Funções Recursivas de Kleene

Teoria da Computação

INF05501

Funções Recursivas de Kleene

- Construídas sobre **funções básicas**
- Tais construções ocorrem a partir de **três tipos operações**:
  - Composição
  - Recursão
  - Minimização

Definição de Composição de Funções

Sejam g, f_1, f_2, ..., f_k *funções parciais tais que:*

$g = \lambda(y_1, y_2, ..., y_k).g(y_1, y_2, ..., y_k) : \mathbb{N}^k \rightarrow \mathbb{N}$

$f_i = \lambda(x_1, x_2, ..., x_n).f_i(x_1, x_2, ..., x_n) : \mathbb{N}^n \rightarrow \mathbb{N}$, para $i \in \{1, 2, ..., k\}$

A função parcial h *tal que*

$h = \lambda(x_1, x_2, ..., x_n).h(x_1, x_2, ..., x_n) : \mathbb{N}^n \rightarrow \mathbb{N}$

é a ***composição de funções*** *definida a partir de* g, f_1, f_2, ..., f_k *da seguinte forma:*

$h(x_1, x_2, ..., x_n) = g(f_1(x_1, x_2, ..., x_n), f_2(x_1, x_2, ..., x_n), ..., f_k(x_1, x_2, ..., x_n))$

Composição de Funções

- A função parcial h é dita definida para $(x_1, x_2, ..., x_n)$ sss
  - $f_i(x_1, x_2, ..., x_n)$ é definida para todo $i \in \{1, 2, ..., k\}$
  - $g(f_1(x_1, x_2, ..., x_n), f_2(x_1, x_2, ..., x_n), ..., f_k(x_1, x_2, ..., x_n))$ é definida

- A composição de funções generalizada também é chamada de substituição de funções, visto que obtém-se h a partir da substituição de cada y_i em g por $f_i(x_1, x_2, ..., x_n)$

Exemplo de Composição de Funções

- 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)**

  $soma = \lambda(x, y).x + y : \mathbb{N}^2 \rightarrow \mathbb{N}$ **(função soma)**

- Podemos definir as funções abaixo como composições das funções anteriores:

  $um = \lambda x.sucessor(zero(x)) : \mathbb{N} \rightarrow \mathbb{N}$ **(função constante um)**

  $dois = \lambda x.sucessor(um(x)) : \mathbb{N} \rightarrow \mathbb{N}$ **(função constante dois)**

  $tres = \lambda x.soma(um(x), dois(x)) : \mathbb{N} \rightarrow \mathbb{N}$ **(função constante três)**

Definição de Recursão

Sejam f *e* g *funções parciais tais que:*

$$f = \lambda(x_1, x_2, ..., x_n).f(x_1, x_2, ..., x_n) : \mathbb{N}^n \to \mathbb{N}$$
$$g = \lambda(x_1, x_2, ..., x_n, y, z).g(x_1, x_2, ..., x_n, y, z) : \mathbb{N}^{n+2} \to \mathbb{N}$$

A função parcial h *tal que*

$$h = \lambda(x_1, x_2, ..., x_n, y).h(x_1, x_2, ..., x_n, y) : \mathbb{N}^{n+1} \to \mathbb{N}$$

*é definida por **recursão** a partir de* f *e* g *da seguinte forma:*

$$h(x_1, x_2, ..., x_n, 0) = f(x_1, x_2, ..., x_n)$$
$$h(x_1, x_2, ..., x_n, y+1) = g(x_1, x_2, ..., x_n, y, h(x_1, x_2, ..., x_n, y))$$

Recursão

- A função parcial h é dita definida para $(x_1, x_2, ..., x_n, y)$ sss
  - $f(x_1, x_2, ..., x_n)$ é definida
  - $g(x_1, x_2, ..., x_n, i, h(x_1, x_2, ..., x_n, i))$ é definida para todo $i \in \{1, 2, ..., y\}$

- O conceito de recursão é **fundamental na Ciência da Computação**, sendo que grande parte das linguagens de programação possuem mecanismos de recursão e muitas arquiteturas de computadores modernos apresentam estruturas para realizar recursão eficientemente

Exemplo de Recursão

- Suponha as seguintes funções:

  $id = \lambda x.x : \mathbb{N} \to \mathbb{N}$ **(função identidade)**

  $sucessor = \lambda x.x + 1 : \mathbb{N} \to \mathbb{N}$ **(função sucessor)**

  $proj3_3 = \lambda(x, y, z).z : \mathbb{N}^3 \to \mathbb{N}$ **(função projeção da 3a. componente da tripla)**

Exemplo de Recursão (cont.)

- A função soma nos naturais, tal que:

  $soma = \lambda(x, y).x + y : \mathbb{N}^2 \rightarrow \mathbb{N}$

  é definida usando recursão da seguinte maneira

  $soma(x, 0) = id(x)$

  $soma(x, y + 1) = proj3_3(x, y + 1, sucessor(soma(x, y)))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3, 2)$ é como segue:

  $soma(3, 2) =$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3, 2)$ é como segue:

  $soma(3, 2) =$

  $= proj3_3(3, 2, sucessor(soma(3, 1)))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3, 2)$ é como segue:

$soma(3, 2) =$

$= proj3_3(3, 2, sucessor(soma(3, 1)))$

$= proj3_3(3, 2, sucessor(proj3_3(3, 1, sucessor(soma(3, 0)))))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$$soma(3,2) =$$

$$= proj3_3(3,2,sucessor(soma(3,1)))$$

$$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(soma(3,0)))))$$

$$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(id(3)))))$$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$soma(3,2) =$

$= proj3_3(3,2,sucessor(soma(3,1)))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(soma(3,0)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(id(3)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(3))))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$soma(3,2) =$

$= proj3_3(3,2,sucessor(soma(3,1)))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(soma(3,0)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(id(3)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(3))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,4)))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$soma(3,2) =$

$= proj3_3(3,2,sucessor(soma(3,1)))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(soma(3,0)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(id(3)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(3))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,4)))$

$= proj3_3(3,2,sucessor(4))$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$$soma(3,2) =$$

$$= proj3_3(3, 2, sucessor(soma(3, 1)))$$

$$= proj3_3(3, 2, sucessor(proj3_3(3, 1, sucessor(soma(3, 0)))))$$

$$= proj3_3(3, 2, sucessor(proj3_3(3, 1, sucessor(id(3)))))$$

$$= proj3_3(3, 2, sucessor(proj3_3(3, 1, sucessor(3))))$$

$$= proj3_3(3, 2, sucessor(proj3_3(3, 1, 4)))$$

$$= proj3_3(3, 2, sucessor(4))$$

$$= proj3_3(3, 2, 5)$$

Exemplo de Recursão (cont.)

- Por exemplo, $soma(3,2)$ é como segue:

$soma(3,2) =$

$= proj3_3(3,2,sucessor(soma(3,1)))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(soma(3,0)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(id(3)))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,sucessor(3))))$

$= proj3_3(3,2,sucessor(proj3_3(3,1,4)))$

$= proj3_3(3,2,sucessor(4))$

$= proj3_3(3,2,5)$

$= 5$

Definição de Minimização

Seja f uma função parcial tal que:

$$f = \lambda(x_1, x_2, ..., x_n, y).f(x_1, x_2, ..., x_n, y) : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$$

A função parcial h tal que

$$h = \lambda(x_1, x_2, ..., x_n).h(x_1, x_2, ..., x_n) : \mathbb{N}^n \rightarrow \mathbb{N}$$

*é definida por **minimização** de f tal que:*

$h = \lambda(x_1, ..., x_n).min\{y|f(x_1, ...x_n, y) = 0$
e $(\forall z)$ tal que $(z < y), f(x_1, ..., x_n, z)$ é definida $\}$

Minimização

- Portanto, a função h, para o valor $(x_1, ..., x_n)$, é definida como o **menor natural** y **tal que** $f(x_1, ..., x_n, y) = 0$

- Adicionalmente, a condição

  $(\forall z)$ tal que $(z < y), f(x_1, ..., x_n, z)$ é definida

  garante que é possível determinar, **em um tempo finito**, se, para qualquer valor z menor do que y, $f(x_1, ..., x_n, z)$ é diferente de zero

- Por simplicidade, adota-se a seguinte notação:

  $h = \lambda(x_1, x_2, ..., x_n).min\{y | f(x_1, x_2, ..., x_n, y) = 0\}$

Exemplo de Minimização

- Suponha a função constante $zero = \lambda x.0 : \mathbb{N} \rightarrow \mathbb{N}$

- Considere a função que identifica o número zero nos naturais:

  $const_{zero} :\rightarrow \mathbb{N}$

- Note que $const_{zero}$ é uma **função sem argumentos**, ou seja, é uma **constante**

Exemplo de Minimização (cont.)

- Esta função pode ser definida por minimização da seguinte forma:

  $const_{zero} = min\{y | zero(y) = 0\}$

- De fato, o menor natural y tal que $zero(y) = 0$ é 0

Exemplo de Minimização e Recursão

- Sejam a constante zero $const_{zero}$ e a função projeção

  $proj2_1 = \lambda(x, y).x : \mathbb{N}^2 \rightarrow \mathbb{N}$

- A seguinte função antecessor nos naturais:

  $antecessor = \lambda x.antecessor(x) : \mathbb{N} \rightarrow \mathbb{N}$

  pode ser definida usando recursão (supondo-se que $antecessor(0)$ é 0)

  $antecessor(0) = const_{zero}$

  $antecessor(y + 1) = proj2_1(y, antecessor(y))$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

$= proj2_1(1, proj2_1(0, antecessor(0)))$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

$= proj2_1(1, proj2_1(0, antecessor(0)))$

$= proj2_1(1, proj2_1(0, const_{zero}))$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

$= proj2_1(1, proj2_1(0, antecessor(0)))$

$= proj2_1(1, proj2_1(0, const_{zero}))$

$= proj2_1(1, proj2_1(0, 0))$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

$= proj2_1(1, proj2_1(0, antecessor(0)))$

$= proj2_1(1, proj2_1(0, const_{zero}))$

$= proj2_1(1, proj2_1(0, 0))$

$= proj2_1(1, 0)$

Exemplo de Minimização e Recursão (cont.)

$antecessor(2) =$

$= proj2_1(1, antecessor(1))$

$= proj2_1(1, proj2_1(0, antecessor(0)))$

$= proj2_1(1, proj2_1(0, const_{zero}))$

$= proj2_1(1, proj2_1(0, 0))$

$= proj2_1(1, 0)$

$= 1$

Exercício

- Sejam a constante zero $const_{zero}$ e as seguintes funções:

  $id = \lambda x.x : \mathbb{N} \rightarrow \mathbb{N}$

  $proj3_3 = \lambda(x, y, z).x : \mathbb{N}^3 \rightarrow \mathbb{N}$

- A seguinte função subtração nos naturais:

  $sub = \lambda(x, y).sub(x, y) : \mathbb{N}^2 \rightarrow \mathbb{N}$

  pode ser definida usando recursão:

  $sub(x, 0) = id(x)$

  $sub(x, y + 1) = proj3_3(x, y + 1, antecessor(sub(x, y)))$

- Como se dá o processamento de $sub(3, 2)$?

Funções Recursivas Parciais

- Funções recursivas parciais são definidas a partir de três funções básicas:
  - constante zero
  - sucessor
  - projeção sobre o conjunto dos números naturais

- Note que **projeção não é uma função, mas uma família de funções**, pois depende do número de componentes, bem como de qual a componente que se deseja projetar

Definição de Função Recursiva Parcial

Uma função recursiva parcial é indutivamente definida como segue:

- As seguintes funções são recursivas parciais:

  $zero = \lambda x.0 : \mathbb{N} \rightarrow \mathbb{N}$ **(função constante zero)**

  $sucessor = \lambda x.x + 1 : \mathbb{N} \rightarrow \mathbb{N}$ **(função sucessor)**

  $projn_i = \lambda(x_1, ..., x_n).x_i : \mathbb{N}^n \rightarrow \mathbb{N}$ **(função projeção da i-ésima componente da n-upla)**

- Também são funções recursivas parciais funções construídas a partir destas usando composição, recursão ou minimização

Exemplos de Funções Recursivas Parciais

- A função identidade

  $id = \lambda x.x : \mathbb{N} \rightarrow \mathbb{N}$

  é recursiva parcial, pois pode ser construída como uma função básica de projeção:

  $id = proj1_1$

- Outros exemplos:

  $soma = \lambda(x, y).x + y : \mathbb{N}^2 \rightarrow \mathbb{N}$ (função soma)

  $sub = \lambda(x, y).sub(x, y) : \mathbb{N}^2 \rightarrow \mathbb{N}$ (função subtração)

  $um = \lambda x.sucessor(zero(x)) : \mathbb{N} \rightarrow \mathbb{N}$ (função constante um)

$dois = \lambda x.sucessor(um(x)) : \mathbb{N} \rightarrow \mathbb{N}$ (função constante dois)

$tres = \lambda x.soma(um(x), dois(x)) : \mathbb{N} \rightarrow \mathbb{N}$ (função constante três)

$const_{zero} : \rightarrow \mathbb{N}$ (valor constante zero)

$antecessor = \lambda x.antecessor(x) : \mathbb{N} \rightarrow \mathbb{N}$ (função antecessor)

Definição de Função Recursiva Total

Uma função recursiva total é uma função recursiva parcial definida para todos os elementos do domínio

Teorema: Funções Recursivas X Funções Turing-Computáveis

As seguintes classes de funções são equivalentes:

Funções Recursivas Parciais e Funções Turing-Computáveis

Funções Recursivas Totais e Funções Turing-Computáveis Totais

Logo, a seguinte relação entre classes também pode ser estabelecida:

Funções Recursivas Parciais $\Leftrightarrow$ Linguagens Enumeráveis Recursivamente

Funções Recursivas Totais $\Leftrightarrow$ Funções Turing-Computáveis Totais e Linguagens Recursivas