MATA51 – Teoria da Computação
Profa. Laís Salvador

# TEORIA DAS FUNÇÕES RECURSIVAS - Funções Recursivas Parciais

## INTRODUÇÃO

- As Funções Primitivas Recursivas (F. R. P.) falham em capturar todas as funções que poderíamos considerar como computáveis.

- Um exemplo é a função de Ackerman.

- Segue demonstração por diagonalização.

**Demonstração**

A definição de funções recursivas primitivas implica que cada função pode ser resolvida com um conjunto finito de equações. Ao adotar uma codificação adequada para símbolos de variáveis e funções, numerais, subescritos e sinais de pontuação, é possível expressar qualquer sistema de equações como uma cadeia sore um alfabeto.

Isto é, o conjunto de funções primitivas recursivas é enumerável.

**(A mesma ideia foi utilizada para mostrar que o conjunto de máquinas de Turing é enumerável)**

Para esta demonstração, é suficiente considerar funções de uma variável (unárias).

Suponha então a seguinte lista de todas as F.R.P unárias na forma de strings em ordem lexicográfica: $f_0, f_1, f_2, \dots f_z$

Considere agora a função $g$ definida por:

$$g\,(z) = f_z\,(z) + 1,\ z \in N$$

claramente, $g$ é computável – aplicação do sucessor sobre uma função recursiva primitiva. Mas $g$ não pode estar na enumeração de F.R.P. unárias, senão teremos:

$$g = f_y \text{ para algum } y \in N, \text{ e assim:}$$

$$f_y(y) = g(y) = f_y(y) + 1$$

o que é uma contradição, então $g$ não é recursiva primitiva.

## Funções Recursivas Parciais

A classe das funções recursivas pode ser ampliada com a introdução do μ-operador ou operador de minimização, como base de uma 3ª regra para construção de funções recursivas.

## Minimização:

Seja: $g: N^{n+1} \rightarrow N$ uma função total e computável, não necessariamente primitiva recursiva. Defina a função $f: N^n \rightarrow N$ da seguinte forma:

$$f(x) = \mu\, y\, [g\,(x, y) = 0]$$

É o menor $y \geq 0$ para o qual $g\,(x, y) = 0$.

Então $f$ é obtida de $g$ por minimização.

Obs.: $x = x_1, x_2, ..., x_n$.

A função $f$ pode ser avaliada, para um dado argumento $a$, avaliando-se $g\,(a, y)$ para $y = 0, 1, 2, ...$;

Isto é, computando sequencialmente os valores:

$$g\,(a, 0), g\,(a, 1), g\,(a, 2), ...$$

Se $y_0$ é o 1º valor de $y$ para o qual $g(a, y) = 0$, atribuímos $f(a) = y_0$.

Exercício:

Construir um procedimento efetivo para computar $f(a)$:

f(a)

$$\left[\begin{array}{l} \quad\quad y_0 = 0 \\ while\ g(a, y_0) \neq 0\ do \\ \quad\quad y_0 = y_0 + 1 \\ return\ y_{0.} \end{array}\right.$$

- Qual a diferença entre procedimento efetivo e algoritmo?
- Como para algumas escolhas de *a* pode não existir valor de y tal que *g(a, y) = 0*, é possível que a sequencia de computações pode ser executada indefinidamente sem produzir o valor de *f(a)*. Portanto *f* não necessita ser uma função total.
  - Como veremos no **Exemplo (1)**
- Por esse motivo, as funções caracterizadas pela adição do operador de minimização aos construtores das funções recursivas primitivas são denominadas Funções Recursivas Parciais.

**Exemplo (1)**:

Consideremos a função recursiva primitiva:

$$g\ (x, y) = x + y$$

Ao aplicar o operador de minimização temos:

$$f\ (x) = \mu\ y\ [\ g\ (x, y) = 0\ ]$$

$$f(x) = \begin{cases} 0\ , se\ x = 0 \\ \uparrow (indefinida), se\ x > 0 \end{cases}$$

Logo f(x) é uma Função Recursiva Parcial (função parcial).

**<u>Exemplo (2):</u>**

A função $max(x, y)$ é recursiva primitiva pois

$max\ (x, y) = y + (x \sim y)$

Ela também pode ser expressa usando o μ-operador:

$$max(x, y) = \mu z\ [\ (x \sim z) + (y \sim z) = 0]$$

A função $g\ (x, y, z) = (x \sim z) + (y \sim z)$ é zero se, somente se, $z = max\ (x, y)$.

Para avaliar $max\ (x, y)$, avalia-se $g(x, y, z)$ para $z = 0, 1, 2, \ldots$ até se obter o valor $0$.

Exemplo: max (2, 4) repetir as seguintes avaliações:

$$g\ (2, 4, 0) = 6$$
$$g\ (2, 4, 1) = 4$$
$$g\ (2, 4, 2) = 2$$
$$g\ (2, 4, 3) = 1$$
$$g\ (2, 4, 4) = 0$$

O argumento $z_0 = 4$ é o menor $z$ tal que $g(2, 4, z) = 0$, portanto, $max\ (2, 4) = 4$.

➔ A função $max\ (x, y)$ é total.

**EXERCÍCIOS:**

1) Seja g (x, y) = $|x - y^2|$
   1.1) Prove que g (x, y) é primitiva recursiva.
   1.2) Aplique o operador de minimização sobre *g*. Qual é a função obtida? É uma função total?
2) Construa um algoritmo para calcular o max(x,y) com base na sua definição com o μ-operador.

## Referências


1. **Modelos Clássicos de Computação.** Flavio Soares Correa da Silva e Ana Cristina Vieira de Melo. Cengage, 2010.
2. **Elementos de teoria da computação.** LEWIS, Harry R.; PAPADIMITRIOU, Christos H. Bookman, 2000.
3. **Machines, languages, and computation.** Peter J. Denning, Jack Bonnell Dennis, Joseph E. Qualitz
   Prentice-Hall, 1978.