Máquinas de Turing: Variantes e a Definição de Algoritmo

Juliana Kaizer Vizzotto

Universidade Federal de Santa Maria

Disciplina de Teoria da Computação

Roteiro

- Reconhecedores versus Decisores
- Máquinas de Turing com Múltiplas Fitas
- Máquinas de Turing Não-Determinísticas
- Enumeradores

Reconhecedores versus Decisores

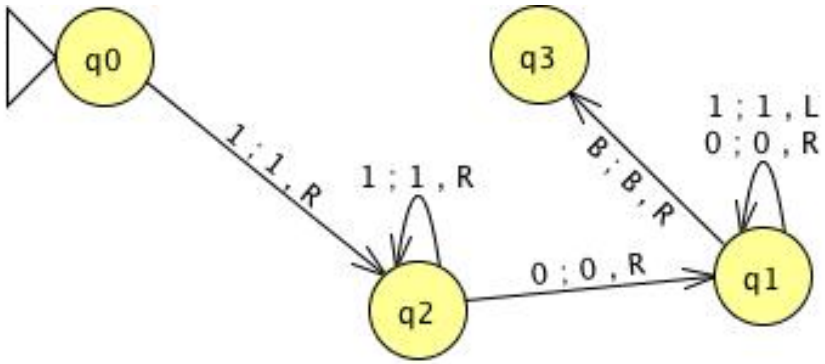
Ao executarmos uma MT, M, sobre uma entrada w, há três possíveis resultados:

- A MT eventualmente entra em um estado de aceitação, q_{ac} e pára. Assim, $w \in L(M)$.
- A MT eventualmente entra em estado de rejeição, q_{rej} ou falha em algum ponto, i.e., $w \notin L(M)$.
- Nenhum das duas situações ocorrem. Ou seja, M entra em um *loop infinito*. Nessa caso, w não é aceita nem rejeitada.

Reconhecedores versus Decisores

Uma máquina de Turing, M, é um **reconhecedor** se **reconhece** $L(M)$. Se, além disso, M nunca entra em loop infinito então M é dito um **decisor** e diz-se que decide $L(M)$.

Exemplo de MT. Considere a seguinte MT, M:



Reconhecedores versus Decisores

- O que é a linguagem L(M) da MT anterior?
- A MT M é um reconhecedor?
- A MT M é um decisor?
- A linguagem L(M) é decidível?

Respostas

M é um reconhecedor, mas não um decisor. A entrada 101, por exemplo, causa um loop infinito. A linguagem L(M) é 1^+0^+ é decidível, pois toda linguagem regular é decidível.

Variantes da MT

- Definições alternativas de Máquinas de Turing.
- Modelo original e suas variantes têm o mesmo poder de expressão, i.e., reconhecem a mesma classe de linguagens.
- Robustez da máquina de Turing.
- As MTs têm um grau surpreendente de robustez!

Exemplo

- Considere uma variação da função de transição
- Na definição que vimos até então, a cabeça obrigatóriamente deve mover para à esquerda ou para à direita.
- Suponha que tivéssemos permitido à MT ficar parada.
- A função de transição teria então a forma:

$$\sigma : Q \times \Gamma \rightarrow Q \times \Gamma \times \{E, D, P\}.$$

- Você acha que essa MT reconheceria linguagens adicionais, aumentando o poder do modelo?

Exemplo (continuação)

- A resposta é não!
- Podemos converter qualquer MT com a característica "permaneça parada" para uma que não a tem!
- Cada transição "permaneça parada" pode ser substituída por duas transições, uma que move para à direita e uma que move para à esquerda.
- Esse exemplo contém a chave para mostrar a equivalência de variantes de MTs.
- Para mostrar que dois modelos são equivalentes, simplesmente precisamos mostrar que podemos simular um pelo outro.

Máquina de Turing MultiFita

- Uma **máquina de Turing Multifita** é como uma máquina de Turing comum com várias fitas.
- Cada fita tem sua própria cabeça de leitura e escrita.
- Inicialmente a entrada aparece sobre a fita 1, e as outras iniciam em branco.
- A função de transição é modificada para permitir ler, escrever e mover as cabeças em algumas ou todas as fitas simultaneamente.
- Formalmente:

$$\sigma : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times E, D, P^k,$$

tal que k é o número de fitas.

Máquina de Turing Multifita

- A expressão

$$\sigma(q_i, a_1, \ldots, a_k) = (q_j, b_1, \ldots, b_k, E, D, \ldots, E)$$

significa que, se a máquina está no estado q_i e as cabeças de 1 a k estão lendo símbolos a_1 a a_k, a máquina vai para o estado q_j, escreve os símbolos b_1 a b_k e direciona a cabeça para mover para a esquerda ou direita, ou permanecer parada, conforme especificado.

- Máquinas de Turing multifita parecem ser mais poderosas que as máquinas de Turing comuns, mas podemos mostrar que elas são equivalentes se elas reconhecem a mesma linguagem.

Máquina de Turing Multifita

Teorema: Toda máquina de Turing multifita tem uma máquina de Turing de uma única fita que lhe é equivalente.
Prova:

- Mostramos como converter uma máquina de Turing multifita M para uma máquina de Turing equivalente S de uma única fita. A ideia chave é mostrar como simular M com S.
- Digamos que M tenha k fitas. Então, S simula o efeito de k fitas armazenando sua informação na sua única fita.
- Ela usa um novo símbolo # como delimitador para separar o conteúdo das fitas diferentes.
- Além do conteúdo dessas fitas S tem de manter o registro das posições das cabeças.

Máquina de Turing Multifita

Prova (cont.):

- Ela faz isso escrevendo um símbolo de fita *com um ponto acima dele* para marcar o local onde a cabeça estaria naquela fita.
- Os símbolos da fita *marcados com um ponto* são simplesmente novos símbolos que foram adicionados ao alfabeto da fita.
- Por exemplo: S = Sobre a entrada $w = w_1 \ldots w_n$:
  - Primeiro S põe sua fita no formato que representa todas as k fitas de M. A fita formatada contém:

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

Máquina de Turing Multifita

Prova (cont.):

- Para simular um único movimento, S faz uma varredura na sua fita desde o primeiro #, que marca a extremidade esquerda, até o $(k+1)$-ésimo #, que marca a extremidade direita, de modo a determinar os símbolos sobre as cabeças virtuais. Então, S faz uma segunda passagem para atualizar as fitas conforme a maneira pela qual a função de transição de M estabelece.
- Se em algum ponto S move uma das cabeças virtuais sobre um #, essa ação significa que M moveu a cabeça correspondente para a parte previamente não lida em branco daquela fita. Portanto, S escreve um símbolo em branco nessa célula da fita e desloca o conteúdo da fita, a partir dessa célula até o # mais à direita, uma posição para a direita. Então ela continua a simulação tal qual como anteriormente.

Máquina de Turing Multifita

Teorema: Uma Linguagem é Turing-reconhecível se e somente se alguma máquina de Turing multifita a reconhece.
Prova:

- Uma linguagem Turing-reconhecível é reconhecida por uma máquina de Turing comum (com uma única fita), o que é um caso especial de uma máquina de Turing multifita. Isso prova uma direção desse corolário. A outra direção segue do Teorema anterior.

Máquina de Turing Não-Determinísticas

- Uma máquina de Turing não-determinística é aquela que em qualquer ponto em uma computação, a máquina pode proceder de acordo com várias possibilidades.

- A função de transição para uma máquina de Turing não-determinística tem a forma:

$$\sigma : Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{E, D\})$$

Máquina de Turing Não-Determinísticas

- A computação de uma máquina de Turing não-determinística é uma árvore cujos ramos correspondem a diferentes possibilidades para a máquina. Se algum ramo da computação leva ao estado de aceitação, a máquina aceita a sua entrada.
- **Teorema:** Toda máquina de Turing não-determinística tem uma máquina de Turing determinística que lhe é equivalente.
- **Ideia da Prova:** Podemos simular qualquer MT não-determinística N com uma MT determinística D. A ideia por trás da simulação é fazer D tentar todos os possíveis ramos da computação não-determinística de N. Se D em algum momento encontra um estado de aceitação em algum desses ramos, D aceita. Caso contrário a simulação de D não terminará.

Máquina de Turing Não-Determinísticas

- **Ideia da Prova (Cont):** Vemos a computação de N sobre uma entrada w como uma árvore. Cada ramo da árvore representa um dos ramos do não-determinísmo.
- Cada nó da árvore é uma configuração de N. A raiz da árvore é a configuração inicial.
- A MT D busca nessa árvore uma configuração de aceitação.
- Ideia: busca em largura, em vez de profundidade. Essa estratégia explora todos os ramos na mesma profundidade antes de continuar a explorar qualquer ramo na próxima profundidade.
- Esse método garante que D visitará todo o nó da árvore até que ela encontre uma configuração de aceitação.

Máquina de Turing Não-Determinísticas

- **Prova Formal:** A MT determinística simuladora D tem três fitas. Pelo Teorema anterior, esse arranjo é equivalente a se ter uma única fita. A máquina D usa suas três fitas de uma maneira específica:
- A fita 1 sempre contém a cadeia de entrada e nunca é alterada.
- A fita 2 mantém uma cópia da fita de N em algum ramo da sua computação não-determinística.
- A fita 3 mantém o registro da posição de D na árvore de computação não-determinística de N (fita de endereço).

Máquina de Turing Não-Determinísticas

- **Prova Formal (Cont):** Considerando a representação de dados na fita 3, todo nó na árvore pode ter no máximo b filhos, onde b é o tamanho do maior conjunto de possíveis escolhas dado pela função de transição de N.
- A cada nó na árvore associamos um endereço que é uma cadeia sobre o alfabeto $\Sigma_b = \{1, 2, \ldots, b\}$.
- Associamos o endereço 231 ao nó ao qual chegamos iniciando da raiz, indo para seu segundo filho, indo para o terceiro filho desse nó, e finalmente indo para o primeiro filho desse nó.
- Cada símbolo na cadeia nos diz que escolha fazer a seguir quando simulamos um passo em um ramo da computação não-determinística de N.
- A fita 3 contém uma cadeia sobre Σ_b. Ela representa o ramo da computação de N da raiz para o nó endereçado por essa cadeia, a menos que o end. seja inválido.

Máquina de Turing Não-Determinísticas

- **Prova Formal (Cont):** 1. Inicialmente a fita contém a entrada w e as fitas 2 e 3 estão vazias.
- 2. Copie a fita 1 para a fita 2.
- 3a. Use a fita 2 para simular N como a entrada w sobre um ramo de sua computação não determinística. Antes de cada passo de N, consulte o próximo símbolo na fita 3 para determinar qual escolha fazer entre aquelas permitidas pela função de transição de N.
- 3b. Se não restam mais símbolos na fita 3 ou se essa escolha não-determinística for inválida, aborte esse ramo indo para o estágio 4. Tb vá para o estágio 4 se uma configuração de rejeição for encontrada. Se uma configuração de aceitação for encontrada, aceita a entrada.

Máquina de Turing Não-Determinísticas

- **Prova Formal (Cont):**
- 4. Substitua a cadeia da fita 3 pela próxima cadeia na ordem lexicográfica.
- Simule o próximo ramo da computação de N indo para o estágio 2.

Máquina de Turing Não-Determinísticas

- Chamamos uma máquina de Turing de **decisor** se todos os ramos param sobre todas as entradas.

- *Corolário*: Uma linguagem é decidível se e somente se alguma máquina de Turing não-determinística a decide.

Enumeradores

- *Linguagem recursivamente enumerável* $\equiv$ linguagem Turing-reconhecível.
- Esse termo se originou de uma variante de máquina de Turing chamada **enumerador**.
- Um enumerador pode ser visto como uma máquina de Turing com uma impressora anexa.
- A MT pode usar essa impressora como um dispositivo de saída para imprimir cadeias.
- Toda vez que a máquina de Turing quiser adicionar uma cadeia á lista, ela envia a cadeia para impressora.
- **Exercício:** dê uma definição formal para enumerador.

Enumeradores

- Um enumerador, E, inicia com uma fita de entrada em branco.
- Se o enumerador não pára, ele pode imprimir uma lista infinita de cadeias.
- A linguagem enumerada por E é a coleção de todas as cadeias que ela em algum momento imprime.
- Além disso, E pode gerar as cadeias da linguagem em qualquer ordem, possivelmente com repetições.

Enumeradores

Theorem

Uma linguagem é Turing-reconhecível somente se um enumerador a enumera.

PROVA:

- Primeiro mostramos que, se tivermos um enumerador E, que enumere uma linguagem A, uma MT M reconhece A. A MT M funciona da seguinte forma: M="sobre a cadeia de entrada w:
  1. Rode E. Toda vez que E dá como saída uma cadeia, compare-a com w.
  2. Se w em algum momento aparece na saída de E, aceite.

  Claramente, M aceita aquelas cadeias que aparecem na lista de E

Enumeradores

Theorem

Uma linguagem é Turing-reconhecível somente se um enumerador a enumera.

PROVA:

- Agora, para a outra direção. Se a MT M reconhece a linguagem A, podemos construir o seguinte enumerador E para$A$. Digamos que $s_1, s_2, s_3, \ldots,$ é uma lista de todas as possíveis cadeias em Σ^* M=" Ignore a entrada:
  1. Repita o seguinte para $i = 1, 2, 3, \ldots$
  2. Rode M por i passos sobre cada entrada, $s_1, s_2, s_3, \ldots$
  3. Se quaisquer computações aceitam, imprima a s_j correspondente.

  Claramente, M aceita uma cadeia específica s, em algum momento ela vai aparecer na lista gerada por E.

Algoritmo

- Muito embora algoritmos tenham tido uma longa história na matemática, a noção em si de algoritmo não foi definida precisamente até o século XX.
- 1900, David Hilbert identificou 23 problemas matemáticos e colocou-os como um desafio para o século vindouro.
- O décimo problema na lista dizia respeito a algoritmos: "conceber um algoritmo que testasse se um polinômio tinha uma raiz inteira".
- 1936, Alonso Church e Alan Turing: : Cálculo-λ e Máquina de Turing.

A Tese de Church-Turing

Noção intuitiva de algoritmo é igual a algoritmos de máquina de Turing.