Reduções

Disclaimer

Slides baseados nos slides dos Prof.

- Ichilton Pedrosa
- Carla N. Lintzmoyer

Redução

- Redução entre problemas é uma técnica muito importante de algoritmos
  - A ideia é usar um algonitmo que existe, ou que ainda será criado, para um problema para resolver outro.

Exemplo:

- Problema da seleção
- Problema de ordenação

Exemplo

Problema de Seleção

Entrada: $\langle V, n, k \rangle$, onde V é um vetor de tamanho n e $k \in \mathbb{N}$

Saída: o k-ésimo menor elemento que está armazenado em V.

Exemplo de Instância

Entrada:

- $V = (3, 7, 12, 6, 8, 234, 9, 78, 45)$
- $k = 5$

Saída:

- 9

Exemplo

Problema da Ordenação

Entrada: $\langle V, n \rangle$ um vetor V de n elementos

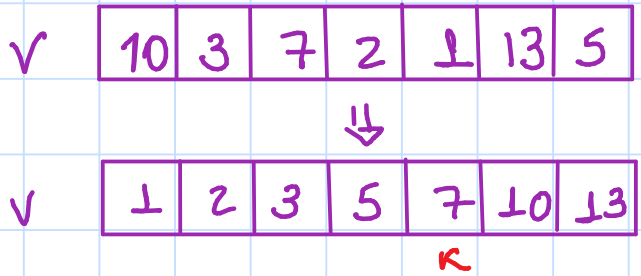
Saída: uma permutação $V' = (a_1, a_2, \cdots, a_n)$ dos elementos de V tal que $a_1 \leq a_2 \leq \cdots \leq a_n$

Redução do Problema da Seleção para Ordenação

Alg (V, n, k)

Algoritmo-Ordenação (V, n)

Retorna V[k]



- Acabamos de reduzir o problema da seleção no de ordenação, i.e., transformamos o problema de resolver o prob. da seleção no prob. de resolver a ordenação.

Formalizando Problema Computacional

Um **Problema Computacional** é uma relação $P \subseteq I \times S$, onde

- I é o conjunto das entradas
- S é o conjunto das saídas

Problema Quadrado

Entrada: $x \in \mathbb{N}$

Saída: x^2

$$P = \{ (1,1), (2,4), (3,9), (4,16), \ldots \}$$

Algoritmo p/ um Problema

Dizemos que um algoritmo ALG **resolve** um problema $P \subseteq (\mathcal{I}, \mathcal{S})$ se, para toda entrada $i \in \mathcal{I}$, ele devolve $s \in \mathcal{S}$, tal que $(i, s) \in P$.

- escrevemos ALG(i) para representar a saída do algoritmo para a instância i

Redução

Uma **redução** do problema A ao problema B é um par de sub-rotinas σ_1 e σ_S tais que

- σ_1 transforma instância I_A de A em uma instância I_B de B
- σ_S transforma a solução S_B de I_B em uma solução S_A de I_A

Significa que existe uma redução do prob. A ao prob. B.

- Se A é **redutível** à B, então escrevemos

$$A \leq B$$

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

$$I_A \xrightarrow{\sigma_I} I_B \longrightarrow ALG_B \longrightarrow S_B \xrightarrow{\sigma_S} S_A$$

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

Redução (I_A)

$I_B \leftarrow \sigma_I(I_A)$

$S_B \leftarrow ALG_B(I_B)$

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

Redução (I_A)

$I_B \leftarrow \sigma_I(I_A)$

$S_B \leftarrow ALG_B(I_B)$ ← caixa-preta

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

Redução (I_A)

$I_B \leftarrow \sigma_I(I_A)$

$S_B \leftarrow ALG_B(I_B)$ ← caixa-preta

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

Tempo Execução: tempo σ_I + tempo ALG_B + tempo σ_S

Tempo de Redução

- Tempo de redução é o tempo gasto apenas com a redução
- A complexidade de uma redução é a soma $f(n)$ dos tempos das transformações σ_1 e σ_S
- Escrevemos $A \preceq_{f(n)} B$

$$I_A \xrightarrow{\sigma_1} I_B \longrightarrow ALG_B \longrightarrow S_B \xrightarrow{\sigma_S} S_A$$

Redução (I_A)

$I_B \leftarrow \sigma_1(I_A)$

$S_B \leftarrow ALG_B(I_B)$

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

$$f(n) = T(\sigma_1) + T(\sigma_2)$$

Reduções Polinomiais

- Queremos construir algoritmos eficientes (polinomiais)
  - queremos reduções de tempo polinomiais
  - nesse caso, escrevemos $A \leq_{poli} B$

- Consequência de $A \leq_{poli} B$
  1. Se B tem algoritmo polinomial, então A também tem
  2. Se A não tem, tempouco B

- Isto é útil para distinguir problemas fáceis de difíceis
  - mais sobre isso nos próximos capítulos.

Redução (I_A)

$I_B \leftarrow \sigma_1(I_A)$

$S_B \leftarrow ALG_B(I_B)$

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

Usando Redução para Criar um Algoritmo

- Queremos resolver um problema A
- Sabemos reduzir A para um problema B
- Suponha que existe um algoritmo ALG_B para B

$$I_A \xrightarrow{\sigma_I} I_B \longrightarrow ALG_B \longrightarrow S_B \xrightarrow{\sigma_S} S_A$$

Redução (I_A)

$I_B \leftarrow \sigma_I(I_A)$

$S_B \leftarrow ALG_B(I_B)$

$S_A \leftarrow \sigma_S(S_B)$

devolva S_A

Conclusão:

- Se sei resolver B, então tbm sei resolver A
- A é mais "fácil" que B
- denotamo $A \leq B$

Problema da Seleção de Atividades (SA)

Entrada: $\langle T, n, s, f \rangle$, onde $T = \{a_1, a_2, \ldots, a_n\}$ é um conjunto com n atividades onde cada a_i tem tempo inicial s_i e final f_i.

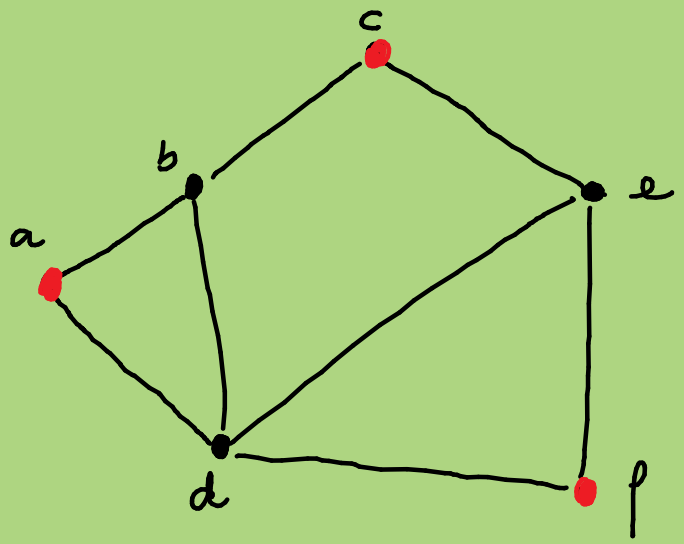
Saída: Maior subconjunto $S \subseteq T$ de tarefas mutuamente compatíveis

Vamos assumir que $f_1 \leq f_2 \leq f_3 \leq \cdots \leq f_n$

Problema do Conjunto Independente Máximo (CIM)

Entrada: $\langle G \rangle$, onde G é um grafo

Saída: Conjunto independente $S \subseteq V(G)$ de tamanho máximo



$S = \{a, c, f\}$

Reduzindo o SA p/ CIM

$\sigma_1(T, n, s, f)$

Seja G um grafo tal que $V(G) = \{v_1, v_2, \ldots, v_n\}$

para $i \leftarrow 1$ até n

para $j \leftarrow i$ até n

se $s_j < f_i$

$G \leftarrow G \cup ij$

Retorna G

Tempo: $\Theta(n^2)$

Reduzindo o SA p/ CIM

$\sigma_S(G, S)$

Retorna S

Tempo: $\Theta(1)$

Reduzindo o SA p/ CIM

Redução-SA-CIM (T, n, s, f)

1. $G \leftarrow \sigma_1(T, n, s, f)$
2. $S_{CIM} \leftarrow ALG_{CIM}(G)$
3. $S_{SA} \leftarrow \sigma_2(G, S_{CIM})$
4. Devolva S_{SA}

$SA \leq_{n^2} CIM$

Tempo Total: $\underbrace{\Theta(n^2)}_{\text{tempo Redução}}$ + tempo de ALG_{CIM}

Reduzindo o SA p/ CIM

Lema Seja $I_{SA} = (T, n, s, f)$ uma instância do problema SA e seja $G = \sigma_1(I_{SA})$. Se S_G é uma solução ótima para CIM com entrada G, então $\sigma_S(G)$ é solução ótima para I_{SA}.

Esse resultado segue diretamente do seguinte lema

Lema Seja $I_{SA} = (T, n, s, f)$ uma instância do problema SA e seja $G = \sigma_1(I_{SA})$. S é um conjunto de atividades compatíveis em I_{SA} sse S é um conj. independente em G.

Lema Seja $I_{SA} = (T, n, s, f)$ uma instância do problema SA e seja $G = \sigma_1(I_{SA})$. S é um conjunto de atividades compatíveis em I_{SA} sse S é um conj. independente em G.

($\Rightarrow$)
- Seja S um conj. de atividades compatíveis
- Sejam $a_i, a_j \in S$ e assuma que $i < j$
- Como são compatíveis $f_i \leq s_j$
- Então σ_1 não colocou aresta entre i e j
- Logo S é conj. independente

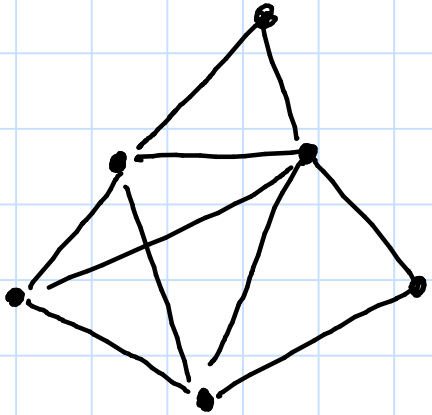
($\Leftarrow$)
- S conj. independente em G
- $i, j \in S$ e suponha $i < j$
- Como σ_1 não colocou a aresta $ij \Rightarrow f_i \leq s_j$
- Portanto i e j são compatíveis

□

Problema Clique Máxima (CIM)

Entrada: um grafo G

Saída: o tamanho da clique Máxima

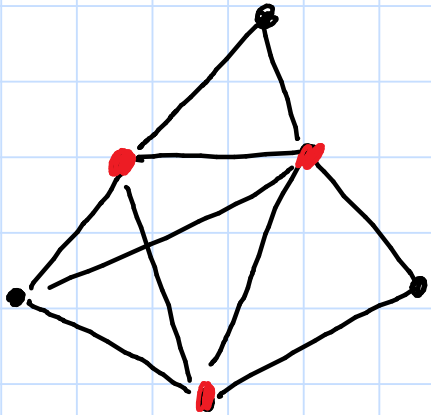


Saída: 4

Dado um grafo G, vamos denotar por cli(G) o maior tamanho de uma clique em G.

Uma cobertura de Vértices é um conjunto $S \subseteq V(G)$ tal que, para toda aresta $xy \in E(G)$, vale que

$$S \cap \{x, y\} \neq \emptyset.$$

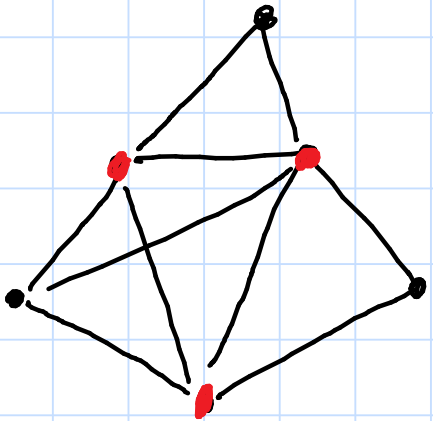


Dado um grafo G, vamos denotar por $cob(G)$ o menor tamanho de uma cobertura de vértices

Problema Cobertura de Vértices (CV)

Entrada: grafo G

Saída: tamanho da cobertura de vértices mínima



Saída: 3

$\sigma_1(G)$

Devolva $\overline{G}$

$\sigma_S(G, x)$

Devolva $|V(G)| - x$

Redução-CV-CIM (G)

$G' \leftarrow \sigma_1(G)$

$x \leftarrow ALG_{CIM}(G')$

$y \leftarrow \sigma_S(G, x)$

Devolve y

Lema Seja G um grafo com n vértices

$$cob(G) = n - cli(\bar{G})$$

Demonstração

- Duas etapas
  - (a) $cob(G) \leq n - cli(\bar{G})$
  - (b) $cob(G) \geq n - cli(\bar{G})$

(b) Seja F uma cobertura de G. Seja $S = V(G) \setminus F$ e note que S é conj. independente em G e que $|S| = n - |F|$.

- note que S é uma clique em $\bar{G}$ e $|S| \leq cli(\bar{G})$
- Quando $|F| = cob(G)$,

$$cli(\bar{G}) \geq |S| = n - |F| = n - cob(G)$$

$$cob(G) \geq n - cli(\bar{G})$$

@ Seja S uma clique em $\bar{G}$ e seja $F = V(\bar{G}) \setminus S$. Note que S é um conjunto independente em G, então todas as arestas de G possuem ao menos um extremo em F, portanto F é um vertex cover de G. Assim $cob(G) \leq |F| = n - |S|$

Sabemos que $|F| = n - |S|$ inclusive qndo $|S| = cli(\bar{G})$. Portanto

$$cob(G) \leq |F| = n - |S| = n - cli(G)$$

□

$\sigma_1(G)$

Devolva $\overline{G}$

$\sigma_S(G, x)$

Devolva $|V(G)| - x$

Redução-CV-CIM (G)

$G' \leftarrow \sigma_1(G)$

$x \leftarrow ALG_{CIM}(G')$

$y \leftarrow \sigma_S(G, x)$

Devolve y

Em um problema de Otimização, para provar a corretude da redução, você precisa ser capaz de estabelecer uma relação entre o valor ótimo do problema original e do problema reduzido

$SA \leq CIM$: $OPT_{SA} = OPT_{CIM}$

$CV \leq Cli$: $OPT_{CV} = n - OPT_{Cli}$

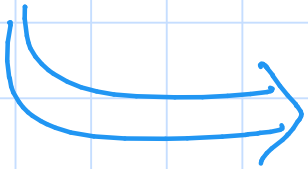
Problema de Decisão

Um problema de decisão é um problema cuja resposta é sim ou não

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $k \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um uv-caminho P tal que $w(P) \leq k$,
não caso contrário.

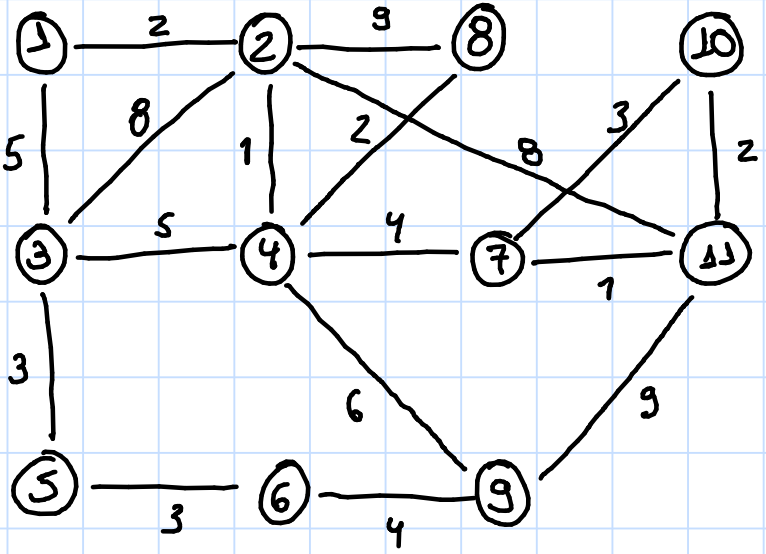


Next

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $K \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um uv-caminho P tal que $w(P) \leq K$,
não caso contrário.



$s = 3$

$t = 10$

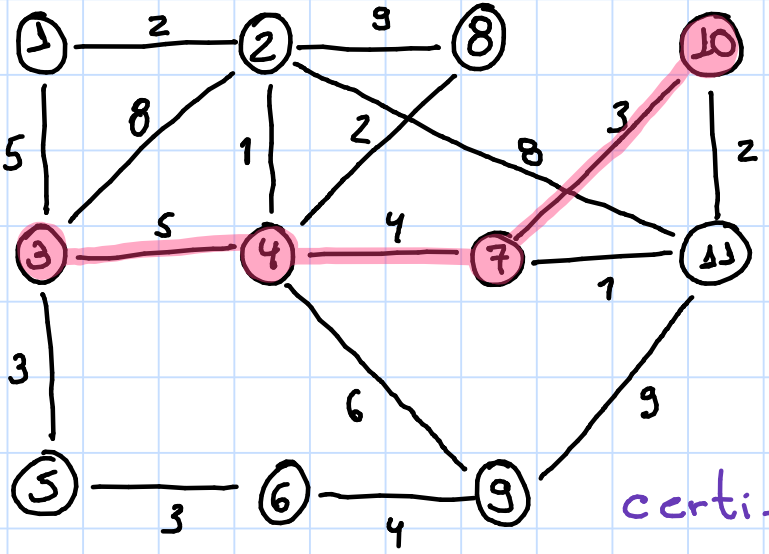
$K = 20$

Saída: SIM

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $K \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um uv-caminho P tal que $w(P) \leq K$,
não caso contrário.



$u = 3$

$v = 10$

$K = 20$

Saída: SIM

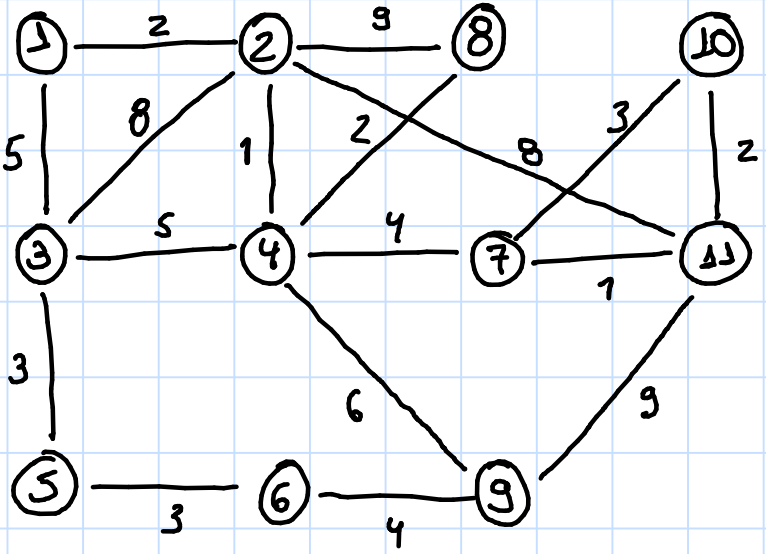
$P = 3, 4, 7, 10$

certificado

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $K \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um uv-caminho P tal que $w(P) \leq K$,
não caso contrário.



$s = 3$

$t = 10$

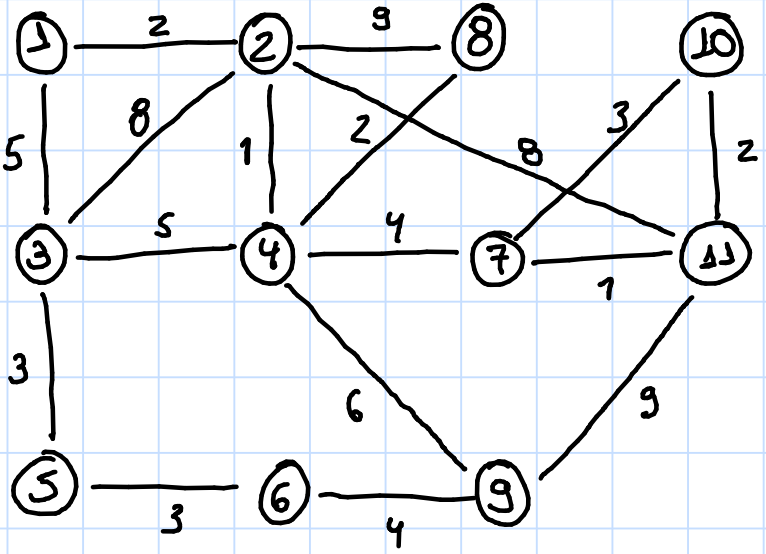
$K = 9$

Saída: NÃO

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $K \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um uv-caminho P tal que $w(P) \leq K$,
não caso contrário.



$s = 3$
$t = 10$
$K = 9$

Saída: NÃO

Certificado?

Problemas de Decisão

Por que estudar problemas de Decisão?

- São mais simples de estudá-los
- Várias situações podem ser modeladas como problema de decisão
- Às vezes, decidir se existe uma solução para um problema é tão difícil quanto encontrá-la.

Instâncias Sim e Não

- Seja $P \subseteq \mathcal{I} \times \mathcal{S}$ um problema de decisão
- Seja $(i, s) \in P$
  - i é uma instância (entrada) do problema P
  - s é a saída esperada (sim ou não)

- Podemos dividir as instâncias de P em dois tipos:

  - instâncias sim : $\{ i : (i, sim) \in P \}$

  - instâncias não : $\{ i : (i, não) \in P \}$

Certificados Positivos e Negativos

- Um certificado de uma instância sim é chamado de certificado positivo

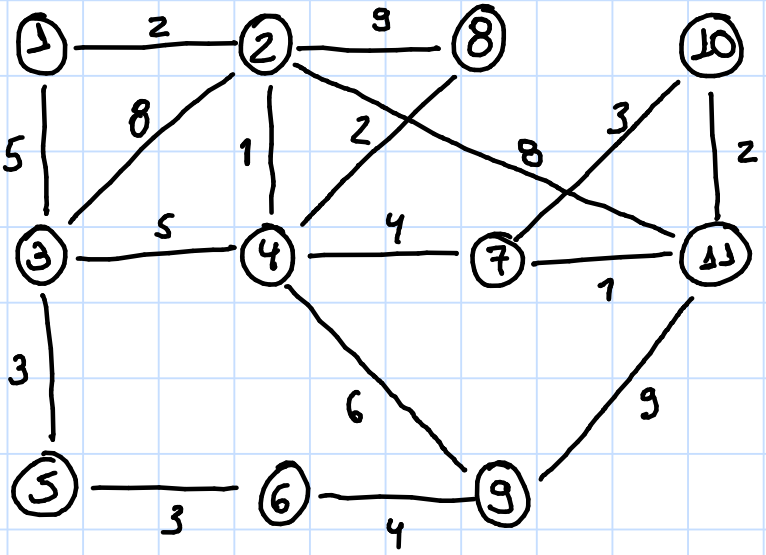
- Um certificado de uma instância não é chamado de certificado negativo

Problema do caminho mínimo

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: $\min \{ w(P) : P \text{ é um } st\text{-caminho} \}$

← prob. de otimização



$s = 3$

$t = 10$

Saída: 12

Certificado?

Problema Decisão vs Otimização

Problema do Caminho

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um valor $k \in \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: sim se existe um st-caminho P tal que $w(P) \leq k$, não caso contrário.

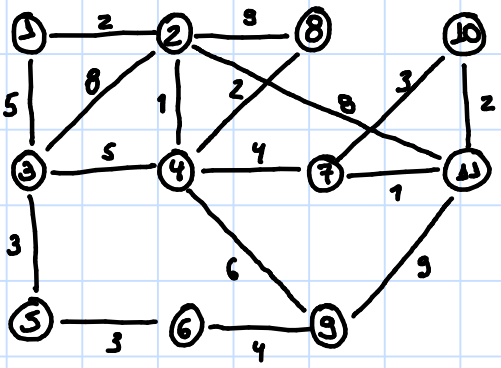
Problema do Caminho mínimo

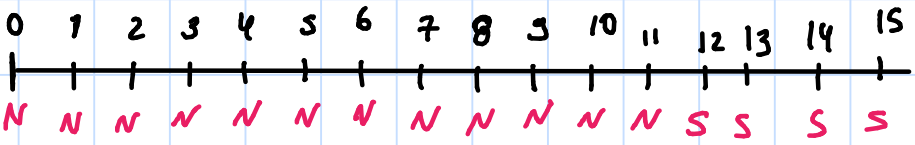
Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, e dois vértices distintos $s, t \in V(G)$

Saída: $\min \{ w(P) : P \text{ é um } st\text{-caminho} \}$

$s = 3$

$t = 10$

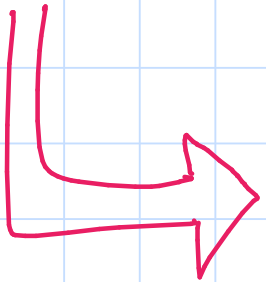




Redução (Versão simplificada p/ prob Decisão)

→ Redução de Karp

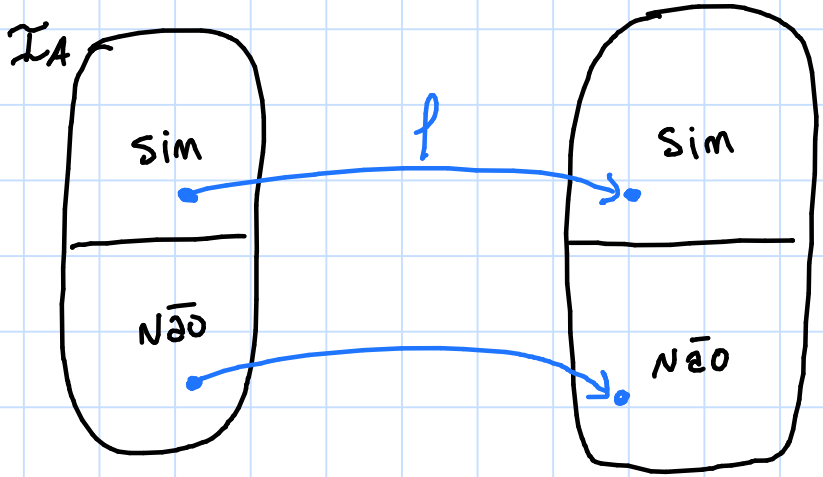
Uma **redução** do problema A ao problema B é uma função f tal que uma instância I_A de A é sim se e somente se $f(I_A)$ é uma instância sim de B.



Next

Redução (Versão simplificada p/ prob Decisão)

Uma **redução** do problema A ao problema B é uma função f tal que uma instância I_A de A é sim se e somente se $f(I_A)$ é uma instância sim de B.

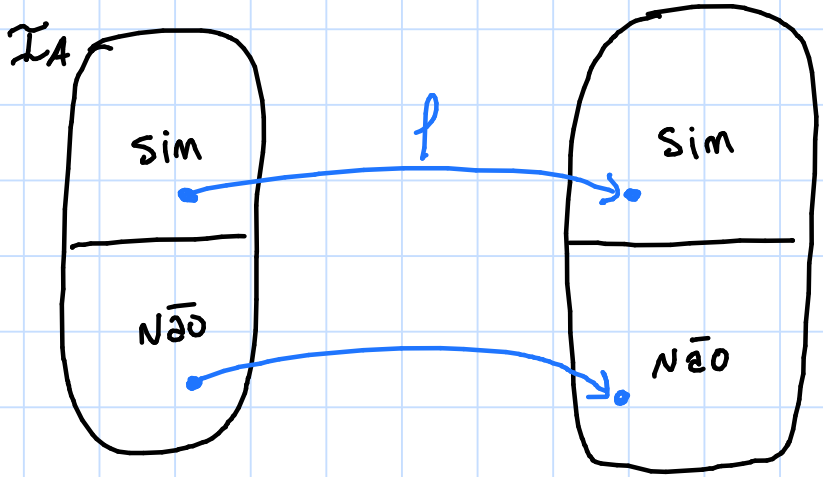


$\mathcal{I}_X$ é o conj. das instâncias do prob X

$$\{f(i) : i \in \mathcal{I}_A\} \subseteq \mathcal{I}_B$$

Redução (Versão simplificada p/ prob Decisão)

Uma **redução** do problema A ao problema B é uma função f tal que uma instância I_A de A é sim se e somente se $f(I_A)$ é uma instância sim de B.



$\mathcal{I}_X$ é o conj. das instâncias do prob X

I_A é sim $\Leftrightarrow$ $f(I_A)$ é Sim

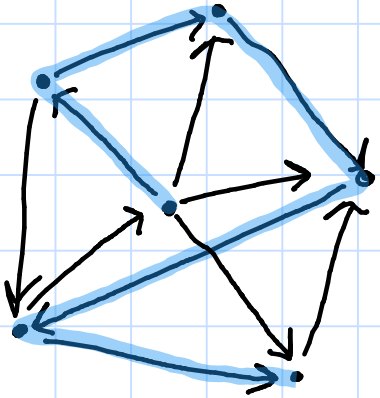
Um caminho Hamiltoniano P de um digrafo D é um caminho gerador de D

→ caminho que contém todos os vértices.

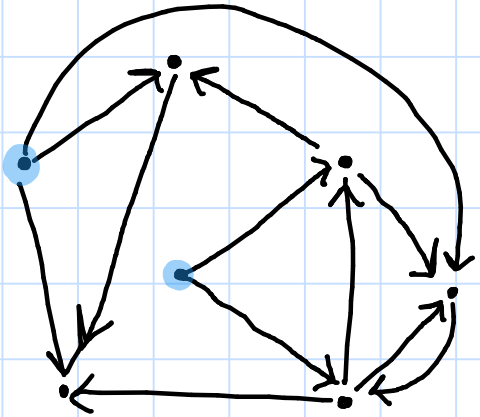
Problema Caminho Hamiltoniano (CaH)

Entrada: um digrafo D

Saída: sim se D contém um caminho Hamiltoniano
não caso contrário



sim



não

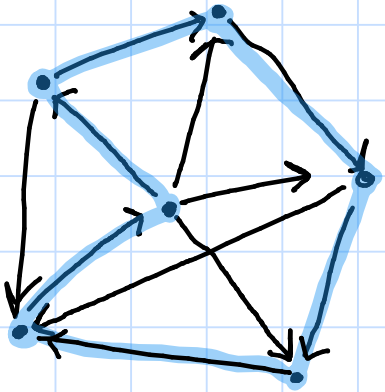
Um ciclo Hamiltoniano P de um digrafo D é um ciclo gerador de D

ciclo que contém todos os vértices.

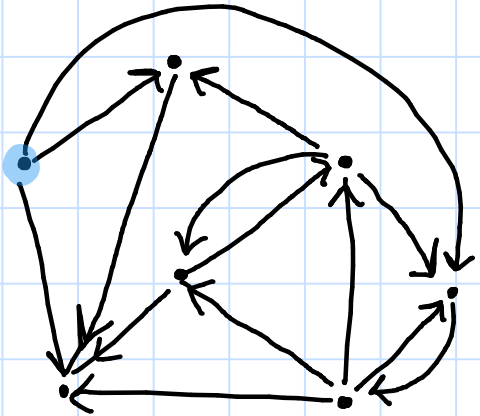
Problema Ciclo Hamiltoniano (CH)

Entrada: Um digrafo D

Saída: sim se D contém um ciclo Hamiltoniano
não caso contrário



sim



não

Lema $CaH \leq_{poli} CH$

Demonstração

Objetivo:

- mostrar redução de Karp:

  Instância sim $CaH \Leftrightarrow$ Instância sim CH

- analisar o tempo gasto na redução e concluir que é polinomial.

Lema $CaH \leq_{poli} CH$

Demonstração

- Seja f o procedimento

$f(D)$ {

Seja D' o grafo obtido a partir de D inserindo um novo vértice w e adicionando as arestas uw e wu à D', para todo $u \in V(D)$

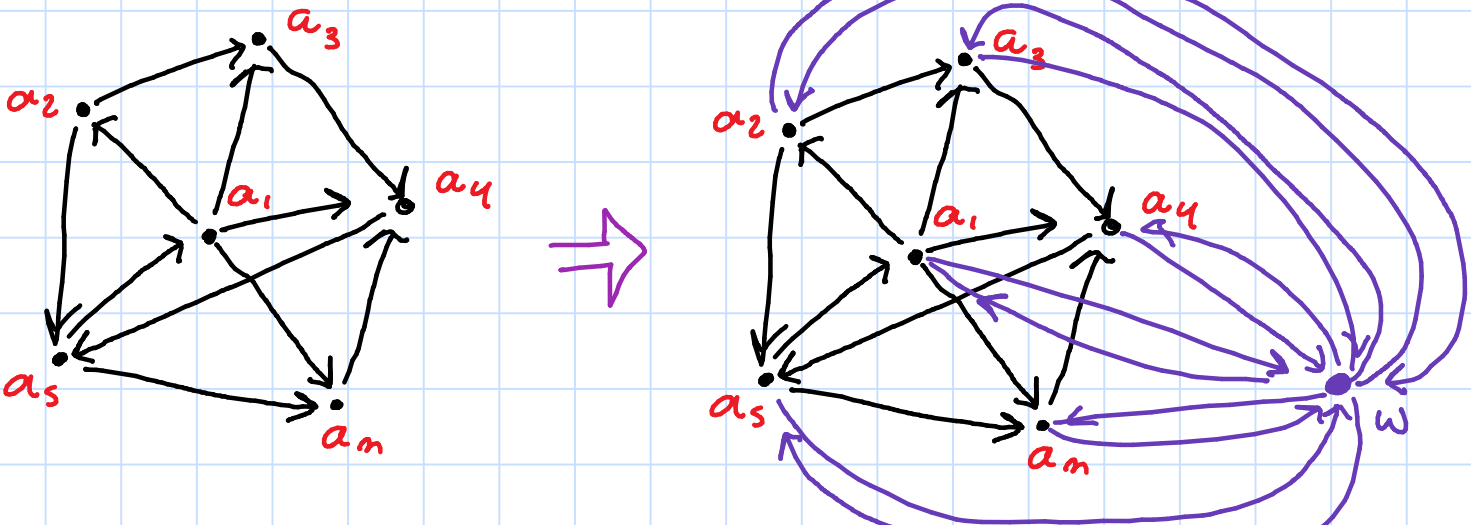
Retorna D

}

Next Slide

- Podemos criar D' em $O(|V(D)|)$, então f é polinomial
- Vemos mostrar que f é uma redução

```
f(D) {
    Seja D' o grafo obtido a partir de D
    inserindo um novo vértice w e adicionando
    as arestas uw e wu à D', para todo
    u ∈ V(D)
    Retorna D
}
```



Lema $CaH \leq_{poli} CH$

Demonstração

- Seja f o procedimento

$f(D)$ {

Seja D' o grafo obtido a partir de D inserindo um novo vértice w e adicionando as arestas uw e wu à D', para todo $u \in V(D)$

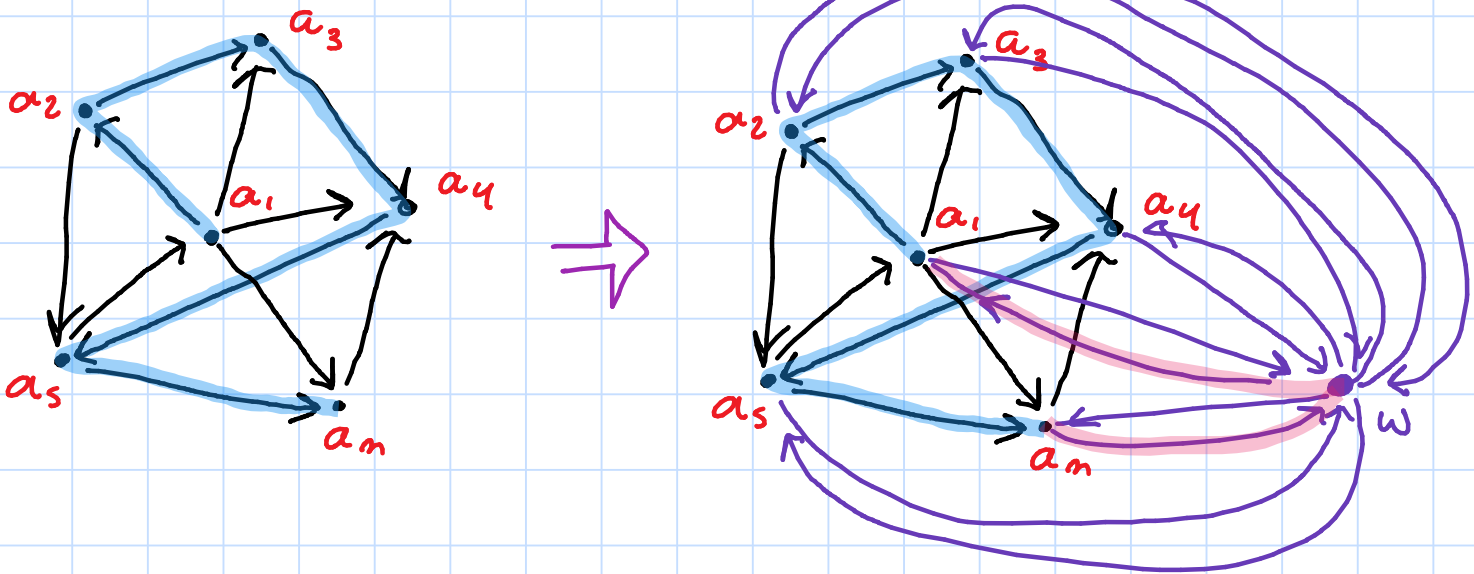
Retorna D

}

- Podemos criar D' em $O(|V(D)|)$, então f é polinomial
- Vamos mostrar que f é uma redução

($\Rightarrow$)

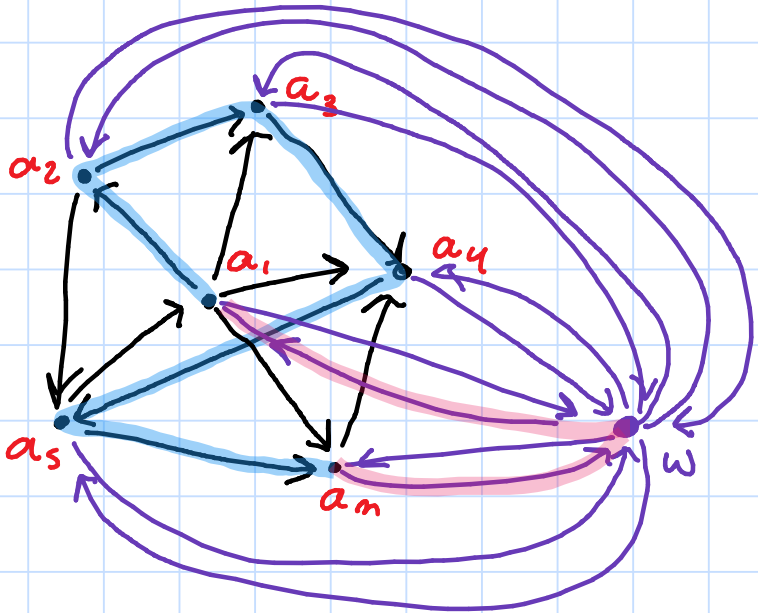
- Suponha que D seja uma instância sim para C2H
- Seja $D' = f(D)$ e seja $P = a_1 a_2 \dots a_n$ um caminho Hemiltoniano em D
- $C = w a_1 a_2 \dots a_n w$ é um ciclo Hamiltoniano em D'

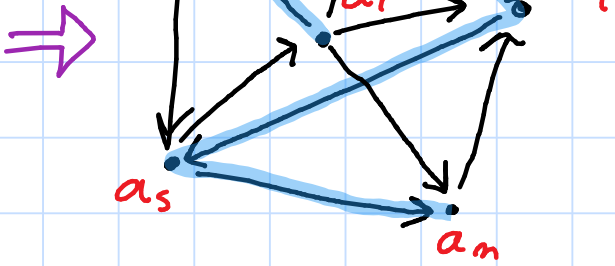


(⇐)

- Suponha que $D' = f(D)$ seja uma instância sim para CH e seja C um ciclo Hemiltoniano em D'
- $P = C - w$ é um caminho Hemiltoniano em D

□





ALG-CaH-CH (D)

D' ← ρ(D)

devolva ALG_{CH} (D')

} θ(V)

Problema da Barra (Bar)

Entrada: $\langle n, p, k \rangle$, onde n é um inteiro positivo indicando o tamanho da barra, sendo que cada tamanho $i \in \mathbb{N}$ de barra, com $1 \leq i \leq n$, tem um preço p_i de venda e k é um inteiro.

Saída: sim se é possível cortar uma barra de tamanho n e vender os pedaços obtendo um faturamento de valor ao menos k. não caso contrário.

$n = 4$

	1	2	3	4	5	6
p	3	2	10	11	12	14

$k = 12$

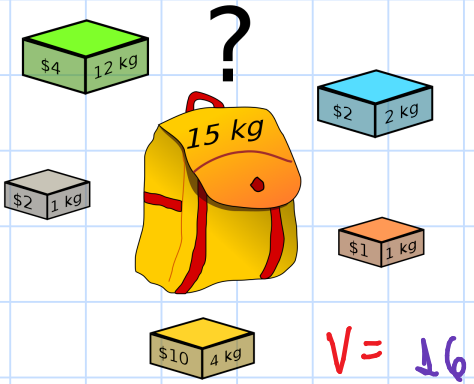
sim

{ 3 , 1 }

Problema Mochila (Mo)

Entrada: $\langle I, n, w, v, W, V \rangle$, onde $I = \{1, 2, \ldots, n\}$ é um conjunto de n itens, sendo que cada item $i \in I$ tem peso w_i e valor v_i. W é a capacidade da mochila e V é um valor.

Saída: sim se existe $S \subseteq I$ tal que $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i \geq V$. não caso contrário.

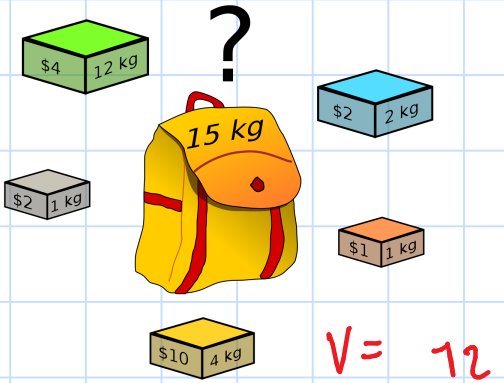


$V = 16$

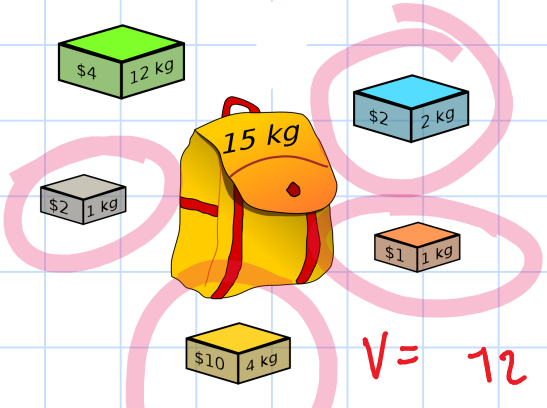
Problema Mochila (Mo)

Entrada: $\langle I, n, w, v, W, V \rangle$, onde $I = \{1, 2, \ldots, n\}$ é um conjunto de n itens, sendo que cada item $i \in I$ tem peso w_i e valor v_i. W é a capacidade da mochila e V é um valor.

Saída: sim se existe $S \subseteq I$ tal que $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i \geq V$. não caso contrário.



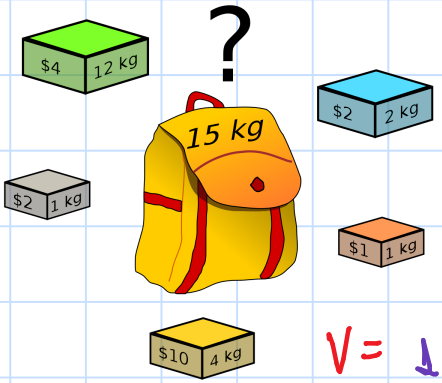
sim!



Problema Mochila (Mo)

Entrada: $\langle I, n, w, v, W, V \rangle$, onde $I = \{1, 2, \ldots, n\}$ é um conjunto de n itens, sendo que cada item $i \in I$ tem peso w_i e valor v_i. W é a capacidade da mochila e V é um valor.

Saída: sim se existe $S \subseteq I$ tal que $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i \geq V$. não caso contrário.

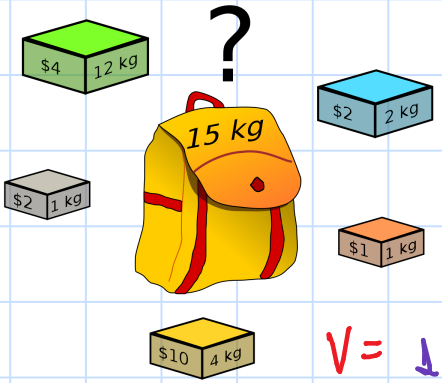


$V = 16$

Problema Mochila (Mo)

Entrada: $\langle I, n, w, v, W, V \rangle$, onde $I = \{1, 2, \ldots, n\}$ é um conjunto de n itens, sendo que cada item $i \in I$ tem peso w_i e valor v_i. W é a capacidade da mochila e V é um valor.

Saída: sim se existe $S \subseteq I$ tal que $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i \geq V$. não caso contrário.



$V = 16$

⇒ NÃO

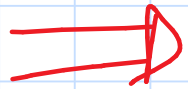
Teo. Bor $\preccurlyeq$ Mo

Demonstração

Seja f definido como

$f(n, p, k)$

- Crie $\lfloor n/i \rfloor$ itens de peso i e valor p_i, para $1 \leq i \leq n$
- $m = \sum_{i=1}^{n} \lfloor n/i \rfloor$ e $I = \{1, \ldots, m\}$
- $W = n$
- $V = k$
- Devolva (I, n, w, v, W, V)



next slide

$f(n, p, \kappa)$

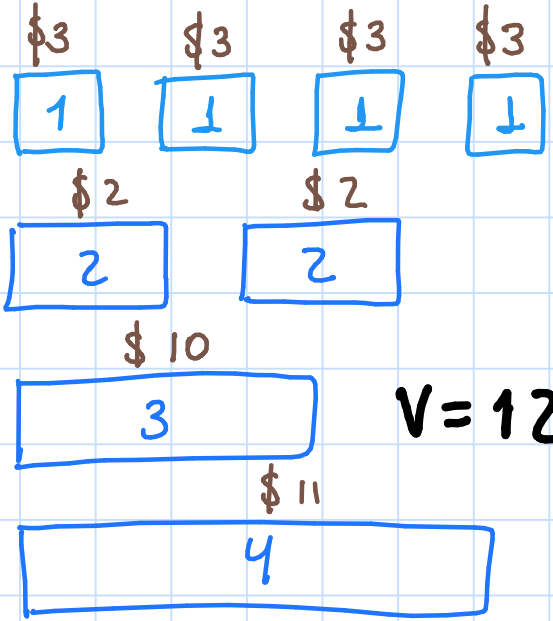
- Crie $\lfloor n/i \rfloor$ itens de peso i (w) e valor p_i (v), para $1 \leq i \leq m$
- $m = \sum_{i=1}^{n} \lfloor n/i \rfloor$ e $I = \{1, \ldots, m\}$
- $W = n$
- $V = \kappa$
- Devolva (I, n, w, v, W, V)

$n = 4$

	1	2	3	4	5	6
p	3	2	10	11	12	14

$\kappa = 12$

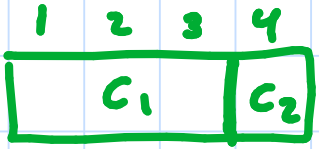
f



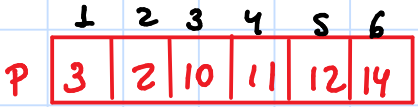
(⇒)

- Suponha que $\langle n, p, k \rangle$ é <u>sim</u> para Bar
- Então existem pedaços $(c_1, c_2, \ldots, c_x)$ tais que

$$\sum_{i=1}^{x} c_i = n \quad \text{e} \quad \sum_{i=1}^{x} p_{c_i} \geq k$$

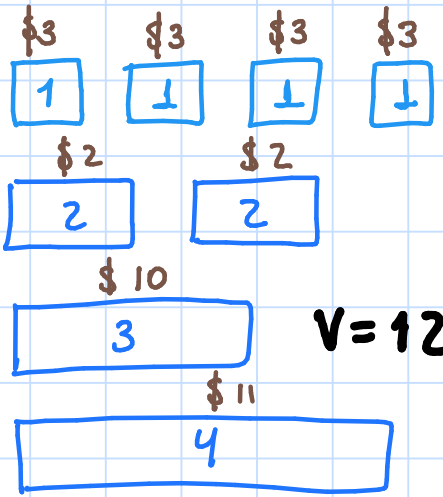


$n = 4$



$k = 12$

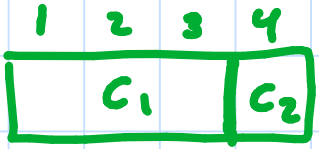
f



V=12

- Para cada pedaço c_i, coloque um item de peso c_i em um conjunto S. Note que

$$W = n = \sum_{i=1}^{x} c_i = \sum_{j \in S} w_j \quad \text{e} \quad V = k \leq \sum_{i=1}^{x} p_{c_i} = \sum_{j \in S} v_j$$

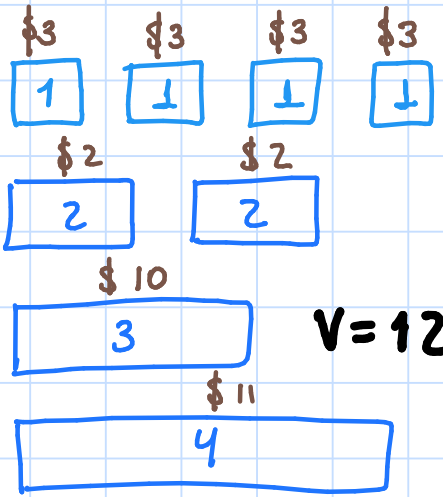


$n = 4$

	1	2	3	4	5	6
P	3	2	10	11	12	14

$k = 12$

f



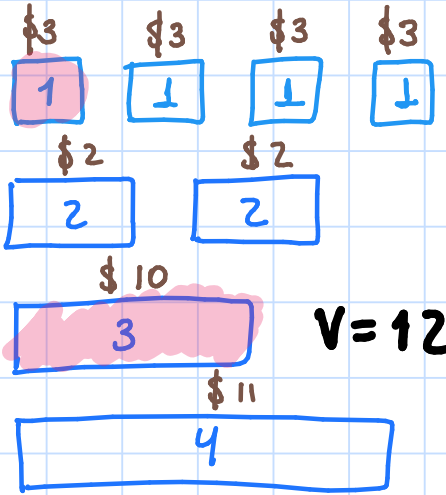
$V = 12$

(⇐)

- Agora suponha que $\langle I, m, w, v, W, V\rangle = f(m, p, K)$ é uma instância sim para mochila.

- Existe um conj. $S \subseteq I$ de itens tal que

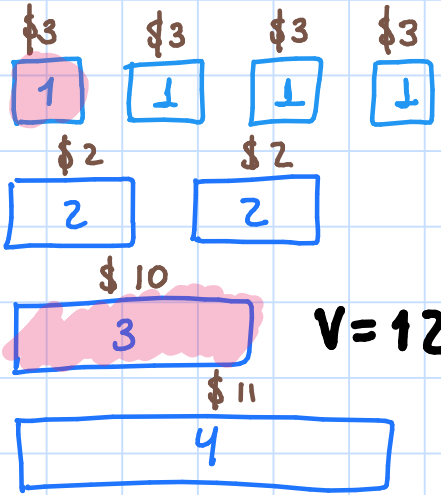
$$\sum_{j \in S} w_j \leq W \quad e \quad \sum_{j \in S} v_j \geq V$$

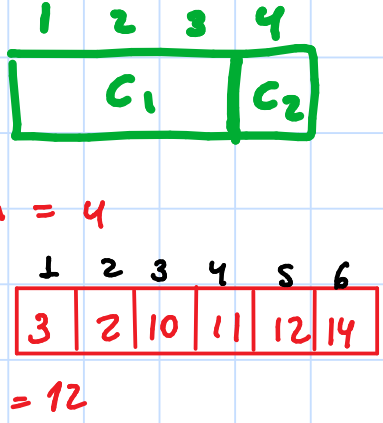


V=12

- Para cada item $j \in S$, corte e barra em um tamanho w_j. Sejam $(c_1, c_2, \ldots, c_{|S|})$ os pedaços contados. Note que

$$\sum_{i=1}^{|S|} c_i = \sum_{j \in S} w_j \leq W = n \quad \text{e} \quad \sum_{i=1}^{|S|} p_{c_i} = \sum_{j \in S} v_j \geq V = k$$





- Para cada item $j \in S$, corte e barra em um tamanho w_j. Sejam $(c_1, c_2, \ldots, c_{|S|})$ os pedaços cortados. Note que

$$\sum_{i=1}^{|S|} c_i = \sum_{j \in S} w_j \leq W = n \quad \text{e} \quad \sum_{i=1}^{|S|} p_{c_i} = \sum_{j \in S} v_j \geq V = k$$

- Seja $t = n - \sum_{i=1}^{|S|} c_i$ e note que $p_t \geq 0$

- Assim $(c_1, c_2, \ldots, c_{|S|}, t)$ é uma solução p/ o corte de barras, pois

$$\sum_{i=1}^{|S|} c_i + t = \sum_{i=1}^{|S|} c_i + n - \sum_{i=1}^{|S|} c_i = n \quad \text{e} \quad \sum_{i=1}^{|S|} p_{c_i} + p_t \geq k + 0 \geq k$$

□

The End

Cota Superior e Inferior de um Problema

- Seja P um problema e seja n um parâmetro

Cota Superior

Uma função $f(n)$ é chamada de cota superior para P se existe algum algoritmo que resolve P em $O(f(n))$.

Cota Inferior

Uma função $g(n)$ é chamada de cota inferior para P se todo algoritmo resolve P em $\Omega(f(n))$.

$\Theta(n \lg n)$ é uma cota superior e inferior para a ordenação

- Merge-Sort: $\Theta(n \lg n)$
- Nenhum algoritmo leve menos do que $n \lg n$ p/ ordenar

Cota Superior e Inferior de um Problema

- Seja P um problema e seja n um parâmetro

Cota Superior

Uma função $f(n)$ é chamada de cota superior para P se existe algum algoritmo que resolve P em $O(f(n))$.

Cota Inferior

Uma função $g(n)$ é chamada de cota inferior para P se todo algoritmo resolve P em $\Omega(f(n))$.

Complexidade de Algoritmo

Seja ALG um algoritmo para um problema P e n um parâmetro

- Se o algoritmo leva tempo **no máximo** $f(n)$ para **toda entrada** de tamanho n, então dizemos que ALG **resolve** P em $O(f(n))$.

- Se o algoritmo leva tempo **pelo menos** $g(n)$ para **alguma entrada** de tamanho n, então dizemos que ALG **resolve** P em $\Omega(g(n))$.

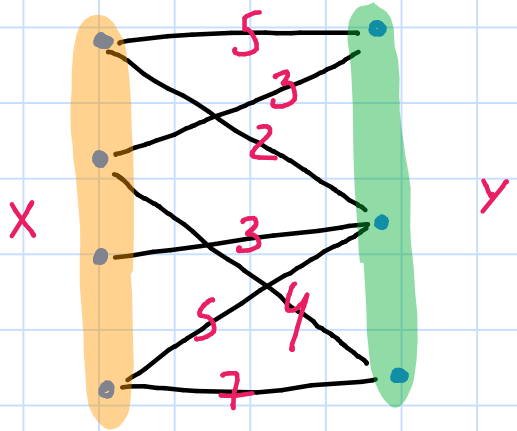
Problema de Alocação de Centros (AC)

Entrada: um grafo bipartido $G = ((X \cup Y), E)$ e uma função $w : E \to \mathbb{R}^+$

Saída: Associar cada vértice $u \in X$ a um vértice $\phi(u) \in Y$ tal que

$$\sum_{u \in X} w(u\phi(u))$$

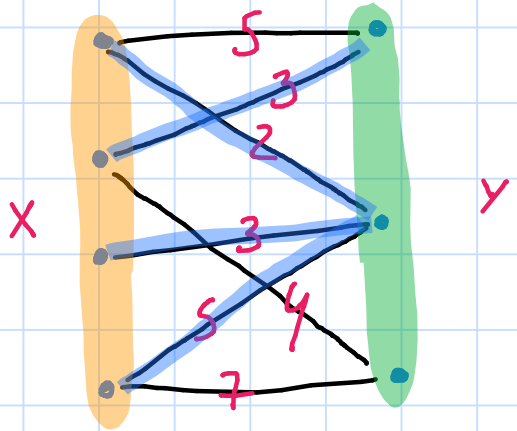
seja mínimo.



Problema de Alocação de Centros (AC)

Entrada: um grafo bipartido $G = ((X \cup Y), E)$ e uma função $w : E \to \mathbb{R}^+$

Saída: Associar cada vértice $u \in X$ a um vértice $\phi(u) \in Y$ tal que $w(u\phi(u))$ seja <u>mínimo</u>.

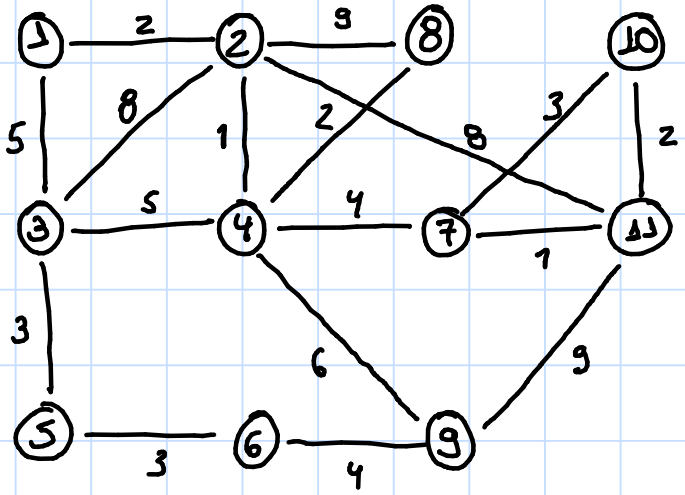


Problema do Caminho mínimo Raiz Única (Cmim)

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um vértice $s \in V(G)$

Saída: vetor d tal que $d[u] = dist^w(s,u)$ para todo $u \in V$

vetor pred da árvore de caminhos mínimo



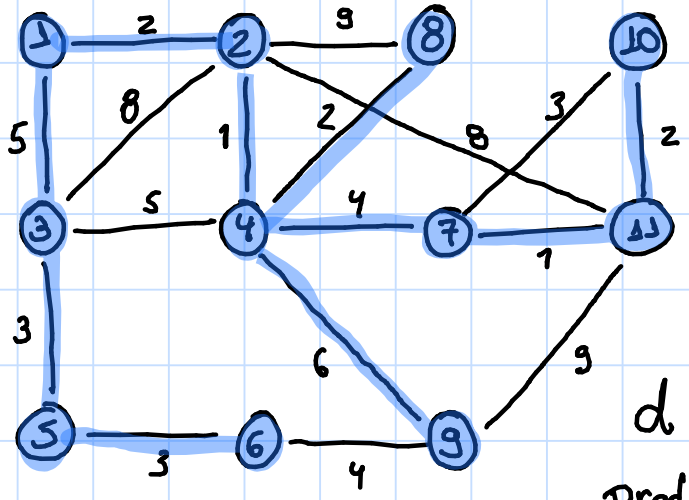
$s = 1$

Problema do Caminho mínimo Raiz Única (Cmim)

Entrada: um grafo G, uma função $w: E(G) \to \mathbb{R}^+$, um vértice $s \in V(G)$

Saída: vetor d tal que $d[u] = dist^w(s,u)$ para todo $u \in V$

vetor pred da árvore de caminhos mínimo



$s = 1$

	1	2	3	4	5	6	7	8	9	10	11
d	0	2	5	3	8	11	7	5	9	10	8
Pred	1	1	1	2	3	5	4	4	4	11	7

Seja σ_i definido como

σ_i (G, w)

$G' \leftarrow G$, $w' \leftarrow w$

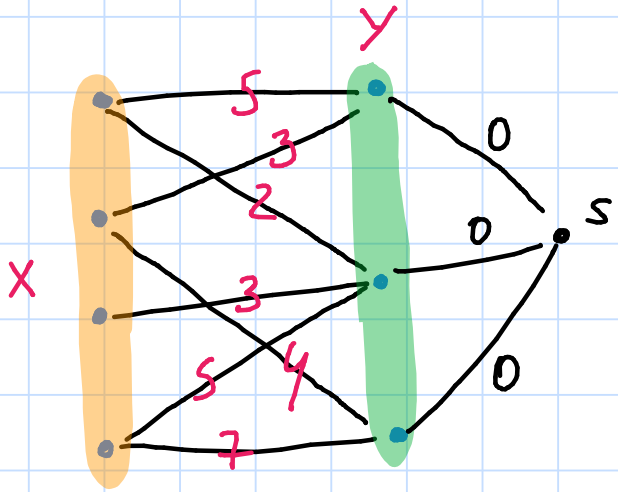
Adicione um novo vértice s a G'

Para cada $u \in Y$ faça

Adicione a aresta su a G'

$w'(sv) \leftarrow 0$

Retorne (G', w', s)



Tempo: $O(Y)$

Seja σ_i definido como

σ_i (G, w)

$G' \leftarrow G$, $w' \leftarrow w$

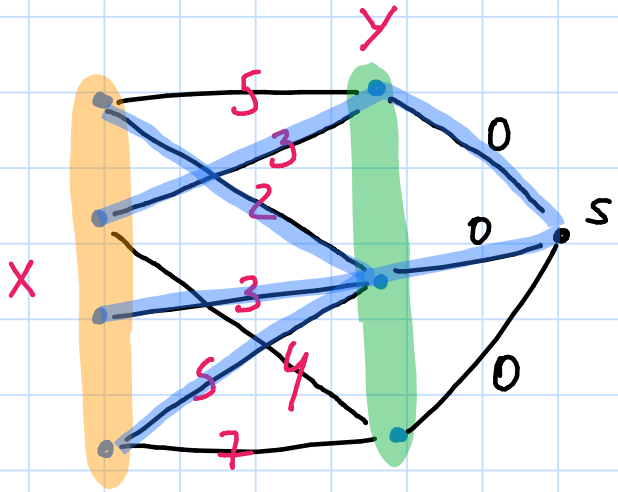
Adicione um novo vértice s a G'

Para cada $u \in Y$ faça

Adicione a aresta su a G'

$w'(sv) \leftarrow 0$

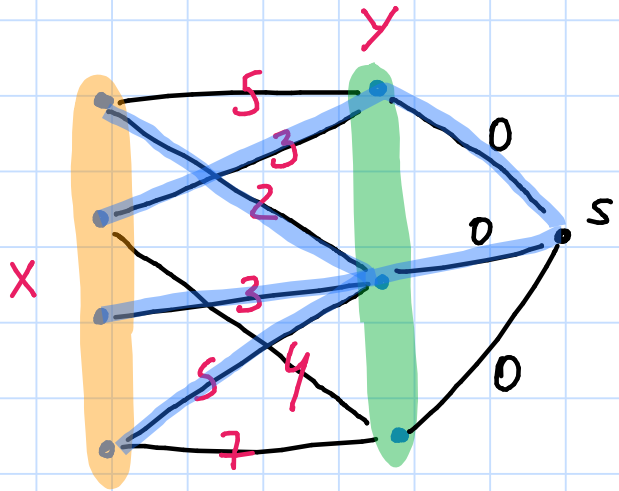
Retorne (G', w', s)



Tempo: $O(Y)$

```
Gs ( G, w, d, pred )
    para cada u ∈ X faça
        φ(u) ← pred[u]
    devolva φ
```

Tempo: $O(X)$



Redução-AC-Cmin (G, w)

$(G', w', s) \leftarrow \sigma_i(G, w)$

$(d, pred) \leftarrow ALG_{Cmin}(G', w', s)$

$\phi \leftarrow \sigma_s(G, w, d, pred)$

Devolva ϕ

Lema: Seja $I_{AC} = \langle G, w \rangle$ uma entrada para o AC e seja $I_{CMin} = \langle G', w', s \rangle = \sigma_1(I_{AC})$. Seja S_{AC} e S_{cmin} soluções ótimas de I_{AC} e I_{cmin}, respect. Então S_{AC} e S_{cmin} tem o mesmo valor.

Demontração

($\Rightarrow$) • Seja $S_{AC} = \phi$

- para cada $u \in X$, $d[u] = c(u\phi(u))$ e $pred[u] = \phi(u)$
  para cada $v \in Y$, $d[v] = 0$ e $pred[v] = 0$
- $d[s] = 0$ e $pred[s] = s$
- Claramente d, $pred$ é uma solução viável de I_{cmin}

C

Algoritmo Ótimo

Um algoritmo ALG é ótimo para um problema P se:

1. Alg resolve P em tempo $O(f(n))$ e
2. $f(n)$ é uma cota inferior para P

Heap-Sort e Merge-Sort são ótimos para a ordenação

- Eles tem complexidade $O(n \lg n)$
- O ordenação tem cota inferior $\Omega(n \lg n)$

Busca-Binária é ótimo para busca em vetor ordenado:

- tem complexidade $O(\lg n)$
- cota inferior é $\Omega(\lg n)$

Comparando Problemas

Como comparemos dois algoritmos para um único problema?

- Comparamos a complexidade de cada algoritmo
- Descobrimos se um algoritmo é mais rápido do que o outro

Como comparamos dois problemas A e B?

- queremos descobrir se A é mais fácil do que B
- podemos comparar as cotas de cada algoritmo

Exemplo: Achar o máximo em um vetor é mais fácil que ordenar

- Máximo tem cota superior $O(n)$
- Ordenação tem cota inferior $\Omega(n \lg n)$

($\Rightarrow$)

- Suponha que $\langle n, p, K \rangle$ é <u>sim</u> para Bar
- Então existem pedaços $(c_1, c_2, \ldots, c_x)$ tais que

$$\sum_{i=1}^{x} c_i = n \quad \text{e} \quad \sum_{i=1}^{x} p_i \geq K$$

- Para cada pedaço c_i, coloque um item de peso c_i em um conjunto S. Note que

$$W = n = \sum_{i=1}^{x} c_i = \sum_{j \in S} w_j \quad \text{e} \quad V = k \leq \sum_{i=1}^{x} p_{c_i} = \sum_{j \in S} v_j$$

($\Leftarrow$)

- Agora suponha que $\langle I, m, w, v, W, V\rangle = f(n, p, K)$ é uma instância <u>sim</u> para mochila.

- Existe um conj. $S \subseteq I$ de itens tal que

$$\sum_{j \in S} w_j \leq W \quad \text{e} \quad \sum_{j \in S} v_j \geq V$$

- Para cada item de peso $j \in S$, corte a barra em um tamanho j. Seja $(c_1, c_2, \ldots, c_{|S|})$ o pedaços cortados. Note que

$$\sum_{i=1}^{|S|} c_i = \sum_{j \in S} w_j \leq W = n \quad \text{e} \quad \sum_{i=1}^{|S|} p_i = \sum_{j \in S} v_j \geq V = K$$

- Então $\langle n, p, K\rangle$ é uma instância <u>sim</u>. □