Teoria da Computabilidade

Subseção 1

Decidibilidade

seção 4.1
Introdução à Teoria da Computação. Michael Sipser. Thomson Learning, 2007.

Objetivos

- Investigar o poder de algoritmos para a solução de problemas.
- Explorar os limites da possibilidade de solução de problemas por processos algorítmicos.
- Demonstrar que certos problemas podem ser resolvidos por processos algorítmicos e outros não.

Notação

- Entrada para uma Máquina de Turing é sempre uma cadeia definida sobre um alfabeto.
- Se a entrada for outro objeto, o mesmo deve ser representado como uma cadeia.
  - Cadeias podem representar polinômios, grafos, gramáticas, autômatos e combinações de tais objetos.
- Codificação:
  - $\langle O \rangle$ representa o objeto O.
  - $\langle O_1, O_2, \ldots, O_k \rangle$ representa os objetos $O_1, O_2, \ldots, O_k$.
- A codificação em si pode ser feita de diversos modos.
  - Uma Máquina de Turing sempre pode traduzir uma codificação para outra qualquer!

Máquina de Turing Universal

- Máquina de Turing capaz de simular qualquer outra máquina de Turing.
- A máquina deve conter na fita:
  - O conjunto de instruções sobre o comportamento da máquina a ser simulada;
  - O conteúdo da fita da máquina a ser simulada.
- Possibilita respotas questões sobre o comportamento de outras máquinas de Turing.
  - Muitas dessas questões são indecidíveis, ou seja, a função em questão não pode ser calculada por nenhuma máquina de Turing.
  - Ex: Problema de determinar se uma máquina de Turing em particular vai parar para uma entrada dada (ou para qualquer entrada) é indecidível.
  - Ex: O teorema de Rice mostra que qualquer questão não-trivial sobre o comportamento ou saída de uma máquina de Turing é indecidível.

Linguagens Decidíveis

Linguagens Regulares

- Problemas decidíveis:
  - Um dado autômato finito aceita uma cadeia em particular?
  - A linguagem de um autômato finito é vazia?
  - Dois autômatos finitos são equivalentes?
- Outros problemas computacionais podem ser formulados como a pertinência a uma certa linguagem.
  - Mostrar que a linguagem é decidível equivale a mostrar que o problema computacional é decidível.

Problema da aceitação de uma cadeia w por um AFD A
Como escrever esse problema em forma de uma linguagem?

Linguagens Decidíveis

Linguagens Regulares

Problema da aceitação para DFA's

- $\mathcal{L}_{DFA} = \{\langle A, w\rangle \mid A \text{ é um } DFA \text{ que aceita a cadeia } w\}$.
  - Codificações de todos os DFA's com as cadeias que os mesmos aceitam.
- Testar se $\langle A, w\rangle$ pertence à linguagem $\mathcal{L}_{DFA}$ equivale a testar se o DFA A aceita a cadeia w.

Linguagens Decidíveis

Linguagens Regulares

Teorema 3.1

A linguagem $\mathcal{L}_{DFA}$ é decidível.

Esquema da prova.

- Máquina de Turing M_1 que decide $\mathcal{L}_{DFA}$:
  1. Simular o DFA A com a cadeia w.
  2. Se a simulação termina em um estado que não é final, `rejeite`.
  3. Caso contrário, `aceite` (parou em um estado de aceitação).

□

Linguagens Decidíveis

Linguagens Regulares

Teorema 3.1

A linguagem $\mathcal{L}_{DFA}$ é decidível.

Esquema da prova.

- Codificação $\langle A, w \rangle$: entrada da máquina M_1:
  - Representação do DFA A com a cadeia w.
  - A : lista de seus componentes, ou seja, Σ, S, s_0, δ e F.
  - w : cadeia de entrada de A.

□

Linguagens Decidíveis

Linguagens Regulares

Teorema 3.1

A linguagem $\mathcal{L}_{DFA}$ é decidível.

Esquema da prova.

- Comportamento da máquina M_1:
  - M_1 recebe $\langle A, w \rangle$ e testa a codificação. Se inválida, `rejeita`.
  - Fita é usada para acompanhar o estado corrente do DFA e símbolo corrente da cadeia w.
  - Inicialmente, s_0 é o estado inicial de A e o símbolo corrente é o símbolo mais a esquerda de w.
  - Estado e símbolo atualizados de acordo com função de transição δ.
  - M_1 `aceita` a entrada $\langle A, w \rangle \Rightarrow M_1$ processa o último símbolo de w e A está em um estado final.
  - M_1 `rejeita` a entrada $\langle A, w \rangle \Rightarrow M_1$ processa o último símbolo de w e A não está em um estado final.

□

Linguagens Decidíveis

Linguagens Regulares

$\mathcal{L}_{NFA} = \{\langle N, w\rangle \mid N \text{ é um } NFA \text{ que aceita a cadeia } w\}$.

Teorema 3.2

A linguagem $\mathcal{L}_{NFA}$ é decidível.

Esquema da prova.

- Máquina de Turing M_2 que decide $\mathcal{L}_{NFA}$:
  1. M_2 converte o NFA N em um equivalente DFA A.
  2. M_2 chama a máquina M_1 com a codificação $\langle A, w\rangle$.
     - Máquina de Turing M_2 pode usar a máquina M_1 como subrotina!!!
  3. Se M_1 aceita, M_2 `aceita`. Caso contrário, `rejeita`.

□

Linguagens Decidíveis

Linguagens Regulares

$\mathcal{L}_{ER} = \{\langle \mathcal{R}, w \rangle \mid \text{Cadeia } w \text{ é gerada pela expressão regular } \mathcal{R}\}$.

Teorema 3.3

A linguagem $\mathcal{L}_{ER}$ é decidível.

Esquema da prova.

- Máquina de Turing M_3 que decide $\mathcal{L}_{ER}$:
  1. M_3 converte a expressão regular $\mathcal{R}$ em um equivalente DFA A.
  2. M_3 chama a máquina M_1 com a codificação $\langle A, w \rangle$.
  3. Se M_1 aceita, M_3 `aceita`. Caso contrário, `rejeita`.

□

Linguagens Decidíveis

Linguagens Regulares

$\mathcal{L}_\emptyset = \{\langle A \rangle \mid A \text{ é um } DFA \text{ e } \mathcal{L}(A) = \emptyset\}$.

Teorema 3.4

A linguagem $\mathcal{L}_\emptyset$ é decidível.

Esquema da prova.

- Um DFA aceita uma cadeia se e somente se é possível alcançar, a partir do estado inicial, um estado final.
- Máquina de Turing M_4 que decide $\mathcal{L}_\emptyset$:
  1. Marque o estado inicial de A.
  2. Repita o passo 3 enquanto possível:
  3. Para cada estado marcado, marque aqueles alcançáveis com a função de transição.
  4. Se nenhum estado final está marcado, `aceite`. Caso contrário, `rejeite`.

□

Linguagens Decidíveis

Linguagens Regulares

$\mathcal{L}_{AB} = \{\langle A, B\rangle \mid A \text{ e } B \text{ são } DFA\text{'s e } \mathcal{L}(A) = \mathcal{L}(B)\}$.

Teorema 3.5

A linguagem $\mathcal{L}_{AB}$ é decidível.

Esquema da prova.

- DFA C aceita cadeias aceitas só por A ou só por B.
- $\mathcal{L}(A) = \mathcal{L}(B) \Rightarrow \mathcal{L}(C) = \emptyset$.
- $\mathcal{L}(C) = \Big(\mathcal{L}(A) \cap \overline{\mathcal{L}(B)}\Big) \cup \Big(\overline{\mathcal{L}(A)} \cap \mathcal{L}(B)\Big)$.

□

Linguagens Decidíveis

Linguagens Regulares

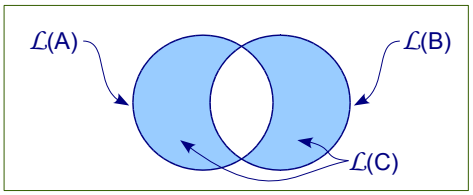
$\mathcal{L}_{AB} = \{\langle A, B\rangle \mid A \text{ e } B \text{ são } DFA\text{'s e } \mathcal{L}(A) = \mathcal{L}(B)\}$.

Teorema 3.5

A linguagem $\mathcal{L}_{AB}$ é decidível.

Esquema da prova.

- Diferença simétrica de $\mathcal{L}(A)$ e $\mathcal{L}(B)$:



□

Linguagens Decidíveis

Linguagens Regulares

$\mathcal{L}_{AB} = \{\langle A, B\rangle \mid A$ e B são DFA's e $\mathcal{L}(A) = \mathcal{L}(B)\}$.

Teorema 3.5

A linguagem $\mathcal{L}_{AB}$ é decidível.

Esquema da prova.

- Construção do DFA C:
  - Idéia usada para mostrar que linguagens regulares são fechadas para as operações de complementação, união e interseção.
- Máquina de Turing M_5 que decide $\mathcal{L}_{AB}$:
  1. Construir o DFA C como descrito.
  2. M_5 chama a máquina M_4 com a codificação $\langle C\rangle$.
  3. Se M_4 aceita, M_5 `aceita`. Caso contrário, `rejeita`.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

Codificações de todas as Gramáticas com as cadeias que as mesmas geram.

$\mathcal{L}_{GLC} = \{\langle G, w\rangle \mid G$ é uma GLC que gera a cadeia $w\}$.

Teorema 3.6

A linguagem $\mathcal{L}_{GLC}$ é decidível.

Esquema da prova.

- Usar G para listar todas as derivações e determinar se alguma leva a w.
  - Infinitas derivações podem ter de ser verificadas.
  - Se G não gera w, o algoritmo nunca termina.
  - Gera máquina de Turing que reconhece $\mathcal{L}_{GLC}$, mas não a decide.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

$\mathcal{L}_{GLC} = \{\langle G, w\rangle \mid G \text{ é uma } GLC \text{ que gera a cadeia } w\}$.

Teorema 3.6

A linguagem $\mathcal{L}_{GLC}$ é decidível.

Esquema da prova.

- G na forma normal de Chomsky $\Rightarrow$ derivação de w é feita em $2.|w| - 1$ passos.
- Máquina de Turing M_6 que decide $\mathcal{L}_{GLC}$:
  1. Converter G para a forma normal de Chomsky.
  2. Se $|w| > 0$, listar todas as derivações com $2.|w| - 1$ passos, senão listar todas com 1 passo.
  3. Se qualquer de tais derivações gera w, M_6 `aceita`. Caso contrário, `rejeita`.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

$\mathcal{L}_{LC_\emptyset} = \{\langle G \rangle \mid G \text{ é uma } GLC \text{ e } \mathcal{L}_G = \emptyset\}$.

Teorema 3.7

A linguagem $\mathcal{L}_{LC_\emptyset}$ é decidível.

Esquema da prova.

- Usar máquina de Turing M_6 para verificar se G gera alguma cadeia w?
  - Para determinar que $\mathcal{L}_{LC_\emptyset} = \emptyset$, todas as possíveis cadeias w devem ser testadas.
  - Processo pode não terminar, dado o número infinito de possíveis cadeias w.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

$\mathcal{L}_{LC_\emptyset} = \{\langle G \rangle \mid G \text{ é uma } GLC \text{ e } \mathcal{L}_G = \emptyset\}$.

Teorema 3.7

A linguagem $\mathcal{L}_{LC_\emptyset}$ é decidível.

Esquema da prova.

- Testar se variável inicial pode gerar uma cadeia de símbolos terminais.
- Determinar quais variáveis podem gerar cadeias de terminais.
- Marcar tais variáveis.
- No final verificar se a variável inicial está marcada.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

$\mathcal{L}_{LC_\emptyset} = \{\langle G \rangle \mid G \text{ é uma } GLC \text{ e } \mathcal{L}_G = \emptyset\}$.

Teorema 3.7

A linguagem $\mathcal{L}_{LC_\emptyset}$ é decidível.

Esquema da prova.

- Máquina de Turing M_7 que decide $\mathcal{L}_{LC_\emptyset}$:
  1. Marcar todos os símbolos terminais de G.
  2. Repetir o passo 3 enquanto possível.
  3. Marcar as variáveis A tais que a regra $A \to U_1 U_2 \ldots U_k$ pertence a G e cada símbolo $U_1 U_2 \ldots U_k$ já está marcado.
  4. Se o símbolo inicial não está marcado, `aceite`. Caso contrário, `rejeite`.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

$\mathcal{L}_{GH} = \{\langle G, H\rangle \mid G \text{ e } H \text{ são } GLC\text{'s e } \mathcal{L}(G) = \mathcal{L}(H)\}$.

Teorema 3.8

A linguagem $\mathcal{L}_{GH}$ não é decidível.

Esquema da prova.

- A classe de linguagens livres de contexto não é fechada para as operações de complementação e interseção.
- Demonstração do teorema será abordada no tópico `Reduções`.

□

Linguagens Decidíveis

Linguagens Livres de Contexto

Teorema 3.9

Toda linguagem livre de contexto é decidível.

Esquema da prova.

- Considerar uma LLC arbitrária $\mathcal{L}_{llc}$.
- Mostrar que $\mathcal{L}_{llc}$ é decidível.
- Considerar uma GLC G_{llc} para $\mathcal{L}_{llc}$.
- Máquina de Turing M_9 que decide $\mathcal{L}_{llc}$:
  1. M_9 chama a máquina M_6 com a codificação $\langle G_{llc}, w \rangle$.
  2. Se M_6 aceita, M_9 `aceita`. Caso contrário, `rejeita`.

□

Relação entre Classes de Linguagens