NP Completude

Prof. Fábio Dias

18 de fevereiro de 2025

Sumário

Introdução

- Nós vimos que é uma problema de otimização.
- Exemplos: Problema da Árvore Geradora Mínima, Problema do Caminho Mínimo, Problema da Mochila, etc.
- Também vimos que a dificuldade de um problema está relacionado ao tempo computacional que precisamos para resolver esse problema.
- E isso está diretamente relacionado a complexidade dos algoritmos que o resolve:
  $O(\log n), O(n), O(n \log n), O(n^2), O(n^k), O(2^n), O(n!)$.
- Em geral, pensamos que problemas que podem ser resolvidos por algoritmo de tempo polinomial $O(n^k)$ são tratáveis, ou fáceis, e que problemas que exigem tempo superpolinomial $O(2^n)$ são intratáveis ou difíceis.

Introdução

- Classes de problemas dividem os problemas computacionais de acordo com a complexidade do melhor algoritmos **conhecido** que o resolve: **Classes P, NP e NP-Completo**;
- A classe mais importante é a chamada NP-Completo, que ninguém descobriu um algoritmo de tempo polinomial e nem provou que não exista;
- Em cada um dos pares de problemas a seguir, um deles pode ser resolvido em tempo polinomial e o outro é NP-completo, mas a diferença entre eles parece insignificante.

Caminhos simples mínimos e caminhos simples de comprimento máximo

- No problema de caminho mínimo, vimos que até mesmo com pesos de arestas negativos podemos encontrar caminhos mínimos que partem de uma única origem em um grafo dirigido $G = (V, E)$ no tempo $O(VE)$, usando o algoritmo Dijkstra ou Bellman-Ford.
- Contudo, determinar o caminho simples de comprimento máximo entre dois vértices em um grafo não ponderado é difícil. Simplesmente determinar se um grafo contém um caminho simples com pelo menos um determinado número de arestas é NP-completo.

Passeio de Euler e ciclo hamiltoniano

- Um passeio de Euler de um grafo conexo dirigido $G = (V, E)$ é um ciclo que percorre cada aresta de G exatamente uma vez, embora possa visitar cada vértice mais de uma vez. Podemos determinar se um grafo tem um passeio de Euler no tempo de apenas $O(E)$ e, de fato, podemos encontrar as arestas do passeio de Euler no tempo $O(E)$.
- Um ciclo hamiltoniano de um grafo dirigido $G = (V, E)$ é um ciclo simples que contém cada vértice em V. Determinar se um grafo dirigido tem um ciclo hamiltoniano é NP-completo.

Satisfazibilidade 2-CNF e satisfazibilidade 3-CNF

- Uma fórmula booleana contém variáveis cujos valores são 0 ou 1; conectivos booleanos como $\wedge$ (AND), $\vee$ (OR) e $\neg$ (NOT); e parênteses.
- Uma fórmula booleana é satisfazível se existe alguma atribuição dos valores 0 e 1 às suas variáveis que faça com que ela seja avaliada como 1.
- Uma fórmula booleana está em forma normal k-conjuntiva, ou k-CNF, se for o AND de cláusulas OR de exatamente k variáveis ou de suas negações. Por exemplo, a fórmula booleana $(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_3)$ está em 2-CNF.
- Embora possamos determinar em tempo polinomial se uma fórmula 2-CNF é satisfazível, determinar se uma fórmula 3-CNF é satisfazível é NP-completo.

Classe P

- Consiste nos problemas que podem ser resolvidos em tempo polinomial $(O(n^k))$, para k inteiro positivo.
- Informalmente: Consiste nos problemas de otimização que possuem algoritmos **conhecidos** com tempo polinomial.
- Classe P $=$ Classes dos problemas polinomiais.
- Exemplos: Problema da Árvore Geradora Mínima, Problema do Caminho Mínimo, Problema da Mochila Fracionado, etc.

Classe NP

- A classe NP consiste nos problemas que são **verificáveis** em tempo polinomial;
- O que significa um problema ser verificável?
- Se tivéssemos algum tipo de "certificado" de uma solução, poderíamos verificar se o certificado é correto em tempo polinomial para o tamanho da entrada para o problema;
- Por exemplo, no problema do ciclo hamiltoniano, dado um grafo dirigido $G = (V, E)$, um certificado seria uma sequência $<v_1, v_2, v_3, ..., v_n>$ de $|V|$ vértices. É fácil verificar em tempo polinomial que $(v_i, v_{i+1}) \in E$ para $i = 1, 2, 3, ..., n–1$ e também que $(v_n, v_1) \in E$;
- Existe um algoritmo de tempo polinomial que verificar se esse certificado é verdadeiro ou não.

Classe NP

- **CUIDADO**: NP não significa não polinomial, mas não determinístico em tempo polinomial.
- Qualquer problema em P também está em NP, ou seja, $P \subseteq NP$.
- Se um problema está em P então podemos resolvê-lo em tempo polinomial sem sequer receber um certificado.
- A grande questão é $NP \subseteq P$, porque dessa forma teríamos $P = NP$. Essa é uma questão em aberto em ciência da computação.

Classe NP - Completo

- Informalmente, um problema está na classe NP-Completo se **ele está em NP** ($NP - C \subset NP$) e **é tão "difícil" quanto qualquer problema em NP**.
- O que significa ser tão difícil quanto qualquer problema em NP?
- Daqui a pouco explico, por enquanto, ficaremos com: Se qualquer problema NP-Completo pode ser resolvido em tempo polinomial, então todo problema NP também pode ser resolvido em tempo polinomial, isto é, $P = NP$.
- Como provamos que um problema está na classe NP-Completo?

Problema de Decisão

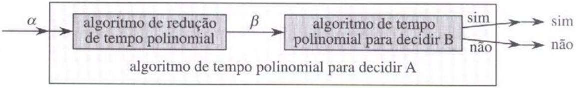
- Toda a teoria sobre classes de problemas se aplica a **problema de decisão**.
- No problemas de decisão a resposta é simplesmente "sim"ou "não".
- Podemos definir um problema de decisão impondo um limite sobre o valor a ser otimizado do problema de otimização:
  - (Problema da Árvore Geradora Mínima) AGM_d: Existe uma árvore geradora de custo no máximo d?
  - (Problema do Caminho Mínimo) CM_d: Existe um caminho do vértice s até o vértice t de custo no máximo d?
  - (Problema da Mochila) PM_d: Posso adicionar uma certa quantidade de objetos na mochila de valor no máximo d?
- Todo problema de otimização possui um problema de decisão associado.

Problema de Decisão

- O problema de decisão é de certo modo "mais fácil" ou, pelo menos, "não mais difícil" que o problema de otimização.
- Por exemplo, podemos resolver o problema de decisão da AGM resolvendo AGM, e depois comparando o valor da solução ótima ao valor do parâmetro entrada d do problema de decisão.
- Se um problema de otimização é fácil, seu problema de decisão relacionado também é fácil.
- Se pudermos fornecer evidências de que um problema de decisão é difícil, também forneceremos evidências de que seu problema de otimização relacionado é difícil.

Reduções

- Considere um problema de decisão A que gostaríamos de resolver em tempo polinomial.
- Suponha que exista um problema de decisão diferente B que já sabemos como resolver em tempo polinomial.
- Suponha que temos um procedimento que transforma qualquer instância α de A em alguma instância β de B com as seguintes características.
  1. A transformação demora tempo polinomial.
  2. As respostas são as mesmas, isto é, a resposta para α é "sim" se e somente se a resposta para β também é "sim".



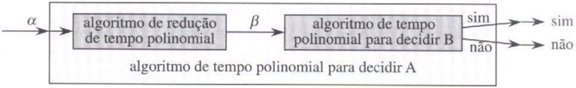
- Essa transformação damos o nome de redução polinomial $A \leq_p B$.

Reduções

- NP-completude consiste em mostrar o quanto um problema é difícil, em vez de mostrar o quanto ele é fácil;
- Iremos usar reduções de tempo polinomial ao contrário para mostrar que um problema é NP-completo;
- Como usar reduções de tempo polinomial para demonstrar que não pode existir nenhum algoritmo de tempo polinomial para um determinado problema B?

Reduções

- Suponha que tenhamos um problema de decisão A para o qual já sabemos que não pode existir nenhum algoritmo de tempo polinomial;
- Suponha ainda que tenhamos uma redução de tempo polinomial que transforma instâncias de A em instâncias de B;
- Suponha o contrário: isto é, suponha que B tenha um algoritmo de tempo polinomial.
- Então, usando o método mostrado anteriormente teríamos um modo de resolver o problema A em tempo polinomial, o que contradiz nossa hipótese da inexistência de algoritmo de tempo polinomial para A.



Classe NP - Completo

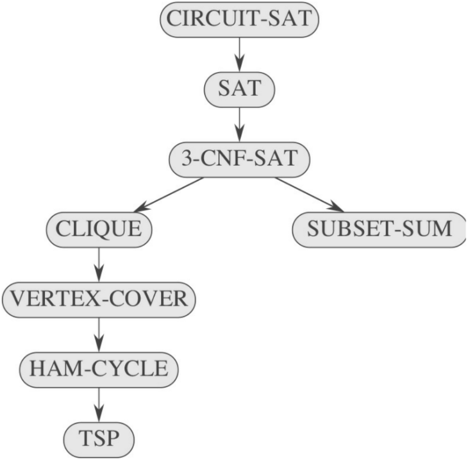
- Formalmente, como é **ele está em NP** e **ser tão "difícil" quanto qualquer problema em NP**?
- Um problema de decisão H é NP-Completo se:
  1. $H \in NP$;
  2. $\forall Q \in NP$, existe $Q \leq_p H$.
- Se qualquer problema NP-Completo pode ser resolvido em tempo polinomial, então todo problema NP também pode ser resolvido em tempo polinomial, isto é, $P = NP$.
- Professor, como provar o item 2?????

Classe NP - Completo

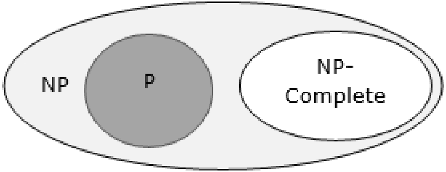
- Suponha que exista um problema W, tal que $W \in NP - Completo$.
- Logo, sabemos que $\forall Q \in NP$, existe $Q \leq_p W$.
- Para provar que $\forall Q \in NP$, existe $Q \leq_p H$, basta mostrar uma redução $W \leq_p H$.
- Com isso, para provar que um problema de decisão H é NP-Completo:
  1. $H \in NP$;
  2. Dado $W \in NP - Completo$, mostrar $W \leq_p H$.

Primeiro Problema na Classe NP - Completo

- O problema de Satisfatibilidade de Circuito foi o primeiro problema a ser provado que pertence a NP-Completo em 1971 por Stephen Cook em um artigo chamado The complexity of theorem-proving procedures.



P vs NP vs NP-Completo



- Questão em aberto é se $P = NP$?
- E por fim, a classe NP-Difícil é formado pelos problemas de otimização que sua versão de decisão são NP-Completo.
- Normalmente usamos NP-Completo também para designar um problema de otimização.

Perguntas?!