - No termo $\lambda x.\lambda k.x^k$, as variáveis x e k são ditas ligadas.

Semântica de um Termo Lambda

- A semântica de um termo lambda dado por uma função aplicada a um parâmetro é dada pelas aplicações sucessivas, das possíveis regras de redução beta.
- Exemplo:
  - $(\lambda k.(\lambda x.x^k)(5))(2)$ aplicação da regra de redução beta em k
  - $= [2/k](\lambda x.x^k)(5)$ substituição da variável k
  - $= (\lambda x.x^2)(5)$ aplicação da regra de redução beta em x
  - $= [5/x](x^2)$ substituição da variável x
  - $= 5^2$
  - $= 25$

Funções Recursivas de Kleene

- As funções recursivas parciais propostas por Kleene são funções construídas sobre funções básicas, usando três tipos de construções denominadas:
  - Composição
  - Recursão
  - Minimização

Composição

- Suponha as seguintes funções:
  - zero = $\lambda x.0: \mathbb{N} \rightarrow \mathbb{N}$ função constante zero
  - sucessor = $\lambda x.x + 1: \mathbb{N} \rightarrow \mathbb{N}$ função sucessor
  - adição = $\lambda(x, y).x + y: \mathbb{N}^2 \rightarrow \mathbb{N}$ função adição
- As seguintes funções são definidas, usando composição de funções:
  - um = $\lambda x.$sucessor(zero(x)): $\mathbb{N} \rightarrow \mathbb{N}$ função constante um
  - dois = $\lambda x.$sucessor(um(x)): $\mathbb{N} \rightarrow \mathbb{N}$ função constante dois
  - três = $\lambda x.$adição(um(x), dois(x)): $\mathbb{N} \rightarrow \mathbb{N}$ função constante três

Recursão (1/2)

- Suponha as seguintes funções:
  - id = $\lambda x.x: \mathbb{N} \rightarrow \mathbb{N}$ função identidade
  - sucessor = $\lambda x.x + 1: \mathbb{N} \rightarrow \mathbb{N}$ função sucessor
  - $proj3_3 = \lambda(x, y, z).z: \mathbb{N}^3 \rightarrow \mathbb{N}$ função projeção do 3ª componente
- A função adição = $\lambda(x, y).x + y: \mathbb{N}^2 \rightarrow \mathbb{N}$ é definida, usando recursão, como sendo:
  - adição(x, 0) = id(x)
  - adição(x, y + 1) = $proj3_3$(x, y + 1, sucessor(adição(x, y)))

Recursão (2/2)

- Por exemplo, adição(3, 2) é:
  - = adição(3, 2)
  - = $proj3_3$(3, 2, sucessor(adição(3, 1)))
  - = $proj3_3$(3, 2, sucessor($proj3_3$(3, 1, sucessor(adição(3, 0)))))
  - = $proj3_3$(3, 2, sucessor($proj3_3$(3, 1, sucessor(id(3)))))
  - = $proj3_3$(3, 2, sucessor($proj3_3$(3, 1, sucessor(3))))
  - = $proj3_3$(3, 2, sucessor($proj3_3$(3, 1, 4)))
  - = $proj3_3$(3, 2, sucessor(4))
  - = $proj3_3$(3, 2, 5)
  - = 5

Minimização (1/2)

- Por exemplo, a constante $const_{zero}: \rightarrow \mathbb{N}$ é definida usando minimização, como segue:
  - $const_{zero} = min\{y \mid zero(y) = 0\}$
- De fato, o menor natural y tal que zero(y) = 0 é 0.
- Nota-se que é uma função sem variáveis, ou seja, é uma constante.
- Suponha as seguintes funções:
  - $proj2_1 = \lambda(x, y).x: \mathbb{N}^2 \rightarrow \mathbb{N}$ função projeção do 1ª componente
  - $antecessor(0) = const_{zero}$
  - $antecessor(y + 1) = proj2_1(y, antecessor(y))$

Minimização (2/2)

- Por exemplo, antecessor(2) é:
  - = antecessor(2)
  - = $proj2_1(1, antecessor(1))$
  - = $proj2_1(1, proj2_1(0, antecessor(0)))$
  - = $proj2_1(1, proj2_1(0, const_{zero}))$
  - = $proj2_1(1, proj2_1(0, 0))$
  - = $proj2_1(1, 0)$
  - = 1

Função Recursiva Parcial e Total

- Funções recursivas parciais são definidas a partir de três funções básicas sobre o conjunto dos números naturais, como segue:
  - Constante zero
  - Sucessor
  - Projeção
- Mais precisamente, projeção não é uma função, mas uma família de funções, pois depende do número de componentes, bem como de qual componente deseja-se projetar.
- É interessante observar que, somente com estas três funções, bem como com as construções de composição, recursão e minimização, é possível definir qualquer função intuitivamente computável.
- Uma função recursiva total é uma função recursiva parcial definida para todos os elementos do domínio.

Definições Recursivas de Bird (1/2)

- Formalismo de R. Bird para o tratamento do conceito de recursão através de definições recursivas.
- Verifica-se que a Classe das Funções com Definição Recursiva é a mesma Classe das Funções Computáveis.
- Representação:
  - $(p \rightarrow e1, e2)$
    - Informalmente, o valor de tal expressão é e1, se p é verdadeiro, e e2, caso contrário.

Definições Recursivas de Bird (2/2)

- Exemplos:
  - fatorial = λx.(x = 0 → 1, x * fatorial(x - 1))
  - adição = λ(x, y).(x = 0 → y, adição(x - 1, y) + 1)
  - multiplicação = λ(x, y).(x = 0 → 0, adição(multiplicação(x - 1, y), y))
  - potencia = λ(x, y).(y = 0 → 1, multiplicação(potencia(x, y - 1), x))