ACH2043 INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 26

Cap 7.3 – A classe NP

Profa. Ariane Machado Lima
ariane.machado@usp.br

A classe NP

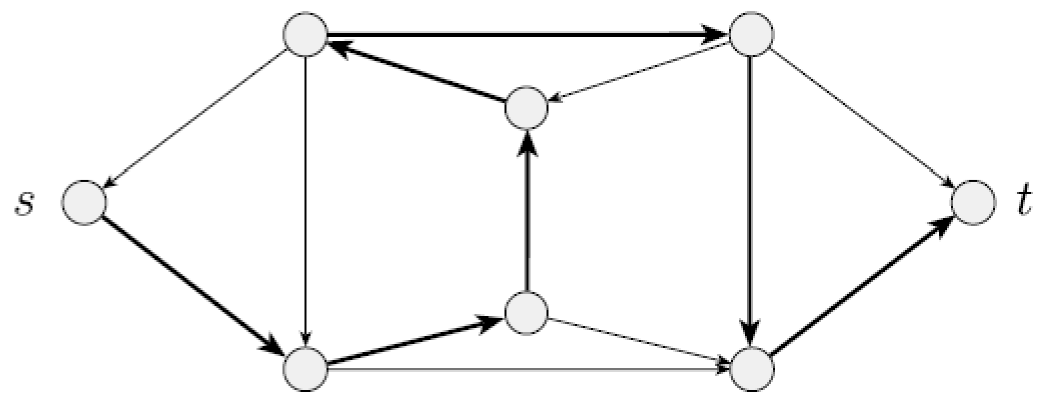
- Um problema está na classe P se existe uma MT determinística que o resolva e que rode em tempo polinomial.
- E se um algoritmo de tempo polinomial não é conhecido?
    -
    -
-
-
-

A classe NP

- Um problema está na classe P se existe uma MT determinística que o resolva e que rode em tempo polinomial.
- E se um algoritmo de tempo polinomial não é conhecido?
  - Ou o problema é intrinsicamente difícil
  - Ou um algoritmo polinomial não é conhecido
- Há vários problemas nesta situação
- Não sabemos distinguir entre os dois casos
- Para certos problemas, embora para DECIDIR conheça-se apenas algoritmos exponenciais, VERIFICAR se uma solução candidata é mesmo uma solução pode ser feito em tempo polinomial

Ex: Caminho Hamiltoniano (em um grafo direcionado)

- Dado um grafo direcionado G, um caminho Hamiltoniano do nó s ao nó t é um caminho que, partindo do nó s, chega ao nó t após passar por todos os nós do grafo exatamente uma vez.



Ex: Caminho Hamiltoniano (em um grafo direcionado)

- CAMHAM = { <G, s, t>: G é um grafo direcionado com um caminho Hamiltoniano do nó s ao nó t}
- Aplicações?

Ex: Caminho Hamiltoniano (em um grafo direcionado)

- CAMHAM = { <G, s, t>: G é um grafo direcionado com um caminho Hamiltoniano do nó s ao nó t}
- Aplicações?
  - Rotas de distribuição
  - Itinerários de ônibus
  - Etc.

Ex: Caminho Hamiltoniano (em um grafo direcionado)

- Algoritmo exponencial que decide CAMHAM:

  Gere todos os caminhos possíveis

  Verifique se um deles é:

    - de s a t
    - passa por cada nó (todos eles) apenas uma vez

-
-
-

Ex: Caminho Hamiltoniano (em um grafo direcionado)

- Algoritmo exponencial que decide CAMHAM:

  Gere todos os caminhos possíveis

  Verifique se um deles é:

    - de s a t
    - passa por cada nó (todos eles) apenas uma vez

- Não se conhece uma solução polinomial
- NINGUÉM SABE SE ESSA SOLUÇÃO EXISTE
- Mas dado um caminho, é possível VERIFICAR se ele é hamiltoniano em tempo polinomial.

Ex: Caminho Hamiltoniano (em um grafo direcionado)

- Para alguns problemas, nem para a VERIFICAÇÃO se conhece um algoritmo polinomial

- Ex: o complemento de CAMHAM (grafos que não possuem nenhum caminho hamiltoniano entre s e t)

Verificador

- Um verificador para uma linguagem A é um algoritmo V, onde

A = { w | V aceita <w,c> para alguma cadeia c}

- O c é o certificado ou prova de que w pertence a A
- Tempo do verificador medido em termos apenas do comprimento de w (não de c)
- Uma linguagem A é polinomialmente verificável se ela tem um verificador de tempo polinomial
- Para verificadores polinomiais, c deve ter comprimento polinomial (no tamanho de w)

Verificador - Exemplo

- O que seria um certificado para CAMHAM?

Verificador - Exemplo

- O que seria um certificado para CAMHAM?
  - Um caminho hamiltoniano de s a t
- Como seria o verificador?

Verificador - Exemplo

- O que seria um certificado para CAMHAM?
  - Um caminho hamiltoniano de s a t
- Como seria o verificador?
  - Verifica se não há repetição de nós no caminho
  - Verifica se começa com s e termina com t
  - Verifica se entre dois nós há uma aresta no grafo

A classe NP

- NP é a classe de todas as linguagens polinomialmente **verificáveis**
- Um problema NP também pode ser P?
  -
  -
  -
  -
  -

A classe NP

- NP é a classe de todas as linguagens polinomialmente **verificáveis**
- Um problema NP também pode ser P?
  - Sim!
  - Ex: COMPOSTOS = { x | x = pq, para inteiros p,q >1}
  - Certificado?
  -
  -

A classe NP

- NP é a classe de todas as linguagens polinomialmente **verificáveis**
- Um problema NP também pode ser P?
  - Sim!
  - Ex: COMPOSTOS = { x | x = pq, para inteiros p,q >1}
  - Certificado? Um dos divisores
  - Há um algoritmo polinomial que o resolve (está em P)
  - Há um algoritmo polinomial que o verifique (está em NP)

A classe NP

- NP vem de tempo polinomial não-determinístico
- Teorema: Uma linguagem está em NP se e somente se ela é decidida por alguma máquina de Turing não-determinística (MTN) de tempo polinomial
  - Lembrando que o tempo de uma MTN é o tempo do ramo mais longo...
  - O certificado é uma solução aceita em um dos ramos
- Antes de provar o teorema, um exemplo

MTN que decide CAMHAM

MTN que decide CAMHAM

- “Sobre a entrada <G, s, t>, onde G é um grafo direcionado com nós s e t:

1. Escreva uma lista de m números, $p_1, ..., p_m$, onde m é o número de nós em G. Cada número na lista é selecionado não-deterministicamente entre os números 1 a m.
2. Verifique se há repetições na lista. Se houver, *rejeite*.
3. Teste se $s = p_1$ e $t = p_m$. Se um dos testes falhar, *rejeite*.
4. Para cada i entre 1 e m-1, verifique se (p_i, p_{i+1}) é uma aresta de G. Se alguma não for, *rejeite*. Caso contrário, *aceite*.”

MTN que decide CAMHAM

- “Sobre a entrada <G, s, t>, onde G é um grafo direcionado com nós s e t:
  1. Escreva uma lista de m números, p_1, ..., p_m, onde m é o número de nós em G. Cada número na lista é selecionado não-deterministicamente entre os números 1 a m.
  2. Verifique se há repetições na lista. Se houver, *rejeite*.
  3. Teste se $s = p_1$ e $t = p_m$. Se um dos testes falhar, *rejeite*.
  4. Para cada i entre 1 e m-1, verifique se (p_i, p_{i+1}) é uma aresta de G. Se alguma não for, *rejeite*. Caso contrário, *aceite*.”

Tempo polinomial não-determinístico

A classe NP

- Teorema: Uma linguagem está em NP se e somente se ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial
- Ideia da prova: mostrar como converter um **verificador** de tempo polinomial para uma MTN **decisora** de tempo polinomial equivalente e vice-versa

Prova

Uma linguagem está em NP => ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial

- Seja V o verificador de tempo polinomial (n^k) de uma linguagem A em NP. A MTN N será:

Prova

Uma linguagem está em NP => ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial

- Seja V o verificador de tempo polinomial (n^k) de uma linguagem A em NP. A MTN N será:

N = “Sobre a entrada w de comprimento n:

1. Não-deterministicamente selecione uma cadeia c de comprimento no máximo n^k.
2. Rode V sobre a entrada <w,c>
3. Se V aceita, *aceite*; caso contrário *rejeite*.”

Prova

Uma linguagem está em NP <= ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial

-

Prova

Uma linguagem está em NP <= ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial

- Seja N uma MTN que decide uma linguagem A em NP. V o verificador de tempo polinomial de A será:

Prova

Uma linguagem está em NP <= ela é decidida por alguma máquina de Turing não-determinística de tempo polinomial

- Seja N uma MTN que decide uma linguagem A em NP. V o verificador de tempo polinomial de A será:

V = “Sobre a entrada <w,c> onde w e c são cadeias:

1. Simule N sobre a entrada w, tratando cada símbolo de c como uma descrição da escolha não-determinística a fazer a cada passo.
2. Se esse ramo da computação de N aceita, *aceite*; caso contrário *rejeite*.”

Classe de complexidade de tempo não-determinístico

- NTIME(t(n)) = { L | L é uma linguagem decidida por uma máquina de Turing não-determinística de tempo O(t(n))}

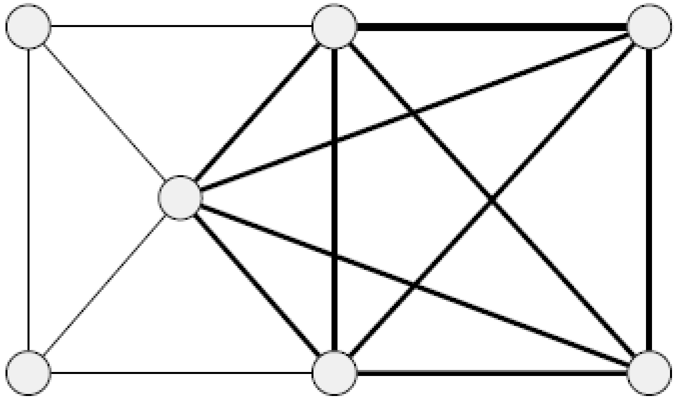
- NP = U_k NTIME(n^k).

- P = U_k TIME(n^k)   (relembrando...)

Exemplos de problemas em NP

Dado um grafo não direcionado G:

- **Clique**: subgrafo no qual todo par de nós está conectado por uma aresta (subgrafo completo)
- **K-clique**: clique de k nós
- Exemplo de 5-clique:



Exemplos de problemas em NP

Linguagem de um grafo com um clique (de qualquer tamanho:)

Exemplos de problemas em NP

Linguagem de um grafo com um clique (de qualquer tamanho:)

CLIQUE = {<G, k> | G é um grafo não-direcionado com um k-clique}

Clique está em NP

- Ideia da prova: quem é o certificado?
-

Clique está em NP

- Ideia da prova: quem é o certificado? O clique
- Prova: verificador V:

Clique está em NP

- Ideia da prova: quem é o certificado? O clique
- Prova: verificador V:

V = "Sobre a entrada <<G,k>,c>:

Clique está em NP

- Ideia da prova: quem é o certificado? O clique
- Prova: **verificador** V:

V = “Sobre a entrada <<G,k>,c>:

1. Teste se c é um conjunto de k nós em G
2. Teste se G contém todas as arestas conectando cada par de nós em c
3. Se ambos os testes forem positivos, *aceite*; caso contrário, *rejeite*.”

Clique está em NP

- Prova alternativa: a MTN N que **decide** CLIQUE:

Clique está em NP

- Prova alternativa: a MTN N que **decide** CLIQUE:

N = “Sobre a entrada <G,k>, onde G é um grafo:

1. Não-deterministicamente, selecione um subconjunto c de k nós de G
2. Teste se G contém todas as arestas conectando cada par de nós de c
3. Se sim, *aceite*; caso contrário, *rejeite*.”

SOMA-SUBC

- Coleção de números x1, ..., xk e um número alvo t. Há uma subcoleção cuja soma seja t?

SOMA-SUBC = {<S, t>| S = {x1, ..., xk} e para algum {y1, ..., yl} subconjunto de {x1, ..., xk}, ∑ yi = t }

SOMA-SUBC está em NP

- Certificado?
-

SOMA-SUBC está em NP

- Certificado? O subconjunto
- Prova: Verificador V:

SOMA-SUBC está em NP

- Certificado? O subconjunto
- Prova: Verificador V:

V = “Sobre a entrada <<S,t>,c>:

1. Teste se c é uma coleção de números que somam t.
2. Teste se S contém todos os números de c.
3. Se ambos os testes forem positivos, *aceite*; caso contrário, *rejeite*.”

SOMA-SUBC está em NP

- Prova alternativa: MTN N decisora:

N = “Sobre a entrada <S,t>:

1. Não-deterministicamente selecione um subconjunto c dos números em S.
2. Teste se c é uma coleção de números que somam t.
3. Se sim, *aceite*; caso contrário, *rejeite*.”

coNP

- Os complementos de CLIQUE e de SOMA-SUBC estão em NP?

coNP

- Os complementos de CLIQUE e de SOMA-SUBC estão em NP?
- Não se sabe...
- Verificar que algo NÃO está presente parece ser mais difícil...
- CoNP = { L | L é uma linguagem que é o complemento de uma linguagem que está em NP}

P versus NP

- P é um subconjunto de NP
- Questão: P é igual ou diferente a NP?

P versus NP

- P é um subconjunto de NP
- Questão: P é igual ou diferente a NP?
  - Um dos maiores problemas não-resolvidos da computação
- Acredita-se que sejam diferentes
- Muitos esforços para encontrar algoritmos polinomiais para certos problemas em NP
- Mas provar que P ≠ NP também é complicado (provar que não existe um tal algoritmo...)
- NP subconjunto de EXPTIME = $\cup_k$ TIME(2^{nk})
- Mas não sabemos se NP é subconjunto de uma classe de complexidade menor