DCC207 – Algoritmos 2

Aula 09 – Introdução a Teoria da Complexidade (Parte 03)

Professor Renato Vimieiro

DCC/ICEx/UFMG

Introdução

- Como vimos na aula passada, existe uma subclasse de problemas NP que são pelo menos tão difíceis quanto quaisquer outros da classe
- Essa subclasse é conhecida como a classe dos problemas NP-Completos
- Vimos a demonstração do teorema de Cook-Levin que estabeleceu o ‘primeiro’ problema NP-Completo
- Vimos também que, pela definição de redução polinomial, a demonstração de pertinência de novos problemas à subclasse pode ser feita reduzindo-se um NP-completo a esses problemas
- Hoje veremos algumas reduções que, além de demonstrar que alguns problemas são NP-Completos, podem servir de base para construção de algoritmos

Vertex Cover é NP-Completo

- Vertex cover (cobertura de vértices) é um problema bastante estudado em teoria dos grafos
- Ele possui aplicações em diversas áreas da computação como: mineração de dados, otimização, e bioinformática.
- Dado um grafo G=(V,E), uma cobertura de vértices de G é um subconjunto de vértices tal que toda aresta do grafo é incidente em algum vértice desse conjunto
  - $V' \subseteq V$
  - $\forall (u,v) \in E, (u \in V') \lor (v \in V')$
- A versão de otimização desse problema é encontrar o menor conjunto V’ que seja uma cobertura de vértices de G
- O problema de decisão equivalente é determinar se G possui uma cobertura de k vértices
  - VERTEX_COVER = {<G,k> | O grafo G possui uma cobertura de k vértices}

Vertex Cover é NP-Completo

- É perceptível que o problema pode ser resolvido em tempo polinomial usando não-determinismo
  - Escolhe-se não-deterministicamente um conjunto de vértices
  - Verifica-se se toda aresta é incidente em algum vértices desse conjunto
- Para demonstrar que o problema é NP-difícil (todo problema NP se reduz polinomialmente a ele), basta reduzir um problema NP-completo a Vertex Cover
- Vamos demonstrar que podemos reduzir CLIQUE a VERTEX_COVER em tempo polinomial

Vertex Cover é NP-Completo

- Nossa redução será baseada no complemento do grafo
- Seja <G,k> uma instância do problema de Clique. Obtemos uma instância de Vertex Cover da seguinte forma:
  - G’ = (V,E’)
  - $\forall u,v \in V, u \neq v\ [(u,v) \in E' \longleftrightarrow (u,v) \notin E]$
  - A instância do problema é <G’, |V|-k>
- Agora precisamos demonstrar que G possui uma clique de tamanho k sse G’ possui uma cobertura com |V|-k vértices
- Suponha que G possua uma clique de tamanho k. Suponha que V’ seja tal clique.

Vertex Cover é NP-Completo

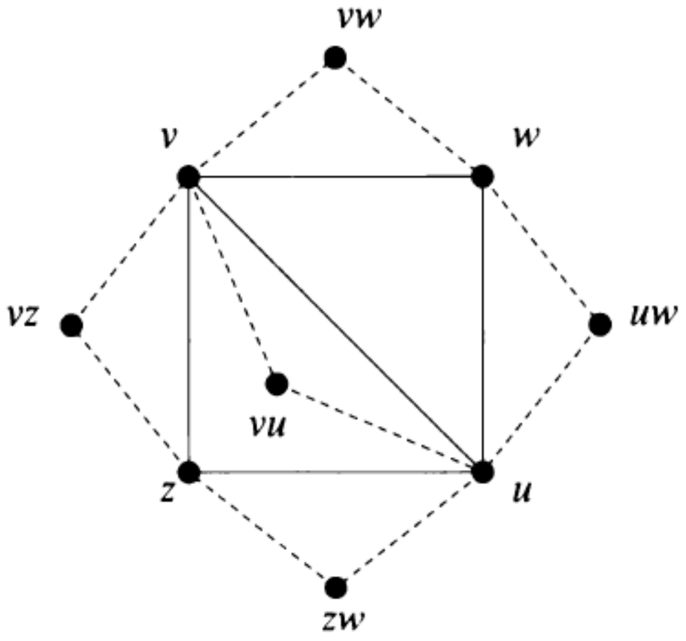
- Agora temos que demonstrar que G’ possui uma cobertura de vértices de tamanho $|V|-|V'| = |V|-k$.
- Seja $(u,v) \in E'$ uma aresta arbitrária. Sabemos que, nesse caso, u e v não podem ambos pertencer a V’. Ou seja, $u \in V-V'$ ou $v \in V-V'$. Como (u,v) é arbitrária, V-V’ é uma cobertura de vértices.
- Agora suponha que G’ tenha uma cobertura de vértices V’’ de tamanho $|V|-k$. Nesse caso, se $(u,v) \in E'$, então $u \in V''$ ou $v \in V''$. A contrapositiva dessa implicação é: se $u \notin V''$ e $v \notin V''$, então $(u,v) \in E$. Portanto, V-V’’ é uma clique em G de tamanho k.

Dominating set é NP-completo

- Dominating set (conjunto dominante) é um outro problema sobre grafos bastante estudado
- O problema consiste em encontrar um subconjunto mínimo de vértices tal que todos os outros são adjacentes a pelo menos um desse conjunto
  - $D \subseteq V$
  - $\forall\, v \in V\ [v \in D \lor \exists u \in D\ (u,v) \in E]$
  - Problema de decisão {<G,k> | G possui um conjunto dominante de tamanho k}
- Aplicações:
  - Redes de computadores -> protocolos de roteamento
  - (Quimio)Bioinformática -> menor conjunto de drogas que afeta diferentes células cancerígenas
- Novamente, é fácil notar que o problema é NP
- Vamos reduzir o Vertex Cover a ele para demonstrar que ele é também NP-completo

Dominating set é NP-Completo

- Dada uma instância <G,k> do Vertex Cover, devemos mapeá-la a uma instância do Dominating Set
- Construímos a instância da seguinte forma:
  - $G' = (V',E')$
  - $V' = V \cup E$
  - $E' = E \cup \{((u,v),u), ((u,v),v) \mid (u,v) \in E\}$
- Suponha que C seja uma cobertura de vértice de tamanho k em G. Por definição, C cobre todas as arestas de G. Portanto, todos os novos vértices são dominados pelos vértices de C. Da mesma forma, os vértices de G que não pertencem à C são dominados porque são incidentes em alguma aresta coberta.
- Suponha que D seja um conjunto dominante de G' com tamanho k. Como os novos vértices representam arestas, se eles pertencerem a D, então podem ser substituídos por um dos terminais. Logo, pode-se assumir que D contenha somente vértices de V sem perda de generalidade. Como qualquer vértice de G' é adjacente a um vértice de D, temos que todos os novos vértices são adjacentes a algum de D. Portanto, D cobre todas as arestas de G.



3-Coloração é NP-Completo

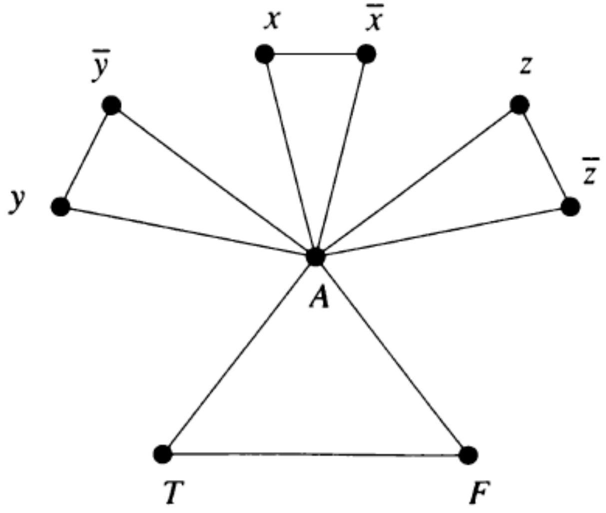
- Já nos deparamos anteriormente com o problema de 3-coloração em grafos
- Uma instância do problema ocorreu quando quisemos resolver o problema da galeria de arte, usando geometria computacional
- Embora a instância em particular era simples de resolver, o caso geral, como veremos, é mais complexo
- Formalmente, o problema de coloração de vértices em grafos consiste em encontrar uma função $f: V \rightarrow C$, tal que todo vértice é mapeado para uma cor e se $(u,v) \in E$, então $f(u) \neq f(v)$
  - O caso particular de 3-coloração consiste em encontrar uma função que mapeie os vértices a exatamente 3 cores distintas

3-Coloração é NP-Completo

- O problema pode ser resolvido não-deterministicamente em tempo polinomial, escolhendo-se três conjuntos disjuntos de vértices, e verificando se vértices adjacentes pertencem a conjuntos distintos.
- Para demonstrar que o problema também é NP-difícil, vamos reduzir 3SAT a ele.
- Dada uma instância de 3SAT, isto é, uma função booleana com k cláusulas, vamos construir uma instância de 3-Coloração equivalente.
- Note que não é necessário que a nova instância mantenha qualquer semântica do problema original. A única restrição é que a função é satisfatível sse o grafo equivalente admita uma 3-coloração.

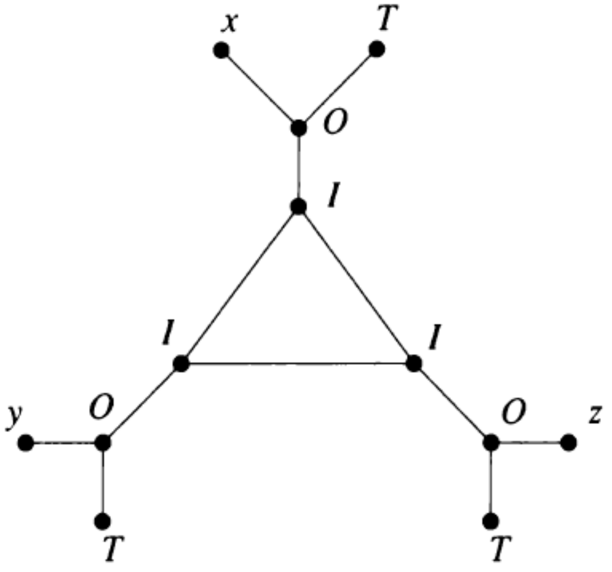
3-Coloração é NP-Completo

- Vamos construir a instância da seguinte forma:
  - Primeiro, criamos um triângulo com os vértices T, F, A, o qual será colorido exatamente com essas três cores.
  - Depois, para cada variável na função, criamos dois vértices, representando os possíveis literais obtíveis a partir dela.
  - Esses dois vértices devem ser conectados por uma aresta, e cada um deles deve ser conectado ao vértice A
  - Veja que essa situação garante que somente um dos dois literais de uma variável seja verdadeiro. Porém, não garante que toda cláusula seja verdadeira



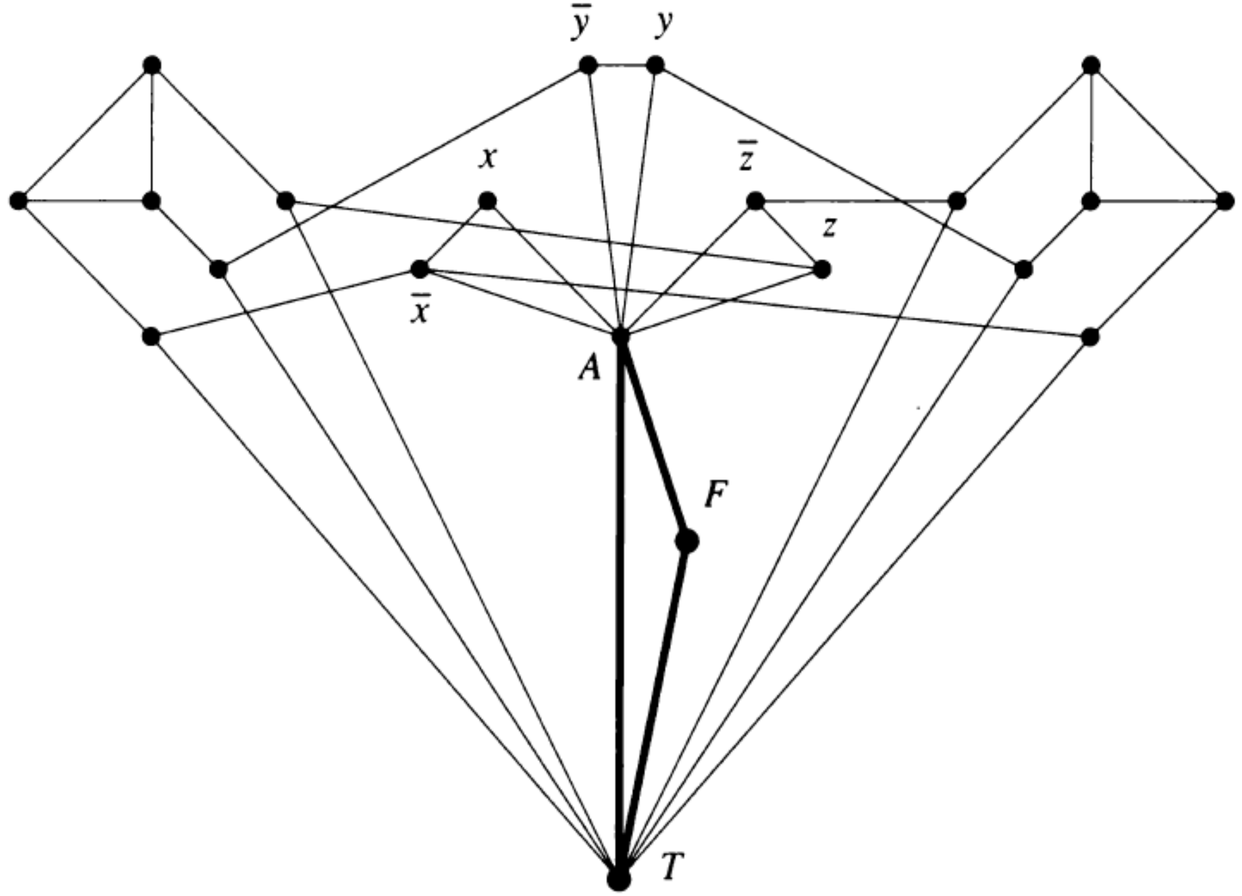
3-Coloração é NP-Completo

- O próximo passo é, então, garantir que pelo menos um literal de cada cláusula seja verdadeiro.
- Para cada cláusula, vamos adicionar três novos vértices chamados de **externos**. Cada vértice externo deve estar conectado a um literal que aparece na cláusula e também ao vértice T.
- Além disso, cada um desses vértices externos estará conectado a um vértice **interno** de um novo triângulo criado para a cláusula
    - O exemplo ao lado foi construído para a cláusula (x ∨ y ∨ z)
    - Deve-se imaginar que os Ts na figura são o mesmo vértice criado anteriormente
- Note que essa configuração implica que pelo menos 1 vértice de cada cláusula seja verdadeiro.
    - Se todos forem falso, então todos os vértices externos devem ser coloridos com A. Consequentemente, o triângulo interno não admite uma 3-coloração



3-Coloração é NP-Completo

- O grafo equivalente à função f(x,y,z) = (¬x ∨ y ∨ ¬z) ∧ (¬x ∨ ¬y ∨ z) é:



3-Coloração é NP-Completo

- Veja que se a função é satisfazível, então podemos colorir os vértices de cada cláusula conforme a atribuição dos valores
  - Pelo menos 1 literal de cada cláusula será verdadeiro. Logo, colorimos o vértice externo conectado a ele com falso, os demais externos da cláusula com A, e o triângulo interno conforme a atribuição de cores.
- Analogamente, se o grafo tem uma 3-coloração, então cada vértice terá uma das 3 cores do triângulo inicial. Por construção, vimos que, pelo menos, um literal de cada cláusula deve ser colorido com T. Logo, usamos as cores dos vértices para fazer a atribuição de valores às variáveis.

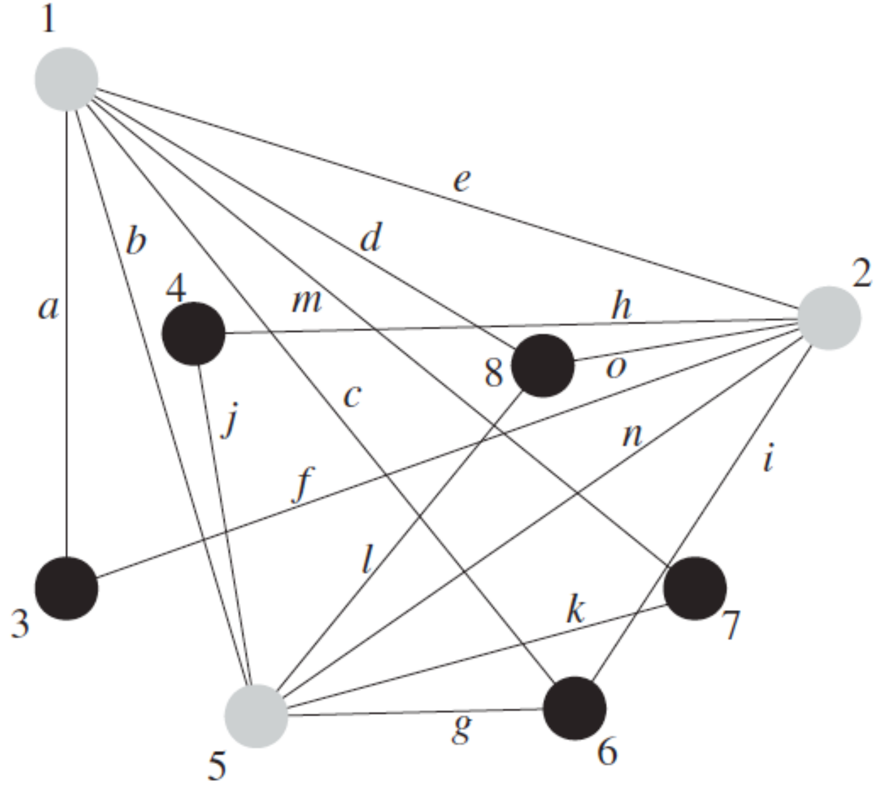
Set Cover é NP-Completo

- O problema Set Cover consiste em encontrar uma sub-família de conjuntos cuja união contenha todos os elementos dos conjuntos da família:
  - Seja S = {S1, S2, S3, ..., Sm} uma família de conjuntos.
  - O problema consiste em encontrar a menor sub-família $F \subseteq S$, tal que $\bigcup F = \bigcup S$.
  - O problema de decisão equivalente é: $\{\langle S,k \rangle \mid \exists\, F \subseteq S\ [|F| = k \land \bigcup F = \bigcup S]\}$
- Fica como exercício demonstrar que Set Cover é NP.
- Agora reduziremos Vertex Cover a ele para demonstrar que ele é NP-Completo.

Set Cover é NP-Completo

- Dada uma instância <G,k> de Vertex Cover, construímos a instância <S,k> de Set Cover da seguinte forma:
  - Para cada $v \in V$, criamos o conjunto $Sv = \{(u,v) \in E \mid u \in V\}$
- A demonstração agora é imediata. Se V’ é uma cobertura de k vértices em G, então $F = \{Sv \mid v \in V'\}$ é uma cobertura de conjuntos.
- Analogamente, se F é uma cobertura de conjuntos de tamanho k, os conjuntos em F cobrem todas as arestas de G. Portanto, os vértices equivalentes formam uma cobertura de k vértices em G.

Set Cover é NP-Completo



$$
\begin{aligned}
S_1 &= \{a, b, c, m, d, e\} \\
S_2 &= \{e, h, o, n, i\} \\
S_3 &= \{a, f\} \\
S_4 &= \{j, m\} \\
S_5 &= \{b, j, l, n, k, g\} \\
S_6 &= \{g, c, i\} \\
S_7 &= \{k, m\} \\
S_8 &= \{d, o, l\}
\end{aligned}
$$

Leitura

- Seção 34.5.2 (CLRS)
- Seção 11.4 (Manber)
- Seção 17.4-Set Cover (Goodrich e Tamassia)