Lista de Exercícios de Funções Parciais de Kleene

Consultar arquivos:

Lambda_Kleene.pdf

lam2010cap8 12152010com exercícios.pdf

Complementando Cálculo Lambda e Funções Recursivas:

8.2.7 Cálculo Lambda e Computabilidade

O λ-Cálculo é equivalente a qualquer linguagem de programação

Se um λ-termo for visto como um programa, e o processo de redução como sua execução, pode-se associar a cada termo da forma λx.M um programa que computa uma função com entrada x e corpo de comandos M (normalmente dependente de x).

O resultado de tal programa seria o termo obtido após um número finito de reduções e tal que não possa ser mais reduzido

a) O processo de reduções, em alguns casos, não pára;
b) O processo de reduções não é deterministico, pois as reduções não têm ordem preferencial de aplicação.

Teorema 8.20
Redução Cálculo Lambda e Função Computável

Uma função $f: N \rightarrow N$ é computável se e somente se existe um termo lambda F tal que, para qualquer $x, y \in N$:

$F(x) = y$ se e somente se $x =_\beta y$

O teorema estabelece que o formalismo Cálculo Lambda é equivalente ao formalismo Máquina de Turing bem como aos demais formalismos equivalentes estudados.

Complementando Funções Recursivas;

8.3 Funções Recursivas e Ciência da Computação

8.3.1 Importância das Funções Recursivas

- O estudo das funções recursivas e da recursão em geral é de fundamental importância na Ciência da Computação.
- Não só são formalismos tão poderosos como as máquinas universais como fornecem uma abordagem (denotacional) diferente da operacional.
- A quase totalidade das linguagens de programação modernas como Pascal ou C possui recursão como um construtor básico de programas.
- As arquiteturas da maioria dos atuais computadores possuem facilidades para implementar recursão.

8.3.2 Linguagem de Programação Funcional

- aplicações das funções recursivas e do Cálculo Lambda, no contexto da programação funcional, usando como exemplo a linguagem *Haskell*.

- *Programação funcional* é um estilo de programação baseada em funções e composição de funções em vez de comandos.
- um **programa** é uma expressão funcional e não uma seqüência de comandos a serem executados.
- Uma ***linguagem de programação*** é ***funcional*** se suporta esse estilo de programação.
  - operações tipadas (ou seja, com tipos associados)
  - construtores que permitem definir novos tipos e operações mais complexos.
- Uma linguagem de programação funcional considerada *pura* não possui variável e nem atribuições:
- tipos primitivos de dados da linguagem;
- constantes de cada tipo primitivo de dado;
- operações as quais são funções sobre os tipos primitivos de dados da linguagem;
- construtores que permitem definir novos tipos e operações derivados dos tipos e operações da linguagem.

8.4 Conclusões

Neste capítulo foram estudados dois formalismos do tipo denotacional, baseados em funções recursivas:

- Funções Recursivas Parciais de Kleene as quais são funções parciais definidas recursivamente sobre três funções naturais básicas: zero, sucessor e projeção, juntamente com as operações substituição composicional, recursão primitiva e minimização;
- Cálculo Lambda, formalismo para definição de função, aplicação de função e recursão.
  - O cálculo Lambda é uma linguagem e um cálculo.
  - O cálculo objetiva verificar a igualdade de termos da linguagem e pode ser interpretado como um "passo computacional" na busca de um "valor" representativo para um λ-termo qualquer.
  - Portanto, a redução pode ser vista como uma operacionalização do λ-Cálculo.

A importância do Cálculo Lambda transcende o estudo da computabilidade, tendo importantes aplicações no estudo da lógica e das linguagens de programação.

No contexto das linguagens de programação funcionais, uma breve introdução da linguagem Haskell é apresentada.

Ambos formalismos são equivalentes ao formalismo Máquina de Turing bem como aos demais formalismos equivalentes estudados o que significa dizer que formalizaram o que é possível computar em um computador.

1) O que significa : Cálculo Lamba é equivalente à Máquina de Touring.

2) O que significa: Funções parciais recursivas são equivalentes ao formalismo Máquina de Turing. Ou seja,

foi provado que a Classe das Funções Turing-Computáveis era igual à Classe das Funções Recursivas Parciais.

3) Quando uma função é dita ser parcial e quando é total.

4) O que é um funcional

5) Defina as funções básicas abaixo:

a) função sucessor: s(x)

b) função predecessor (anterior): p(x)

c) função projeção (seleção sobre uma lista): $\mathbf{p}^{i}_{n(x1,x2,...xn)}$

6) Usando a função básica sucessor, defina as funções abaixo:

a) soma2(x) = x + 2 ou seja, retorna o valor de x soma com dois.

b) soma5(x) = x + 5 ou seja, retorna o valor de x somado com cinco.

7) Defina a função condicional.

8) Usando a função condicional defina a função maior(x,y) que retorna o maior de dois elementos x e y.

Da mesma forma, defina a função menor: menor(x,y)

9) Descreva o conceito de função recursiva:

10) Defina a função constante zero(x) que retorna a constante zero usando as funções cond (condicional) e Projeção.

11) Usando a função zero(x) e a função s(x), sucessor, implemente as funções abaixo:

a) função constante um(x): retorna o valor constante um.

b) função constante dois(x): retorna o valor constante dois.

c) função constante três(x): retorna o valor constante 3.

d) faça uma segunda definição da função constante três(x) usando a função soma(x,y) e as funções constantes já vistas.

12) Defina a recursão while.

13) Defina as funções abaixo usando a função while recursiva:

a) soma(x,y) = x + y

b) sub(x,y) = x – y

c) mult(x,y) = x * y

d) fat(x) = x!