Universidade Federal de Ouro Preto
Departamento de Computação

decom
departamento
de computação

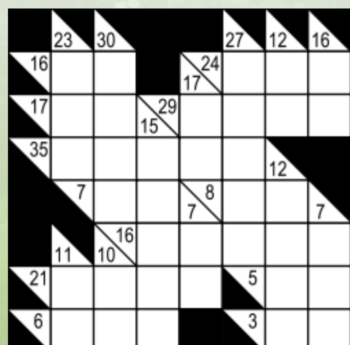
P e NP

Haroldo Gambini Santos
Projeto e Análise de Algoritmos

Busca e Decisão

- **Decisão**:
  - Respostas SIM ou NÃO
    - Existe uma clique de tamanho *k* no grafo ?
    - Existe um preenchimento da mochila com lucro *z* ?
- **Busca**:
  - Determine uma clique de tamanho *k* no grafo ou diga que a mesma não existe.
  - Determine um preenchimento da mochila com lucro *z* ou diga que o mesmo não existe.

Kakuro



Pode-se resolver o Kakuro somente resolvendo problemas de decisão ?

Problemas de Otimização

- Solução que satisfaça as restrições
- Minimizar/Maximizar função $f(x)$
- Também facilmente vistos como problemas de Busca → Decisão

NP – *Nondeterministic Polynomial Time*

- Problemas de Decisão
- Instância *I*
- Solução Sim ou Não
- Provas (solução) pode ser ***checada*** em tempo polinomial

P

Problemas em NP para os quais se conhece algum algoritmo de solução com complexidade de ***tempo polinomial***

Exemplos:
- Caminho Mínimo
- Casamento em grafos bipartidos
- Programação linear

Questão: os avanços em algoritmos irão fazer com que P = NP ?

Conjectura

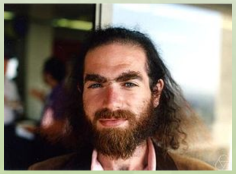
$$P \neq NP$$

Ganhe US$ 1.000.000

- The Clay Mathematics Institute Millenium Prize Problems
  - Birch and Swinnerton-Dyer Conjecture
  - Hodge Conjecture
  - Navier-Stokes Equations
  - **P vs NP**
  - Poincaré Conjecture
  - Riemann Hypothesis
  - Yang-Mills Theory

Digressão

- Os problemas do *Millenium Prize* são factíveis de resolução ?
- 2006 - Grigori Perelman resolveu a conjectura de Poincaré



- Recusou o prêmio alegando que a felicidade por resolver o problema era mais importante do que o prêmio – se disse “não interessado em dinheiro ou fama”

Está o P = NP para ser resolvido ?

- Se sim, muitos outros problemas serão resolvidos
  - Prova automática de teoremas

Situação Atual

Alguns Problemas **NP-Completos**	Alguns Problemas em **P**
3SAT	2SAT, Horn SAT
Caixeiro Viajante	Árvore de cobertura com peso mínimo
Caminho mais longo	Caminho mínimo
Casamento 3D	Casamento bipartido
Conjunto independente	Conjunto independente em árvores
Programação Linear Inteira	Programação Linear
Ciclo Hamiltoniano	Ciclo Euleriano

NP-Completude

- Os problemas mais difíceis em NP são ditos **NP-Completos**
- Esses problemas são intimamente relacionados, de modo que a resolução de 1 implica a resolução de *todos*

Redução entre problemas

- Informalmente: resolver um novo problema usando um antigo que já sabemos resolver
- Para problemas de decisão ***A*** e ***B***, ***A*** é dito <u>redutível em tempo polinomial</u> para ***B*** (denotado por $\boldsymbol{A \le_p B}$) se existe um uma função computável em tempo polinomial tal que:

q é uma instância de $A \Leftrightarrow B$ $f(q)$ é uma instância de B

Lemma: a) se $\boldsymbol{A \le_p B}$ e ***B*** é da classe ***P*** então ***A*** também é da classe ***P***
b) se $\boldsymbol{A \le_p B}$ e ***A*** não é da classe ***P*** então ***B*** também não será da classe ***P***

Redução entre problemas

- Provando que um problema é fácil, exemplo
  - Buscar o menor elemento em um vetor está em *P* (O(n)))
  - Ordenação é redutível ao problema da busca do menor elemento, de modo que também está em *P*
- Redução vetor v[] de entrada e s[] de saída:
  - Repita n-1 vezes:
    - encontre o valor mínimo em v[]
    - coloque em na última posição em s[]
    - remova o valor de v[]

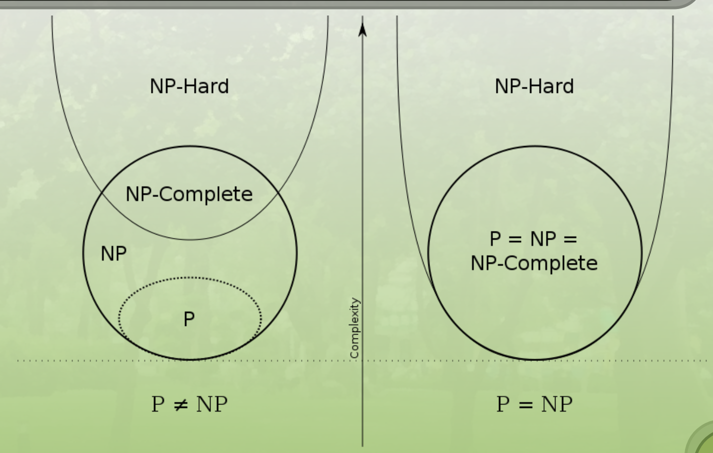
NP-Completude

- Um problema de decisão ***S*** é considerado **NP-Completo** se
  a)estiver em NP e
  b)para todo *A* em NP temos que $\boldsymbol{A \le_p S}$

Provando que um problema é NP-Completo:
• selecione um problema já conhecido como NP-Completo
• Pegue esse problema e faça uma redução em tempo polinomial para o seu problema

- **NP-Difícil** (*NP-Hard*) problema de busca, decisão ou otimização que satisfaz as condições acima

NP-Completude



fonte: wikipedia

SAT

- SAT – Problema de **Satisfatibilidade Booleana**
- Resultado original: Cook, 1971, problema *mãe* de todos os problemas **NP-Completos**
- Decidir valores para variáveis em fórmulas lógicas de modo que a forma dê como resultado **verdadeiro**
- Ex. de instância SAT:

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge \ldots$$

$$x_i \in \{verdadeiro, falso\}$$

$$\overline{x_i} = \text{negação de } x_i$$

SAT - Modelos

- Variáveis:

$$x_i \in \{verdadeiro, falso\}$$

$$\overline{x_i} = \text{negação de } x_i$$

- Forma normal conjuntiva:
  - Conjunção $\wedge$ **(e)** de várias ...
  - Cláusulas **∨ (ou)** disjuntivas
- Exemplo

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge \ldots$$

3SAT

- Versão do SAT onde temos no máximo 3 variáveis por cláusula
- Essa restrição deixa o 3SAT mais fácil ?
- Na verdade, o 3SAT é ***tão difícil quanto*** o SAT, podemos mostrar isso reduzindo o SAT ao 3 SAT

SAT → 3SAT

- Transformando cláusulas com mais de 3 valores

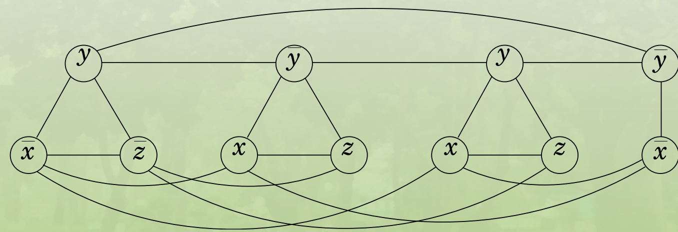
$$(x_1 \vee x_2 \vee \ldots x_k)$$

$$\Leftrightarrow$$

$$(x_1 \vee x_2 \vee y_1) \wedge (\overline{y_1} \vee x_3 \vee y_2) \cdots (\overline{y_{k-3}} \vee x_{k-1} \vee x_k)$$

3SAT → Conjuntos Independentes

$(\bar{x} \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee y \vee z) \wedge (\bar{x} \vee \bar{y})$



Note que vértices ***diferentes*** podem representar a mesma variável.

Note que encontrar o conjunto independente máximo corresponde a encontrar a ***Clique*** máximo no grafo complementar.

Clique de tam. *k* ≤ *MeiaClique*

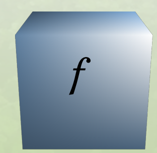
- Problema MeioClique (MC):
  - Dado um grafo $G=(V,E)$, com $n=|V|$ vértices e $m=|E|$ arestas, encontrar uma clique com exatamente a metade dos vértices
- Vamos provar que MC é NP-Completo através da redução do problema de encontrar uma clique de tamanho *k* (CK) para MC, ou seja mostrar que CK ≤ MC

Clique de tam. *k* ≤ *MeiaClique*

- Prova:
- Inicialmente, mostramos que MC está em NP.
  - Solução pode ser verificada em tempo polinomial?
    - Solução $S \subseteq V$: verificar se $|S|=|V|/2$ e para cada (v_1,v_2) S checar se $(v_1,v_2) \in E$ – $O(n^2)$ – está em NP

Transformação Instância CK → MC

$X \rightarrow$



$\rightarrow f(X)$

Instância da Problema da Clique de tam. *k*

Instância da Problema da *MeiaClique*

Função de transformação *f* deve rodar em tempo polinomial

Transformação Instância CK → MC

$X \rightarrow$

$\rightarrow f(X)$

Instância da Problema de encontrar clique de tam. k em G

Instância da Problema da *MeiaClique*: encontrar clique de tam. $|G|/2$

3 casos:
$k = |G|/2$
$k > |G|/2$
$k < |G|/2$

Clique de tam. *k* ≤ *MeiaClique*

- Transformações:
  - $k = |G|/2$: criar $G' = G$ **$O(n^2)$**
  - $k < |G|/2$: criar G' inicialmente igual a G e adicionar $(|G|/2) - 2k$ vértices em G' , todos conectados com todos os vértices do grafo **$O(n^2)$**
  - $k > |G|/2$: criar G' inicialmente igual a G e adicionar $2k-(|G|/2)$ vértices em G' , todos sem conexão alguma **$O(n^2)$**
- Solução de MC em G' contém clique de tamanho k se o mesmo existir, vértices artificiais podem ser ignorados