Introdução à Teoria da Computação

Complexidade Computacional - Definições básicas

Professor Murilo V. G. da Silva

Departamento de Informática

Universidade Federal do Paraná

06/03/2021

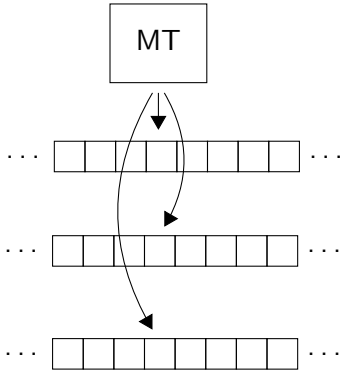
Complexidade Computacional

(1) Dados dois números inteiros, calcular a soma dos dois números;
(2) Dado um tabuleiro de xadrez em que as peças brancas têm a vez de jogar, determinar a jogada ótima para as peças brancas.

- Parece "óbvio" que $z = x + y$ é um problema simples.
- Porém, z tem pelo menos 64 dígitos (existem 10^{64} números com o tamanho de z).
- Mais importante: Somar com n dígitos *vs* "xadrez generalizado" (tamanho n).
- Obs: Não confundir *problemas indecidíveis* com *problemas intratáveis*.

Complexidade de Tempo e de Espaço de Máquinas de Turing

Modelo de computação:



Complexidade de Tempo de Máquinas de Turing

Seja M uma MT,

Complexidade de tempo

A complexidade de tempo de M é uma função $t_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer string de entrada de tamanho n, a máquina para depois de executar no máximo $t_{\mathrm{M}}(n)$ transições.

Exemplo: Se $\forall w \in \Sigma^*$ com $|w| = n$, M sempre para depois de fazer, no máximo, $n^2 + 3n$ transições, dizemos que a complexidade de tempo de M é $n^2 + 3n$.

Complexidade de Espaço de Máquinas de Turing

Seja M uma MT,

Complexidade de espaço

A complexidade de espaço de M é uma função $s_{\mathrm{M}} : \mathbb{N} \to \mathbb{N}$ tal que, para qualquer string de entrada de tamanho n, a máquina M para usando no máximo $s_{\mathrm{M}}(n)$ posições da fita 2.

Exemplo: Se $\forall w \in \Sigma^*$ com $|w| = n$, M sempre para usando no máximo $\log n + 7$ posições da fita 2, dizemos que a complexidade de espaço de M é $\log n + 7$.

Mais definições

Notação: $poli(n)$

Se $f(n) = O(n^r)$ para algum $r \in \mathbb{N}$ constante, então dizemos que $f(n) = poli(n)$.

MT com complexidade polinomial

Se M tem complexidade de tempo $poli(n)$, então dizemos que *M é polinomial*.

Se M tem complexidade de espaço $poli(n)$, dizemos que *M é de espaço polinomial*.

Complexidade em MTs não determinísticas

Uma MTN N é polinomial se dada uma entrada de tamanho n, *todos os ramos da árvore de computações possíveis tem profundidade* $poli(n)$.

Note: Independente das escolhas não determinísticas, ela sempre faz $poli(n)$ transições

Decidindo problemas

Seja L uma linguagem.

Decisão em tempo polinomial

- Se $\exists$ MT polinomial que decide L, então dizemos que L pode ser *decidida deterministicamente em tempo polinomial*
- Se $\exists$ MTN polinomial que decide L, então dizemos que L pode ser *decidida não deterministicamente em tempo polinomial*.

Decisão em espaço polinomial

- Se $\exists$ MT de espaço polinomial que decide L, então dizemos que L pode ser *decidida deterministicamente em espaço polinomial*
- Se $\exists$ MTN de espaço polinomial que decide L, então dizemos que L pode ser *decidida não deterministicamente em espaço polinomial*.

Decisão em tempo ou espaço exponencial: definições análogas

As classes **P**, **NP** e **PSPACE** e **EXP**

A classe **P**

Linguagens decidíveis deterministicamente em tempo polinomial.

A classe **NP**

Linguagens decidíveis não deterministicamente em tempo polinomial.

A classe **EXP**

Linguagens decidíveis deterministicamente em tempo exponencial.

A classe **PSPACE**

Linguagens decidíveis deterministicamente em espaço polinomial.

O problema **P** *vs* **NP**

Teorema

$P \subseteq NP$

Prova: Seja $L \in P$.

1. Existe uma MT polinomial $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ que decide L.
2. Agora considere a Máquina de Turing não determinística $N = (Q, \Sigma, \Gamma, (\delta, \delta), q_0, B, F)$.
3. A máquina N comporta-se exatamente da mesma maneira que M, portanto N também decide L em tempo polinomial.
4. Logo $L \in NP$ e consquentemente $P \subseteq NP$.

- $P \neq NP$?
- E as classes P-space e NP-space?

TIME($f(n)$) e SPACE($f(n)$)

TIME($f(n)$)

Dada uma função $f : \mathbb{N} \to \mathbb{N}$, o conjunto de toda linguagem que pode ser decidida por uma MT com complexidade de tempo $c \cdot f(n)$, para $c > 0$, é denotado por TIME($f(n)$).

Exemplo: TIME(n^3) é a classe de toda linguagem que pode ser decidida por uma MT que execute $c \cdot n^3$ transições, $c > 0$, sendo n o tamanho da string de entrada.

Exemplo: A classe TIME(2^n), que é conjunto de todas as linguagem que podem ser decidida por uma MT que execute $c \cdot 2^n$ transições, $c > 0$, transições, tal que n é o tamanho da string de entrada.

SPACE($f(n)$)

Dada uma função $f : \mathbb{N} \to \mathbb{N}$, o conjunto de toda linguagem que pode ser decidida por uma MT com complexidade de espaço $c \cdot f(n)$, para $c > 0$, é denotado por SPACE($f(n)$).

Definições auxiliares

Conjuntos, Listas, Permutações:

- Como de custume, usamos chaves para denotar conjuntos
  Exemplo: $A = \{1, 5, 9, 7\}$
- No caso de listas, usamos colchetes
  Exemplo: $A = [1, 5, 9, 7, 1]$ (existe ordem e possibilidade de repetição)

  Usamos a notação $A[1] = 1$, $A[2] = 5$, $A[3] = 9$, ... (semelhante a vetor)
  Também usamos a notação $A_1 = 1$, $A_2 = 5$, $A_3 = 9$, ...
- Listas, vetores e tuplas são tratadas como equivalentes
  (strings também tratadas como tuplas quando conveniente)
- Permutação de um conjunto S com n elementos:
  Definido como uma lista de n elementos de S sem repetição.

Notação para Intervalo Inteiro: Dados $a, b \in \mathbb{Z}$, $[a..b] = \{z \in \mathbb{Z} \mid a \leq z \leq b\}$

Grafos, grafos direcionados e grafos ponderados

Um *grafo* G é um par $(V(G), E(G))$ onde

- $V(G)$ é um conjunto finito, chamado de conjunto de vértices
- $E(G)$ é um conjunto onde cada elemento é um conjunto de dois vértices, chamado de conjunto de arestas

Um *grafo direcionado* G é um par $(V(G), E(G))$ onde

- $V(G)$ é um conjunto finito, chamado de conjunto de vértices
- $E(G)$ é um conjunto de pares de vértices, chamado de conjunto de arcos

Notação simplificada: $G = (V, E)$ ao invés de $G = (V(G), E(G))$

Quando o $V(G)$ não é definido explicitamente, a convenção é $V(G) = [1..n]$.

Notação simplificada: uv denota a aresta $\{u, v\}$

Grafos, grafos direcionados e grafos ponderados

Um *grafo ponderado* é um par (G, w) onde G é um grafo e w é uma função que associa a cada aresta a de G um *peso* $w(a)$.

- Grafos direcionados ponderados são definidos de maneira análoga.
- Para simplificar, às vezes diremos "grafo ponderado G" ao invés de "grafo ponderado (G, w)".

- Convenção: Se G não é um grafo ponderado, $\forall e \in E(G)$, $w(e) = 1$.
- Notação simplificada: Se $\{u, v\} \in E(G)$, escrevemos $w(u, v)$ para o peso de $\{u, v\}$. (ao invés de escrever $w(\{u, v\})$

Matriz de Adjacência, propriedades de grafos

Matriz de Adjacência de um grafo (ponderado) G: A *matriz de adjacência* de G é a matriz M_G indexada por $V(G) \times V(G)$ dada por

$$\mathrm{M}_G[u, v] = \begin{cases} w(u, v), & uv \in E(G), \\ 0, & uv \notin E(G). \end{cases}$$

Uma clique em um grafo G: um conjunto $S \subseteq V(G)$ tal que
$\forall u, v \in S$, $uv \in E(G)$ (i.e., todo par de vértices em S é adjacente)

Um conjunto independente em um grafo G: um conjunto $S \subseteq V(G)$ tal que $\forall u, v \in S$, $uv \notin E(G)$

(i.e., nenhum par de vértices em S é adjacente)

Matriz de Adjacência, propriedades de grafos

Grafo completo: existe uma aresta entre cada par de vértices do grafo
(i.e., $V(G)$ é uma clique)

Grafo conexo: existe um caminho entre cada par de vértices

Grafo hamiltoniano: existe ciclo passando por todos os vértices (sem repetir)
(definição precisa no próximo slide)

Grafo euleriano: existe passeio vizitando todas as arestas (sem repetir)
(definição precisa no próximo slide)

Árvore: grafo conexo e acíclico

Grafos Hamiltonianos

Seja G um grafo com n vértices

Seja $\pi = [\pi_1, ..., \pi_n]$ uma permutação de $V(G)$.

- obs: o elemento π_{n+1} se refere à π_1.

Definição de Circuito Hamiltoniano: A permutação π é um **circuito hamiltonino** de G se $\{\pi_i, \pi_{i+1}\} \in E(G)$, $i = 1, ..., n$.

- Isto é, existe um ciclo passando por todos os vértices do grafo
- Chamados de ciclos hamiltonianos.
- Grafos que admitem circuitos hamiltonianos são chamados de grafos hamiltonianos.

Definição de Caminho Hamiltoniano: A permutação π é um **caminho hamiltonino** de G se $\{\pi_i, \pi_{i+1}\} \in E(G)$, $i = 1, ..., n-1$.

- Isto é, existe um caminho passando por todos os vértices do grafo (não necessariamente voltando ao vértice inicial)

Grafos Eulerianos

Seja G um grafo com m arestas

Seja $\pi = [\pi_1, ..., \pi_m]$ uma permutação de $E(G)$.

- obs: o elemento π_{n+1} se refere à π_1.

Definição de Circuito Euleriano: A permutação π é um **circuito euleriano** de G se para todo $i \in [1..n]$, $\pi_i = uv$ e $\pi_{i+1} = vw$, para vértices u, v, w de G.

- Isto é, existe um passeio visitando todas as arestas do grafo
- Chamados de circuitos eulerianos.
- Grafos que admitem circuitos eulerianos são chamados de grafos eulerianos.

Problema de Decisão em Grafos

Exemplos de problemas de decisão em grafos:

- $L_{\text{CNX}} = \{\llcorner G \lrcorner \in \Sigma^* : G \text{ é um grafo conexo}\}$.
- $L_{\text{EU}} = \{\llcorner G \lrcorner \in \Sigma^* : G \text{ é um grafo euleriano}\}$
- $L_{\text{HAM}} = \{\llcorner G \lrcorner \in \Sigma^* : G \text{ é um grafo hamiltoniano}\}$.

Para problemas de decisão em geral (não apenas sobre grafos):

x é uma instância verdadeira do problema L: sinônimo de $x \in L$.
x é uma instância falsa do problema L: $x \notin L$.

Exercícios

Responda as questões abaixo:

1. Quantas arestas contém um grafo completo com n vértices?
2. É verdade que todo grafo completo também é conexo?
3. É verdade que todo grafo conexto também é uma árvore
4. Desenhe todos os grafos de 4 vértices[1].
5. Apresente um grafo de 4 vértices que seja conexo, mas que não seja nem árvore nem grafo hamiltoniano.
6. Enumere todas as cliques fo grafo da resposta da Questão 5.
7. É verdade que todo grafo completo também é um grafo hamiltoniano?

[1] Lembre: quando conjunto de vértices do grafo não é especificado, por convenção usamos naturais 1,2,3,...

Problema SAT

Uma fórmula booleana na *forma normal conjuntiva*: formula é uma conjunção de disjunções.

- Diremos "fórmulas booleanas em CNF"

Satisfatibilidade de fórmulas booleanas (SAT): O problema de decidir se uma dada fórmula booleana em CNF é satisfazível

$$L_{\text{SAT}} = \{\llcorner\phi\lrcorner \in \Sigma^* : \phi \text{ é uma fórmula booleana em CNF satisfazível}\}.$$

Como a fórmula $\phi_1 = (x_1 \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_3})$ é satisfazível, dizemos que $\llcorner\phi_1\lrcorner$ é uma instância verdadeira de L_{SAT}.

Por outro lado, como $\phi_2 = (\overline{x_1} \vee \overline{x_2}) \wedge (x_1) \wedge (x_2)$ não é satisfazível, dizemos que $\llcorner\phi_2\lrcorner$ é uma instância falsa de L_{SAT}.

- Em fórmulas CNF, cada x_i ou $\overline{x_i}$ é chamado de literal
  e.g., em ϕ_1 os literais são: x_1, x_2 $\overline{x_2}$ e $\overline{x_3}$
- Cada disjunção na fórmula é chamada de cláusula
  e.g. as cláusulas de ϕ_2 são $(\overline{x_1} \vee \overline{x_2})$, (x_1) e (x_2)