TEORIA DA COMPUTAÇÃO I

Aula 03 – Máquinas

Prof. Dr. Guilherme Pina Cardim

guilhermecardim@fai.com.br

28 de fevereiro de 2020

Máquinas

O objetivo de uma *máquina* é suprir todas as informações necessárias para que a computação de um programa possa ser descrita.

(Diverio e Menezes, 2011)

Cabe à *máquina* suprir as funções de entrada e saída, o armazenamento e recuperação de informações na memória e dar significado semântico aos identificadores das operações e testes.

(Diverio e Menezes, 2011)

Máquinas

Uma máquina é definida por uma 7-upla ordenada:

$$\boldsymbol{M = (V, X, Y, \pi_X, \pi_Y, \Pi_F, \Pi_T)}$$

- V – Conjunto de valores de memória;
- X – Conjunto de valores de entrada;
- Y – Conjunto de valores de saída;
- π_X – Função de entrada, tal que: $\pi_X: X \longrightarrow V$;
- π_Y – Função de saída, tal que: $\pi_Y: V \longrightarrow Y$;
- Π_F – Conjunto de interpretações de operações. Para cada **F** interpretada por **M**, existe uma única função:

$$\pi_F: V \longrightarrow V \text{ em } \Pi_F$$

- Π_T – Conjunto de interpretações de testes. Para cada **T** interpretado por **M**, existe uma única função:

$$\pi_T: V \longrightarrow \{verdadeiro, falso\} \text{ em } \Pi_F$$

Máquinas

- **Exemplo:** *Máquina de dois registradores.*
  - Considerando uma máquina com duas posições de memória (registradores **a** e **b**) que assumem valores em **N**, com duas operações e um teste:
    - subtração de **1** em **a**, se **a > 0**;
    - adição de **1** em **b**;
    - teste se **a** é **zero**.
  - A entrada é constituída de um único valor armazenado em **a**, zerando **b** e a saída retorna o valor de **b**.

Máquinas

$$dois\_reg = (N^2, N, N, armazena\_a, retorna\_b, \{subtrai\_a, adiciona\_b\}, \{a\_zero\})$$

onde:

N^2 é conjunto de valores de memória
N é conjunto de valores de entrada, bem como o de saída

$armazena\_a: N \to N^2$ é a função de entrada tal que, $\forall n \in N$:

$$armazena\_a(n) = (n, 0)$$

$retorna\_b: N^2 \to N$ é a função de saída tal que, $\forall (n, m) \in N^2$:

$$retorna\_b(n, m) = m$$

$subtrai\_a: N^2 \to N^2$ é interpretação tal que, $\forall (n, m) \in N^2$:

$$subtrai\_a(n, m) = (n\text{-}1, m), \text{ se } n \neq 0; \; subtrai\_a(n, m) = (0, m), \text{ se } n = 0$$

$adiciona\_b: N^2 \to N^2$ é interpretação tal que, $\forall (n, m) \in N^2$:

$$adiciona\_b(n, m) = (n, m+1)$$

$a\_zero: N^2 \to \{ \text{verdadeiro}, \text{falso} \}$ é interpretação tal que, $\forall (n, m) \in N^2$:

$$a\_zero(n, m) = \text{verdadeiro}, \text{ se } n = 0; \; a\_zero(n, m) = \text{falso}, \text{ se } n \neq 0$$

Máquinas

- **Exemplo:** *Máquina de dois registradores.*

Programa iterativo itv_b←a
até **a_zero** faça (**subtrai_a**; **adiciona_b**)

Programa recursivo rec_b←a
rec_b←a é *R* onde *R* def (se **a_zero** então ✓ senão *S*;*R*), *S* def **subtrai_a**; **adiciona_b**

Computação

A *computação* de um programa é o histórico das instruções executadas e o correspondente valor de memória.

(Diverio e Menezes, 2011)

A *computação* de um programa pode ser finita ou infinita.

(Diverio e Menezes, 2011)

Computação

Programa monolítico mon_b←a
1: se **a_zero** então vá_para 9 senão vá_para 2
2: faça **subtrai_a** vá_para 3
3: **adiciona_b** vá_para 1

(1,(2,0))	instrução inicial e valor de entrada armazenado
(2,(2,0))	em 1, como **a≠0**, desviou para 2
(3,(1,0))	em 2, subtraiu do registrador **a** e desviou para 3
(1,(1,1))	em 3, adicionou no registrador **b** e desviou para 1
(2,(1,1))	em 1, como **a≠0**, desviou para 2
(3,(0,1))	em 2, subtraiu do registrador **a** e desviou para 3
(1,(0,2))	em 3, adicionou no registrador **b** e desviou para 1
(9,(0,2))	em 1, como **a=0**, desviou para 9

A **computação** foi finalizada, ou seja é **finita**.

Computação

Programa monolítico add_b
1: faça **adiciona_b** vá_para 1

(1,(2,0))	instrução inicial e valor de entrada armazenado
(1,(2,1))	adicionou no registrador **b** e permanece em 1
(1,(2,2))	adicionou no registrador **b** e permanece em 1
(1,(2,3))	adicionou no registrador **b** e permanece em 1
...	repete 1 indefinidamente

Neste caso a **computação** se torna **infinita**.

Função Computada

$$\boldsymbol{um\_reg} = (\boldsymbol{N}, \boldsymbol{N}, \boldsymbol{N}, \boldsymbol{id_N}, \boldsymbol{id_N}, \{\boldsymbol{ad}, \boldsymbol{sub}\}, \{\boldsymbol{zero}\})$$

Onde:

- $\boldsymbol{N}$ corresponde aos conjuntos de valores de memória, entrada e saída
- $\boldsymbol{id_N}: \boldsymbol{N} \longrightarrow \boldsymbol{N}$ é a função de entrada e de saída
- $\boldsymbol{ad}: \boldsymbol{N} \longrightarrow \boldsymbol{N}$ é a interpretação tal que,

$$\forall n \in N, \boldsymbol{ad}(\boldsymbol{n}) = \boldsymbol{n} + \boldsymbol{1}$$

- $\boldsymbol{sub}: \boldsymbol{N} \longrightarrow \boldsymbol{N}$ é a interpretação tal que,

$\forall n \in N: \boldsymbol{sub}(\boldsymbol{n}) = \boldsymbol{n} - \boldsymbol{1}$, se $n \neq 0$; ou $\boldsymbol{sub}(\boldsymbol{n}) = \boldsymbol{0}$ se $n = 0$

- $\boldsymbol{zero}: \boldsymbol{N} \longrightarrow \{\boldsymbol{verdadeiro}, \boldsymbol{falso}\}$ é a interpretação tal que,

$$\forall n \in N: \boldsymbol{zero}(\boldsymbol{n}) = \boldsymbol{verdadeiro}, \text{ se } n = 0$$

ou $\mathbf{zero}(\boldsymbol{n}) = \boldsymbol{falso}$ se $n \neq 0$

Função Computada

Programa recursivo duplica
duplica é R onde R def (se **zero** então ✓ senão **sub**;**R**;**ad**;**ad**)

- Qual a função computada do programa duplica na máquina um_reg quando o valor de entrada for 3?

- O conceito de **função computada** está diretamente ligado com a **computação** de um programa **P** em uma máquina **M** para determinado valor de entrada.

Função Computada

A *função computada* de um programa é a resposta (valor de saída) obtida após a computação finita de um programa associado a uma determinada entrada.

(Diverio e Menezes, 2011)

$$\langle P, M\rangle : X \longrightarrow Y$$

$$\langle duplica, um\_reg\rangle : N \longrightarrow N$$

$$\langle duplica, um\_reg\rangle(n) : 2n$$

Relações de Equivalência

Equivalência Forte de Programas
Dois programas possuem equivalência forte se as correspondentes funções computadas coincidem para qualquer máquina.

- Sejam **P** e **Q** dois programas arbitrários, então o par (P, Q) terá uma relação do tipo *equivalência forte de programas* se, e somente se, para qualquer máquina **M**, as funções parciais computadas sejam iguais:

$$P \equiv Q \iff \langle P, M \rangle = \langle Q, M \rangle$$

Relações de Equivalência

Programa monolítico P1
1: se **T** então vá_para 2 senão vá_para 5 2: faça **F** vá_para 1

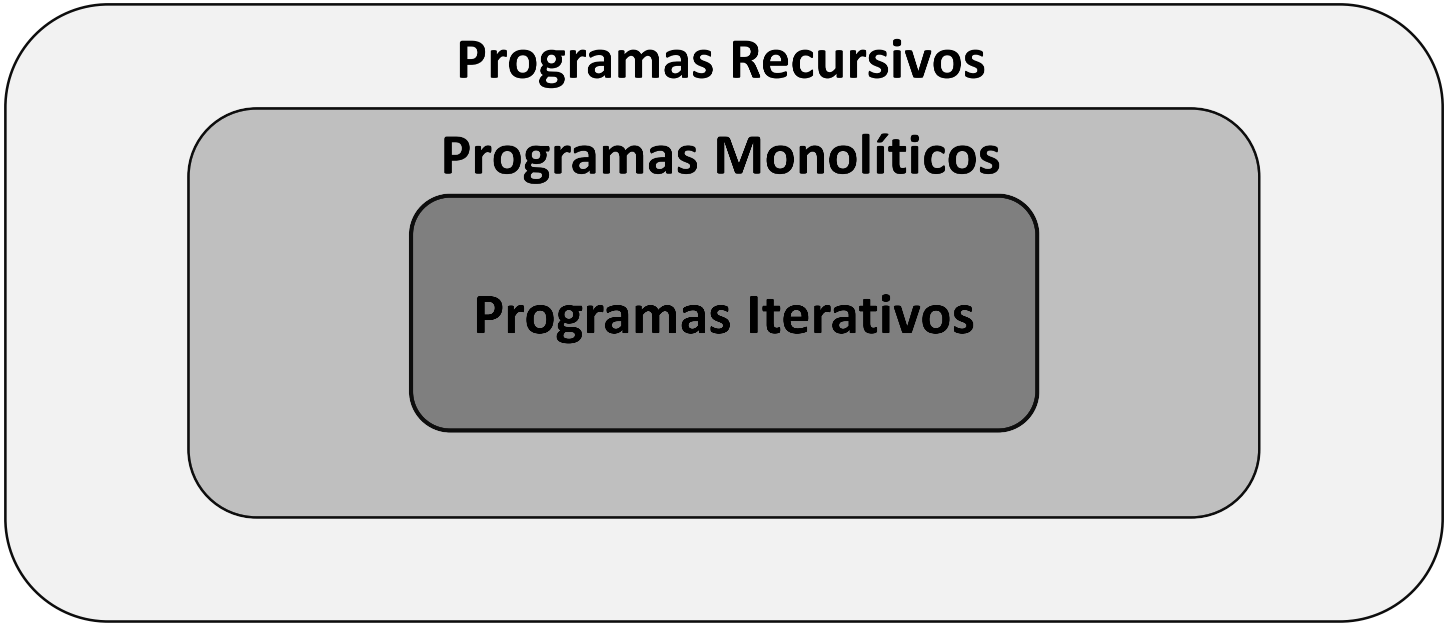
Programa iterativo P2
enquanto **T**   faça **F**

Programa recursivo P3
P3 é R onde   R def (se **T** então **F;R** senão ✓)

$$P1 \equiv P2 \equiv P3$$

Relações de Equivalência

- Para todo programa iterativo, existe um programa monolítico fortemente equivalente;
- Para todo programa monolítico existe um programa recursivo fortemente equivalente.



Relações de Equivalência

Equivalência de Programas (em uma Máquina)
Dois programas possuem equivalência em uma máquina se as correspondentes funções computadas coincidem para uma determinada máquina.

- Sejam **P** e **Q** dois programas arbitrários, então o par (P, Q) terá uma relação do tipo *equivalência de programas* se, e somente se, para determinada máquina **M**, as funções computadas sejam iguais:

$$P \equiv_M Q \iff \langle P, M \rangle = \langle Q, M \rangle$$

Relações de Equivalência

Equivalência de Máquinas
Duas máquinas são ditas equivalentes se elas forem capazes de se simular mutuamente. A simulação de uma máquina por outra pode ser feita utilizando programas diferentes.

Sejam $\mathbf{M}$ e $\mathbf{N}$ duas máquinas arbitrárias, $\mathbf{N}$ simula fortemente $\mathbf{M}$ se, e somente se, para qualquer programa $\mathbf{P}$ para $\mathbf{M}$, existe um programa $\mathbf{Q}$ para $\mathbf{N}$ tais que as correspondentes funções parciais coincidem, ou seja:

$$\langle \boldsymbol{P}, \boldsymbol{M} \rangle = \langle \boldsymbol{Q}, \boldsymbol{N} \rangle$$

Dúvidas?

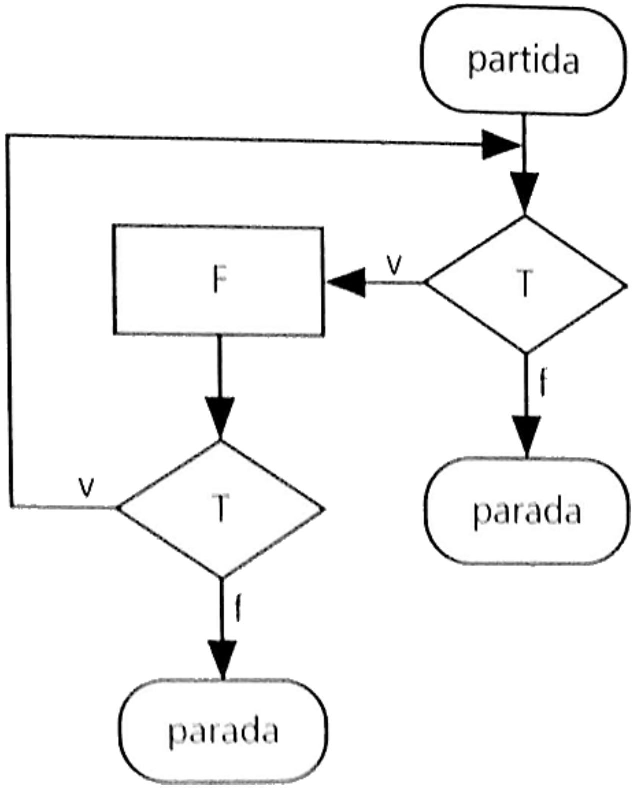
Programa Monolítico?

Programa Iterativo?

Programa Recursivo?

Exercícios

1) Traduza o programa monolítico representado por fluxograma em instruções rotuladas e programa recursivo:



Exercícios

- *2) Traduza o programa iterativo representado abaixo em programa monolítico nas formas de fluxograma e instruções rotuladas:*

Programa iterativo P1

(se T_1
então enquanto T_2
    faça (até T_3
        faça (V;W))
senão (✓))

Exercícios

- *3) Traduza o programa recursivo representado abaixo em iterativo:*

Programa recursivo P2
P é R_1 onde R_1 def (se T então F;R_2 senão R_1), R_2 def G;(se T então F;R_1 senão ✓)

Material Referência

- DIVERIO, Tiarajú A. e MENEZES, Paulo B. **Teoria da computação: máquinas universais e computabilidade.** 3ed. Porto Alegre: Bookman, 2011.
- TANENBAUM, Andrew S. **Organização estruturada de computadores**. 5.ed. São Paulo: Pearson Prentice Hall, 2007.
- OLIVETE JR, Celso. **Teoria da computação: Aula 01**. Presidente Prudente: FCT/UNESP, 2019.