DCC206 – Algoritmos 1

Aula 20 – Introdução a Teoria da Complexidade (Parte 02)

Professor Renato Vimieiro

DCC/ICEx/UFMG

Introdução

- Na aula passada, vimos três classes de complexidade de tempo em que problemas podiam ser classificados
- Vimos que a classe dos problemas NP são aqueles que possuíam certificadores polinomiais
- Uma pergunta que surge é: todos os problemas de uma mesma classe possuem a mesma dificuldade?
- Nessa aula, veremos como certos problemas resumem a dificuldade da classe:
  - Alguns problemas dessa classe são pelo menos tão difíceis quanto qualquer outro problema dela
- Esse conceito está apoiado na definição de redução de problemas
  - Focaremos no caso particular de redução polinomial entre problemas NP

Redução de problemas

- A ideia de se reduzir um problema (A) a outro (B) consiste na computação de uma função (algoritmo) f: A -> B que:
  - Dada uma entrada do problema A, a converte para uma do problema B
  - Existe solução para o problema A, se existe solução para o problema f(A)
- Do ponto de vista de paradigmas de desenho de algoritmos, essa técnica pode ser usada também na construção de algoritmos
  - Reduz-se um problema para o qual não se conhece uma solução a outro que possua solução conhecida
- Em outras palavras, essa técnica possui utilidade tanto prática quanto teórica

Redução polinomial de problemas

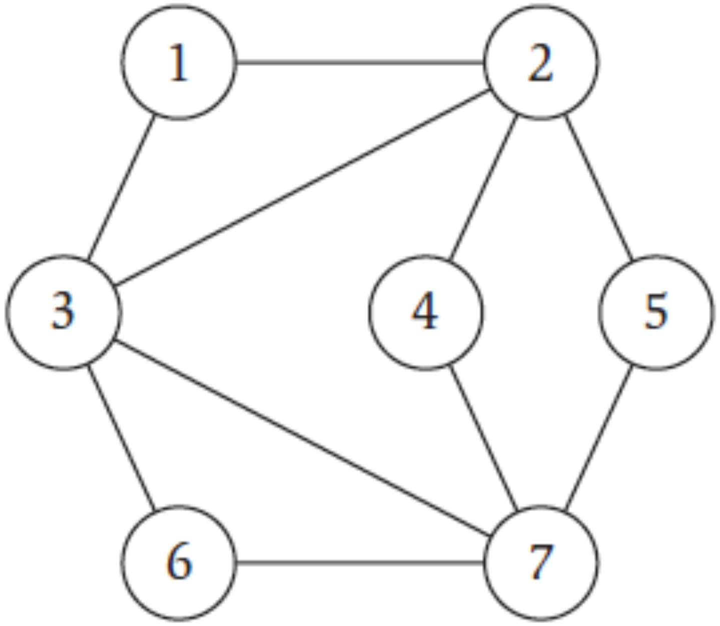
- Uma linguagem (problema) A é redutível a outra linguagem B em tempo polinomial, $A \leq_p B$, se existe uma função f: A -> B, computável deterministicamente em tempo polinomial, tal que, para toda palavra w:
  - $w \in A \leftrightarrow f(w) \in B$
- Veja que se A é redutível em tempo polinomial a B, então B é pelo menos tão difícil quanto A (pode ser até mais difícil que A)
  - Mas A não é mais difícil que B (Por quê?)
- Teorema: Se $A \leq_p B$ e $B \in P$, então $A \in P$.
  - Prova: Segue direto da definição de redução.

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Vamos estudar um exemplo de redução polinomial entre problemas
- Um problema teórico envolvendo grafos com bastante apelo prático é o de encontrar o maior subconjunto de vértices não adjacentes entre si
- Esse problema é conhecido como o problema do Conjunto Independente (Independent Set)
- Veja que esse é um problema de otimização, porém ele pode ser transformado em um problema de decisão se perguntarmos se existe um conjunto independente de tamanho k em um grafo

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Considere o grafo ao lado, o problema de decisão de conjunto independente para esse grafo é:
  - Existe um conjunto independente nesse grafo de tamanho 4?
- A resposta é sim: S = {1, 4, 5, 6}
- É fácil perceber que uma solução ingênua para esse problema teria custo exponencial

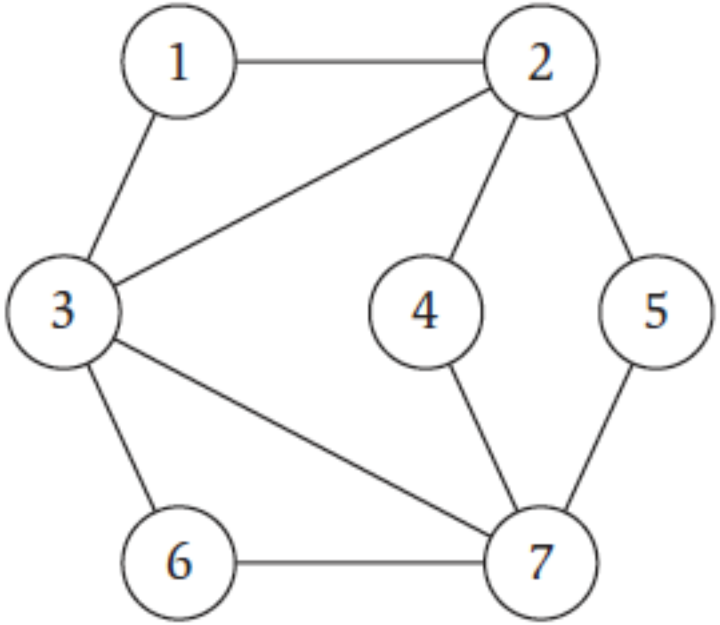


Conjunto Independente $\leq_p$ Cobertura de Vértices

- Agora considere outro problema em grafos também bastante estudado: o problema de determinar o menor conjunto de vértices que domina todas as arestas do grafo
  - Ou seja, determinar o menor subconjunto de vértices C tal que toda aresta possui pelo menos um dos terminais em C
- Esse problema é chamado de Cobertura de Vértices (Vertex Cover)
  - Para lembrar da definição, queremos encontrar vértices que cobrem/dominam arestas
- Novamente, esse é um problema de otimização que pode ser transformado em um de decisão, basta especificar que queremos um conjunto de tamanho k

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Vamos usar o mesmo grafo do exemplo de conjunto independente para exemplificar esse problema
- Agora, perguntamos se existe uma cobertura de vértices de tamanho 3?
- De novo, a resposta é sim: C = {3, 2, 7}



Conjunto Independente $\leq_p$ Cobertura de Vértices

- Agora, queremos mostrar que o problema da Cobertura de Vértices é pelo menos tão difícil quanto o problema do Conjunto Independente
- Para tal, precisamos reduzir polinomialmente o primeiro para o segundo
  - Note a direção da redução: **nos casos de demonstração de complexidade, é sempre daquele que já sabemos para aquele que queremos determinar**
- Como foi dito, para reduzir um problema ao outro, precisamos mostrar como as instâncias de um são mapeadas para instâncias do outro de forma que a resposta é sim para um sse é sim para o outro

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Instâncias do CI são da forma <G,k>, e instâncias do CV são da forma <G,k>
- Assim, recebemos um grafo e um inteiro e devemos transformá-lo em um (outro) grafo e um (outro) inteiro
- Vamos fazê-lo da seguinte forma: <G,k> -> <G,|V|-k>
- Agora precisamos mostrar que G possui um conjunto independente de tamanho k sse G possui uma cobertura de vértices de tamanho |V|-k

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Suponha que S seja um conjunto independente de tamanho k em G
- Por definição, para todo u, v ∈ S, (u,v) ∉ E
- A contrapositiva dessa afirmação é que se (u,v) ∈ E, então u ∉ S ou v ∉ S
- Logo, para toda aresta, pelo menos um dos dois terminais pertence ao complemento de S
- Ou seja, o complemento de S é uma cobertura de vértice de G
- Como esse conjunto é C = V – S, concluímos que se G possui um conjunto independente de tamanho k, ele possui uma cobertura de vértices de tamanho |V|- k

Conjunto Independente $\leq_p$ Cobertura de Vértices

- Para mostrar a volta, suponha que C seja uma cobertura de vértices de G com tamanho $|V| - k$
- Analogamente, para toda aresta $(u,v) \in E$, $u \in C$ ou $v \in C$
- Novamente, a contrapositiva diz que $u \notin C$ e $v \notin C$, então $(u,v) \notin E$
- Seja $S = V - C$. Nesse caso, $u \in S$ sse $u \notin C$. Logo, para todo $u, v \in S$, $u \neq v$, $(u,v) \notin E$.
- Portanto, S é um conjunto independente. Como $|C| = |V| - k$, $|S| = k$
- Concluímos que CI é redutível em tempo polinomial a CV

Problemas NP-Completos

- Existem problemas na classe NP que são pelo menos tão difíceis quanto qualquer outro da classe.
- Esses problemas são conhecidos como problemas NP-completos
- Formalmente, um problema B é NP-completo se:
  - $B \in NP$
  - Todo problema $A \in NP$ é redutível em tempo polinomial a B, $A \leq_p B$. (**NP-Difícil**)
- Teorema: Se B é NP-completo e $B \in P$, então P=NP.
- Prova: Segue da definição de redução.

Problemas NP-completos

- Um corolário do teorema anterior é que, se A é NP-completo, B $\in$ NP, e A $\leq_p$ B, então B também é NP-completo.
- Esse corolário permite demonstrar que um problema é NP-Completo a partir de um outro conhecido.
- No entanto, como determinar o ‘primeiro’ problema NP-Completo, a partir do qual os outros podem ser determinados?
- Esse é o teorema postulado, de forma independente, pelos cientistas da computação Stephen Cook (americano-canadense) e Leonid Levin (ucraniano-soviético) na década de 1970

Problema da satisfabilidade

- Cook e Levin demonstraram que o problema da satisfabilidade era NP-Completo
- Seja ϕ uma função booleana com n variáveis. O problema da satisfabilidade (SAT) consiste em verificar a existência de uma atribuição de valores às variáveis de forma que a função avalie para 1
- Em geral, consideramos que ϕ esteja na forma normal conjuntiva, ou seja a função é um produto de somas (maxtermos)
- Exemplo: $\phi(a,b,c) = (a \lor \neg b) \land (\neg a \lor \neg c) \land (b \lor \neg c)$
- A função acima é satisfazível?

Problema da satisfabilidade

- Veja que uma solução para o problema seria verificar todas as atribuições possíveis
  - Isso teria um custo $O(2^n)$
- Se nos for dada a função e uma atribuição de valores, podemos verificar se a resposta é sim em tempo polinomial
- Logo, SAT pertence à classe NP
- O que Cook-Levin mostraram foi que todos os problemas da classe NP podem ser reduzidos em tempo polinomial ao SAT
  - Ou seja, SAT é NP-difícil

Leitura

- Seções 8.1 – 8.4 (Kleinberg e Tardos)