Universidade de Santa Cruz do Sul – UNISC
Departamento de Engenharias, Arquitetura e Computação

TEORIA DA COMPUTAÇÃO
Máquinas Universais

Prof. Ivan L. Süptitz
ivansuptitz@unisc.br

Introdução

“Há um teorema conhecido que diz que qualquer computador é capaz de emular qualquer outro computador”

Astronauta Frank Poole ao explicar o princípio usado por Halman (computador HAL/astronauta Bowman) para impedir o Monólito de executar qualquer ordem que ameaçasse a humanidade.
Do livro 3001 - A Odisséia Final da Série iniciada por 2001 - Uma Odisséia no Espaço
Arthur C. Clarke

Introdução

MÁQUINA UNIVERSAL

Torna possível representar qualquer algoritmo como um programa

Evidências

Evidências:

a) Evidência Interna:
demonstração de que qualquer extensão das capacidades da máquina universal não aumenta o seu poder computacional.

b) Evidência Externa:
exame de outros modelos que definem a noção de algoritmo, juntamente com a prova de que são, no máximo, computacionalmente equivalentes.

MÁQUINA NORMA (Máquina de Registradores) – Richard Bird

- Possui como memória um conjunto "infinito" de registradores naturais e três instruções sobre cada registrador:
  - adição do valor "um"
  - subtração do valor "um"
  - teste se o valor armazenado é "zero"

MÁQUINA NORMA (Máquina de Registradores) – Richard Bird

Algumas características de máquinas reais que podem ser simuladas:

a) ***Operações e Testes*.** Definição de operações e testes mais complexos como adição, subtração, multiplicação e divisão de dois valores e tratamento de valores diversos como os números primos;

b) ***Valores Numéricos*.** Armazenamento de tratamento de valores numéricos de diversos tipos como inteiros (negativos e não-negativos) e racionais;

c) ***Dados Estruturados*.** Armazenamento de tratamento de dados estruturados como em arranjos (vetores uni e multidimensionais), pilhas, etc;

d) ***Endereçamento Indireto e recursão*.** Desvio para uma instrução determinada pelo conteúdo de um registrador;

e) ***Cadeia de Caracteres*.** Definição e manipulação de cadeias de caracteres.

a) Operações e Testes

Exemplos de operações e testes não definidos na Máquina Norma:

Atribuição de um Valor Natural a um Registrador

A macro de atribuição de um valor natural n a um registrador A,

A := n,

é a generalização do programa abaixo.

```
Programa Iterativo n := 3
    A := 0;
    A ++;
    A ++;
    A ++;
```

a) Operações e Testes

Atribuição do Valor Zero a um Registrador A

```
Programa Iterativo A := 0
    até A = 0
        faça( A -- )
```

- Pode-se tratar a operação A := 0 como uma *macro*, ou seja, um trecho de programa que é substituído pela sua definição sempre que referenciado.

a) Operações e Testes

Adição de Dois Registradores

A macro correspondente à operação de adição do valor do registrador B ao do registrador A, denotada por: A := A + B

```
Programa Iterativo A := A + B
      até B = 0
            faça(A := A + 1; B := B - 1)
```

Obs.: ao somar o valor de B em A, o registrador B é zerado! Vamos pensar em um algoritmo para fazer a soma preservando B.

a) Operações e Testes

Adição de Dois Registradores, Preservando B

A macro correspondente à esta operação necessita usar um registrador auxiliar C

Programa Iterativo A := A + B usando C

```
C := 0;
até B = 0
      faça(A := A + 1; C := C + 1; B := B - 1);

até C = 0
      faça(B := B + 1; C := C - 1)
```

b) Valores Numéricos

- Exemplo de tipos de dados numéricos não definidos na **Máquina Norma**:

Inteiros

Um valor inteiro m pode ser representado como um par ordenado:

$$(s, |m|)$$

$|m|$ = magnitude / valor absoluto de m;

s = sinal de m: se m < 0, então s = 1 (negativo)

senão s = 0 (positivo)

b) Valores Numéricos

Racionais

Um valor **racional r** pode ser denotado como um **par ordenado** (a, b):

tal que:
b > 0
r = a/b

c) Dados Estruturados

Arranjo Unidimensional

Esta estrutura pode ser definida por um único registrador A na forma A(1), A(2), ..., usando **Codificação de Conjuntos Estruturados***

*Dados estruturados podem ser armazenados como **números naturais** baseado no fato que qualquer nº natural pode ser obtido pela multiplicação de números primos

Exemplo: arranjo C = (1, 2, 4)

$p1^1 * p2^2 * p3^4$ onde p1, p2, p3, ... são os números primos

$2^1 * 3^2 * 5^4$

2 * 9 * 625 = **11.250**

c) Dados Estruturados

Arranjo Unidimensional

Caminho inverso: Decomposição em fatores primos (tentar sempre o menor)

```
11250 | 2
 5625 | 3
 1875 | 3
  625 | 5
  125 | 5
   25 | 5
    5 | 5
    1
```

Observe que:

- o primeiro nº primo aparece **1** vez
- o segundo nº primo aparece **2** vezes
- o terceiro nº primo aparece **4** vezes

Ou seja:

$$2^1 * 3^2 * 5^4$$

Recuperei o arranjo:

C = (1, 2, 4)

Codificação de Programas Monolíticos

Da mesma forma que um arranjo, também um programa monolítico completo pode ser codificado como um número natural.

`r1: faça Fk vá_para r2` (**Operação - Instrução do tipo 0**)

(0, k, r2, r2) = $2^0 * 3^1 * 5^2 * 7^2$ = 1 * 3 * 25 * 49 = 3.675

`r2: se Tk então vá_para r3 senão vá_para r4` (**Teste – Instrução do tipo 1**)

(1, k, r3, r4) = $2^1 * 3^1 * 5^3 * 7^4$ = 2 * 3 * 125 * 2.401 = 1.800.750

p = $(2^{3.675}) * (3^{1.800.750})$

Codificação de Programas Monolíticos

Podemos sempre fazer o caminho inverso. Dado um número natural, podemos decodificar o programa monolítico correspondente. Exemplo: **p =** $(\mathbf{2^{150}}) * (\mathbf{3^{105}})$

Observamos que o programa possui duas instruções rotuladas correspondentes aos números **150** e **105**.

150 = $2^1 * 3^1 * 5^2 * 7^0$ 105 = $2^0 * 3^1 * 5^1 * 7^1$

o que corresponde às quádruplas: (1, 1, 2, 0) e (0, 1, 1, 1)

Logo, as instruções rotuladas decodificadas são como segue:

```
1: se T1 então vá_para 2 senão vá_para 0
2: faça F1 vá_para 1
```

d) Endereçamento indireto e Recursão

Exemplo de definição de desvios usando endereçamento indireto:

Endereçamento Indireto usando Prog. Monolítico

Uma operação com endereçamento indireto da seguinte forma, onde A é um registrador:

r: faça F vá_para A

pode ser definida pelo seguinte programa monolítico:

r: faça F vá_para End_A

d) Cadeia de caracteres

- Cadeia de caracteres é outro tipo de dado não pré-definido na Máquina Norma.

- O tratamento da definição e da manipulação de cadeias de caracteres será realizado através da Máquina de Turing, a qual é **equivalente** à Norma.

MÁQUINA DE TURING

- Proposta por Alan Turing;
- é universalmente conhecida e aceita como **formalização de algoritmo**;
- trata-se de um mecanismo simples que formaliza a ideia de uma **pessoa que realiza cálculos**;
- possui, no mínimo, o **mesmo poder computacional** de qualquer computador de propósito geral.

MÁQUINA DE TURING - Noção Intuitiva

- O ponto de partida de Turing foi analisar a situação na qual uma pessoa, equipada com um instrumento de escrita e um apagador, realiza cálculos em uma folha de papel organizada em quadrados.

- **Obs:** Inicialmente, a folha de papel contém somente os dados iniciais do problema.

MÁQUINA DE TURING - Noção Intuitiva

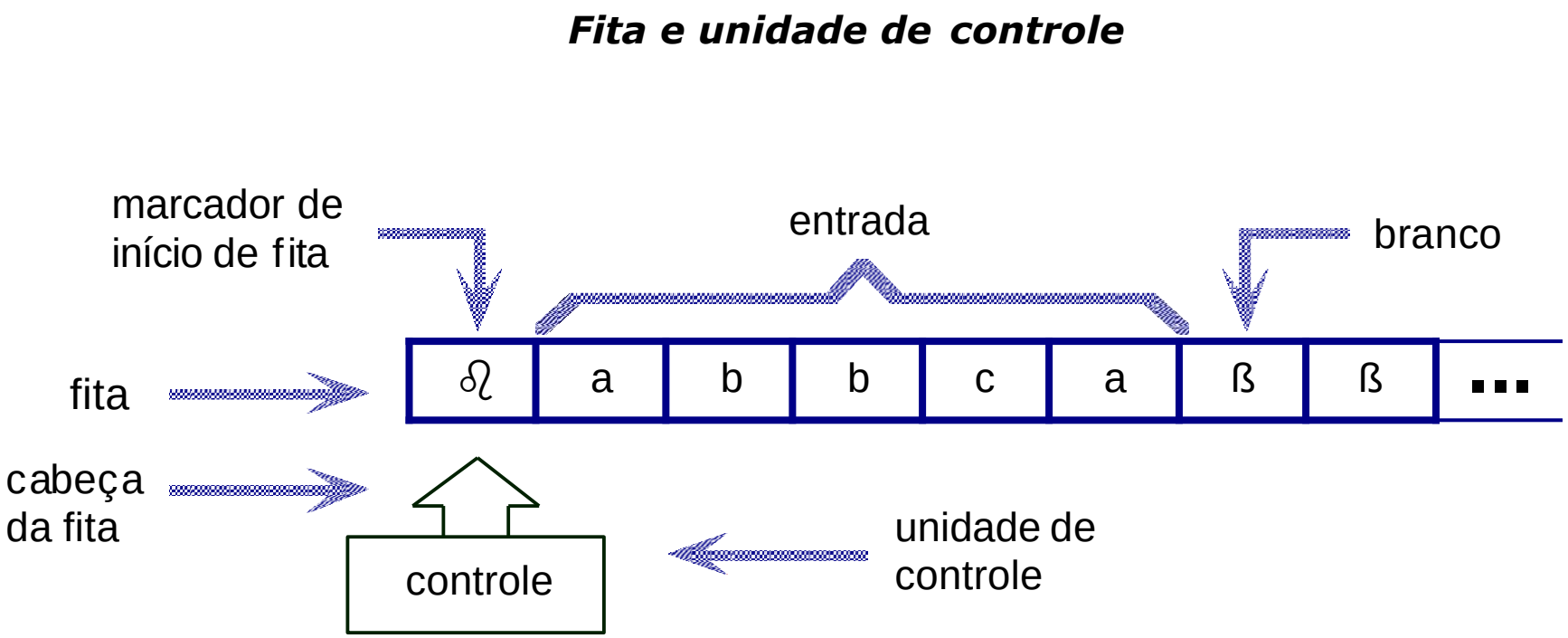
- O trabalho da pessoa pode ser resumido em sequências de operações simples como segue:
  - ler um símbolo de um quadrado de papel;
  - alterar um símbolo em um quadrado (usando lápis e borracha);
  - mover os olhos para outro quadrado;
  - quando é encontrada alguma representação satisfatória para a resposta desejada, a pessoa termina seus cálculos.

MÁQUINA DE TURING - Noção Intuitiva

Para viabilizar esse procedimento, as seguintes hipóteses são aceitáveis:

- Assume-se que o papel consiste de uma fita **infinita** organizada em quadrados (células);
- O conjunto de **símbolos** é **finito**;
- O conjunto de **estados** da mente da pessoa durante o processo de cálculo é **finito**;
- Existem dois estados em particular: ***estado inicial*** e ***estado final***, correspondendo ao início e ao fim dos cálculos, respectivamente;
- O comportamento da pessoa a cada momento é determinado somente pelo seu **estado presente** e pelo **símbolo** para o qual sua atenção está voltada;
- A pessoa é capaz de observar e alterar o símbolo de **apenas um quadrado de cada vez**, bem como de transferir sua atenção somente para um dos **quadrados adjacentes**.

MÁQUINA DE TURING – Noção como máquina



MÁQUINA DE TURING – Noção como máquina

Fita:

- É usada simultaneamente como dispositivo de **entrada**, de **saída** e de **memória de trabalho**;

- É finita à esquerda e **infinita** (tão grande quanto necessário) à direita, sendo dividida em células, cada uma das quais armazenando um símbolo.

- Os símbolos podem pertencer:
  ⇒ ao **alfabeto de entrada**;
  ⇒ ao **alfabeto auxiliar**;
  ⇒ **ß** branco;
  ⇒ ♌ marcador de início de fita.

- Inicialmente, a palavra a ser processada ocupa as células mais à esquerda, após o marcador de início de fita, ficando as demais com *branco*.

MÁQUINA DE TURING – Noção como máquina

Unidade de Controle:

- Reflete o **estado corrente** da máquina;
- Possui um número **finito** e predefinido de **estados**;
- Possui uma unidade de leitura e gravação (cabeça da fita), a qual acessa uma célula da fita **de cada vez**;
- A cabeça da fita **lê** o símbolo de uma célula de cada vez e **grava** um novo símbolo.;
- Após a leitura/gravação (a gravação é realizada na mesma célula de leitura), a cabeça **move-se** uma célula para a direita ou esquerda;

MÁQUINA DE TURING – Noção como máquina

Programa ou Função de Transição

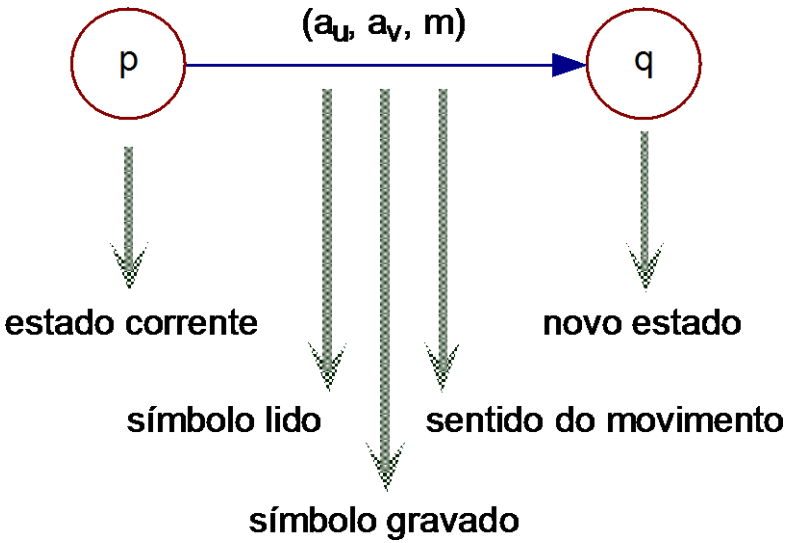
- O programa **comanda** as leituras e gravações, o sentido de **movimento** da cabeça e **define o estado** da máquina;

- Poderíamos dizer que o Programa é uma função que, dependendo do estado corrente da máquina e do símbolo lido, determina o símbolo a ser gravado, o sentido do movimento da cabeça e o novo estado.

MÁQUINA DE TURING – Noção como máquina

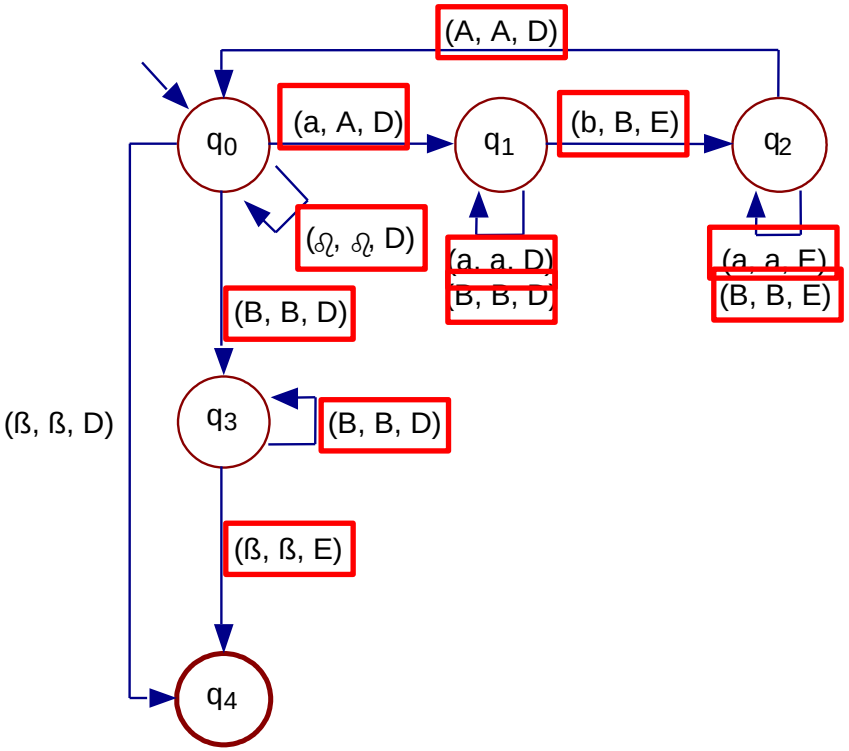
Programa ou Função de Transição

- É representado através de um Grafo

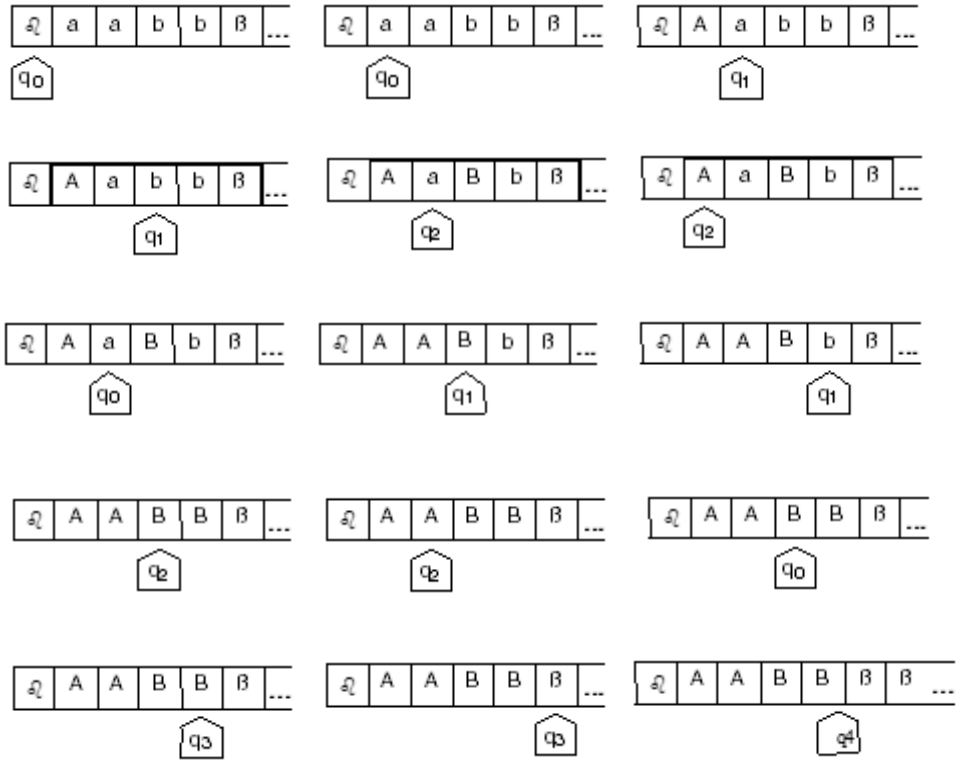


MÁQUINA DE TURING – Noção como máquina

Exemplo:



*Sequência do processamento da Máquina para a entrada **w = aabb**.*



Exercícios - Lista 5

1. Escreva um programa (Java, C, ou outra linguagem de sua preferência) capaz de armazenar um array unidimensional como um número inteiro utilizando codificação de conjuntos estruturados. Exemplo: Entrada **(1, 2, 4)** produz como saída **11.250**

2. Escreva um novo programa (pode ser o mesmo) com a opção de realizar o caminho inverso, ou seja, dado um número inteiro, realizar a decomposição em fatores primos e apresentar como saída o array.

Obs.: Em ambos os casos, apresente um log com os passos intermediários para realização da codificação.