Variantes de Máquinas de Turing

Teoria da Computação – Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

IFB – Instituto Federal de Brasília, Campus Taguatinga

Sumário

Sumário

Variantes

- Existem definições alternativas de Máquinas de Turing.
- Algumas possuem mais de uma fita.
- Outras não são determinísticas.
- Elas são chamadas de variantes do nosso modelo inicial da Máquina de Turing.

Variantes

- O interessante é que, apesar de diferentes elas tem o mesmo **poder** computacional.
- Todas as linguagens reconhecíveis por uma variante, é reconhecida por outra.
- Todas as linguagens decidíveis por uma variante, é reconhecida por outra.

Variantes

- Chamamos esta invariância de **robustez**.
- Vamos mudar a nossa definição de máquina de Turing inicial para ilustrar isto.

Variantes

- Em nossa definição da máquina de Turing, a cabeça de leitura movimentava-se para a esquerda ou para a direita.
- Ou seja:

$$\delta : Q \times \Gamma \Rightarrow Q \times \Gamma \times \{L, R\}$$

Variantes

- Podemos alterá-la levemente de modo a permitir a operação de continuar na mesma célula.
- Ou seja, a cabeça de leitura fica no mesmo lugar.
- A função de transição então seria alterada para:

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\} \tag{1}$$

- S especifica que a cabeça não muda de lugar.

Variantes

- Esta variante possui mais poder computacional?
- Ela consegue resolver mais problemas que a versão original?
- Em outras palavras, esta variante reconhece mais linguagens?

Variantes

- Esta variante possui mais poder computacional?
- Ela consegue resolver mais problemas que a versão original?
- Em outras palavras, esta variante reconhece mais linguagens?
- Na verdade **não**.

Variantes

- Podemos converter qualquer TM' com a característica de ficar imóvel sobre a fita pela nossa.
- Como?
- Convertendo qualquer transição que não mova a cabeça de leitura por uma que mova para a esquerda e outra para a direita.

Variantes

Exercício

Mostre que qualquer TM' pode ser convertida para uma TM com a nossa definição original.

Variantes

- Mostraremos agora alguns modelos que, apesar de diferentes, possuem o mesmo poder computacional do que uma TM ordinária.
- Para mostrar a equivalência, só precisamos mostrar que um modelo consegue simular o outro, e vice-versa.
- Quando falamos em **poder**, estamos falando em termos de computabilidade, e não em "velocidade".
- Ou seja, apresentaremos modelos que podem até reconhecer ou decidir linguagens mais rápido do que nossa TM ou original, mas as linguagens reconhecidas/decididas são as mesmas.

Sumário

Máquinas Multi-fitas

- Uma máquina de Turing multi-fita é como uma máquina de turing com várias fitas.
- Cada fita tem a sua própria cabeça de leitura e escrita.
- Inicialmente, a entrada aparece na fita 1.
- As outras fitas estão em branco.

Máquinas Multi-Fitas

- Supondo que este modelo possui k fitas a função de transição deve ser modificada de modo a possibilitar:
  - Ler.
  - Escrever.
  - Mover.
- Estas operações podem ser suportadas simultaneamente em fitas separadas.
- Podemos operar em algumas fitas e deixar outras de fora.

Máquinas Multi-Fitas

- Formalmente, temos:

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k$$

Máquinas Multi-Fitas

- A computação é dada por:

$$\delta(q_i, a_1, \ldots, a_k) = (q_j, b_1, \ldots, b_k, L, R, \ldots, L)$$

- Ou seja, se a máquina encontra-se no estado q_i e as cabeças de 1 a k estão lendo os símbolos a_1,...,a_k, a máquina vai ao estado q_j, escreve os símbolos b_1,...,b_k em suas respectivas fitas e direciona cada cabeça para esquerda, direita ou parada.

Máquinas Multi-Fitas

- Máquinas multi-fitas aparentam ser mais poderosas do que TM ordinárias.
- Mas na verdade são equivalentes em poder.
- Reconhecem absolutamente as mesmas linguagens.

Máquinas Multi-Fitas

Teorema (Equivalência entre TM^k e TM)

Toda máquina de Turing multi-fita possui uma Máquina de Turing equivalente.

Equivalência entre TM^k e TM

Ideia da Prova

A ideia da prova é mostrar que podemos converter uma máquina multi-fita em uma máquina de Turing usual. Para isso procuraremos simular a máquina multi-fita M através da máquina comum S.

Equivalência entre TM^k e TM

Prova

Suponha M uma máquina multi-fita com k fitas.

Uma máquina de Turing S pode simular M ao armazenar a informação de todas as fitas desta máquina em sua única fita.

Ela utiliza um novo símbolo $\#$ como delimitador do conteúdo de diferentes fitas.

Além disso ela marca as posições das cabeças virtuais ao substituir um símbolo c, em que a cabeça está posicionada sobre, por um símbolo novo $\dot{c}$

Equivalência entre TM^k e TM

Prova

Suponha M uma máquina multi-fita com k fitas.

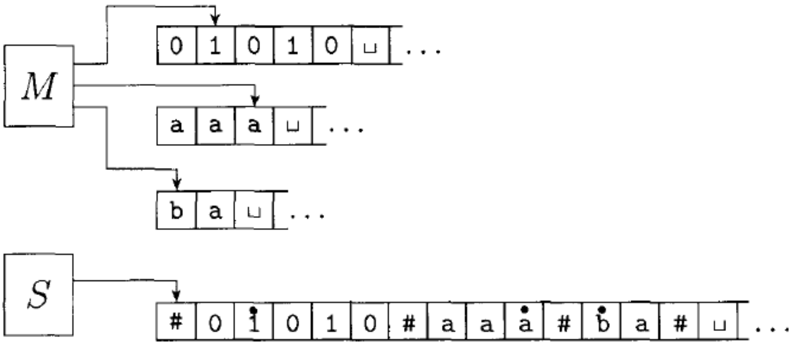
Uma máquina de Turing S pode simular M ao armazenar a informação de todas as fitas desta máquina em sua única fita.

Ela utiliza um novo símbolo $\#$ como delimitador do conteúdo de diferentes fitas.

Além disso ela marca as posições das cabeças virtuais ao substituir um símbolo c, em que a cabeça está posicionada sobre, por um símbolo novo $\dot{c}$

Equivalência entre TM^k e TM

Prova



Equivalência entre TM^k e TM

Prova

Seja a máquina S sobre a entrada $w = w_1 \ldots w_n$:

- Primeiro ela coloca w na fita na codificação proposta, isto é:

$$\#\dot{w}_1 w_2 \ldots w_n \# \dot{\sqcup} \# \dot{\sqcup} \# \ldots \#$$

- Primeiramente a máquina verifica cada cabeça virtual para definir qual a transição a ser aplicada.
- Para simular um movimento, S, para cada cabeça virtual, aplica-se a transição.

Equivalência entre TM^k e TM

Prova

Seja a máquina S sobre a entrada $w = w_1 \ldots w_n$:

- Se em algum momento, alguma cabeça virtual fica sobre o i-ésimo $\#$, significa que a cabeça da virtual fita está sobre uma posição $\sqcup$ da fita i de M.
- Então, S move todo o conteúdo a partir do i-ésimo $\#$ para a direita até o último $\#$ e escreve $\sqcup$ no lugar original do i-ésimo $\#$.

Equivalência entre TM^k e TM

Corolário

Uma linguagem é Turing-reconhecível se, e somente se, alguma máquina de Turing multi-fita a reconhece.

Equivalência entre TM^k e TM

Prova

$\Rightarrow$) Se uma linguagem é reconhecível em uma Máquina de Turing, ela é reconhecível em uma Máquina de Turing multi-fita usando uma única fita.

$\Leftarrow$) Uma linguagem reconhecível em uma máquina de Turing multi-fita também é Turing-reconhecível, uma vez que uma máquina de Turing consegue simulá-la.

Sumário

3 Máquinas Não-determinísticas

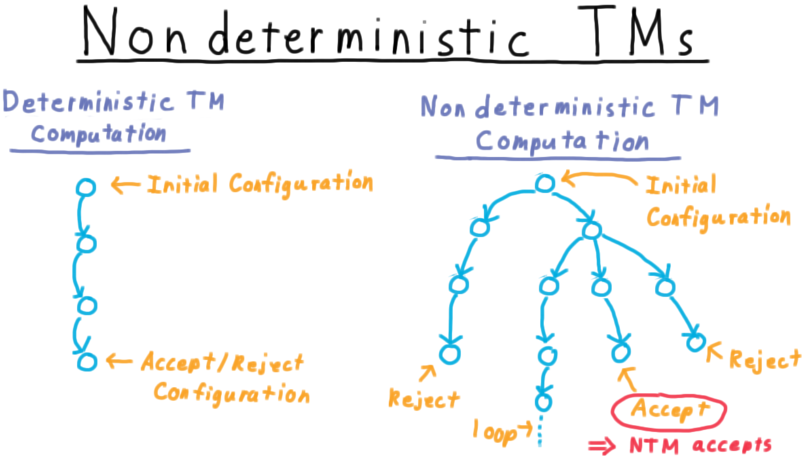
Máquinas de Turing Não-determinísticas

- Em uma máquina de Turing não-determinística, uma computação pode proceder de várias maneiras.
- Computação não determinística:

$$\delta : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

- Dado um estado e um símbolo de lido, podemos ir para vários outros estados, escrever outros símbolos e mover para diferentes direções.

Máquinas de Turing Não-determinísticas



Máquinas de Turing Não-determinísticas

Teorema

Toda máquia de Turing não-determinística possui uma equivalente determinística.

Máquinas de Turing Não-determinísticas

Ideia da Prova

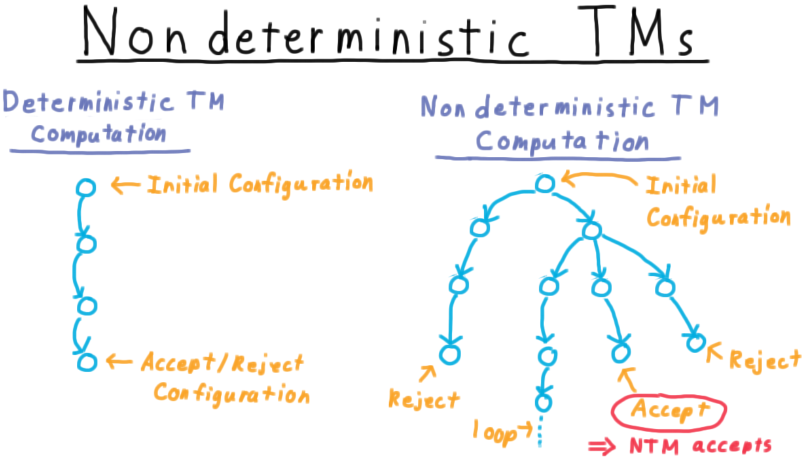
- A ideia da prova é mostrar que é possível simular qualquer TM_{nd} N a partir de uma TM D.
- O objetivo é fazer com que D tente todas as opções não determinísticas serialmente.
- Se D acha pelo menos um ramo que cai no estado q_{aceita}, D aceita.
- Caso contrário, D entra em *loop*.

Máquinas de Turing Não-determinísticas

Ideia da Prova

- Se encaramos a computação de N sobre uma entrada w como uma árvore, cada subárvore representa um ramo não-determinístico.
- Cada nó é uma configuração de N.
- A raiz é a configuração de início.

Máquinas de Turing Não-determinísticas

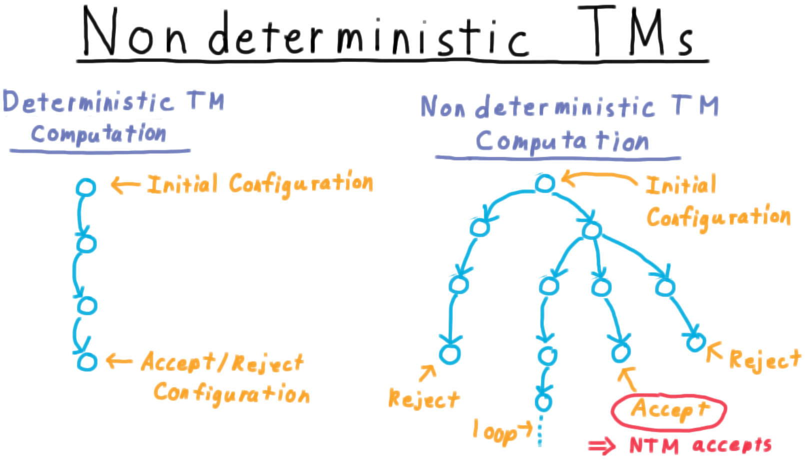


Máquinas de Turing Não-determinísticas

Ideia da Prova

- D deve fazer um percurso nesta árvore em busca de um estado de aceitação.
- A busca deve ser feita de maneira cuidadosa no entanto.
- Uma busca em profundidade funcionaria?
- A busca em profundidade vai até o final de cada ramo para então explorar o próximo.

Máquinas de Turing Não-determinísticas

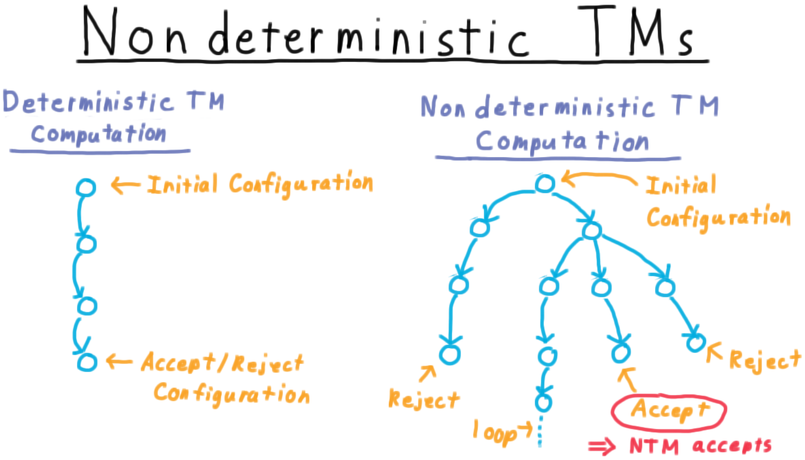


Máquinas de Turing Não-determinísticas

Ideia da Prova

- De jeito maneira.
- Se explorarmos desta forma, e a busca estiver sendo em um ramo que entra em loop, nunca poderemos concluir que a máquina aceita w em outro ramo não-determinístico.
- Utilizaremos busca em largura!
- Explorarmos todas as configurações de um mesmo nível antes de ir para o próximo.

Máquinas de Turing Não-determinísticas



Máquinas de Turing Não-determinísticas

Ideia da Prova

- Este método garante que, se N chegar ao estado de aceitação, D eventualmente também alcançará este estado na simulação.
- Agora vamos para a prova de verdade.

Máquinas de Turing Não-determinísticas

Prova

- A máquina determinística D tem 3 fitas.
- Se provarmos o resultado para uma máquina multi-fita provamos para uma máquina com uma só fita.

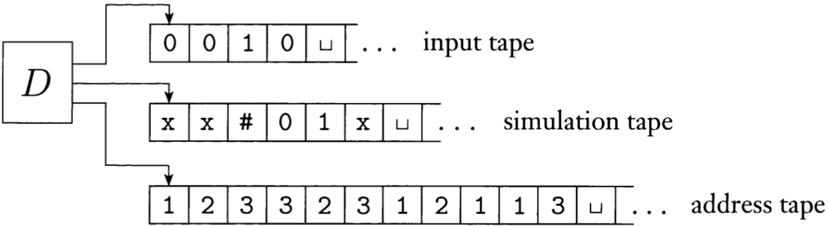
Máquinas de Turing Não-determinísticas

Prova

- Fita 1: contém a palavra w de entrada. Fita read-only, nunca é alterada.
- Fita 2: é a fita de simulação, mantém uma cópia da fita de N em uma computação de um ramo não-determinístico.
- Fita 3: guarda a posição de D na árvore de computação não-determinística.

Máquinas de Turing Não-determinísticas

Prova



Máquinas de Turing Não-determinísticas

Prova

- Vamos entender melhor a fita 3.
- Cada nó da árvore de computação tem b filho, onde b é o tamanho do maior conjunto de escolhas não determinísticas durante a computação de N.
- Cada nó vai ter um endereço na árvore de computação que corresponde a uma palavra sobre o alfabeto $\Sigma_b = \{1, 2, \ldots, b\}$.

Máquinas de Turing Não-determinísticas

Prova

- Por exemplo, o endereço 231 corresponde ao nó alcançável a partir do percurso a partir raiz, pelos nós u, v e w.
  - u é o segundo filho da raiz.
  - v é o terceiro filho de u.
  - w é o primeiro filho de v.
- Obviamente, o endereço da raiz é a palavra ϵ.

Máquinas de Turing Não-determinísticas

Prova

- Cada símbolo fala qual a próxima escolha a fazer quando estamos simulando um passo de computação em um ramo não determinístico de N.
- Às vezes um símbolo pode não corresponder à escolha alguma.
- Neste caso o endereço é inválido e não corresponde a nó algum.

Máquinas de Turing Não-determinísticas

Prova

- Agora podemos descrever como D opera.

Máquinas de Turing Não-determinísticas

Prova

1. Inicialmente, a fita 1 contém w e as fitas 2 e 3 estão vazias.
2. Copie o conteúdo da fita 1 para a fita 2.
3. Use a fita 2 para simular N com a entrada w em um ramo não determinístico. Antes de cada passo de N, devemos consultar o próximo símbolo na fita 3 para determinar qual escolha fazer dentre as possíveis escolhas não determinísticas.

Máquinas de Turing Não-determinísticas

Prova

3.1 Se não temos mais símbolos restantes na fita 3 ou se essa escolha não determinística é inválida, abortamos este ramo de computação e vamos para o passo 4.

3.2 Se o estado de rejeição é encontrado, também vá para o passo 4.

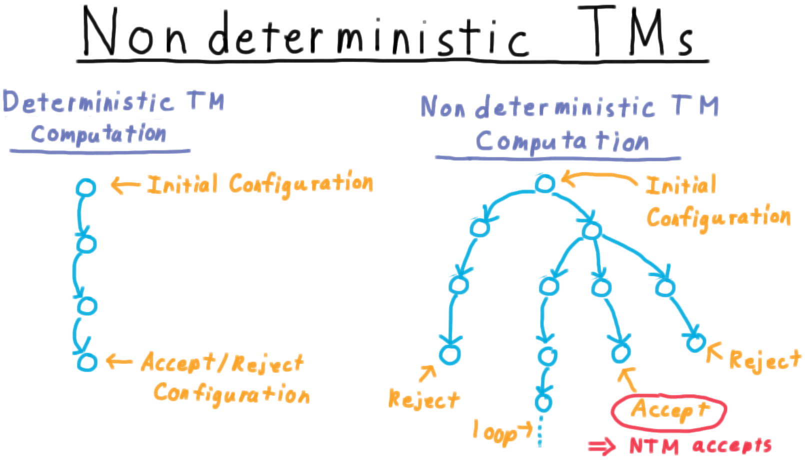
3.3 Se o estado de aceitação é encontrado, aceite w.

Máquinas de Turing Não-determinísticas

Prova

4. Troque a palavra na fita 3 pela próxima palavra na ordem lexicográfica. (Estamos efetivamente testando $(\epsilon, 1, 2, 3, \ldots, b, 11, 12, \ldots, bb, \ldots)$.
   Simule o próximo ramo da computação de N voltando para o passo 2.

Máquinas de Turing Não-determinísticas



Máquinas de Turing Não-determinísticas

Corolário

Uma linguagem e Turing-reconhecível se, e somente se alguma máquina de Turing não-determinística a reconhece.

Máquinas de Turing Não-determinísticas

Prova

$\Rightarrow$) Toda máquina de Turing determinística também é uma não determinística.

$\Leftarrow$)

Decorre imediatamente do teorema anterior.

Máquinas de Turing Não-determinísticas

Corolário

Uma linguagem e Turing-decidível se, e somente se alguma máquina de Turing não-determinística a decide, isto é, para em todos os ramos de computação não determinística (q_{aceita} ou $q_{rejeita}$).

- Para aceitação, basta um ramo atingir o estado q_{aceita}.
- Para rejeição, todos os ramos devem alcançar $q_{rejeita}$.

Sumário

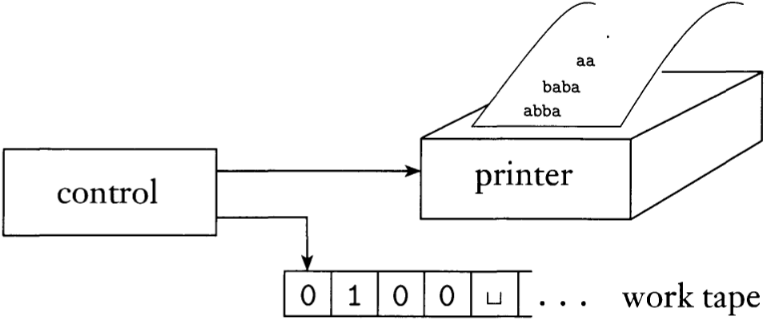
Máquinas Enumeradoras

- A classe das linguagens recursivamente enumeráveis são utilizadas como sinônimo para a classe das linguagens Turing-reconhecíveis.
- O nome recursivamente enumerável veio de uma variante de máquina de Turing, denominada **enumeradora**.

Máquinas Enumeradoras

- Uma máquina enumeradora é basicamente uma máquina de Turing com uma impressora com estoque ilimitado de papel e tinta.
- A máquina de Turing pode utilizar essa impressora para uma lista de palavras.
- Toda vez que a máquina de Turing deseja adicionar a palavra a lista, ela envia tal palavra para a impressora.

Máquinas Enumeradoras



Máquinas Enumeradoras

- Uma máquina Enumeradora E começa com uma fita vazia.
- Se a enumeradora não para, pode imprimir uma lista infinita de palavras.
- A linguagem enumerada por E é a coleção de todas as palavras impressas.
- E pode gerar as palavras em qualquer ordem e com repetições.

Máquinas Enumeradoras

Teorema

Uma linguagem é Turing-reconhecível se, e somente se uma máquina enumeradora a enumera.

Máquinas Enumeradoras

Prova

$\Leftarrow)$

Primeiramente mostraremos que, se temos uma máquina enumeradora E que enumera a linguagem A, existe uma máquina de Turing M, que reconhece a linguagem.

Máquinas Enumeradoras

Prova

A construção de M é a seguinte.

Com a entrada w, M:

1 Roda E. Toda vez que E imprime uma palavra w', M compara esta palavra com w.

1.1 Caso $w' = w$, M aceita w.

Eventualmente, se uma palavra w é impressa por E, ela será aceita por M.

Máquinas Enumeradoras

Prova

$\Rightarrow)$

Agora mostraremos que se M reconhece uma linguagem A, podemos construir uma enumeradora E para A.

Máquinas Enumeradoras

Prova

Suponha que $s_1, s_2, \ldots$ é a lista de todas as possíveis palavras sobre Σ^*.

E faz o seguinte:

1 Para $i = 1, 2, 3 \ldots$.

1.1 Rode M por i passos para cada entrada $s_1, s_2, \ldots, s_i$.

1.2 Se alguma computação é aceita, imprima o s_j correspondente.

Máquinas Enumeradoras

Prova

Se M aceita uma palavra em particular s, eventualmente ela aparecerá na lista de E.

Podemos dizer até que ela aparecerá infinitas vezes, pois M roda todas as entradas para cada elemento em $[1, i]$.

Basicamente estamos obtendo o efeito de rodar M em paralelo sobre todas as entradas possíveis.

Sumário