

Problemas NP-Completo

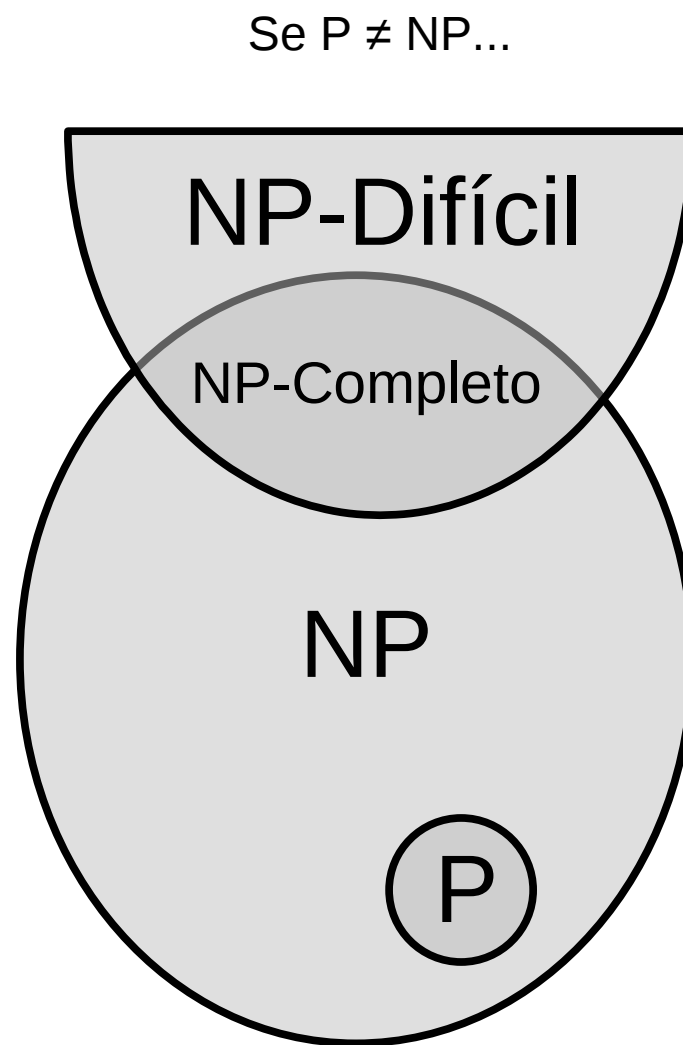
Guilherme D. da Fonseca

www.uniriotec.br/~fonseca

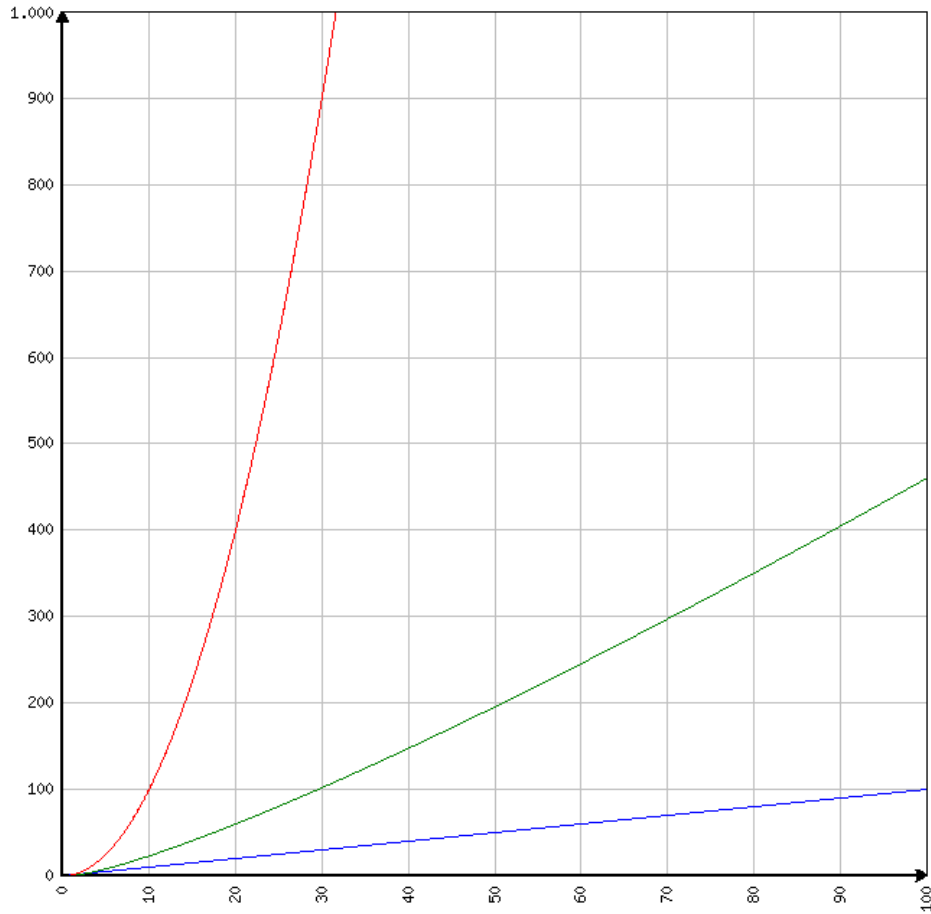
14/11/2009

Aula de Hoje

- Algoritmos polinomiais
- Problemas de decisão
- Classes P e NP
- Redução polinomial
- Classes NP-Difícil e NP-Completo
- Teorema de Cook



Eficiência



- Considere um problema cuja entrada tem tamanho n
- O que é um algoritmo eficiente?
 - Complexidade quase linear $O(n \log^{O(1)} n)$?
 - Complexidade polinomial $O(n^{O(1)})$?
 - Complexidade ótima?

Algoritmos Polinomiais

- Definimos um algoritmo eficiente como um algoritmo que leva tempo polinomial por algumas razões:
 - Conduz a bons teoremas
 - Pouco sensível ao modelo computacional
 - Geralmente os algoritmos polinomiais tem grau relativamente baixo (há exceções!)
 - Polinomios têm boas propriedades
- Se $f(n)$ e $g(n)$ são polinômios, então:
 - $f(n) + g(n)$ é um polinômio
 - $f(n) g(n)$ é um polinômio
 - $f(g(n))$ é um polinômio

Não Polinomialidade

- Quando não encontramos um algoritmo polinomial gostaríamos de provar que não há tal algoritmo
 - Raramente conseguimos tal façanha :(
- E se provarmos que um algoritmo polinomial para um problema implica em algoritmos polinomiais para vários outros problemas considerados difíceis?
 - Conseguimos tal prova para vários problemas importantes :)

Problema de Decisão

Sim

Não

- Um problema de decisão é um problema em que a saída é *sim* ou *não*.
 - Dado um grafo G , G é Euleriano?
 - Dado um grafo G , G é Hamiltoniano?
 - Dados um grafo G e um inteiro k , G possui clique de tamanho k ?
 - Dado um inteiro n , n é primo?

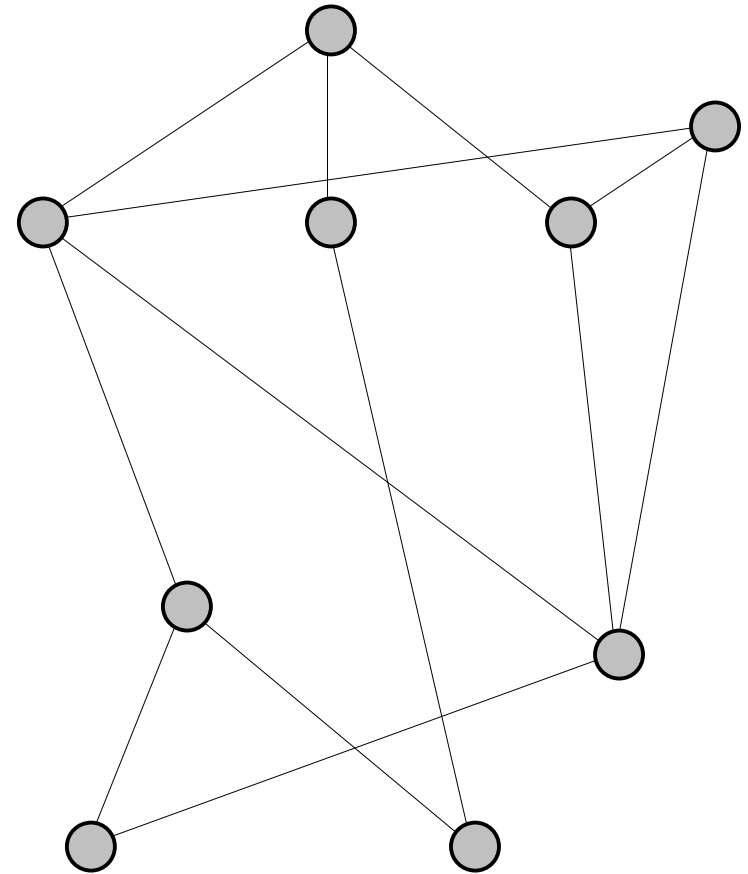
Classe P



- Uma classe de problemas é um conjunto de problemas
- Dizemos que um problema está na classe P se o problema pode ser resolvido em tempo polinomial

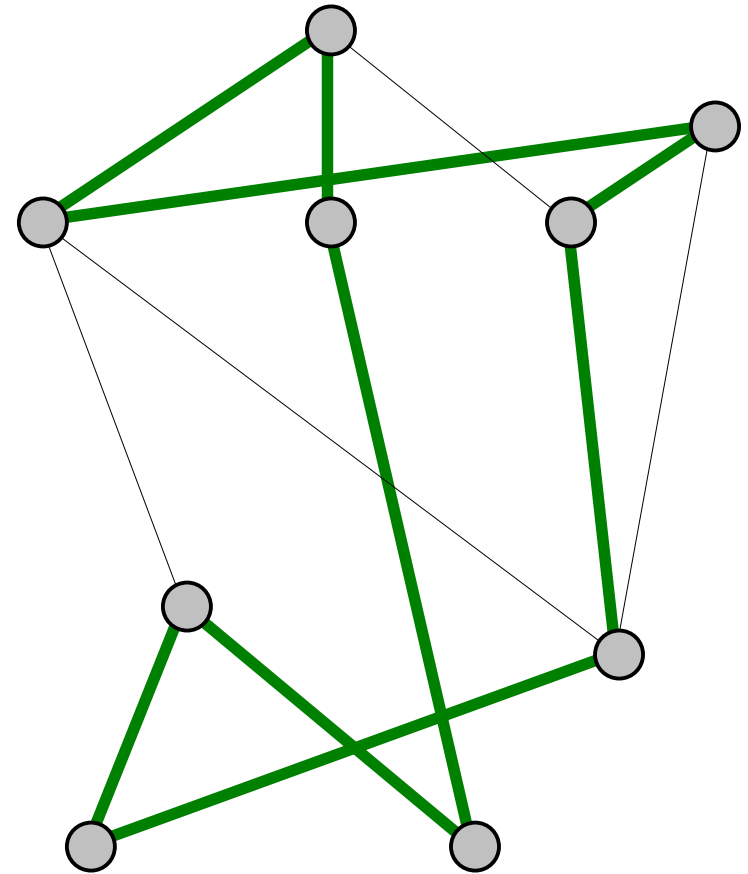
Verificação Polinomial

- Dizemos que uma resposta pode ser verificada em tempo polinomial se existe um certificado com o qual conseguimos verificar, em tempo polinomial, que a resposta está correta
- O grafo ao lado possui ciclo Hamiltoniano?

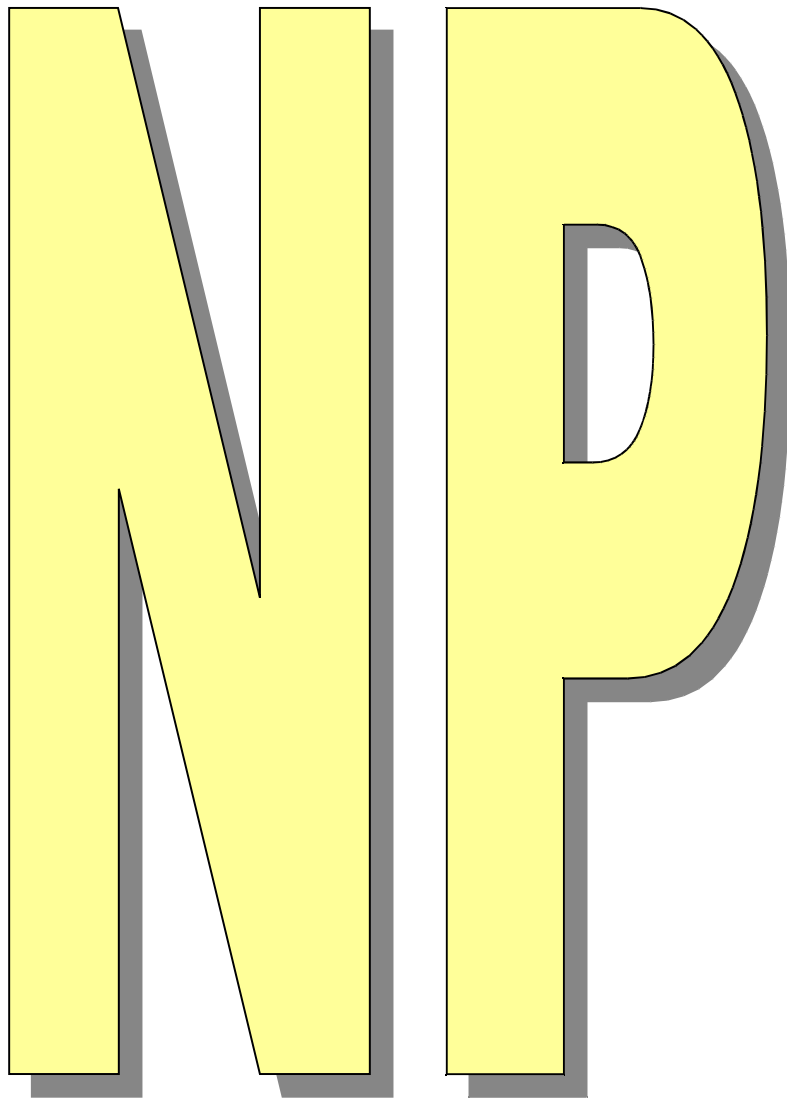


Verificação Polinomial

- Dizemos que uma resposta pode ser verificada em tempo polinomial se existe um certificado com o qual conseguimos verificar, em tempo polinomial, que a resposta está correta
- O grafo ao lado possui ciclo Hamiltoniano?
 - Sim!



Classe NP

The image shows the letters 'NP' in a large, bold, yellow font with a grey drop shadow. The letters are positioned on the left side of the slide.

- Dizemos que um problema está na classe NP se a resposta *sim* pode ser verificada em tempo polinomial
- Claramente $P \subseteq NP$
 - Dispensa certificado
- A pergunta de 1 milhão de dólares:

$P = NP?$

Outra Definição



- Dizemos que um problema está na classe NP se existe máquina de Turing não determinística que resolve o problema em tempo polinomial
- O certificado equivale ao caminho que levou a resposta *sim*
- Uma máquina de Turing não determinística pode tentar verificar todos os certificados possíveis

Redução Polinomial

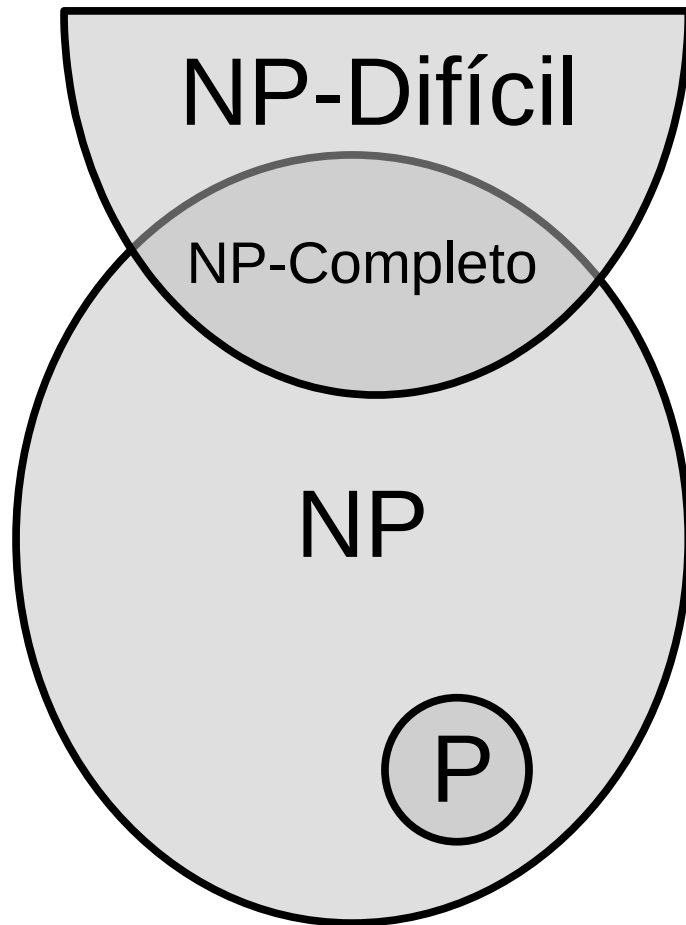
- Uma redução polinomial de um problema X para um problema Y é um algoritmo polinomial tal que:
 - A entrada é uma instância x de X
 - A saída é uma instância y de Y
 - $X(x) = Y(y)$
- Note que se Y é polinomial, então X também é
- Analogamente, se X não é polinomial, então Y também não é

NP-Difícil

- Diz-se que um problema Y é NP-difícil se, para todo problema X em NP, existe redução polinomial de X para Y
 - Um problema NP-Difícil é pelo menos tão difícil quanto qualquer problema em NP
 - Dizer se uma máquina de Turing termina dizendo *sim* é NP-difícil
 - Porém dizer se uma máquina de Turing termina dizendo *sim* não está em NP (de fato, não é nem decidível!)

NP-Completo

Se $P \neq NP \dots$



- Será que um problema pode simultaneamente estar em NP e ser NP-Difícil?
- Surpreendentemente, a resposta é sim!
- Chamamos tais problemas de NP-Completo
- Muitos problemas importantes são NP-Completo

Satisfabilidade

- Problema de satisfabilidade (SAT):
 - Entrada: fórmula lógica composta por operações \wedge , \vee e negação \neg
 - Consideramos que a fórmula está na forma normal conjuntiva: conjunção \wedge de cláusulas que são disjunções \vee de literais
 - Saída: *sim* se existe atribuição de *verdadeiro* e *falso* para as variáveis que façam a fórmula ser verdadeira

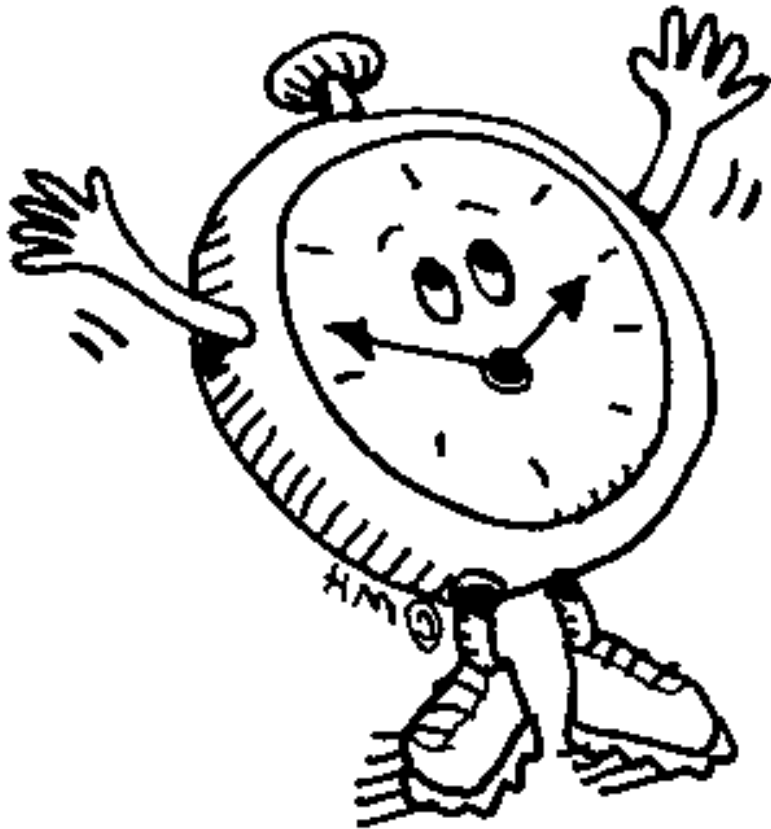
Teorema de Cook

- Teorema: SAT é NP-Completo
- Idéia da prova:
 - SAT está em NP, pois uma atribuição de variáveis é um certificado
 - Dada uma máquina de Turing não determinística de tempo polinomial, constrói-se uma fórmula que é satisfatível se e só se a máquina de Turing responde *sim*
 - As variáveis correspondem ao conteúdo da fita, posição da cabeça e estado em cada instante de tempo

Reduções Novamente

- Sabemos que SAT é NP-Completo
- Podemos provar que um problema X é NP-Completo provando que X está em NP e que SAT reduz polinomialmente para X
- Com isso, conseguimos mostrar que vários problemas são NP-Completos:
 - Clique máxima, ciclo Hamiltoniano, caixeiro viajante, mochila, partição, programação inteira...
- Caso qualquer um desses problemas tenha algoritmo polinomial, $P=NP$

Próxima Aula

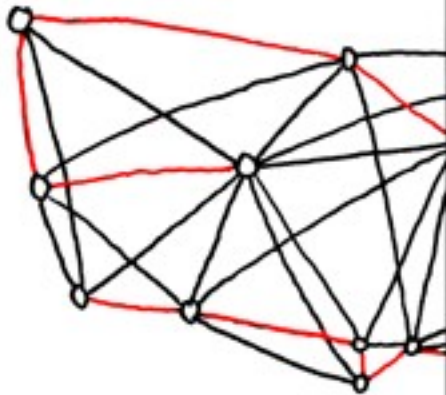


- Provaremos que os seguintes problemas são NP-Completo:
 - Conjunto independente
 - Cobertura por vértices
 - Ciclo Hamiltoniano

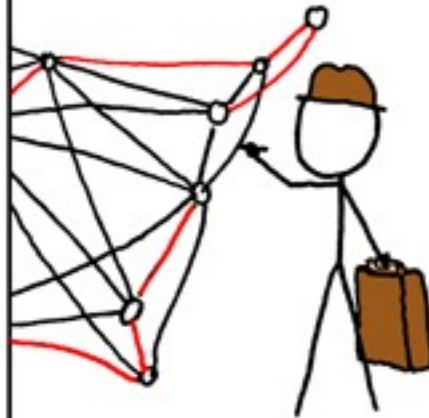
Referências

- *Introduction to Algorithms*, T. H. Cormen, C. Stein, R. L. Rivest, C. E. Leiserson, McGraw-Hill, 2001.
- *Algorithm Design*, Michael T. Goodrich, Roberto Tamassia, John Wiley & Sons, 2001.
- *Algorithms*, Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani, McGraw-Hill, 2006.

BRUTE-FORCE
SOLUTION:
 $O(n!)$



DYNAMIC
PROGRAMMING
ALGORITHMS:
 $O(n^2 2^n)$



SELLING ON EBAY:
 $O(1)$

STILL WORKING
ON YOUR ROUTE?

SHUT THE
HELL UP.

