TEORIA DE COMPLEXIDADE

Fundamentos: classes $\mathcal{P}$ e $\mathcal{NP}$

Mauricio Ayala-Rincón

Grupo de Teoria da Computação
http://ayala.mat.unb.br/TCgroup
Instituto de Ciências Exatas
Universidade de Brasília, Brasília D.F., Brazil

Brasília
Dez. 13, 2006

Organização

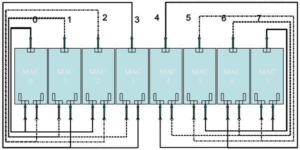
Problemas
- Tratados até o momento
- Problemas de decisão e de otimização

Classes $\mathcal{P}$ e $\mathcal{NP}$
- $\mathcal{P}$ e tratabilidade
- Algoritmos não determinísticos e $\mathcal{NP}$

Problemas $\mathcal{NP}$-completos
- Redução polinomial
- Problemas $\mathcal{NP}$-completos

Problemas Previamente Tratados

- Soluções algorítmicas de ordem limitado polinomialmente



Space efficient FFT by dynamic reconfiguration of interconnections and logical components

$(\Theta(n^3))$

- Solução polinomial: tratável
- Sem solução polinomial conhecida: intratável

Problemas de decisão e de otimização

Coloração de Grafos: dados $G = (V, E)$ e S, achar $C : V \rightarrow S$ tal que se $(u, v) \in V$, então $C(u) \neq C(v)$.
número cromático de G: mínima $|Imagem(C(V))|$, denotado $\mathcal{X}(G)$
Problema de otimização: dado G, determine $\mathcal{X}(G)$ e produza uma coloração ótima
Problema de decisão: dados G e $k \geq 0$, determine se existe C com $|Imagem(C(V))| \leq k$

Problemas de decisão e de otimização

Execução de tarefas com penalidades: dadas seqüências

$$J_1, \ldots, J_n \quad t_1, \ldots, t_n \quad d_1, \ldots, d_n \quad p_1, \ldots, p_n$$

respectivamente, de tarefas, tempos de execução, prazos limite e penalidades, uma execução das tarefas é uma permutação π de $[1, \ldots, n]$. A penalidade de π é

$$P_\pi := \Sigma_{k=1}^{n}[\text{if } t_{\pi(1)} + \cdots + t_{\pi(k)} > d_{\pi(k)} \text{ then } p_{\pi(k)} \text{ else } 0]$$

Problema de otimização: determine a penalidade mínima.
Problema de decisão: dado, adicionalmente $k \in \mathbb{N}$, determine se existe uma execução π com $P_\pi \leq k$

Problemas de decisão e de otimização

Empacotamento (bin packing): Dado um número ilimitado de caixas de capacidade 1 e n objetos com tamanhos $s_1, \dots, s_n$, onde $0 \leq s_i \leq 1$, para todo $1 \leq i \leq n$.

Problema de otimização: determine o menor número de caixas para empacotar os n objetos

Problema de decisão: dado, adicionalmente $k \in \mathbb{N}$, determine se existe um empacotamento dos n objetos em k caixas

Problemas de decisão e de otimização

Problema da Mochila (Knapsack): Dada uma mochila de capacidade c e n objetos com tamanhos $s_1, \ldots, s_n$, com índices de proveito $p_1, \ldots, p_n$.
Problema de otimização: encontre o máximo proveito de subconjuntos de objetos que não excedam a capacidade da mochila
Problema de decisão: dado, adicionalmente $k \in \mathbb{N}$, determine se existe um subconjunto de objetos que não excede a capacidade da mochila e apresenta um índice de proveito de pelo menos k

Problemas de decisão e de otimização

Suma de subconjuntos: Dados $C \in \mathbb{N}$ e n objetos com tamanhos $s_1, \ldots, s_n$.
Problema de otimização: encontre o subconjunto de objetos com máximo tamanho total que não exceda C
Problema de decisão: determine se existe um subconjunto de objetos com tamanho total de exatamente C

Problemas de decisão e de otimização

CNF-Satisfazibilidade (CNF-SAT):
Um literal é uma variável booleana p ou sua negação $\neg p$. Uma cláusula é uma *disjunção* de literais. Uma fórmula lógica em forma normal conjuntiva é uma *conjunção* de cláusulas. Exemplo

$$(p \vee q \vee r \vee \neg s) \wedge (r \vee \neg q \vee \neg p) \wedge (s \vee \neg r)$$

Problema de decisão: determine se existe uma atribuição de valores *true* e *false* para as variáveis booleanas de uma fórmula lógica em FNC tal que a fórmula é *true*

Problemas de decisão e de otimização

Caminhos e circuitos Hamiltonianos: Dado um grafo não dirigido (digrafo), $G = (\{v_1, \ldots, v_n\}, E)$, um caminho Hamiltoniano em G é um permutação π de $[1, \ldots, n]$, tal que para todo $1 \leq i < n$, $(v_{\pi(i)}, v_{\pi(i+1)}) \in E$.
Se adicionalmente $(v_{\pi(n)}, v_{\pi(1)}) \in E$, diz-se que o caminho é um circuito Hamiltoniano.
Problema de decisão: dado um digrafo G, determine se existe um caminho (circuito) Hamiltoniano em G.

Problemas de decisão e de otimização

Tour mínimo - Problema do caixeiro viajante:
Dado um grafo (ou digrafo) com pesos, $G = (\{v_1, \ldots, v_n\}, E, p)$, onde $p : E \rightarrow \mathbb{R}^*$, um circuito Hamiltoniano π em G tem custo

$$\Sigma_{i=1}^{n-1} p(\ (v_{\pi(i)}, v_{\pi(i+1)})\ ) \quad + \quad p(\ (v_{\pi(n)}, v_{\pi(1)})\ )$$

Problema de otimização: dado um grafo (ou digrafo) com pesos, encontre o circuito hamiltoniano de custo mínimo.
Problema de decisão: dado adicionalmente um inteiro k, determine se existe um circuito Hamiltoniano com peso menor ou igual que k.

Classes $\mathcal{P}$ e $\mathcal{NP}$

- Não é conhecida solução *razoável* para nenhum dos problemas da seção precedente!

Def. Um algoritmo $\mathcal{A}$ é polinomialmente limitado se a sua complexidade tempo pior-caso, $W_{\mathcal{A}}(n) \leq p(n)$ para algum polinômio p.

Def. Um problema de decisão é dito pertencer à classe $\mathcal{P}$ se existe um algoritmo limitado polinomialmente que o soluciona.

Nota os algoritmos vistos na disciplina para ordenação, busca, reconhecimento de padrões, multiplicação de matrizes, FFT, etc., exceto empacotamento em caixas, são limitados polinomialmente. Dessa forma todos eles, exceto empacotamento em caixas, pertencem a classe $\mathcal{P}$.

Classes $\mathcal{P}$ e $\mathcal{NP}$

- Não é conhecida solução *razoável* para nenhum dos problemas da seção precedente!

Def. Um algoritmo $\mathcal{A}$ é polinomialmente limitado se a sua complexidade tempo pior-caso, $W_{\mathcal{A}}(n) \leq p(n)$ para algum polinômio p.

Def. Um problema de decisão é dito pertencer à classe $\mathcal{P}$ se existe um algoritmo limitado polinomialmente que o soluciona.

Nota os algoritmos vistos na disciplina para ordenação, busca, reconhecimento de padrões, multiplicação de matrizes, FFT, etc., exceto empacotamento em caixas, são limitados polinomialmente. Dessa forma todos eles, exceto empacotamento em caixas, pertencem a classe $\mathcal{P}$.

Classes $\mathcal{P}$ e $\mathcal{NP}$

Identificar tratabilidade com pertenencia a $\mathcal{P}$ é justificado, sempre que

- $\notin \mathcal{P}$ implica intratabilidade
- Propriedades de fecho de polinômios:
  - $*$, $+/-$, $\circ$ de polinômios é um polinômio
- Definição de $\mathcal{P}$ não depende do modelo computacional

Classes $\mathcal{P}$ e $\mathcal{NP}$

Intuitivamente $\mathcal{NP}$ é a classe de problemas, cujas soluções podem ser verificadas em tempo limitado polinomialmente.

Def. Um algoritmo de decisão não determinístico consiste de duas fases

Fase não determinística: escreve-se uma palavra aleatoria s em algum local da memória.

Fase determinístico: executa-se um algoritmo determinístico sobre a entrada e s. O algoritmo pára com resposta *sim* ou *não*.

O número de passos de um algoritmo não determinístico é a suma de passos nas suas duas fases.

A saída de um algoritmo não determinístico é

- *sim* se para alguma execução a saída e *sim*, e
- *não* se para nenhuma execução a saída é *sim*.

Algoritmos não determinísticos e $\mathcal{NP}$

Exemplo (Coloração de Grafos) Dados um digrafo $G = (\{v_1, \ldots, v_n\}, E)$ e $k \in \mathbb{N}$.
Fase não determinística: "chute" uma seqüência aleatória

$$C_{I_1}, \ldots, C_{I_n}$$

de comprimento n sobre o alfabeto $\{C_1, \ldots, C_k\}$.
Fase determinística:

k-colorável $\leftarrow$ *true*
FOR $i = 1$ to $n - 1$ DO
    FOR $j = i + 1$ TO n DO
        IF $(v_i, v_j) \in E$ AND $C_{I_i} = C_{I_j}$ THEN
            k-colorável $\leftarrow$ *false*;
            EXIT;
RETURN k-colorável

$\mathcal{NP}$

Def. $\mathcal{NP}$ é a classe de problemas de decisão para os quais existe um algoritmo não determinístico limitado polinomialmente.

$\mathcal{NP}$

Teo. Os problemas de decisão:

- Coloração de Grafos;
- Execução de tarefas com penalidades;
- Empacotamento (bin packing);
- Problema da Mochila (Knapsack);
- Suma de subconjuntos;
- CNF-Satisfazibilidade (CNF-SAT);
- Caminhos e circuitos Hamiltonianos e
- Tour mínimo - Problema do caixeiro viajante

são $\mathcal{NP}$.

Dem. Coloração de Grafos resolve-se em $\Theta(n^2)$ com o algoritmo não determinístico apresentado.

Similarmente, soluções para os outros problemas, podem ser "chutadas" em tempo linear e verificadas em tempo polinomial no tamanho da entrada.

$\mathcal{NP}$

Teo. $\mathcal{P} \subseteq \mathcal{NP}$

Dem. Qualquer algoritmo polinomial para solucionar problemas de decisão em $\mathcal{P}$, pode-se conceber como um algoritmo não determinístico com fase inicial vazia. $\square$

A questão relevante é se $\mathcal{NP} \subseteq \mathcal{P}$

É um dos sete problemas do milênio do *Clay Mathematics Institute* e sua solução tem prêmio de \$ $1.000.000,00$.

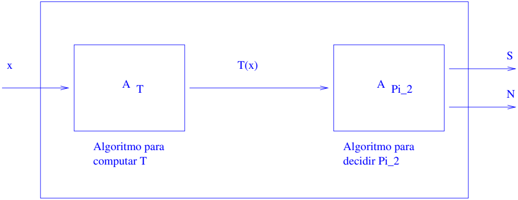
É o não determinismo mais poderoso que o determinismo?

Soluções não determinísticas limitadas polinomialmente traduzem a soluções determinísticas exponenciais: basta listar todos os possíveis "chutes" (normalmente um número exponencial de possibilidades) e verifica-los polinomialmente.

Problemas $\mathcal{NP}$-completos: redução polinomial

Def. Sejam Π_1 e Π_2 problemas de decisão e T uma função que transforma entradas para o problema Π_1 em entradas para o problema Π_2. T é uma redução polinomial ou transformação polinomial de Π_1 em Π_2 se

1. T é computável em tempo limitado polinomialmente e
2. Para cada entrada x do problema Π_1, a resposta correta de Π_2 para $T(x)$ é também correta para Π_1



Problemas $\mathcal{NP}$-completos: redução polinomial

Def. Sejam Π_1 e Π_2 problemas de decisão. Π_1 é polinomialmente redutível a Π_2 se existe uma redução polinomial T de Π_1 em Π_2.

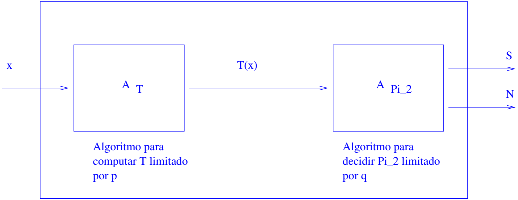
Notação: $\Pi_1 \preceq_{\mathcal{P}} \Pi_2$

Teo.

$$\Pi_1 \preceq_{\mathcal{P}} \Pi_2 \quad \text{e} \quad \Pi_2 \in \mathcal{P} \quad \Rightarrow \quad \Pi_1 \in \mathcal{P}$$

Dem. Seja $\mathcal{A}_T$ um algoritmo polinomial de transformação para T e p um polinômio que limita a sua complexidade $W_{\mathcal{A}_T}$. Seja $\mathcal{A}_{\Pi_2}$ um algoritmo de decisão para Π_2 limitado polinomialmente pelo polinômio q.
Considere o algoritmo esquematizado na figura para decidir Π_2.

Problemas $\mathcal{NP}$-completos: redução polinomial



- Para uma entrada x de tamanho n, $\mathcal{A}_T$ gerá a saída $T(x)$ em tempo limitado por $p(n)$. Dessa forma $|T(x)| \leq p(n)$.
- Subsequentemente, $\mathcal{A}_{\Pi_2}$ recebe como entrada $T(x)$ e gerá uma resposta correta *sim* ou *não* em tempo limitado pelo polinômio q no tamanho da entrada: $q(|T(x)|) \leq q(p(n))$.
- Consequentemtente a composição dos algoritmos assim descrita, decide Π_1 em tempo limitado pelo polinômio

$$p(n) + q(p(n))$$

Problemas $\mathcal{NP}$-completos

Def. Um problema Π é dito $\mathcal{NP}$-completo se

1. $\Pi \in \mathcal{NP}$ e
2. para todo problema $\Gamma \in \mathcal{NP}$, $\Gamma \preceq_{\mathcal{P}} \Pi$.

Teo. Se existe um problema Π, $\mathcal{NP}$-completo e em $\mathcal{P}$, então

$$\mathcal{P} = \mathcal{NP}$$

Dem. Suponha $\Pi \in \mathcal{P}$ e $\mathcal{NP}$-completo. Pela definição de $\mathcal{NP}$-completo, todo problema $\Gamma \in \mathcal{NP}$ é polinomialmente redutível a Π. Pelo Teorema anterior, Γ poderá ser decidido em tempo polinomial utilizando um algoritmo de decisão para Π. Assimm $\Gamma \in \mathcal{P}$. Conclui-se que $\mathcal{NP} \subseteq \mathcal{P}$. $\square$

Problemas $\mathcal{NP}$-completos

Teo.[Cook 1971] CNF-SAT é $\mathcal{NP}$-completo.
Dem. (Sketch) Estabelecem-se TMs como modelo computacional. Assim, se $\mathcal{L}$ é uma linguagem reconhecível polinomialmente com uma TM não determinística $M_{\mathcal{L}}$, então é possível computar uma função $f_{M_{\mathcal{L}}}$ tal que

1. $f_{M_{\mathcal{L}}}(x)$ é uma fórmula abreviada em CNF
2. $f_{M_{\mathcal{L}}}(x)$ é satisfazível *sse* $x \in \mathcal{L}$

$\square$

Problemas $\mathcal{NP}$-completos

Teo. Os problemas de decisão:

Coloração de Grafos;
Execução de tarefas com penalidades;
Empacotamento (bin packing);
Problema da Mochila (Knapsack);
Suma de subconjuntos;
Caminhos e circuitos Hamiltonianos e
Tour mínimo - Problema do caixeiro viajante

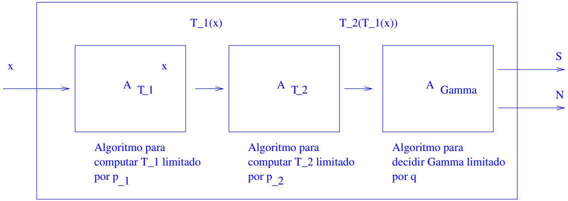
são $\mathcal{NP}$-completos.

Dem. (Sketch) Basta achar uma redução desses problemas a um problema $\mathcal{NP}$-completo. O ponto de partida é o Teorema de Cook; i.e., a $\mathcal{NP}$-completude do problema CNF-SAT. $\square$

Problemas $\mathcal{NP}$-completos

Teo. Se Π é um problema $\mathcal{NP}$-completo e $\Gamma \in \mathcal{NP}$ tal que $\Pi \preceq_{\mathcal{P}} \Gamma$, então Γ é também $\mathcal{NP}$-completo

Dem. Seja $\Delta \in \mathcal{NP}$. $\Delta \preceq_{\mathcal{P}} \Pi$ e $\Pi \preceq_{\mathcal{P}} \Gamma$ via transformações T_1 e T_2 limitadas polinomialmente por polinômios p_1 e p_2, respectivamente. Assim $\Delta \preceq_{\mathcal{P}} \Gamma$ via a transformação $T_2 \circ T_1$, que está limitada pelo polinômio $p_2 \circ p_1$. A figura ilustra como decidir qualquer problema em $\mathcal{NP}$ por redução a Γ. $\square$



Problemas $\mathcal{NP}$-completos

Teo.

$$\text{CNF-SAT} \preceq_{\mathcal{P}} \text{CLIQUE}$$

Teo.

$$\text{Circ. Hamiltonianos em digrafos} \preceq_{\mathcal{P}} \text{Circ. Hamiltonianos em grafos}$$

Teo.

$$\text{Soma de subconjuntos} \preceq_{\mathcal{P}} \text{Exec. Tarefas com penalidades}$$