Máquinas de Turing

Máquinas de Turing

1. Introdução

- Até agora no desenvolvimento da teoria da computação, foram apresentados diversos modelos de dispositivos computacionais como, por exemplo, autômatos finitos.
- Foi mostrado que mesmo alguns problemas muito simples estão além da capacidade destes modelos e, portanto, eles são muito restritos para servir como modelos de *computadores de uso (ou propósito) geral*.
- As Máquinas de Turing (MT) possuem este nome devido a seu inventor, *Alan Turing* (1912-1954).
- As MT são similares aos autômatos finitos, mas possuem uma memória ilimitada e irrestrita, e constituem um modelo de computador de propósito geral muito mais preciso.

- Embora as máquinas de Turing possam fazer tudo que um computador ‘real’ pode fazer, há algumas classes de problemas que elas não podem resolver. Mais especificamente, estes problemas estão além dos limites teóricos da computação.
- Uma máquina de Turing é composta por quatro partes:
    - *Fita (tape)*: responsável por armazenar informação (memória). É dividida em posições discretas ou células e possui tamanho infinito;
    - *Cabeçote (tape head)*: responsável pela leitura e escrita de símbolos na fita e pela movimentação na fita;
    - *Registrador de estados (state register)*: responsável por armazenar o estado da máquina;
    - *Tabela de instruções ou programa*: responsável por controlar a máquina, ou seja, indicar qual símbolo escrever, como mover o cabeçote e qual o novo estado .

- A definição formal de uma MT e sua operação segue o mesmo padrão matemático da definição dos autômatos finitos.
- Entretanto, é possível demonstrar que tentativas de obter ganhos em termos de poder computacional através de melhorias ou combinações de MT resultam em sistemas com poder computacional que não é maior do que o de uma MT básica.
    - Isso pode ser mostrado através de simulação: é possível converter qualquer ‘MT melhorada’ em uma máquina básica que funciona de maneira análoga.
- É possível ir mais longe e mostrar que as MT são equivalentes em poder à formas completamente diferentes de realizar computação que não são baseadas em autômatos com estados e fitas.
    - Também é possível mostrar a *universalidade* de novos paradigmas de computação baseado nos princípios da universalidade das máquinas de Turing.
- As MT são projetadas para satisfazer simultaneamente três critérios:
    - Sua construção e funcionamento seguem o princípio geral dos autômatos;

  - Elas devem ser simples de descrever e definir formalmente;
  - Elas devem ser genéricas em relação às computações que podem executar.
- Resumo das diferenças entre os autômatos finitos e as MT:
  - Uma MT pode escrever e ler a fita;
  - O cabeçote de uma máquina de Turing pode se mover para esquerda ou para a direita;
  - A fita é infinita;
  - Os estados especiais para *rejeitar* e *aceitar* possuem efeito direto.
- Há um estado especial chamado de *halt state*, *h*, que é usado para sinalizar o fim de uma computação.

2. Definição Formal de Máquinas de Turing

- Uma Máquina de Turing é uma quádrupla $(K,\sum,\delta,s)$ onde:
  - K é um conjunto de *estados*, incluindo o *estado de parada* (*halt state – h*);
  - $\sum$ é um alfabeto contendo o símbolo vazio #, mas não contendo os símbolos de movimentação L, R e N;
  - δ é uma função de transição de estados δ: $K \times \sum \rightarrow (K \times \sum \times \{L,R,N\})$, onde $\{L,R,N\}$ indica o movimento do cabeçote para a esquerda (L), direita (R), ou nenhum movimento (N).
  - $s \in K$ é o estado inicial.
- Um movimento da máquina de Turing é função de seu estado atual e do símbolo lido a partir da fita. Em um movimento M poderá:
  - Mudar de estado. O próximo estado pode ser o mesmo estado atual.
  - Escrever um símbolo na célula lida. O novo símbolo pode ser igual ao atual.
  - Mover o cabeçote para a direita, esquerda ou permanecer na mesma posição.

- Exemplo 1:

Dada $M = (K,\sum,\delta,s)$. Se $q \in K$, $a \in \sum$, e $\delta(q,a) = (p,b,L)$, então M enquanto no estado q e lendo o símbolo a, entrará no estado p e moverá o cabeçote para a esquerda.

Dada $M = (K,\sum,\delta,s)$. Se $q \in K$, $a \in \sum$, e $\delta(q,a) = (p,b)$, então M enquanto no estado q e lendo o símbolo a, entrará no estado p e poderá executar uma das duas operações abaixo:

. Se $b \in \sum$, substitua a por b

. Se $b \in \{L,R,N\}$, então mova o cabeçote de acordo com o símbolo b.

Nota: δ é uma função que determina exatamente qual a próxima configuração da MT e, portanto, esta MT é denominada de *determinística*.

2.1. Representações Pictóricas de uma MT

- A máquina de Turing descrita formalmente acima pode ser representada de algumas formas distintas como, por exemplo:

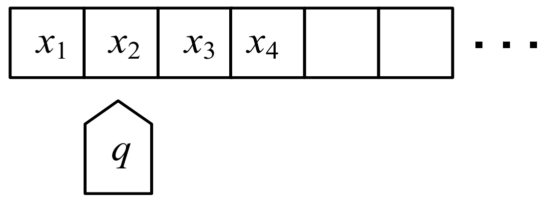


Figura 1: Representação pictórica de uma máquina de Turing M no estado q, lendo o símbolo x_2. (Os espaços em branco correspondem a #.)

- A mesma máquina de Turing ilustrada acima pode ser representada como abaixo:

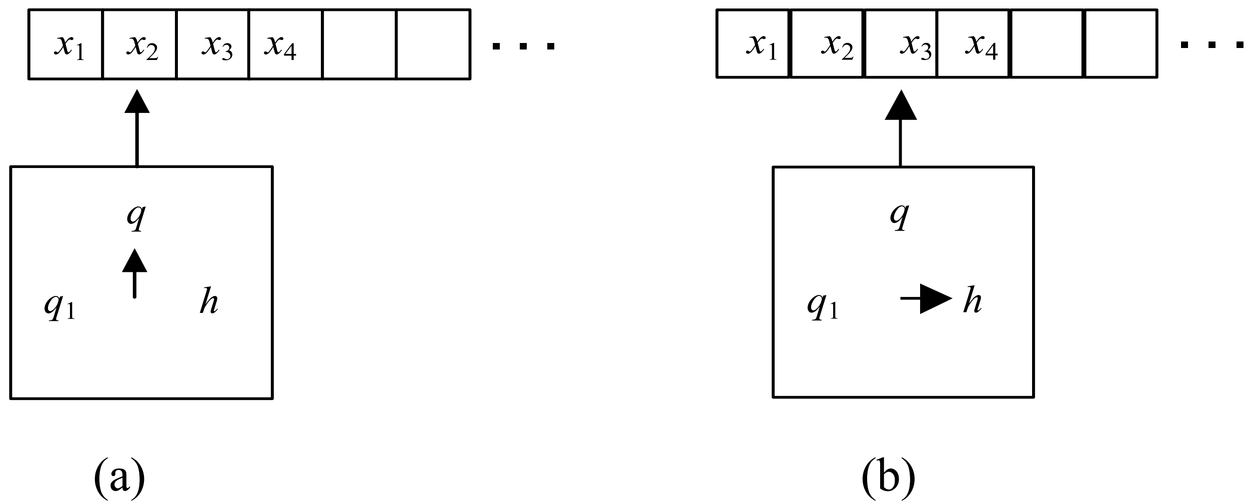


Figura 2: (a) Outra representação pictórica de uma máquina de Turing *M* no estado *q*, lendo o símbolo x_2. Nesta figura aparecem outros possíveis estados de *M*, q_1 e *h*. (Os espaços em branco correspondem a #.) (b) Explicar.

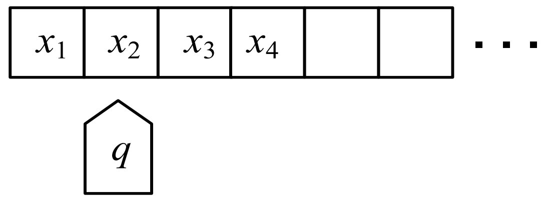
2.2. Configuração de uma MT

- Para especificar o status de uma computação da MT, é preciso especificar o estado da máquina, o conteúdo da fita e a posição do cabeçote.
- Como apenas uma porção finita da máquina conterá o símbolo #, o conteúdo da fita pode ser representado por uma *string* (cadeia de atributos).
- Assim como a própria definição formal de máquina de Turing, a representação de uma configuração também possui variantes. Segue abaixo duas delas:
  - Cada *computação* (*passo* ou *estágio*) de uma máquina de Turing pode ser representada por uma seqüência de *configurações instantâneas* da máquina, onde cada configuração $C = (x_1 \ldots x_{k-1} q x_k x_{k+1} \ldots x_m)$, $x_i \in \sum$ e $q \in K$, codifica um único *passo da computação*. A string $x_1 \ldots x_m$ é a porção da fita entre os símbolos mais a esquerda e mais a direita diferentes de #.

    Formalmente, a configuração de M é um membro de:

    $\sum \times K \times \sum \times \sum$

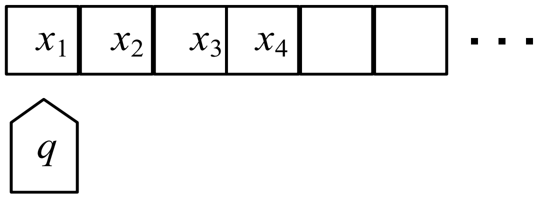
- <u>Exemplo 2</u>:



No caso do exemplo da figura acima tem-se: $C = (x_1qx_2x_3x_4)$.

  - A string que representa a configuração da MT pode ser quebrada em três partes: 1) a primeira (que pode ser vazia), à esquerda da posição sendo lida; 2) o símbolo único sendo lido; e 3) a parte à direita da posição sendo lida. Além disso, para que não haja redundância nas possíveis triplas geradas, a última sub-string não deve terminar com um símbolo # (quaisquer símbolos a direita do último são assumidos conter #).

- <u>Exemplo 3</u>:



$C = (q,x_1,x_2,x_3x_4)$, onde $x_i \in \sum$ e $q \in K$. (Equivalente a anterior.)

2.3. Computando com Máquinas de Turing

- É usual descrever movimentos de uma máquina de Turing $M = (K,\sum,\delta,s)$ utilizando a notação ‘ $\vdash_M$’.
- <u>Exemplo 4</u>:

  Assuma a seguinte função de transição $\delta(q,x_i) = \delta(p,y,L)$. Então a descrição de um movimento (passo) pode ser dada por:

  $x_1\ldots x_{k-1}qx_kx_{k+1}\ldots x_m \vdash_M x_1\ldots x_{k-2}px_{k-1}yx_{k+1}\ldots x_m$
- Este movimento reflete a mudança para o estado p, a escrita de y na posição x_i e o fato de que o cabeçote agora está posicionado na célula $i-1$.

- Há duas exceções para esta representação:
    - Se $i = 1$, então M se move para o vazio a esquerda de x_1:

      $qx_1\ldots x_m \vdash_M p\#yx_2\ldots x_m$
    - Se $i = m$ e $y = \#$, então o símbolo # é escrito sobre x_m e não aparece na configuração:

      $x_1\ldots qx_m \vdash_M x_1\ldots px_{m-1}$
- <u>Exemplo 5</u>:

  Assuma agora a seguinte função de transição com movimento para a direita $\delta(q,x_i) = \delta(p,y,R)$. Neste caso, a descrição de um movimento (passo) pode ser dada por:

  $x_1\ldots x_{k-1}qx_kx_{k+1}\ldots x_m \vdash_M x_1\ldots x_{k-1}ypx_{k+1}\ldots x_m$
- Este movimento reflete a mudança para o estado p, a escrita de y na posição x_i e o fato de que o cabeçote agora está posicionado na célula i+1.

- Também há duas exceções para esta representação:
    - Se $i = m$, então a célula $i + 1$ possui o estado vazio:

      $x_1 \ldots x_{m-1} q x_m \vdash_M x_1 \ldots x_{m-1} y p\#$
    - Se $i = 1$ e $y = \#$, então o símbolo # é escrito sobre x_1 e não aparece na nova configuração:

      $q x_1 \ldots x_m \vdash_M p x_2 \ldots p x_m$
- Uma computação realizada por M é uma seqüência de configurações $C_0, C_1, \ldots, C_n$, $n \geq 0$, tal que:

  $C_0 \vdash_M C_1 \vdash_M C_2 \vdash_M \ldots \vdash_M C_n$
- Neste caso diz-se que a computação tem **comprimento** n, ou n **passos**.
- Exemplo 6:

  Considere a seguinte MT: $M = (K,\sum,\delta,s)$, onde:

  $K = \{q_0,q_1\}$; $\sum = \{a,\#\}$; $s = q_0$

  e δ é dado pela tabela abaixo:

q	σ	$\delta(q,\sigma)$
q_0	a	$(q_1,\#)$
q_0	$\#$	$(h,\#)$
q_1	a	(q_0,a)
q_1	$\#$	(q_0,R)

Descreva a computação realizada por M assumindo que M possui a seguinte seqüência de símbolos escrita na fita (configuração inicial): $aaaa\#$.

$q_0aaa \vdash_M q_1\#aaa$ (passo 1)

$\vdash_M \#q_0aaa$

$\vdash_M \#q_1\#aa$ (passo 3)

$\vdash_M \#\#q_0aa$

$\vdash_M \#\#q_1\#a$ (passo 5)

$\vdash_M \#\#\#q_0a$

$\vdash_M \#\#\#q_1\#$ (passo 7)

$\vdash_M \#\#\#\#q_0$

$\vdash_M \#\#\#\#h$ (passo 9)

Exercício 1:

Considere a seguinte MT: $M = (K,\sum,\delta,s)$, onde:

$K = \{q_0\}$; $\sum = \{a,\#\}$; $s = q_0$

e δ é dado pela tabela abaixo:

q	σ	$\delta(q,\sigma)$
q_0	a	(q_0,L)
q_0	$\#$	$(h,\#)$

Explique informalmente o que M faz e descreva a computação realizada por M assumindo que M possui a seguinte configuração inicial: aaq_0a.

Exercício 2:

Considere a seguinte MT: $M = (K,\sum,\delta,s)$, onde:

$K = \{q_0,q_1,q_2\}$; $\sum = \{a,b,\#\}$; $s = q_0$

e δ é dado pela tabela abaixo:

q	σ	$\delta(q,\sigma)$
q_0	a	(q_1,L)
q_0	b	(q_0,R)
q_0	$\#$	(q_0,R)
q_1	a	(q_1,L)
q_1	b	(q_2,R)
q_1	$\#$	(q_1,L)
q_2	a	(q_2,R)
q_2	b	(q_2,R)
q_2	$\#$	$(h,\#)$

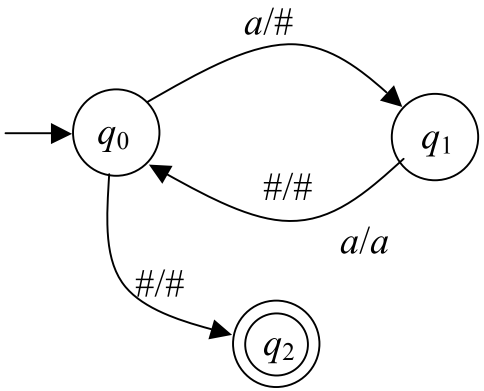
Explique informalmente o que M faz e descreva a computação realizada por M assumindo que M possui a seguinte configuração inicial: $aq_0bb\#bb\#\#aba$.

2.4. Diagramas de Transição para Máquinas de Turing

- É possível representar as transições de estado de uma máquina de Turing pictóricamente como feito para os autômatos finitos.
- Um *diagrama de transição* (de estados) consiste em um conjunto de *nós* correspondente aos estados da máquina de Turing. Um *arco* ligando um estado q a um estado p é nomeado por um ou mais itens da forma x/yD, onde $x, y \in \sum$ (símbolos) e $D \in \{R,L,N\}$ é uma direção de movimentação.
    - Outra representação possível é simplesmente assumir que a direção será dada pela direção do arco no grafo (grafo direcionado). Neste caso, a direção N é representada por um arco que incide no mesmo nó do qual parte.

- Exemplo 7:

  Diagrama de transição de estados da máquina de Turing do Exemplo 6.



2.5. Algumas Extensões das Máquinas de Turing

- As Máquinas de Turing são capazes de realizar computações de considerável complexidade, porém de forma lenta e ineficiente.
- Com o objetivo de investigar a possibilidade de projetar autômatos com maior poder computacional, considera-se o efeito de estender o modelo de Máquinas de Turing de várias formas.
    - Entretanto, boa parte das extensões não aumenta a classe de funções computáveis ou linguagens reconhecidas pela MT.
- Um benefício extra destas extensões é permitir um projeto mais eficiente de MTs para aplicações específicas.
- As principais extensões das MT são:
    - *Fita infinita em ambas as direções*: permitir que a fita tenha tamanho infinito em ambas as direções, e não apenas para a direita.

- *Máquinas de Turing de Múltiplas Fitas*: permitir que a MT tenha mais de uma fita;
- *Máquinas de Turing de Múltiplos Cabeçotes*: permitir que a MT tenha mais de um cabeçote;
- *Fita Multi-Dimensional*: permitir que a MT tenha uma fita de dimensão dois ao invés de um;
- *Máquinas de Turing Não-Determinísticas*: permitir que um dentre um conjunto finito de movimentos seja possível (escolhido probabilisticamente).

Fita Infinita em Ambas as Direções

- Neste caso a fita não possui distinção entre a extremidade esquerda e direita.
- Uma string de entrada pode ser colocada em qualquer posição da fita e o cabeçote posicionado no espaço vazio a sua direita.
- Uma computação procede igualmente para o caso das máquinas com fita infinita em uma única direção, exceto que agora há células disponíveis arbitrariamente tanto para a direita quanto para a esquerda.
- Formalmente estas máquinas são idênticas às máquinas padrão, a diferença é que as noções de configuração e computação mudam um pouco.

<u>Exercício 3</u>: Defina formalmente a configuração e uma computação de uma MT com fita infinita em ambas as direções.

Máquinas de Turing de Múltiplas Fitas

- A diferença principal deste tipo de MT é a existência de mais de uma fita onde armazenar informação. Cada fita pode ser lida ou escrita utilizando um cabeçote distinto.
- Inicialmente:
  - A entrada, uma seqüência finita de símbolos de entrada, é colocada na primeira fita.
  - Todas as outras células de todas as outras fitas estão vazias.
  - O sistema de controle está no estado inicial.
  - O cabeçote da primeira fita está na primeira posição a esquerda da entrada.
  - Todos os outros cabeçotes estão em uma célula arbitrária.
- Um movimento da MT de múltiplas fitas depende do estado e do símbolo lido por cada um dos cabeçotes. Em um movimento (passo), esta máquina faz o seguinte:
  - O controle entra em um novo estado, que pode ser o mesmo do anterior;

- Em cada fita um novo símbolo é escrito na célula lida. Qualquer um destes símbolos pode ser o mesmo que o símbolo atual;
- Cada um dos cabeçotes se move independentemente dos demais.

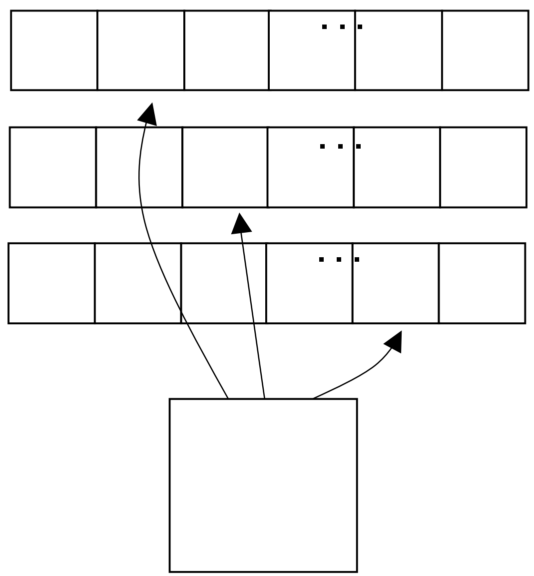


Figura 3: Máquina de Turing de múltiplas fitas.

Máquinas de Turing de Múltiplos Cabeçotes

- O que acontecerá se permitirmos que uma mesma MT tenha vários cabeçotes para ler uma única fita?
- Em um passo, todos os cabeçotes vão ler os símbolos indicados e mover ou escrever independentemente.
    - Obs.: alguma convenção deve ser adotada sobre o que acontece quando dois cabeçotes estão apontando sobre a mesma célula e tentam escrever símbolos distintos nesta posição. Algum mecanismo de decisão deverá ser empregado.
- Assim como no caso da utilização de múltiplas fitas, o uso de múltiplos cabeçotes pode simplificar muito a construção de uma máquina de Turing.

Fita Multi-Dimensional

- Outra generalização de uma máquina de Turing é permitir que sua fita seja um grid infinito bi-dimensional.
    - É possível considerar inclusive um espaço de maior dimensão.
- Máquinas de Turing com fita bi-dimensional podem ser simuladas por máquinas de Turing clássicas.

Máquinas de Turing Não-Determinísticas

- As máquinas de Turing não-determinísticas (MTND) podem ter, dadas algumas combinações de estados e símbolos lidos, mais que um possível comportamento.
- Formalmente, a única diferença de uma MTND para uma MT clássica está na função de transição:

  $\delta: K \times \sum \rightarrow P\{K \times \sum \times \{L,R,N\}\}$
- Outra forma de representar é a seguinte. Dado o estado atual q_0 e o símbolo lido x_0, a função de transição poderá assumir qualquer uma das seguintes triplas: $\{(q_1,x_1,D_1), (q_2,x_2,D_2), ..., (q_k,x_k,D_k)\}$, onde k é um inteiro finito que representa a quantidade de possibilidades de mudança de configuração.

2.6. Máquinas de Turing e Computadores Clássicos

- É possível comparar as máquinas de Turing com os computadores convencionais (domésticos) conhecidos atualmente.
- Ao mesmo tempo em que estes modelos parecem ser muito diferentes, eles podem aceitar exatamente as mesmas linguagens: linguagens recursivamente enumeráveis.
- *Ao mesmo tempo que uma MT não parece nada com um PC e seria grosseiramente ineficiente caso alguma empresa decidisse fabricá-la e vendê-la, ela é reconhecidamente um modelo preciso para o que um dispositivo computacional físico pode fazer.*
- Como a noção do que é um computador convencional não é muito bem definida matematicamente, a discussão a ser apresentada aqui será necessariamente <u>informal</u>.

- A demonstração de equivalência entre uma máquina de Turing e computadores clássicos será feita de duas formas:
  - Mostrando que um computador comum pode simular uma MT; e
  - Mostrando que uma MT pode simular um computador.

Simulando uma MT via um Computador

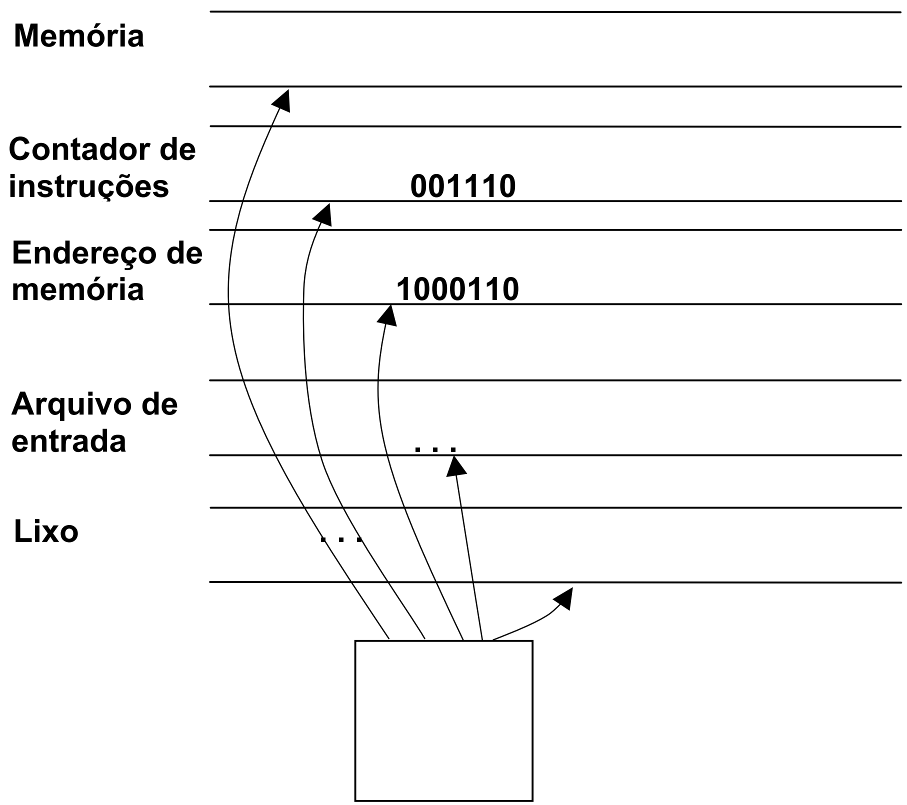
- Dada uma MT qualquer $M = (K,\sum,\delta,s)$, é possível escrever um programa que opera como M. Uma característica importante de M é seu controle finito.
    - Como há um conjunto finito de estados e regras de transição, o programa pode codificar os estados como sendo cadeias de caracteres (strings) e usar uma tabela de transição de estados para determinar cada movimento.
    - Similarmente, os símbolos da fita podem ser codificados como cadeias de caracteres de comprimento finito.
- Questão importante: dado que a fita de M é infinita e que a memória do computador é finita, como o computador poderá simular a fita de M?
    - A resposta para esta questão é simples: substituindo 'infinitamente' a memória do computador. (Desde que isso seja possível!)
    - Sendo assim, é possível ter uma quantidade de memória específica para armazenar os símbolos à direita e à esquerda da fita.

Simulando um Computador via uma MT

- Além de verificar se uma MT pode simular um computador, é importante também comparar a eficiência computacional destes dois modelos.
- Inicialmente, consideremos como um computador clássico opera:
    - Vamos assumir que o armazenamento de um computador consiste em uma seqüência infinitamente longa de *palavras*, cada qual com um *endereço*. Enquanto as palavras possuem um comprimento dado (p. ex. 32 ou 64 bits), os endereços serão considerados *inteiros*.
    - O programa do computador será armazenado em algumas palavras da memória. Estas palavras representam *instruções*, como em uma linguagem assembler. *Endereçamento direto* será assumido possível, ou seja, uma palavra pode referenciar outra palavra.
    - Cada instrução envolve um número limitado (finito) de palavras, e cada instrução muda o valor de no máximo uma palavra.

- o Um computador típico possui *registradores*, que são palavras de memória com acesso especialmente rápido.

- A máquina de Turing a ser usada para simular um computador possui várias fitas, mas como discutido, ela é equivalente a uma máquina de fita única.



- A MT irá simular um ciclo de instrução da seguinte forma:
  - Procure na memória (fita 1) o primeiro endereço que satisfaz a instrução na fita 2.
  - Quando o endereço de instrução for encontrado, examine seu valor. Se uma palavra for uma instrução, então seus primeiros bits representam a ação a ser tomada e os demais endereços envolvidos na ação.
  - Se a instrução requer o valor de algum endereço, então este endereço será parte da instrução. Copie este endereço para a memória de endereço (fita 3) e marque a posição da instrução. Agora procure o endereço de memória na memória (fita 1) e copie seu valor para a memória de endereço (fita 3).
  - Execute a instrução, ou parte da instrução envolvendo este valor. (Obs.: não entraremos em detalhes sobre o que a instrução pode fazer. Exemplos: copiar, escrever, etc.)

    - Após executar a instrução adicione 1 ao contador de instrução na fita 2 e repita o processo.

- Obs.: esta descrição é visivelmente incompleta, mas ela ilustra o potencial de uma MT para simular um computador convencional.

3. A Tese de Church-Turing e Máquinas de Turing Universais

- O principal propósito da Teoria da Computação é apresentar uma perspectiva matemática formal sobre o que é computação e como computar.
- Para isso, foram vistos diversos modelos abstratos de *máquinas*, como autômatos finitos e agora as máquinas de Turing.
  - Basicamente, trata-se de dispositivos com um número finito de estados internos distinguíveis que opera em uma cadeia de símbolos (strings) de acordo com um conjunto finito de regras.
- As MT foram projetadas para aceitar linguagens ‘complicadas’ e também para calcular funções matemáticas não-triviais.
- O fato de que a adição de algumas características às MTs não aumenta o poder computacional delas demonstra parte de seu potencial, e também sugere que um limite superior natural foi alcançado para o tipo de computação que uma máquina foi projetada para realizar.

- A questão da eficiência será tratada no Tópico 6 (Complexidade Computacional).

- Como as MT podem realizar qualquer computação realizável por um autômato similar e como estes autômatos parecem capturar as características essenciais de um computador, a MT é tomada como um equivalente formal preciso da noção intuitiva de *algoritmo*: nada pode ser considerado um algoritmo se ele não puder ser realizado (computado) por uma máquina de Turing.
- De forma simples, um *algoritmo* é uma coleção de instruções com o objetivo de realizar alguma tarefa. Também são denominados de *procedimentos* ou *receitas*.
- *O princípio de que as MT são versões formais de algoritmos e que nenhum procedimento computacional será considerado um algoritmo a não ser que ele possa ser apresentado como uma MT é conhecido como a Tese de Church-Turing*.

Noção intuitiva de algoritmo = Algoritmos das Máquinas de Turing

- É claro que a Tese de Church-Turing pode ser ultrapassada algum dia, quando alguém propuser um novo modelo computacional capaz de realizar computações que não podem ser realizadas por uma MT.
- **Desafio**:

  Proponha um algoritmo que testa se um polinômio possui raiz inteira.

  Exemplo: $2x^2 + y - 4 = 0$.

  Raízes inteiras: $x = 2$, $y = 4$.
- As máquinas de Turing constituem um exemplo de Máquina de Turing Universal (MTU), originalmente proposta por Turing.
- A denominação *universal* é devida a sua capacidade de simular qualquer outra MT a partir da descrição da máquina que se deseja simular.

Bibliografia

J. E. Hopcroft, R. Motwani & J. D. Ullman (2001), *Introduction to Automata Theory, Languages, and Computation*, 2nd ed, Addison Wesley.

M. Sipser (1997), *Introduction to the Theory of Computation*, PWS Publishing Company.

H. R. Lewis & C. H. Papadimitriou (1981), *Elements of the Theory of Computation*, Prentice Hall Inc.