Introdução à classe de problemas NP-Completos: Redução de problemas

R. Rossetti, L. Ferreira, H. L. Cardoso, F. Andrade

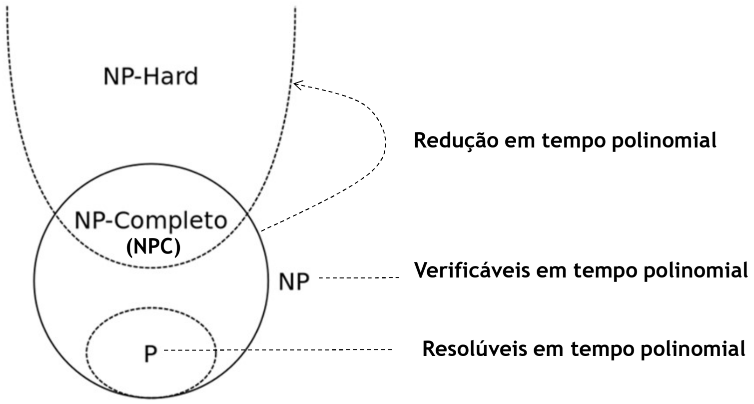
FEUP, MIEIC, CAL

Índice

- Revisão da aula anterior
- Exemplos de reduções

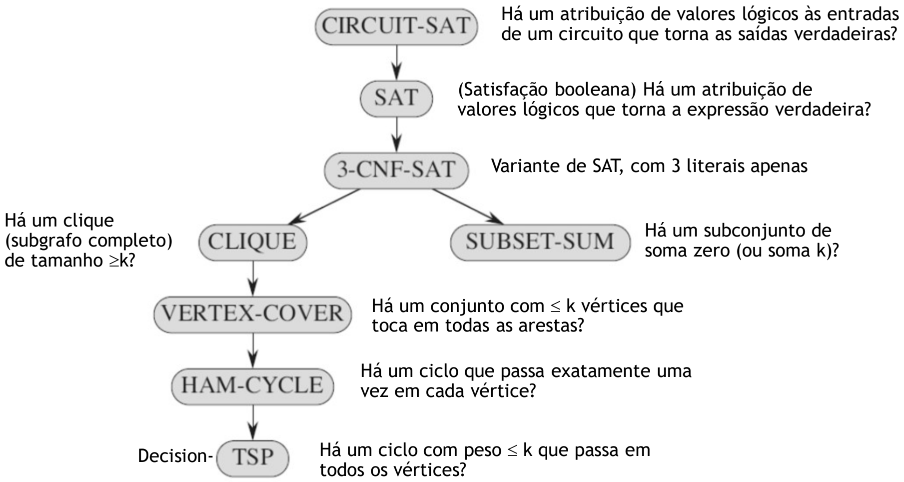
Revisão da aula anterior

Classes P, NP, NP-*complete* e NP-*hard*



Na hipótese P ≠ NP!

Exemplos de problemas NPC e reduções



Redução de Problemas NP

- Definição
  - Dados dois problemas, ***A*** e ***B***, diz-se que ***A*** é polinomialmente redutível a ***B*** se, dada uma subrotina de tempo polinomial para ***B***, pode-se utilizá-la para resolver ***A*** em tempo polinomial. Quando tal se verifica, expressa-se por

    $$A \leq_P B$$
  - **Lema**: Se $A \leq_P B$ e $B \in P$ então $A \in P$
  - **Lema**: Se $A \leq_P B$ e $A \notin P$ então $B \notin P$
  - **Lema**: Se $A \leq_P B$ e $B \leq_P C$ então $A \leq_P C$ (transitividade)

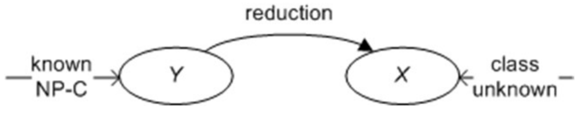
Redução de Problemas NP

- Definição +formal da classe dos problemas NP- completo
  - Um **problema de decisão** $B \in$ NP é NP-completo se
    $A \leq_P B \mid \forall A \in NP$
  - Assim, se *B* pode ser resolvido em tempo polinomial, então qualquer outro problema *A* em NP é resolúvel em tempo polinomial
  - **Lema**: *B* é NP- completo se

    (1) $B \in NP$, e
    (2) $A \leq_P B$ para algum problema *A,* se *A* é *NP-Completo*

Como provar que um problema $X \in$ NPC

1. Provar que *X* está em *NP*
2. Seleccionar um problema *Y* que se sabe ser *NP*-completo
3. Definir uma redução de tempo polinomial de *Y* em *X* *(conversão de entradas)*
4. Provar que, dada uma instância de *Y*, *Y* tem uma solução se, e se somente, *X* tem uma solução *(conversão de saídas)*



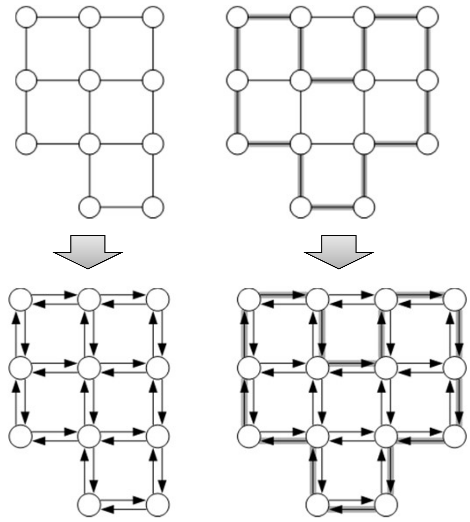
Exemplos de reduções

Directed Hamiltonian Cycle (DHC) é NPC?

- Problema: Sabendo-se que o problema UHC (*Undirected Hamiltonian Cycle*) é NP-completo, provar que o problema DHC (*Directed Hamiltonian Cycle*) é também NP-completo
- Resolução:
  a) Um ciclo Hamiltonian candidato é facilmente verificável em tempo polinomial, logo DHC $\in$ NP
  b) O problema UHC é facilmente redutível ao problema DHC em tempo polinomial (ver slide seguinte), logo DHC $\in$ NPC

Redução de UHC a DHC

- Dado grafo não dirigido $G=(V,E)$, cria-se grafo dirigido $G'=(V,E')$ pela substituição de cada aresta $\{u, v\} \in E$ por duas arestas dirigidas (u, v) e $(v, u) \in E'$
- Cada caminho simples em G é, portanto, um caminho simples em G', e vice-versa. Portanto, G terá um ciclo de Hamilton se, e somente se, G' também o tiver!



Redução de Problemas NP

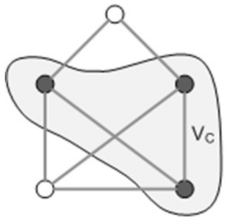
- Solução
  - Considera-se q/ a rotina `bool DHC (aDigraph)` resolve o problema!
  - Redução UHC → DHC

```
bool UHC (G) {
   create digraph G' with the same number of vertices as G
   foreach edge (u, v) in G
      Add edges (u, v) and (v, u) in G'
   return DHC (G')
}
```

  - Note-se que nenhum dos problemas foi efectivamente resolvido. Apenas demonstrou-se como converter uma solução para o DHC numa solução para o UHC. Este procedimento é chamado "**redução**" e é crucial para a teoria dos problemas NP-completos.

Vertex Cover (VC)

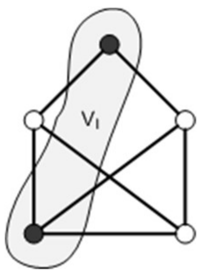
- Uma cobertura de vértices de um grafo $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ é um subconjunto $\boldsymbol{V_C} \subseteq \boldsymbol{V}$, tal que toda aresta $(a, b) \in \boldsymbol{E}$ é incidente em pelo menos um vértice $u \in \boldsymbol{V_C}$.



  - Vértices em $\boldsymbol{V_C}$ "cobrem" todas as arestas em $\boldsymbol{G}$.
- Problema de decisão (*VC*):
  - O grafo $\boldsymbol{G}$ tem uma cobertura de vértices de tamanho $\leq k$?

Independent Set (IS)

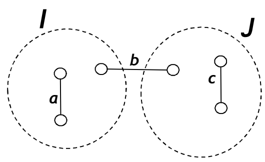
- Um conjunto independente de um grafo $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ é um subconjunto $\boldsymbol{V_I} \subseteq \boldsymbol{V}$, tal que não há dois vértices em $\boldsymbol{V_I}$ que partilham uma aresta de $\boldsymbol{E}$
  - $u, v \in \boldsymbol{V_I}$ não podem ser vizinhos em $\boldsymbol{G}$.



- Problema de decisão (*IS*):
  - O grafo $\boldsymbol{G}$ tem um conjunto independente de tamanho $\geq k$?

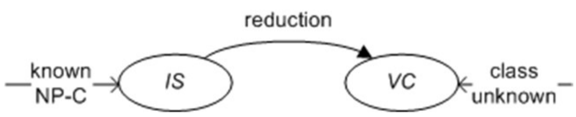
Dualidade VC ↔ IS

- Dado grafo não dirigido $G=(V,E)$, seja I, J uma partição de V em dois subconjuntos disjuntos (i.e., $I \cup J = V$ e $I \cap J = \emptyset$)
- Se I é um conjunto independente de vértices, então não podem existir arestas do tipo a, logo os vértices em J tocam todos as arestas de G, logo J é uma cobertura de vértices
- Se J é uma cobertura de vértices, então não podem existir arestas do tipo a, logo I é um conjunto independente de vértices.
- I é um conj. indep. de vértices $\Leftrightarrow$ $V \setminus I$ é uma cobertura de vértices



Vertex Cover é NPC?

- Problema: Sabendo-se que **IS** $\in$ **NPC**, provar que **VC** $\in$ **NPC**
- Resolução:
  a) Dada um conjunto candidato de vértices V_C, é fácil verificar em tempo polinomial *se* $|V_C| \leq k$ e se toca em todas as arestas, logo **VC∈NP**
  b) Para provar que **VC ∈ NP-hard**, indicamos de seguida uma redução de tempo polinomial de **IS** em **VC**

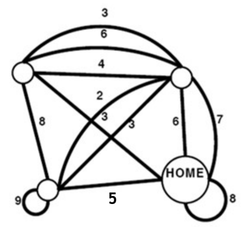


Redução de IS a VC

- Seja uma instância qualquer de IS: ***G*** = (***V***, ***E***), *k*
- Pela propriedade da dualidade, *G* tem um conjunto independente de vértices (V_I) de tamanho $\geq k$ sse tiver uma cobertura de vértices (V_c) de tamanho $\leq k'$, com $k'=|V|-k$
- Assim, a conversão de entradas é trivial:
  - Dada uma instância qualquer de IS: ***G*** = (***V***, ***E***), *k*
  - Constrói-se uma instância de VC: ***G*** = (***V***, ***E***), $k' = |V| - k$
- A conversão de saídas é também trivial:
  - Conversão de 'certificados': $V_c \rightarrow V_I = V \setminus V_c$
  - Conversão de decisão: mantém-se a mesma decisão

Problema do Caminhada (*Jogging (J)*)

- Considere um grafo não dirigido *G*, admitindo arestas paralelas e anéis, com pesos inteiros positivos nas arestas, no qual se distingue um vértice *home*.
- O problema da caminhada (*Jogging (J)*) consiste em verificar se existe um caminho de peso total *k*, começando e terminando em *home*, sem repetir arestas.
- Prove que ***J*** é um problema NPC, sabendo-se que o problema da soma de subconjuntos é NPC.

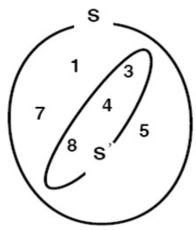


Problema da Soma de Subconjuntos (SS)

- Dado um conjunto de inteiros positivos, ***S***, há um subconjunto, ***S'*** em ***S***, tal que a soma dos elementos de ***S'*** seja *k*?

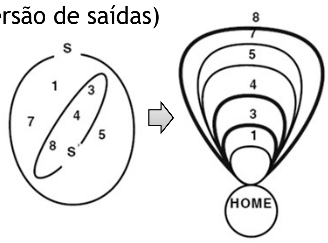
Ex:
S = {1, 3, 4, 5, 7, 8}

Find S' with sum = 15!



Problema da Caminhada é NPC?

- Um caminho candidato é facilmente verificável em tempo polinomial, logo <u>J ∈ NP</u>
- Para provar que <u>J ∈ NP-hard</u>, reduz-se SS a J em tempo polinomial. Como?
  - Dado um conjunto *S*, cria-se um grafo *G* com um único vértice *home* e um anel de peso *x* para cada elemento *x*∈*S* (conversão de entradas)
  - *S* tem um subconjunto de soma *k* sse *G* tem um caminho de peso total *k* sem repetir arestas (conversão de saídas)



Referências e mais informação

- T. Cormen *et al*. (2009) "**Introduction to Algorithms.**" Cambridge, MA: MIT press.
  - Capítulo 34 NP-Completeness
  - Capítulo 35 Approximation Algorithms
- R. Johnsonbaugh & M. Schaefer (2004) "**Algorithms**." Upper Saddle River, NJ: Prentice Hall.
- C.A. Shaffer (2001) "**A Practical Introduction to Data Structures and Algorithm Analysis.**" Upper Saddle River, NJ: Prentice Hall.