Algoritmos Parametrizados

Kernelização

Lehilton Pedrosa

Segundo Semestre de 2016

Instituto de Computação – Unicamp

Roteiro

1. Preprocessamento
2. Cobertura por vértices
3. Conjunto de retroalimentação em torneios
4. Problema da Cobertura de Arestas por Cliques
5. Decomposição em Coroa
6. Lema do Girassol

Preprocessamento

Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

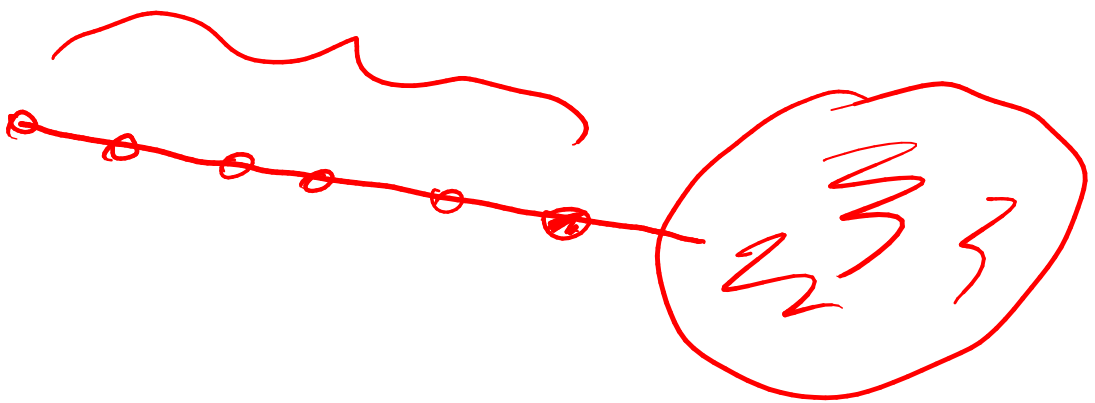
- Nem toda instância é "difícil"

Instância fácil:

Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

- Nem toda instância é "difícil"
- Parte de uma instância pode ser "fácil"



Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

- Nem toda instância é "difícil"
- Parte de uma instância pode ser "fácil"

Ideia: se concentrar na dificuldade

Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

- Nem toda instância é "difícil"
- Parte de uma instância pode ser "fácil"

Ideia: se concentrar na dificuldade

Pré-processamento

- resolver instâncias fáceis
- revolver a parte fácil

Separando a dificuldade

Considere um problema $\mathcal{NP}$-difícil

- Nem toda instância é "difícil"
- Parte de uma instância pode ser "fácil"

Ideia: se concentrar na dificuldade

Pré-processamento

- resolver instâncias fáceis
- revolver a parte fácil
- diminuir o tamanho da instância

Pré-processamento

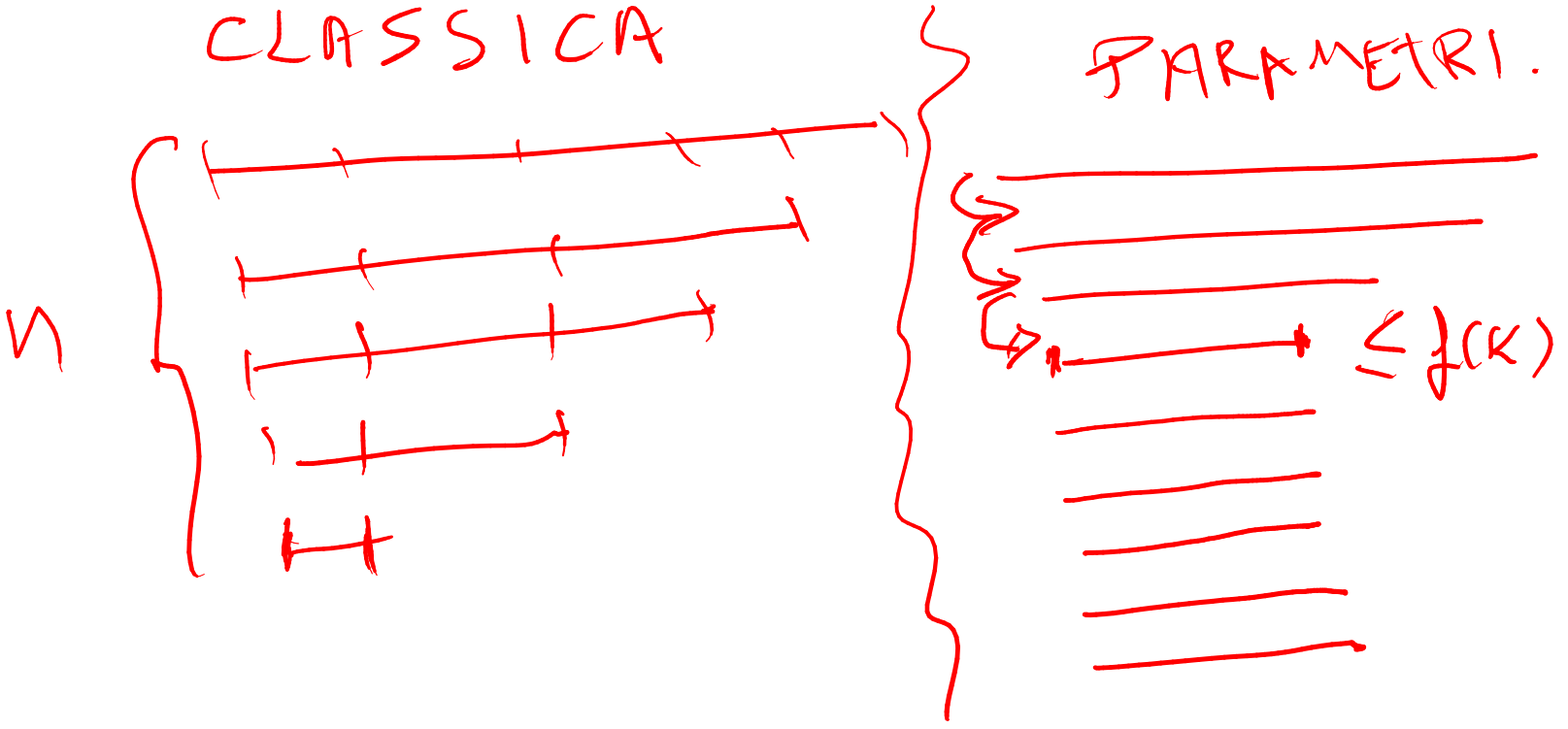
Visão de clássica

- rápido: polinomial

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância



Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil $\Rightarrow \mathcal{P} = \mathcal{NP}$!

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil $\Rightarrow \mathcal{P} = \mathcal{NP}$!

Visão **parametrizada**

- rápido: polinomial $\Rightarrow$ no nº bits $|X| + k$

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil $\Rightarrow \mathcal{P} = \mathcal{NP}$!

Visão **parametrizada**

- rápido: polinomial
- efetivo: diminui o tamanho da instância

nem sempre

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil $\Rightarrow \mathcal{P} = \mathcal{NP}$!

Visão **parametrizada**

- rápido: polinomial
- efetivo: diminui o tamanho da instância (se não for muito pequena)

Pré-processamento

Visão de clássica

- rápido: polinomial
- efetivo: diminui o tamanho da instância

Não é muito útil:

- se P tem pre-processamento $\Rightarrow$ então P é polinomial
- se P é $\mathcal{NP}$-difícil $\Rightarrow \mathcal{P} = \mathcal{NP}$!

Visão **parametrizada**

- rápido: polinomial
- efetivo: diminui o tamanho da instância (se não for muito pequena)

Recuperamos o *lost continent* do pré-processamento!

Formalizando: redução

- Seja $Q \subseteq \Sigma^* \times \mathbb{N}$ um problema parametrizado

Formalizando: redução

- Seja $Q \subseteq \Sigma^* \times \mathbb{N}$ um problema parametrizado

Definição (Regra de redução de dados)

Uma regra de redução é uma transformação $\mathcal{A} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ tal que, dada uma instância $(x, k) \in \Sigma^* \times \mathbb{N}$,

Formalizando: redução

- Seja $Q \subseteq \Sigma^* \times \mathbb{N}$ um problema parametrizado

Definição (Regra de redução de dados)

Uma regra de redução é uma transformação $\mathcal{A} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ tal que, dada uma instância $(x, k) \in \Sigma^* \times \mathbb{N}$,

1. $\mathcal{A}$ executa em tempo polinomial em $|x|$ e k;

Formalizando: redução

- Seja $Q \subseteq \Sigma^* \times \mathbb{N}$ um problema parametrizado

Definição (Regra de redução de dados)

Uma regra de redução é uma transformação $\mathcal{A} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ tal que, dada uma instância $(x, k) \in \Sigma^* \times \mathbb{N}$,

1. $\mathcal{A}$ executa em tempo polinomial em $|x|$ e k;
2. $(x, k) \in Q$ sss $\mathcal{A}(x, k) \in Q$.

Formalizando: redução

- Seja $Q \subseteq \Sigma^* \times \mathbb{N}$ um problema parametrizado

Definição (Regra de redução de dados)

Uma regra de redução é uma transformação $\mathcal{A} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ tal que, dada uma instância $(x, k) \in \Sigma^* \times \mathbb{N}$,

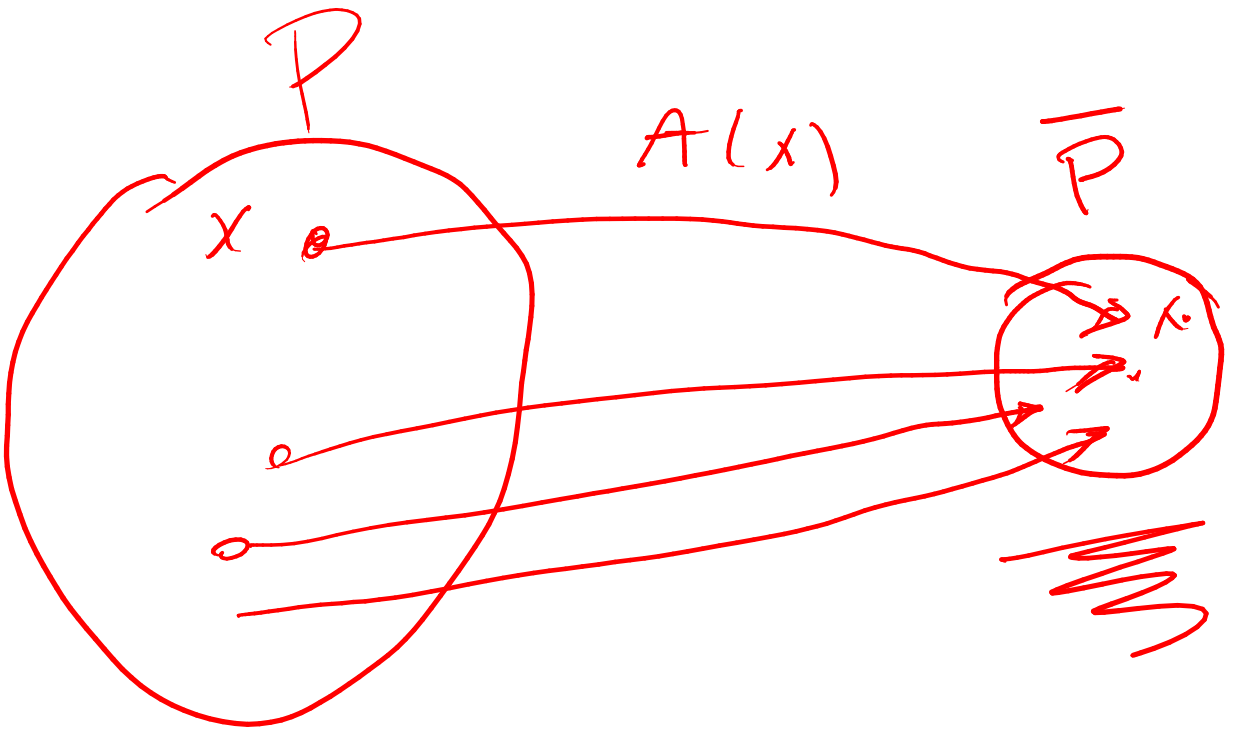
1. $\mathcal{A}$ executa em tempo polinomial em $|x|$ e k;
2. $(x, k) \in Q$ sss $\mathcal{A}(x, k) \in Q$.

Uma transformação que satisfaz a segunda condição é dita segura.

Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\ x \in \Sigma^* \,\}.$$

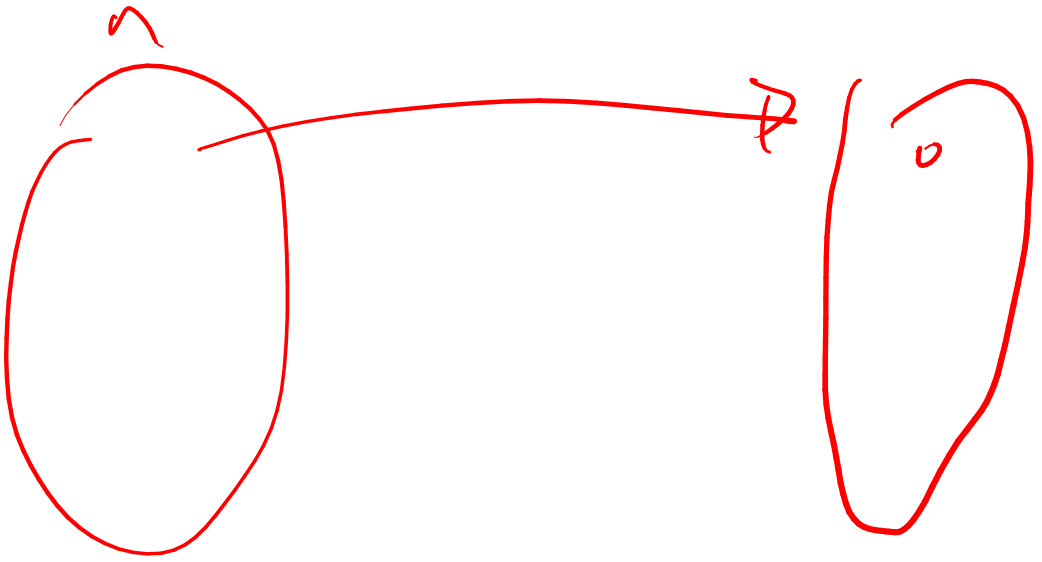


Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\, x \in \Sigma^* \,\}.$$

Note que $\text{size}_{\mathcal{A}}(k) = \infty$ quando $|x'|$ não depende de k.



Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\, x \in \Sigma^* \,\}.$$

Note que $\text{size}_{\mathcal{A}}(k) = \infty$ quando $|x'|$ não depende de k.

Definição (Kernelização)

Um algoritmo de kernelização ou núcleo para um problema parametrizado Q é uma transformação $\mathcal{A}$ que mapeia $(x, k) \in \Sigma^* \times \mathbb{N}$ em $(x', k') \in \Sigma^* \times \mathbb{N}$ tal que

Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\, x \in \Sigma^* \,\}.$$

Note que $\text{size}_{\mathcal{A}}(k) = \infty$ quando $|x'|$ não depende de k.

Definição (Kernelização)

Um algoritmo de kernelização ou núcleo para um problema parametrizado Q é uma transformação $\mathcal{A}$ que mapeia $(x, k) \in \Sigma^* \times \mathbb{N}$ em $(x', k') \in \Sigma^* \times \mathbb{N}$ tal que

1. $\mathcal{A}$ executa em tempo polinomial;

Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\, x \in \Sigma^* \,\}.$$

Note que $\text{size}_{\mathcal{A}}(k) = \infty$ quando $|x'|$ não depende de k.

Definição (Kernelização)

Um algoritmo de kernelização ou núcleo para um problema parametrizado Q é uma transformação $\mathcal{A}$ que mapeia $(x, k) \in \Sigma^* \times \mathbb{N}$ em $(x', k') \in \Sigma^* \times \mathbb{N}$ tal que

1. $\mathcal{A}$ executa em tempo polinomial;
2. $(x, k) \in Q$ sss $\mathcal{A}(x, k) \in Q$;

Formalizando: Kernelização (núcleo)

Dada um algoritmo de pré-processamento A, definimos o tamanho de saída

$$\text{size}_{\mathcal{A}}(k) = \sup\{\, |x'| + k' \;:\; (x', k') = \mathcal{A}(x, k),\, x \in \Sigma^* \,\}.$$

Note que $\text{size}_{\mathcal{A}}(k) = \infty$ quando $|x'|$ não depende de k.

Definição (Kernelização)

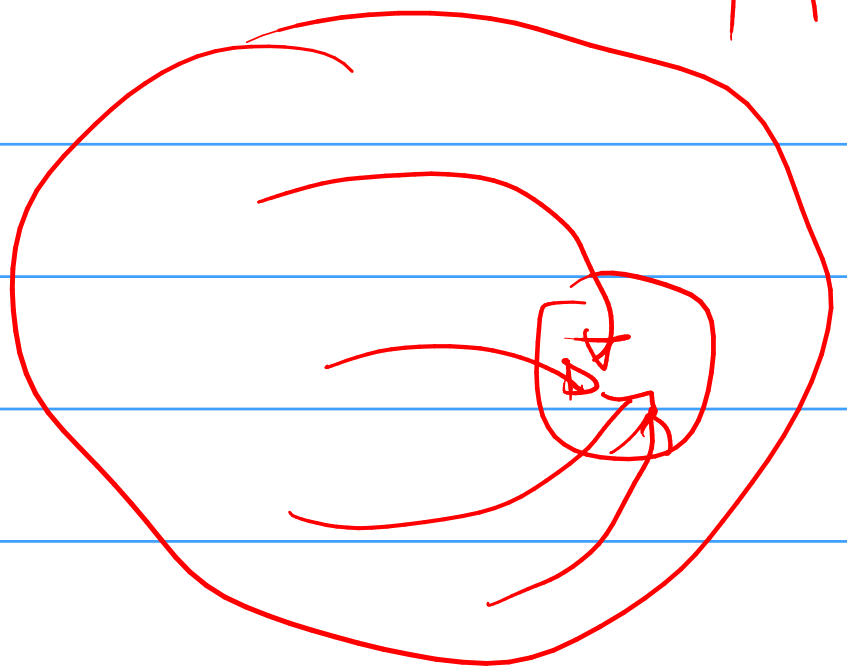
Um algoritmo de kernelização ou núcleo para um problema parametrizado Q é uma transformação $\mathcal{A}$ que mapeia $(x, k) \in \Sigma^* \times \mathbb{N}$ em $(x', k') \in \Sigma^* \times \mathbb{N}$ tal que

1. $\mathcal{A}$ executa em tempo polinomial;
2. $(x, k) \in Q$ sss $\mathcal{A}(x, k) \in Q$;
3. $\text{size}_{\mathcal{A}}(k) \leq g(k)$ para alguma função computável $g(k)$.

$|P| = \infty$

A

$|\bar{P}| \leq g'(K)$



$|x'| \leq g(K)$

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial
- Permitimos que um algoritmo de kernelização devolva sim ou não:

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial
- Permitimos que um algoritmo de kernelização devolva sim ou não:
  - isso quando acontece quando o pré-processamento resolve a instância

Observações

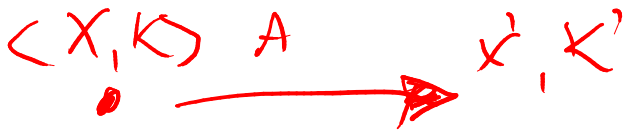
- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial
- Permitimos que um algoritmo de kernelização devolva sim ou não:
  - isso quando acontece quando o pré-processamento resolve a instância
  - formalmente podemos substituir por uma instância trivial
- Enquanto o tamanho do núcleo é o número de “bits”, é comum medir o tamanho por um parâmetro natural da instância reduzida

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial
- Permitimos que um algoritmo de kernelização devolva sim ou não:
  - isso quando acontece quando o pré-processamento resolve a instância
  - formalmente podemos substituir por uma instância trivial
- Enquanto o tamanho do núcleo é o número de "bits", é comum medir o tamanho por um parâmetro natural da instância reduzida
  - ex: $\mathcal{O}(k^3)$ vértices, ou $\mathcal{O}(k^5)$ arestas

Observações

- Queremos um núcleo tal que $g(k)$ seja menor possível
  - Se $g(k)$ é um polinômio, dizemos que o problema admite um núcleo polinomial
- Permitimos que um algoritmo de kernelização devolva sim ou não:
  - isso quando acontece quando o pré-processamento resolve a instância
  - formalmente podemos substituir por uma instância trivial
- Enquanto o tamanho do núcleo é o número de "bits", é comum medir o tamanho por um parâmetro natural da instância reduzida
  - ex: $\mathcal{O}(k^3)$ vértices, ou $\mathcal{O}(k^5)$ arestas

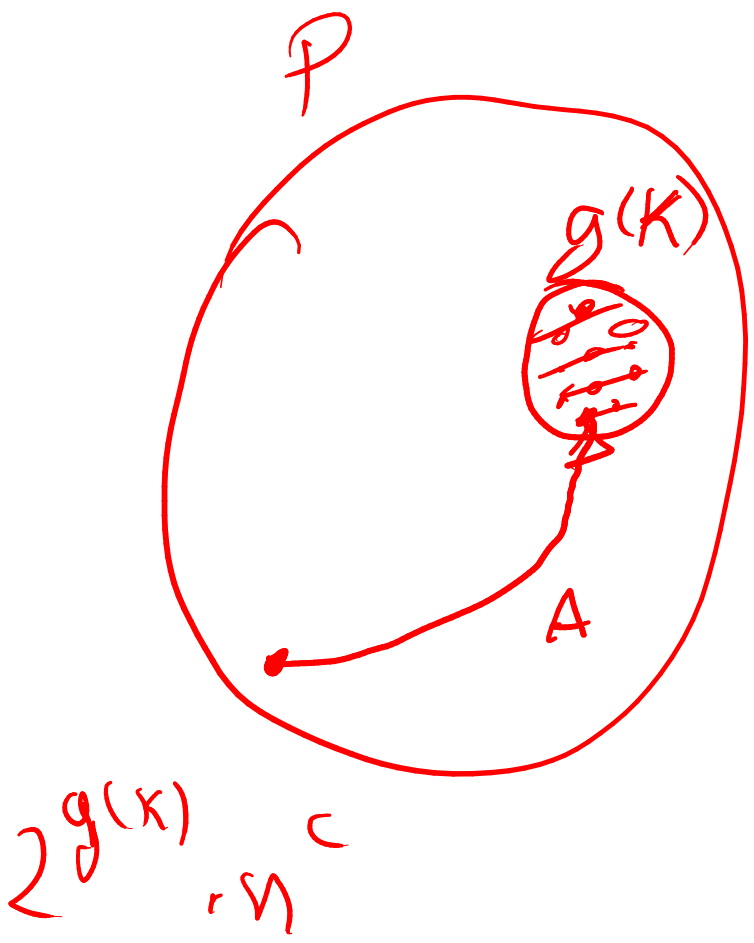


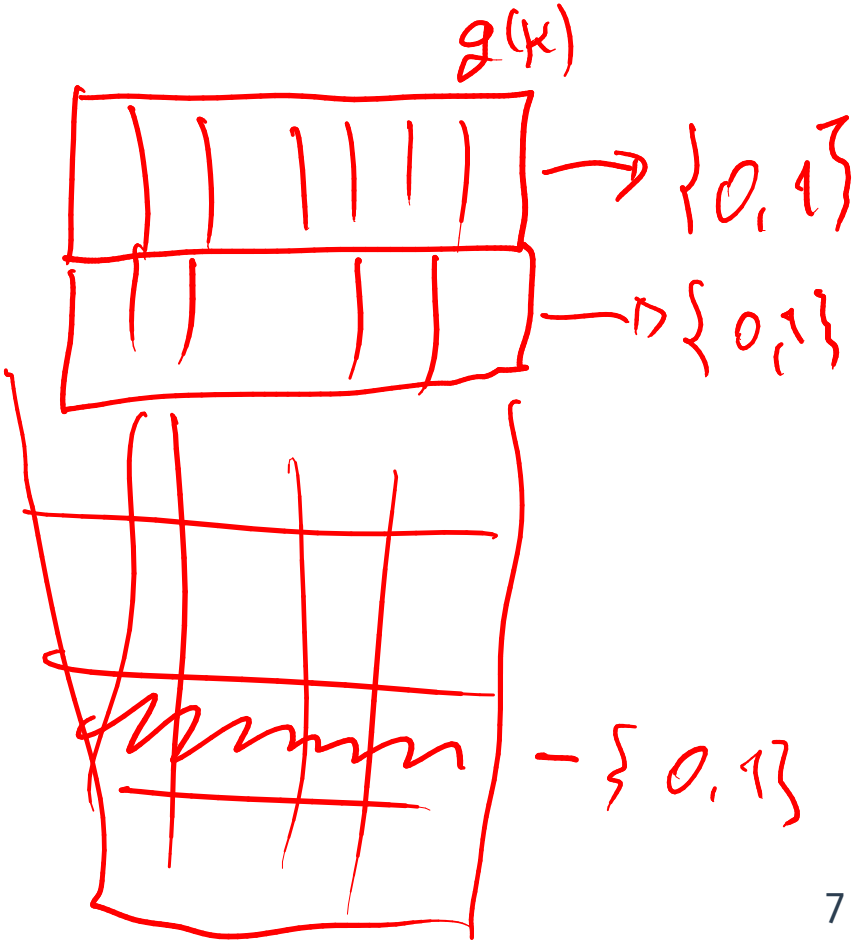
- Embora o valor do parâmetro k' não está necessariamente relacionado a k, normalmente temos $k \leq k'$

FPT e núcleos

Lema

*Se um problema parametrizado Q é **FPT**, então ele admite um algoritmo de kernelização.*





• Como $Q \in FPT \Rightarrow \exists A$ q decide Q em tempo $f(k)n^c$

Redução (x, k): $n^{c'}$

1) Executa $A(x,k)$ por no máx $|x|^{c+1}$

2) Se terminou:
devolve $A(x,k)$

3) Se não terminou
devolve (x,k)

*) Tempo poly: OK

*) Seguro: OK

*) $|x'| \leq g(k)$

• Se terminou

↳ OK

• Se não terminou

$$|x|^{c+1} \leq g(k)\, n^c \Rightarrow |n \leq g(k)$$

Cobertura por vértices

Cobertura por vértices

Problema da Cobertura por Vértices (VC)

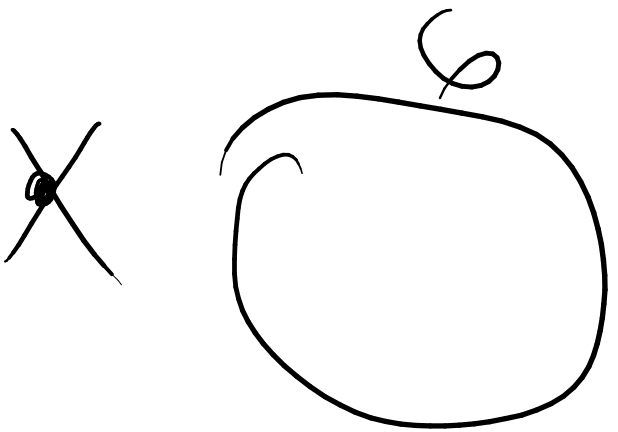
Dado um grafo G e inteiro k, existe uma cobertura de vértices de tamanho k?

Cobertura por vértices

Problema da Cobertura por Vértices (VC)

Dado um grafo G e inteiro k, existe uma cobertura de vértices de tamanho k?

Redução VC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.



Cobertura por vértices

Problema da Cobertura por Vértices (VC)

Dado um grafo G e inteiro k, existe uma cobertura de vértices de tamanho k?

Redução VC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

Reduzimos só a instância.

Cobertura por vértices

Problema da Cobertura por Vértices (VC)

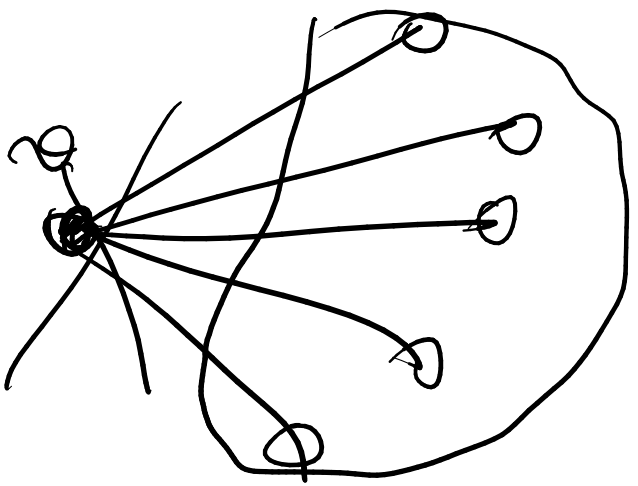
Dado um grafo G e inteiro k, existe uma cobertura de vértices de tamanho k?

Redução VC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

Reduzimos só a instância.

Redução VC.2: Se G contém um vértice v com $d(v) \geqslant k$, devolva $(G - v, k - 1)$.

$d(v) > k$



Cobertura por vértices

Problema da Cobertura por Vértices (VC)

Dado um grafo G e inteiro k, existe uma cobertura de vértices de tamanho k?

Redução VC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

Reduzimos só a instância.

Redução VC.2: Se G contém um vértice v com $d(v) \geq k$, devolva $(G - v, k - 1)$.

Agora também o parâmetro.

$\langle X, k \rangle \in \Sigma^* \times \mathbb{N}$

Outra redução

Usamos as duas primeiras reduções exaustivamente até não serem mais aplicadas.

Outra redução

Usamos as duas primeiras reduções exaustivamente até não serem mais aplicadas.

Lema

Se VC.1 e VC.1 não se aplicam e (G, k) *é instância-sim, então* $|V(G)| \leq k^2 + k$ *e* $|E(G)| \leq k^2$.

Outra redução

Usamos as duas primeiras reduções exaustivamente até não serem mais aplicadas.

Lema

Se VC.1 e VC.1 não se aplicam e (G, k) é instância-sim, então $|V(G)| \leq k^2 + k$ e $|E(G)| \leq k^2$.

O lema permite adicionar mais uma redução:

Outra redução

Usamos as duas primeiras reduções exaustivamente até não serem mais aplicadas.

Lema

Se VC.1 e VC.1 não se aplicam e (G, k) é instância-sim, então $|V(G)| \leq k^2 + k$ e $|E(G)| \leq k^2$.

O lema permite adicionar mais uma redução:

Redução VC.3: Se VC.1 e VC.2 não se aplicam e, além disso: $k < 0$, ou $|V(G)| > k^2 + k$, ou $|E(G)| > k^2$, devolva sim. não

Outra redução

Usamos as duas primeiras reduções exaustivamente até não serem mais aplicadas.

Lema

Se VC.1 e VC.1 não se aplicam e (G, k) é instância-sim, então $|V(G)| \leq k^2 + k$ e $|E(G)| \leq k^2$.

O lema permite adicionar mais uma redução:

Redução VC.3: Se VC.1 e VC.2 não se aplicam e, além disso: $k < 0$, ou $|V(G)| > k^2 + k$, ou $|E(G)| > k^2$, devolva sim.

Teorema

Cobertura por vértices admite um núcleo com $\mathcal{O}(k^2)$ vértices e $\mathcal{O}(k^2)$ arestas.

Conjunto de retroalimentação em torneios

Conjunto de retroalimentação em torneios

Um torneio T é um grafo direcionado tal que para cada par de vértices $u, v \in V(T)$:

Conjunto de retroalimentação em torneios

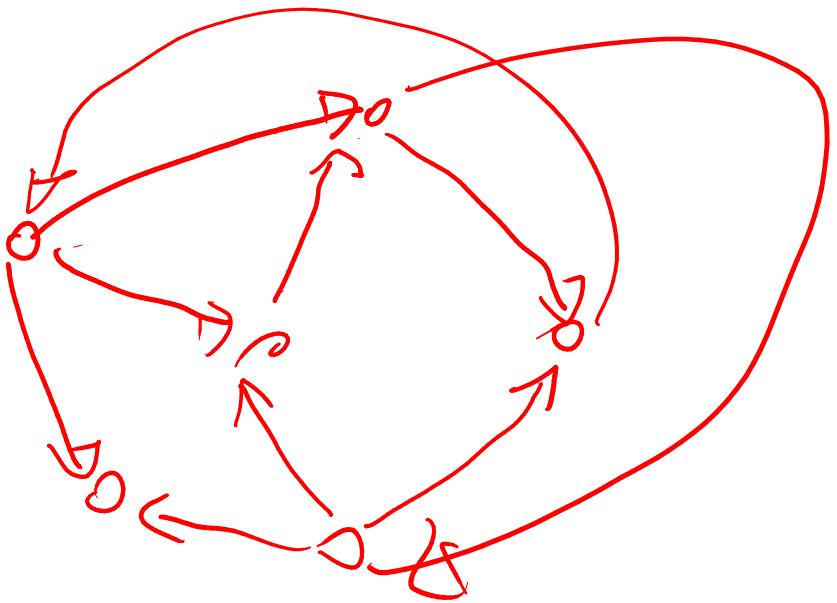
Um torneio T é um grafo direcionado tal que para cada par de vértices $u, v \in V(T)$:

- ou arco $(u, v) \in E(T)$

Conjunto de retroalimentação em torneios

Um torneio T é um grafo direcionado tal que para cada par de vértices $u, v \in V(T)$:

- ou arco $(u, v) \in E(T)$,
- ou arco $(v, u) \in E(T)$.

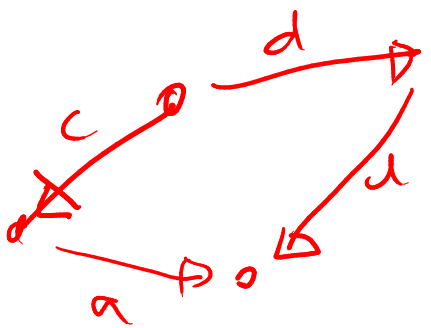


Conjunto de retroalimentação em torneios

Um torneio T é um grafo direcionado tal que para cada par de vértices $u, v \in V(T)$:

- ou arco $(u, v) \in E(T)$,
- ou arco $(v, u) \in E(T)$.

Um conjunto de arcos de retroalimentação de um digrafo G é um conjunto de arcos A tal que $G - A$ é acíclico.



Conjunto de retroalimentação em torneios

Um torneio T é um grafo direcionado tal que para cada par de vértices $u, v \in V(T)$:

- ou arco $(u, v) \in E(T)$,
- ou arco $(v, u) \in E(T)$.

Um conjunto de arcos de retroalimentação de um digrafo G é um conjunto de arcos A tal que $G - A$ é acíclico.

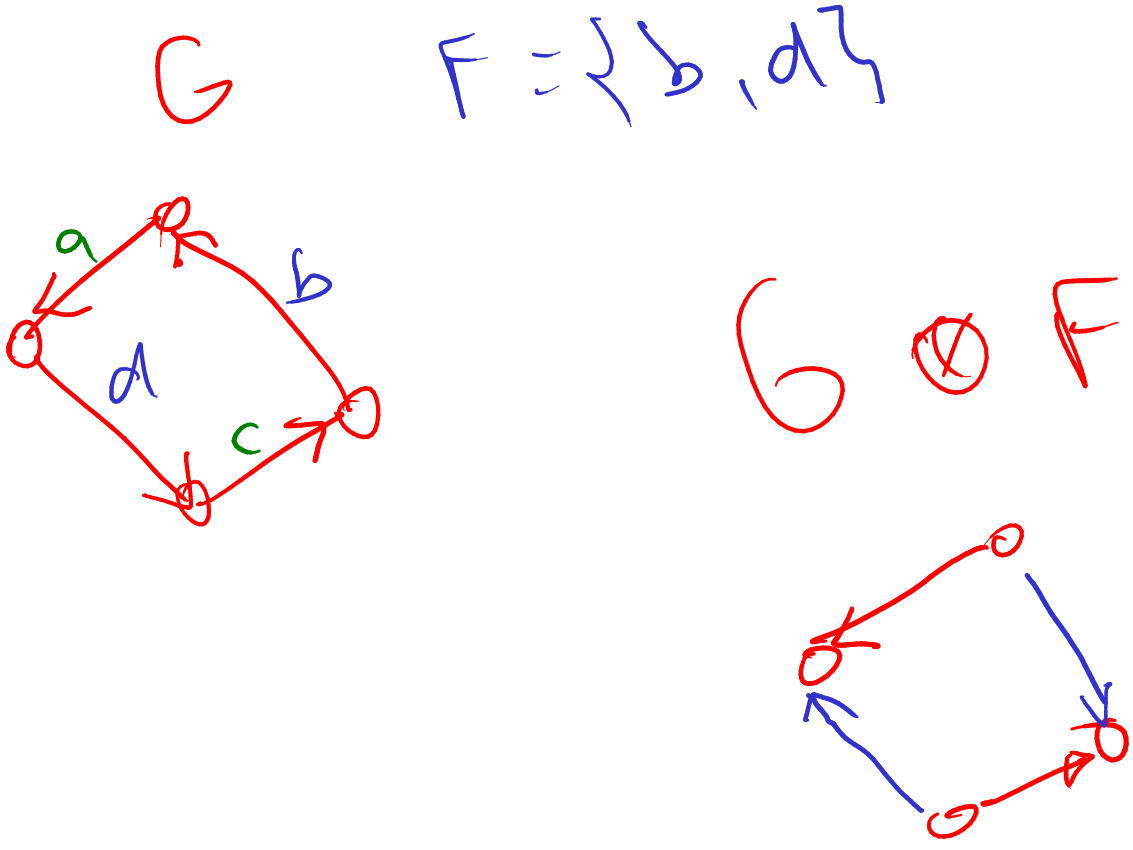
Problema dos arcos de retroalimentação em torneios (FAST)

Dado um torneio T, existe um conjunto de arcos de retroalimentação de tamanho no máximo k?

Revertendo arestas

Definição

Dado um digrafo G e um conjunto de arcos $F \subseteq E(G)$, definimos $G \otimes F$ o digrafo com vértices $V(G)$ e arestas $(E(G) \cup \text{rev}(F)) \setminus F$, onde $(u, v) \in \text{rev}(F)$ sss $(v, u) \in F$.



Revertendo arestas

Definição

Dado um digrafo G e um conjunto de arcos $F \subseteq E(G)$, definimos $G \otimes F$ o digrafo com vértices $V(G)$ e arestas $(E(G) \cup \mathsf{rev}(F)) \setminus F$, onde $(u, v) \in \mathsf{rev}(F)$ sss $(v, u) \in F$.

Relacionando conjunto de retroalimentação e reversão de arcos.

Revertendo arestas

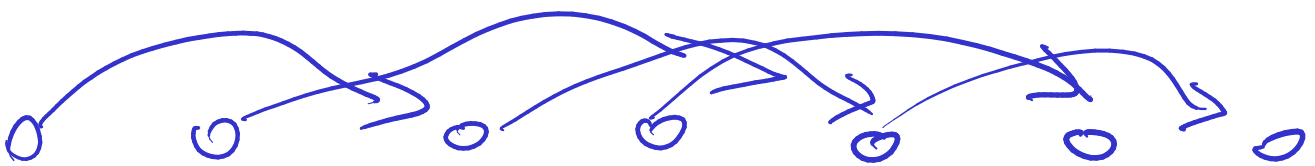
Definição

Dado um digrafo G e um conjunto de arcos $F \subseteq E(G)$, definimos $G \otimes F$ o digrafo com vértices $V(G)$ e arestas $(E(G) \cup \text{rev}(F)) \setminus F$, onde $(u, v) \in \text{rev}(F)$ sss $(v, u) \in F$.

Relacionando conjunto de retroalimentação e reversão de arcos.
Primeiro:

Lema

Um digrafo G é acíclico sss existe uma ordenação dos vértices tal que, se $(u, v) \in E(G)$, então $u < v$.



Revertendo arestas

Definição

Dado um digrafo G e um conjunto de arcos $F \subseteq E(G)$, definimos $G \otimes F$ o digrafo com vértices $V(G)$ e arestas $(E(G) \cup \text{rev}(F)) \setminus F$, onde $(u, v) \in \text{rev}(F)$ sss $(v, u) \in F$.

Relacionando conjunto de retroalimentação e reversão de arcos. Primeiro:

Lema

Um digrafo G é acíclico sss existe uma ordenação dos vértices tal que, se $(u, v) \in E(G)$, então $u < v$.

Lema

Se $G \otimes F$ é acíclico, então F é um conjunto de arcos de retroalimentação.

Problema equivalente

Lema

Seja G um digrafo e $F \subseteq E(G)$. São equivalentes:

Problema equivalente

Lema

Seja G um digrafo e $F \subseteq E(G)$. São equivalentes:

- *F é um conjunto de retroalimentação minimal de G;*

Problema equivalente

Lema

Seja G um digrafo e $F \subseteq E(G)$. São equivalentes:

- *F é um conjunto de retroalimentação minimal de G;*
- *F é conjunto de arcos minimal tal que $G \otimes F$ é acíclico.*

Problema equivalente

Lema

Seja G um digrafo e $F \subseteq E(G)$. São equivalentes:

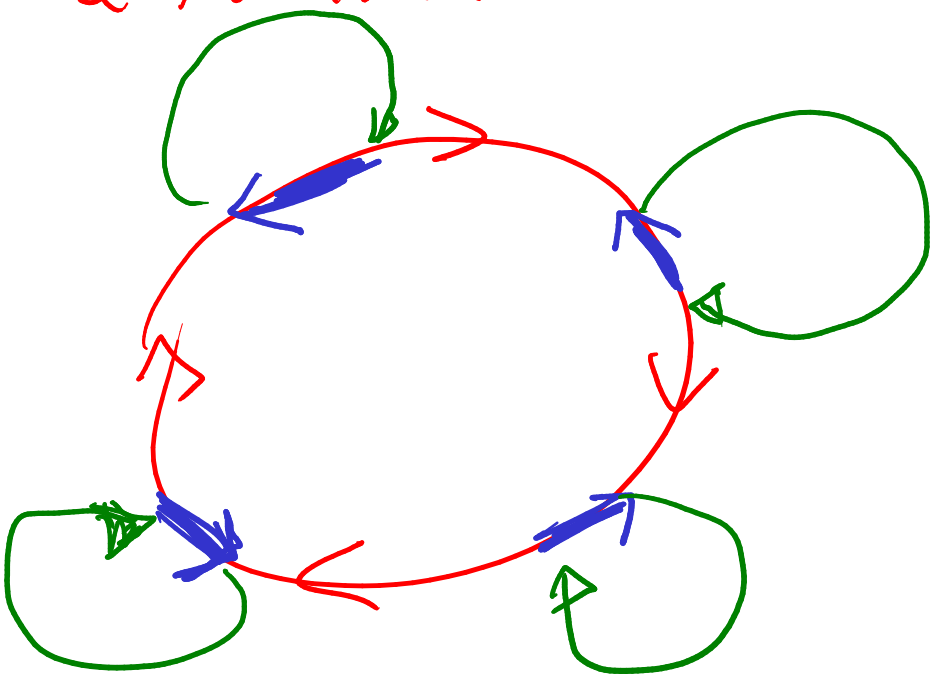
- *F é um conjunto de retroalimentação minimal de G;*
- *F é conjunto de arcos minimal tal que $G \otimes F$ é acíclico.*

Arestas de reversão em torneios

Dado um torneio T, existe um conjunto de arcos $F \leq k$ tal que $G \otimes F$ é acíclico?

$\Rightarrow$ Sup q $\in F$ é conj. minimal retro.

- Sup. por abs q $\delta \notin F$ contém C
- Seja $\{f_1, f_2, \ldots, f_k\} = E(C) \cap rev(F)$
- Como F é minimal $\forall$



$\lightning$

$\Leftarrow$

- Sup. que F minimal tq $G \otimes F$ é ac.

$\Rightarrow$ Pelo Lema: F é conj. de retro.

— Sup. q F não é minimal

— Seja $F' \subseteq F$ que é minimal

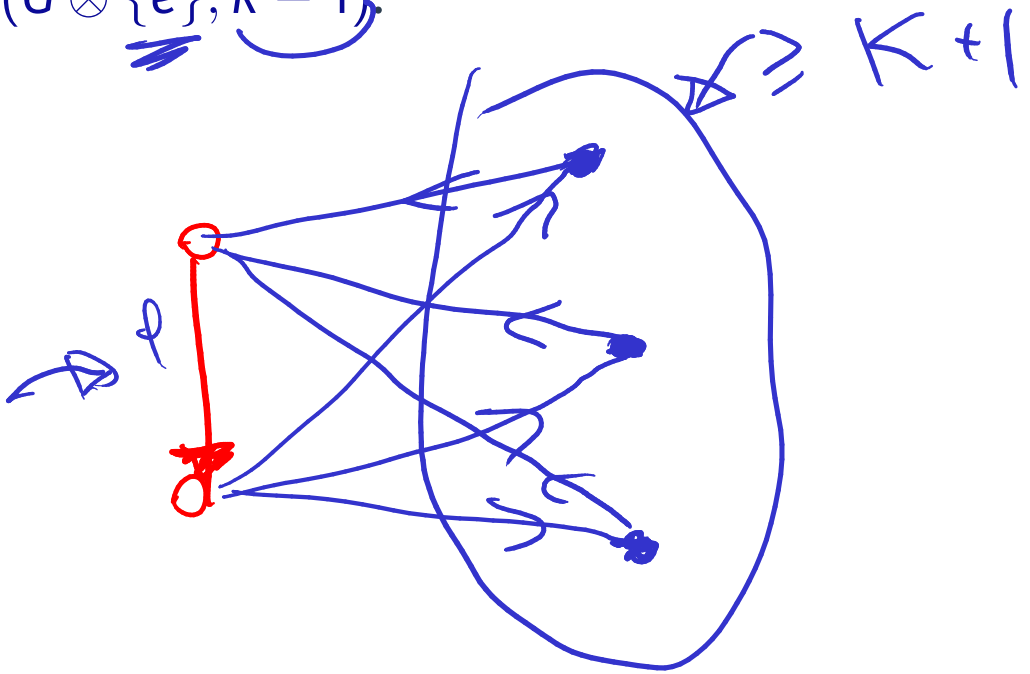
$\Rightarrow G \otimes F'$ é acíclico,

mas $F' \subsetneq F \Rightarrow F$ não é minimal $\lightning$
(tq $G \otimes F$ é ac.)

Reduções

Reduções

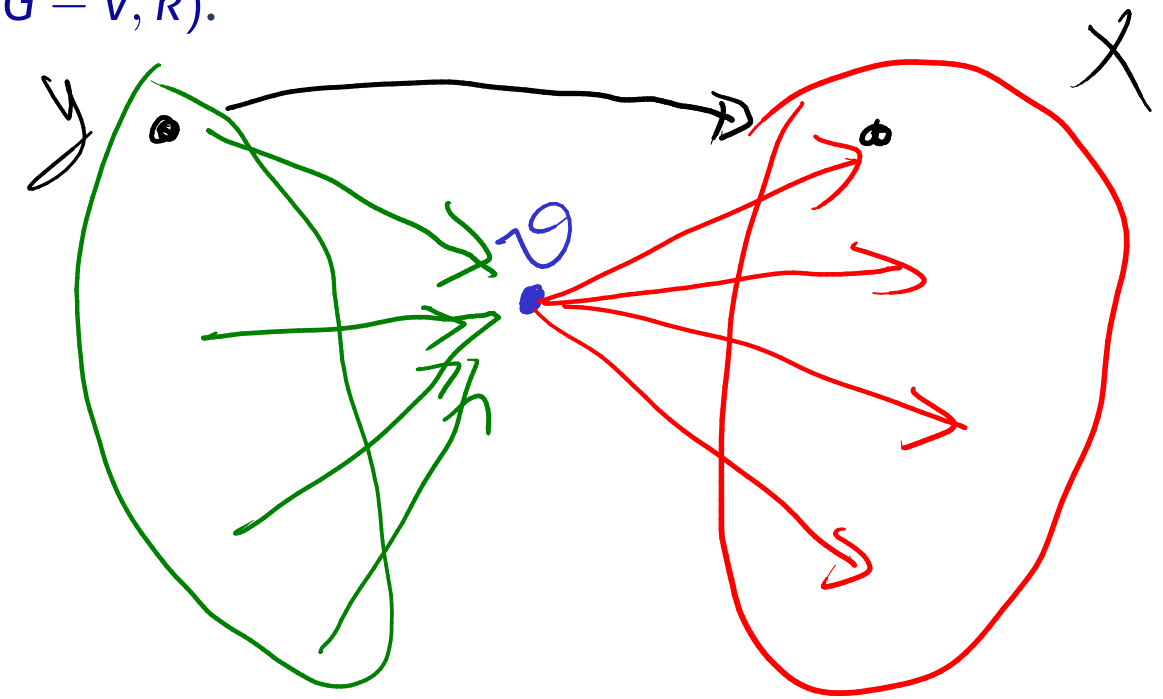
Redução FAST.1: Se uma arco e é contido em pelo menos $k+1$ triângulos, devolva $(G \otimes \{e\}, k-1)$.



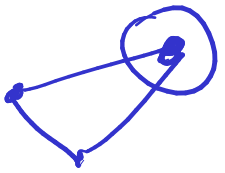
Reduções

Redução FAST.1: Se uma arco e é contido em pelo menos $k+1$ triângulos, devolva $(G \otimes \{e\}, k-1)$.

Redução FAST.2: Se vértice v não é contido em nenhum triângulo, devolva $(G - v, k)$.



Reduções



Redução FAST.1: Se uma arco e é contido em pelo menos $k+1$ triângulos, devolva $(G \otimes \{e\}, k-1)$.

Redução FAST.2: Se vértice v não é contido em nenhum triângulo, devolva $(G - v, k)$.

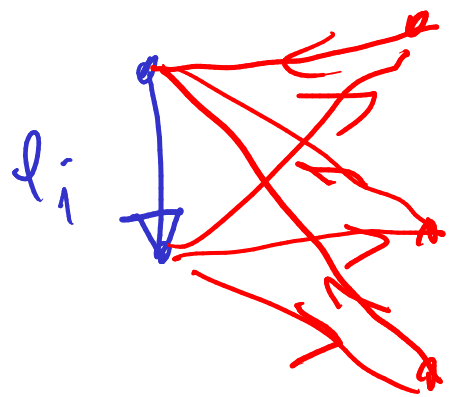
Teorema

O problema de conjunto de retroalimentação em torneios admite um núcleo com no máximo $k^2 + 2k$ vértices.

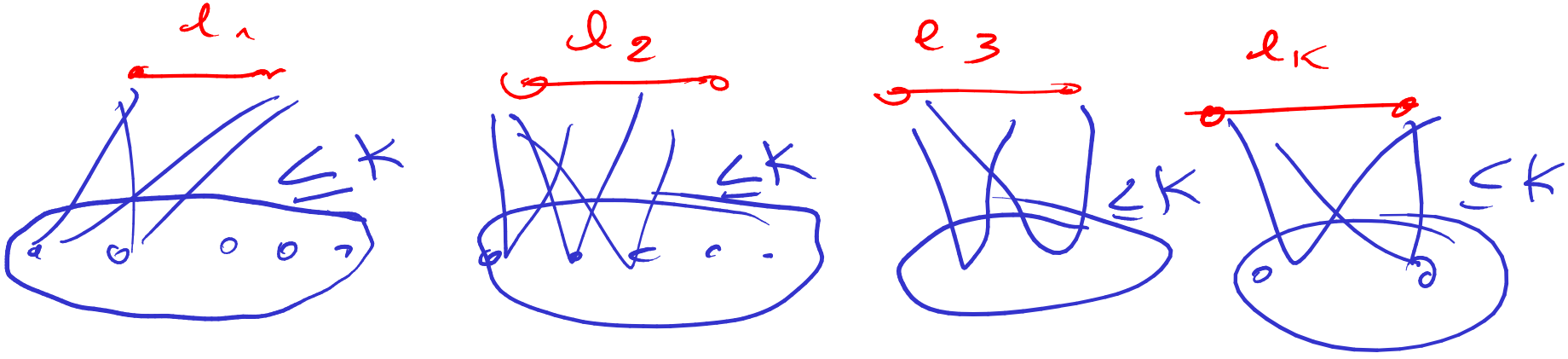
- Sup. q existe F tq. 60 F é ac.
  - $|F| \leq K$
  - Ap. FAST. {1,2}

$$F = \{l_1, \ldots, l_K\}$$

$$X_i = \{ v : (l_i, v) \text{ é triângulo} \}$$



$\bigcup X_i = V$



→ todo vertice faz parte a algum X_i

→ pq todo X_i, existe q_i tq $q_i \in F$

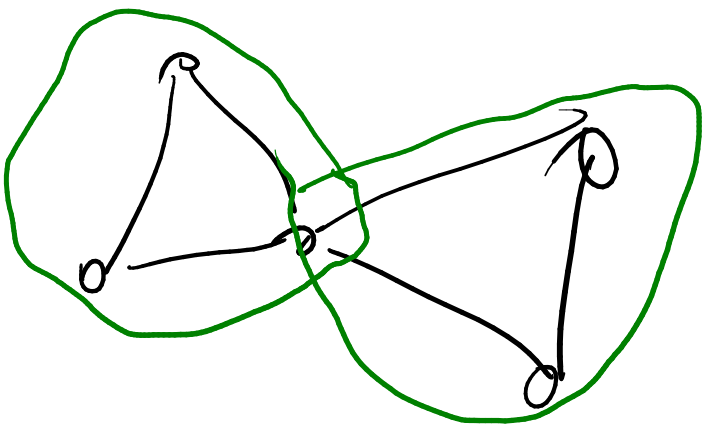
$$|V| \leq \sum_{i=1}^{K} (|X_i| + 2) \leq K^2 + K$$

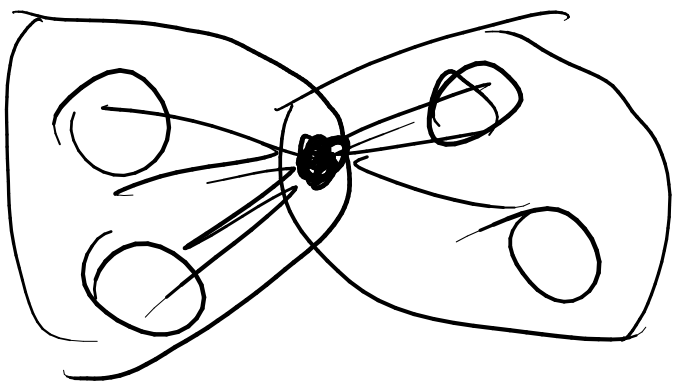
Problema da Cobertura de Arestas por Cliques

Cobertura de Arestas por Cliques

Problema da Cobertura de Arestas por Cliques (ECC)

Dado um gravo G e um inteiro não negativo k, existe um conjunto de k cliques (disjuntos nas arestas) que cobrem as arestas?





Cobertura de Arestas por Cliques

Problema da Cobertura de Arestas por Cliques (ECC)

Dado um gravo G e um inteiro não negativo k, existe um conjunto de k cliques (disjuntos nas arestas) que cobrem as arestas?

Redução ECC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

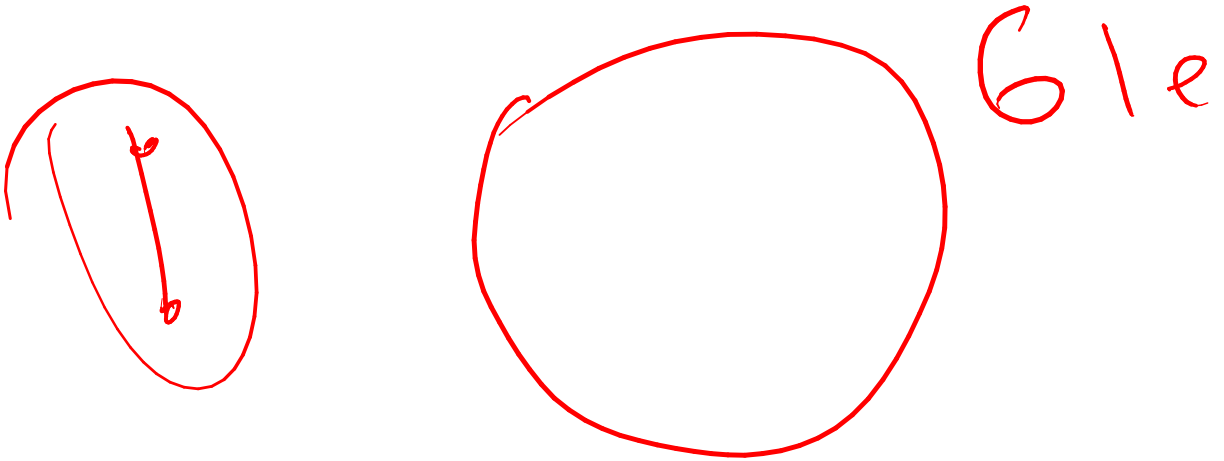
Cobertura de Arestas por Cliques

Problema da Cobertura de Arestas por Cliques (ECC)

Dado um gravo G e um inteiro não negativo k, existe um conjunto de k cliques (disjuntos nas arestas) que cobrem as arestas?

Redução ECC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

Redução ECC.2: Se existe $uv \in E[G]$, tal que $N[u] = N[v] = \{u, v\}$ (isso é, uv é uma componente conexa), devolva $(G - \{u, v\}, k - 1)$.



Cobertura de Arestas por Cliques

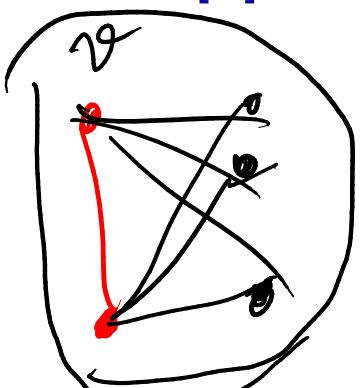
Problema da Cobertura de Arestas por Cliques (ECC)

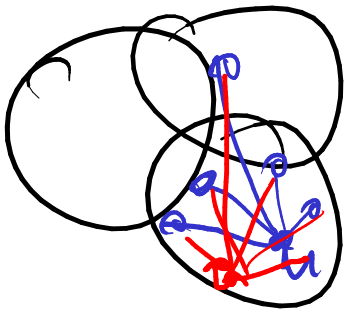
Dado um gravo G e um inteiro não negativo k, existe um conjunto de k cliques (~~disjuntos nas arestas~~) que cobrem as arestas?

Redução ECC.1: Se G contém um vértice isolado v, devolva $(G - v, k)$.

Redução ECC.2: Se existe $uv \in E[G]$, tal que $N[u] = N[v] = \{u, v\}$ (isso é, uv é uma componente conexa), devolva $(G - \{u, v\}, k - 1)$.

Redução ECC.3: Se existe $uv \in E[G]$, tal que $N[u] = N[v]$, devolva $(G - v, k)$.





Núcleo não polinomial

Teorema

O Problema da Cobertura de Arestas por Cliques admite um núcleo com 2^k vértices.

* Sup. q é inst sim

* Sup. q ECC. $\{1,2,3\}$ é aprd.

$(1 \quad 0 \quad 1 \quad 0 \quad 0)$

$\Rightarrow \exists\ C_1, C_2, \ldots, C_K$

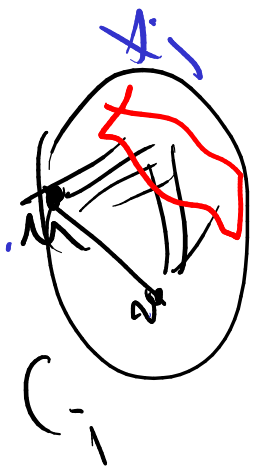
$v \in V$: defina $D_v[i] = \begin{cases} 0, & v \notin C_i \\ 1, & v \in C_i \end{cases}$

Fato: existem 2^K difusores.

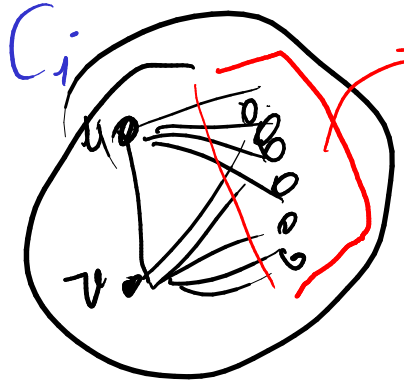
• Sup por abs q $\exists\, u, v \in V$ $u \neq v$

$$b_v = b_u$$

→ como ECC.1 n se aplica: $b_v \neq 0$



→ Daí, para cada clique C_i $b_u \neq 0$ q contém u e v ⇒ $uv \in E(G)$.



$N(v) \setminus u = \bigcup_i X_i =$

$N(u) \setminus v \Rightarrow N[v] = N[v]$

→ daí, ECC.2 ou 3 se aplicam ↯

Núcleo não polinomial

Teorema

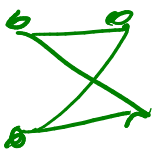
O Problema da Cobertura de Arestas por Cliques admite um núcleo com 2^k vértices.

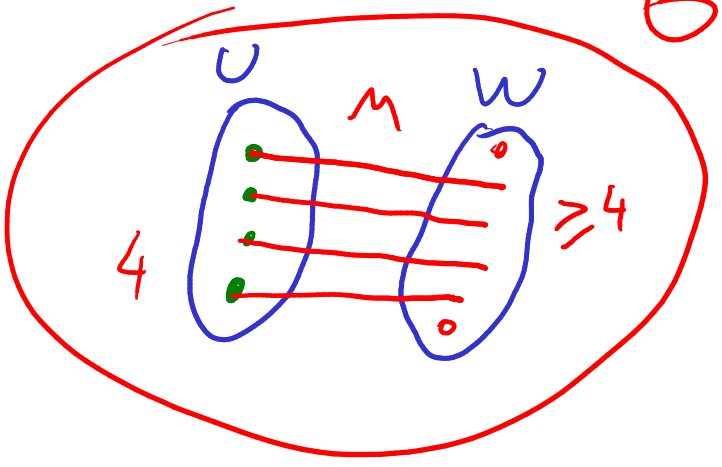
Dificuldade: A não ser que a Hipótese do Tempo Exponencial valha, não existe Kernel assi. melhor.

Decomposição em Coroa

Emparelhamento

Dados dois conjuntos de vértices disjuntos U e W de grafo G, dizemos que um conjunto de arestas M é um emparelhamento de U em W se:

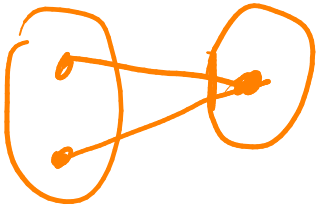




$N_M(U) \subseteq W$

$\Downarrow$

$N_M(U) \subseteq N_G(U)$



$X \subseteq U$. Sup. $|X| > N_G(X)$

Emparelhamento

Dados dois conjuntos de vértices disjuntos U e W de grafo G, dizemos que um conjunto de arestas M é um emparelhamento de U em W se:

- M é um emparelhamento

Emparelhamento

Dados dois conjuntos de vértices disjuntos U e W de grafo G, dizemos que um conjunto de arestas M é um emparelhamento de U em W se:

- M é um emparelhamento
- cada aresta tem exatamente um extremo em U e um em W

Emparelhamento

Dados dois conjuntos de vértices disjuntos U e W de grafo G, dizemos que um conjunto de arestas M é um emparelhamento de U em W se:

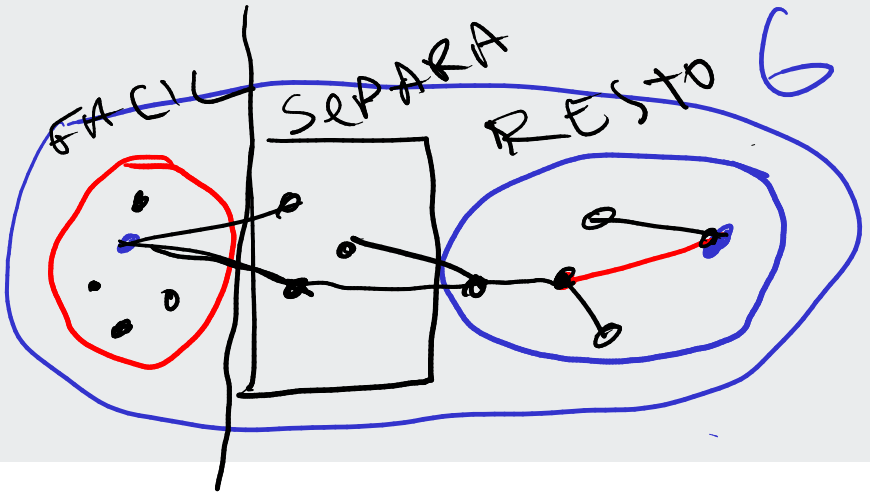
- M é um emparelhamento
- cada aresta tem exatamente um extremo em U e um em W
- M satura U (i.e., todo vértice de U é um extremo de alguma aresta de M)

Decomposição em coroa

Definição (Decomposição em coroa)

Uma decomposição em coroa de um grafo G é uma partição de $V(G)$ em três partes C, H e R tal que:

OBS:

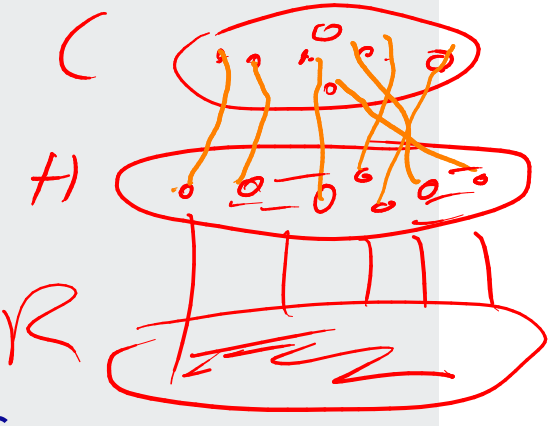


Decomposição em coroa

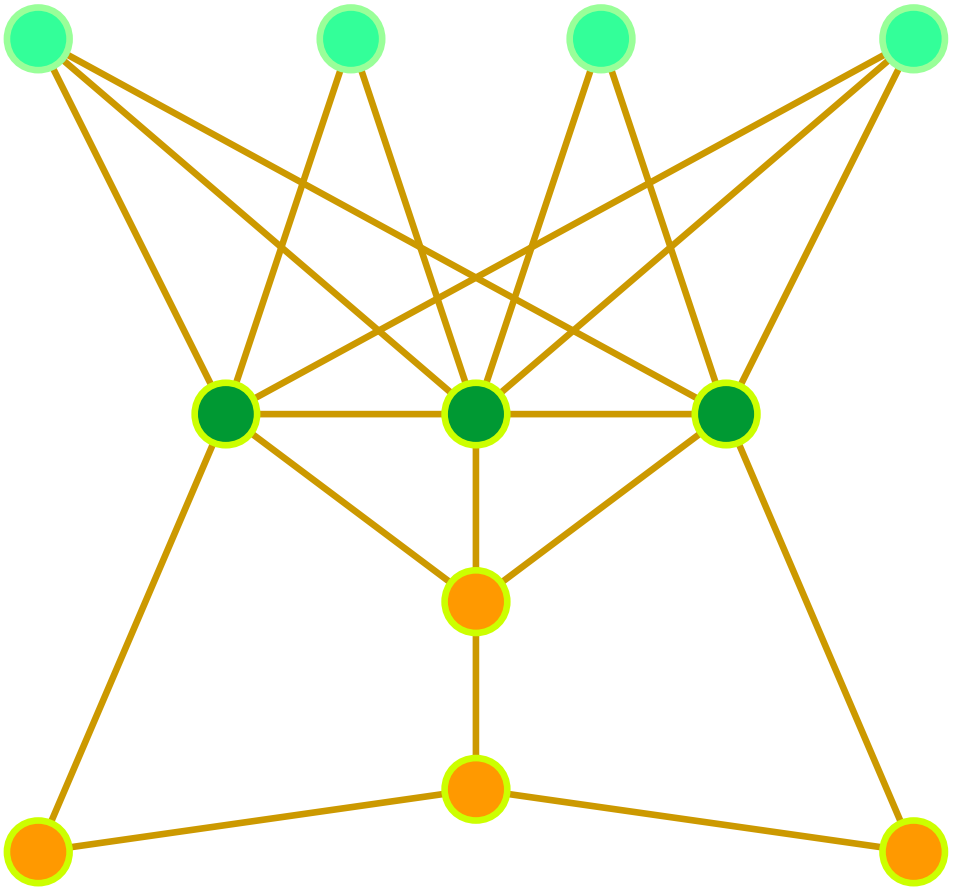
Definição (Decomposição em coroa)

Uma decomposição em coroa de um grafo G é uma partição de $V(G)$ em três partes C, H e R tal que:

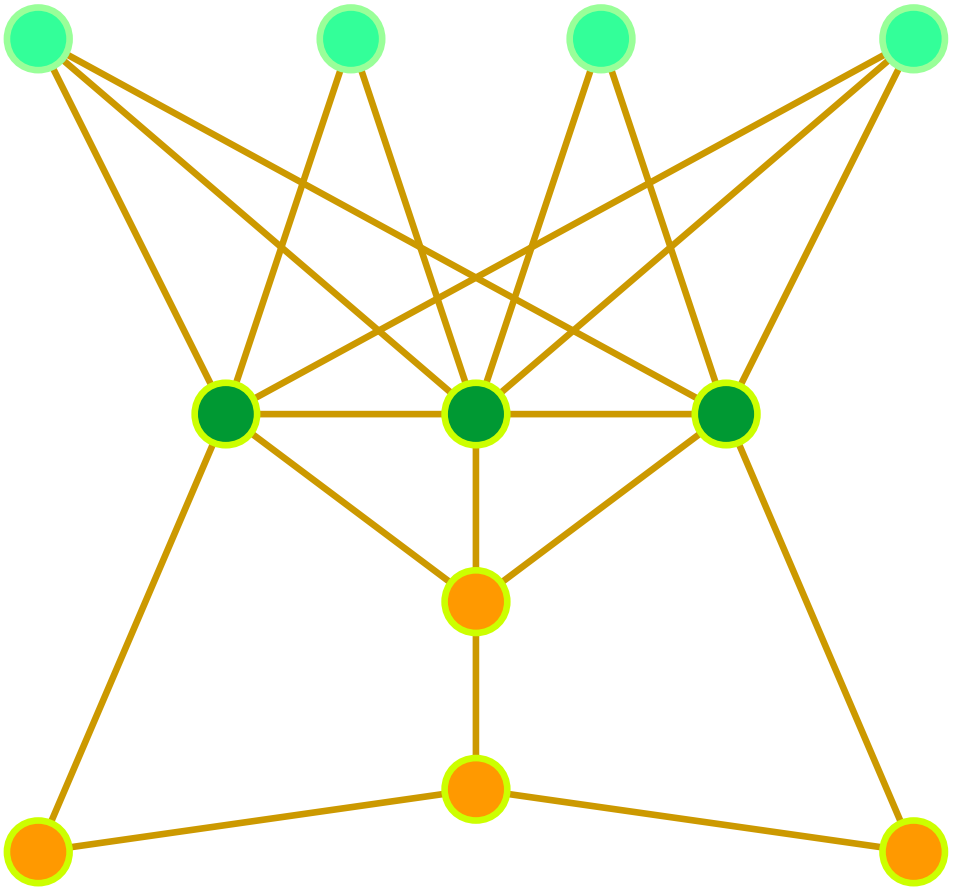
1. C é não vazio;
2. C é um conjunto independente;
3. H é um separador de (C, R);
4. G contém um emparelhamento de H em C



Coroa

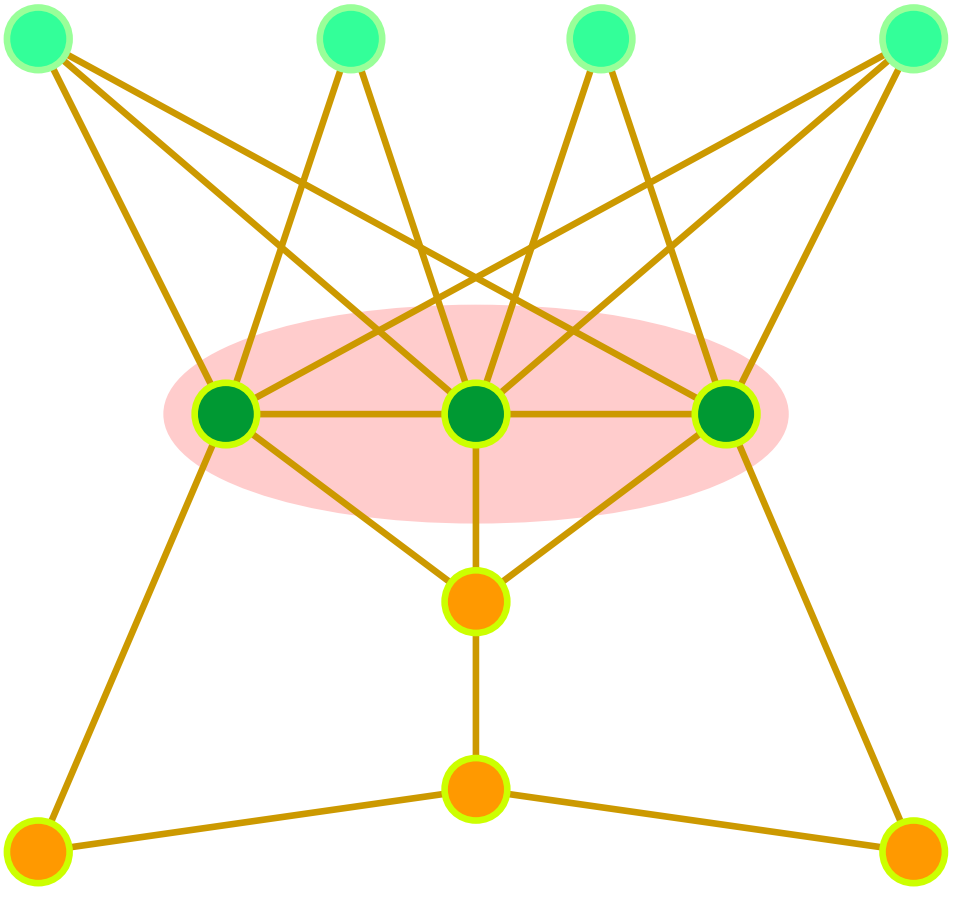


Coroa

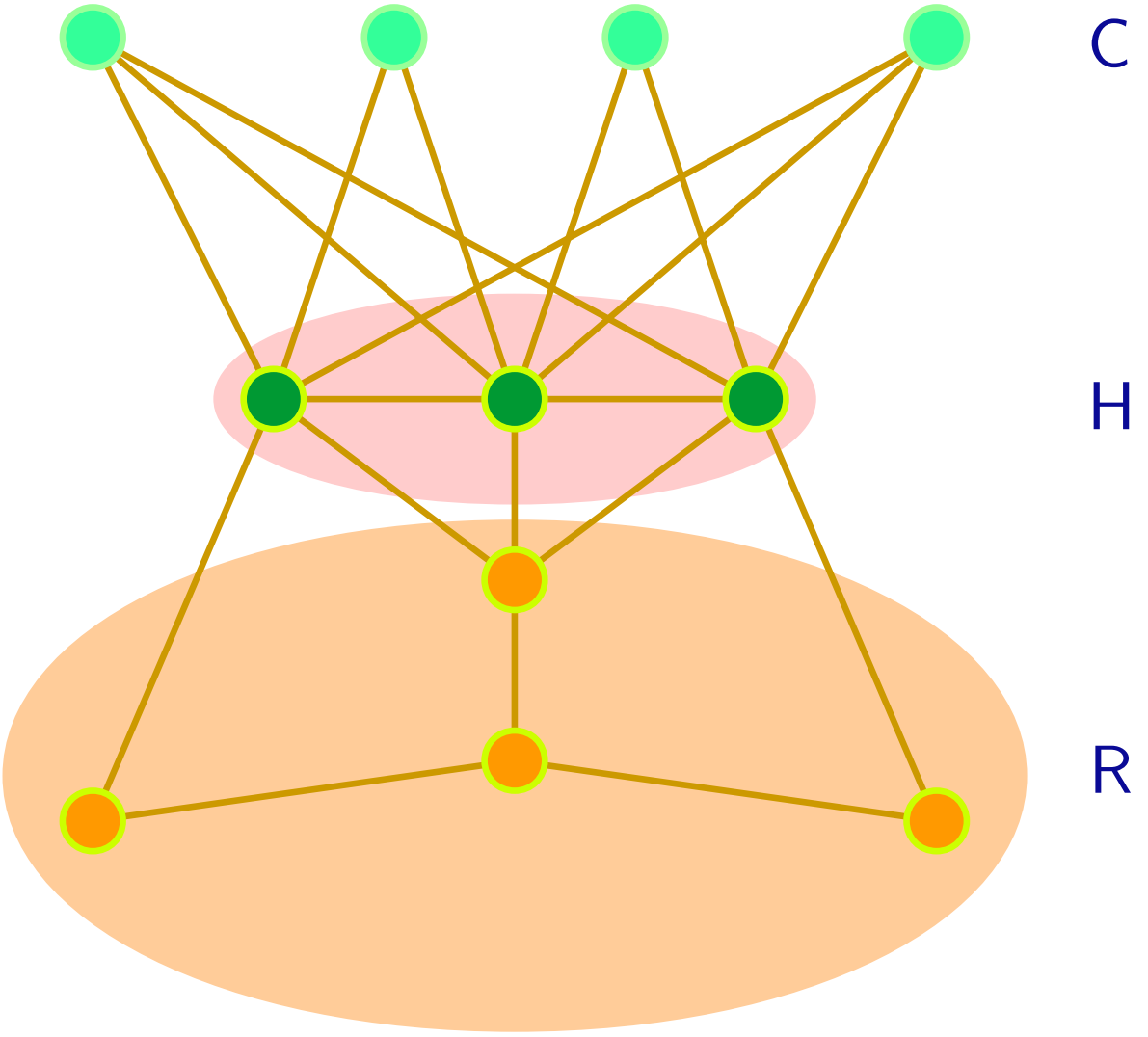


C

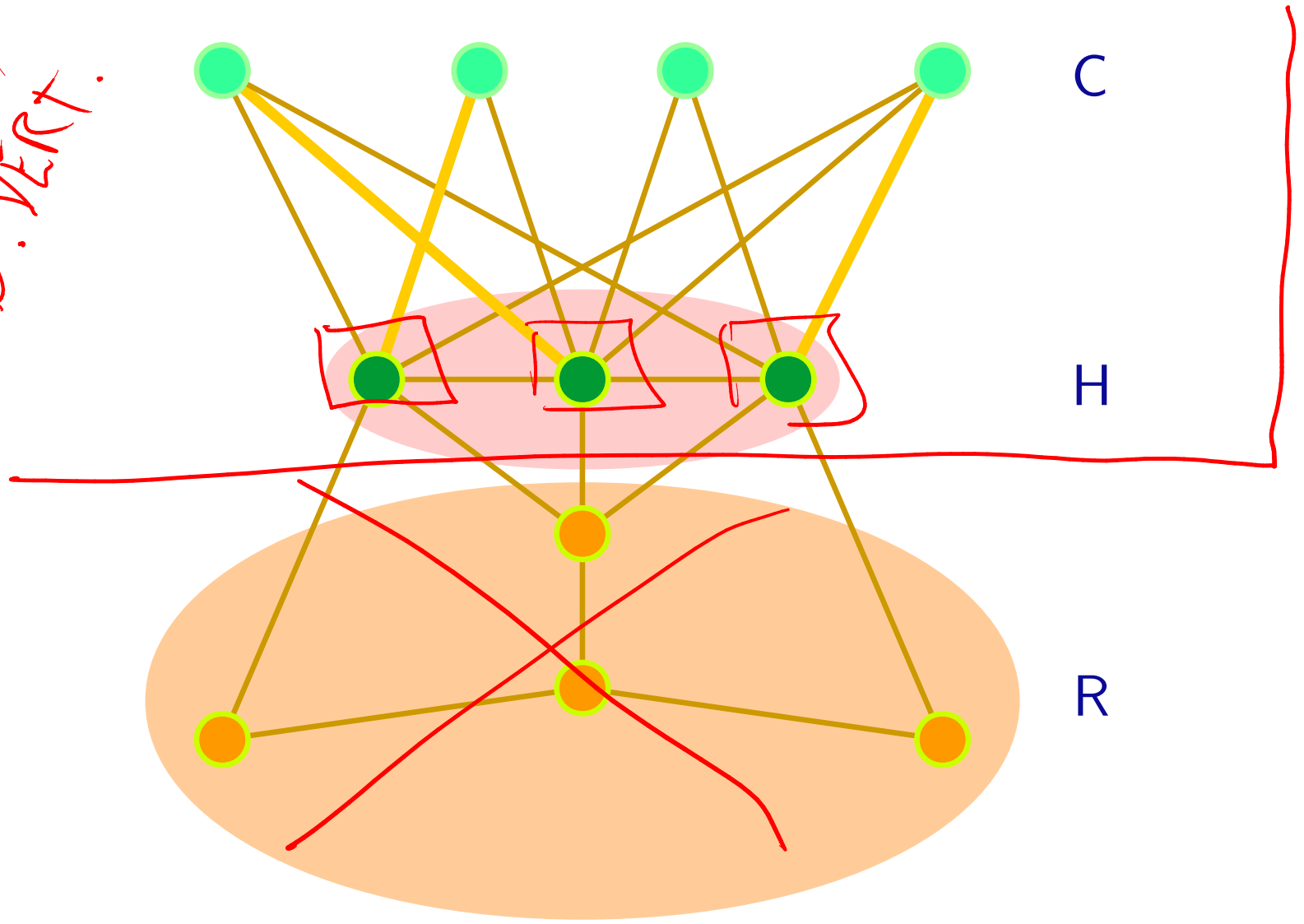
Coroa



Coroa



Coroa

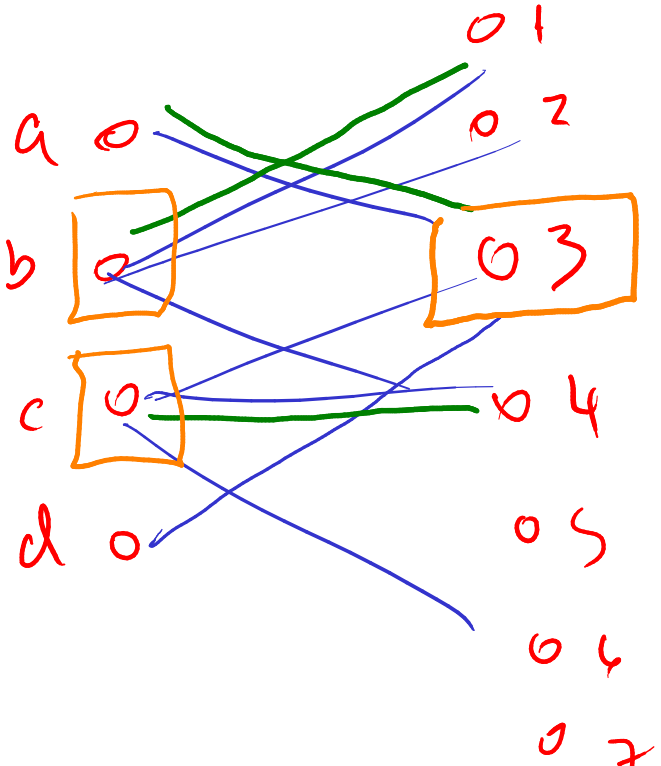


Teoremas estruturais

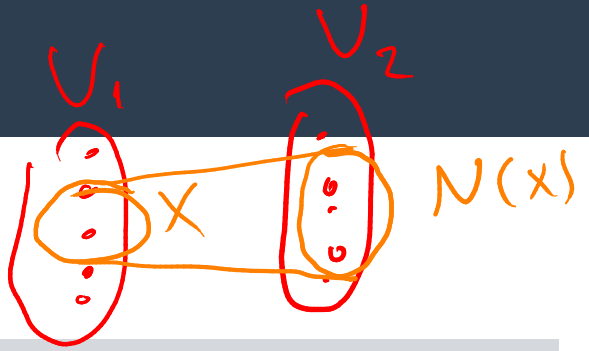
Rei 2.e.

Teorema (~~Kőnoig~~)

Se G é um grafo bipartido, então o tamanho do emparelhamento máximo é igual ao tamanho da cobertura por vértices mínima.



Teoremas estruturais

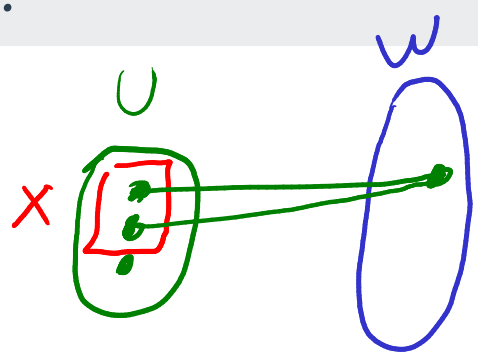


Teorema (Kőnoig)

Se G é um grafo bipartido, então o tamanho do emparelhamento máximo é igual ao tamanho da cobertura por vértices mínima.

Teorema (Hall)

Seja G um grafo bipartido com partição V_1, V_2. Existe um emparelhamento que satura V_1 sss para todo $X \subseteq V_1$ vale $|N(X)| \geq |X|$.



Cond necessária:

$\forall X \subseteq U, |N(X) \cap W| \geq |X|$

Encontrando um emparelhamento

Teorema (Hopcroft-Karp)

Seja G um grafo bipartido com partição V_1, V_2 com n vértices e m arestas.

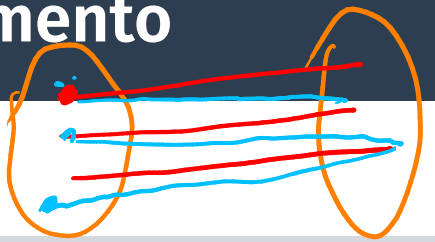
Encontrando um emparelhamento

Teorema (Hopcroft-Karp)

Seja G um grafo bipartido com partição V_1, V_2 com n vértices e m arestas.

- *Existe um algoritmo que encontra um emparelhamento máximo e uma cobertura por vértices mínima em tempo $\mathcal{O}(m\sqrt{n})$.*

Encontrando um emparelhamento



Teorema (Hopcropt-Karp)

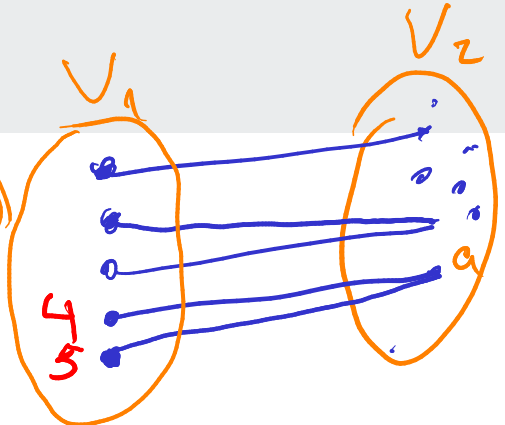
Seja G um grafo bipartido com partição V_1, V_2 com n vértices e m arestas.

- *Existe um algoritmo que encontra um emparelhamento máximo e uma cobertura por vértices mínima em tempo $\mathcal{O}(m\sqrt{n})$.*
- *Além disso, o algoritmo encontra um emparelhamento que satura V_1, ou encontra um conjunto minimal $X \subseteq V_1$ tal que $|N(X)| < |X|$.*

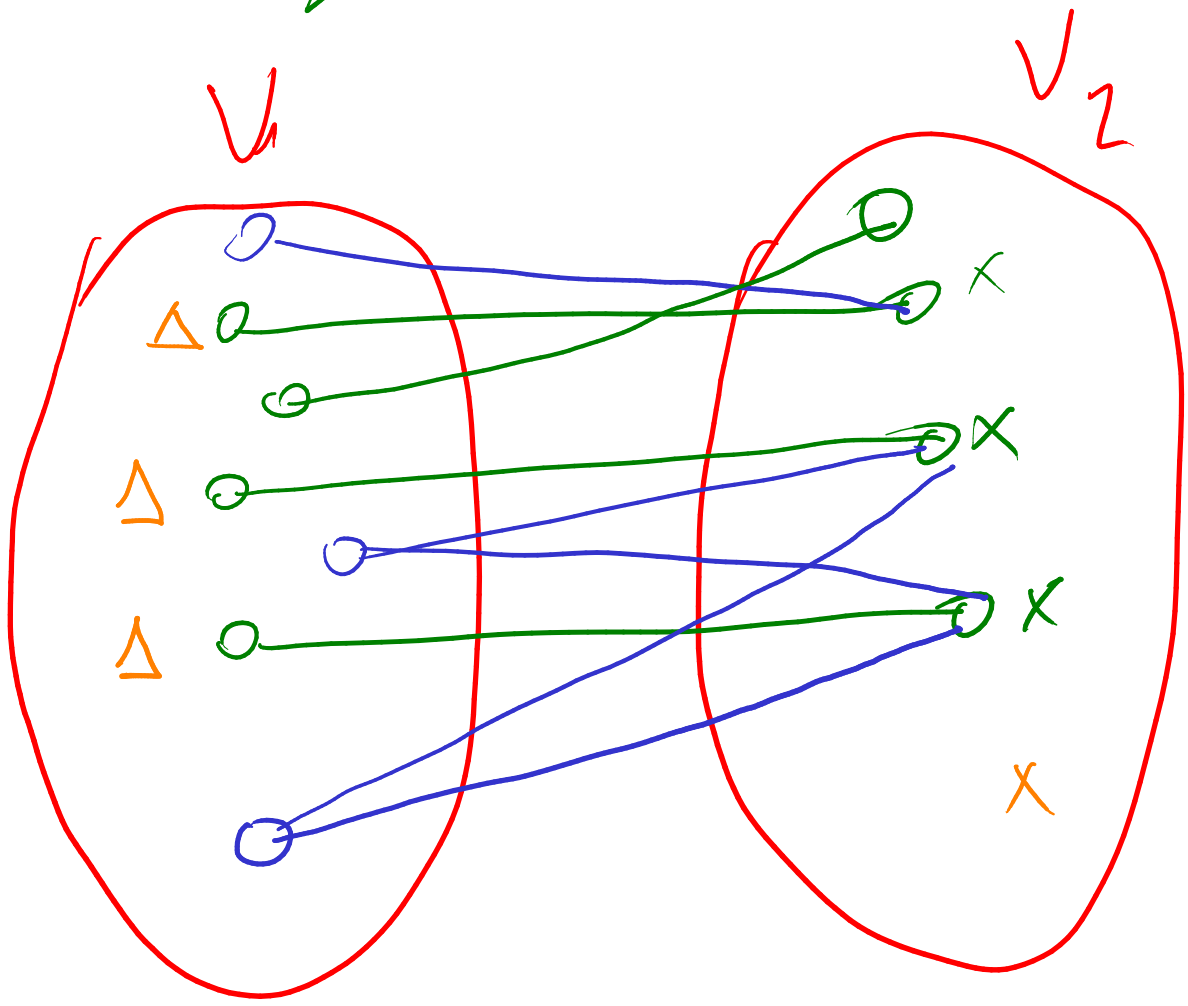
$X = \{4, 5\}$

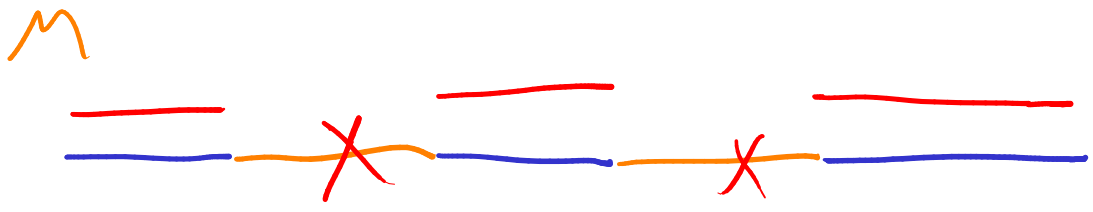
$N(X) = \{a\}$

$|X| = 2 > 1 = |N(X)|$



matching máximo





Lema da coroa

Lema (Lema da coroa)

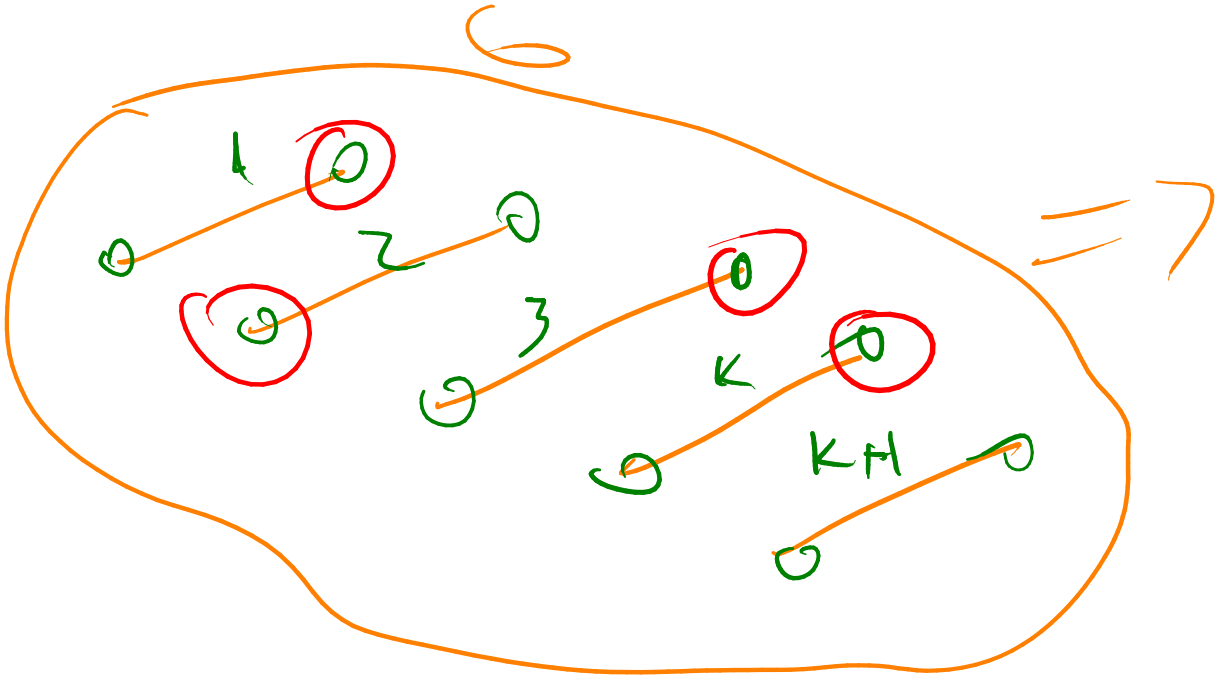
Seja G um grafo sem vértices isolados e com pelo menos $3k + 1$ vértices. Existe um algoritmo polinomial que:

Lema da coroa

Lema (Lema da coroa)

Seja G um grafo sem vértices isolados e com pelo menos $3k + 1$ vértices. Existe um algoritmo polinomial que:

- *encontra um emparelhamento de tamanho $k + 1$ em G; ou*
- *encontra uma decomposição em coroa de G.*



Lema da coroa

Lema (Lema da coroa)

Seja G um grafo sem vértices isolados e com pelo menos $3k + 1$ vértices. Existe um algoritmo polinomial que:

- *encontra um emparelhamento de tamanho $k + 1$ em G; ou*
- *encontra uma decomposição em coroa de G.*

Aplicação: Se um parâmetro k for limitante superior para a cobertura por vértices, então pode ser uma ferramenta para encontrar um núcleo pequeno

$k \geq |C.P.V.M|$

Lema da coroa

Lema (Lema da coroa)

Seja G um grafo sem vértices isolados e com pelo menos $3k + 1$ vértices. Existe um algoritmo polinomial que:

- *encontra um emparelhamento de tamanho $k + 1$ em G; ou*
- *encontra uma decomposição em coroa de G.*

Aplicação: Se um parâmetro k for limitante superior para a cobertura por vértices, então pode ser uma ferramenta para encontrar um núcleo pequeno

$\Rightarrow$ isso é, se k for pequeno, então também o tamanho da cobertura mínima.

Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

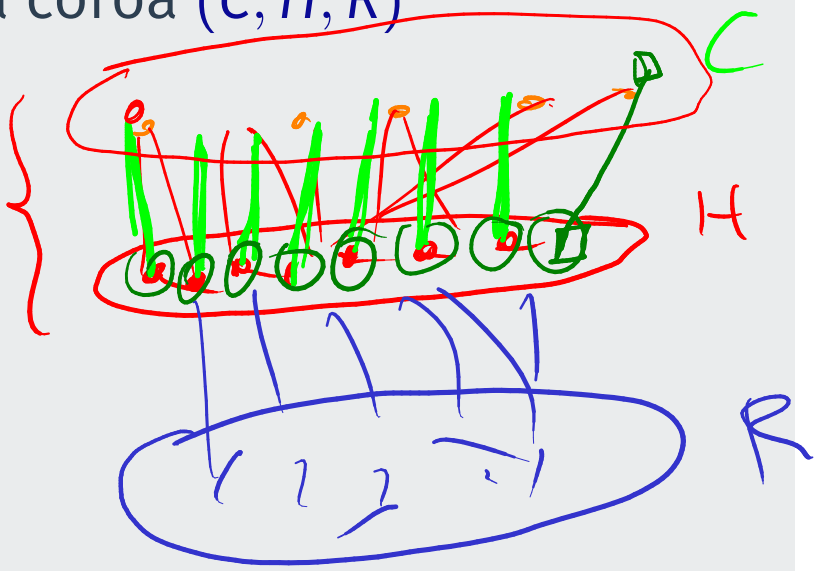
1. Aplicamos VC.1 (removemos vértices isolados) ;

Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)

Ex. p. menos |H| vértices em cobertura ótima.



$|C| > |V|$

Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)
   2.1 Se há um emparelhamento M, com $|M| = k + 1$:

Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)
   - 2.1 Se há um emparelhamento M, com $|M| = k + 1$:
     - responda não;

Aplicação: Cobertura por vértices

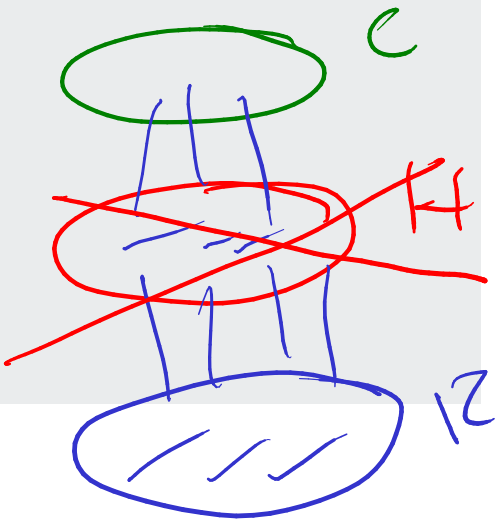
NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)
   - 2.1 Se há um emparelhamento M, com $|M| = k + 1$:
     - responda não;
   - 2.2 Senão:
     - Faça $(G, k) \leftarrow (G - H, k - |H|)$;

Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)
   - 2.1 Se há um emparelhamento M, com $|M| = k + 1$:
     - responda não;
   - 2.2 Senão:
     - Faça $(G, k) \leftarrow (G - H, k - |H|)$;
     - Volte ao passo 1



Aplicação: Cobertura por vértices

NÚCLEO-VC(G, k):

1. Aplicamos VC.1 (removemos vértices isolados) ;
2. Se $|V| \geq 3k + 1$, obtenha uma coroa (C, H, R)
   - 2.1 Se há um emparelhamento M, com $|M| = k + 1$:
     - responda não;
   - 2.2 Senão:
     - Faça $(G, k) \leftarrow (G - H, k - |H|)$;
     - Volte ao passo 1
   - 2.3 Devolva (G, k)

$\Theta(K)$

Problema de satisfatibilidade máxima

Problema de satisfatibilidade máxima

Dada uma fórmula normal conjuntiva (CNF) F, e um inteiro k, existe uma atribuição de verdade nas variáveis que satisfaz pelo menos k cláusulas.

Problema de satisfatibilidade máxima

Problema de satisfatibilidade máxima

Dada uma fórmula normal conjuntiva (CNF) F, e um inteiro k, existe uma atribuição de verdade nas variáveis que satisfaz pelo menos k cláusulas.

Exemplo:

variáveis:

x_1	x_2	val
0	0	3
0	1	3
1	1	3
1	0	3

$$F = (x_0 \vee x_1) \wedge (x_0 \vee \neg x_1) \wedge (\neg x_0 \vee x_1) \wedge (\neg x_0 \vee \neg x_1)$$

$k = 3$

=> SIM!

=> K = 4 => NÃO!

Aplicação: Satisfatibilidade máxima

Vars $O(k)$

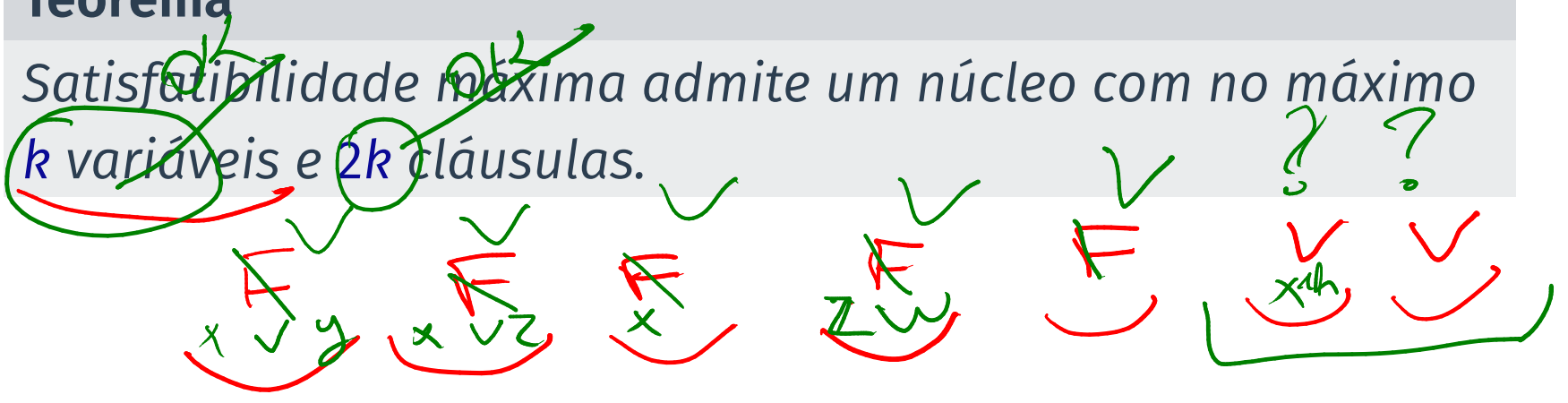
Cláusulas $O(k)$

$\neg x$

$x' = \neg x$

Teorema

Satisfatibilidade máxima admite um núcleo com no máximo k variáveis e $2k$ cláusulas.



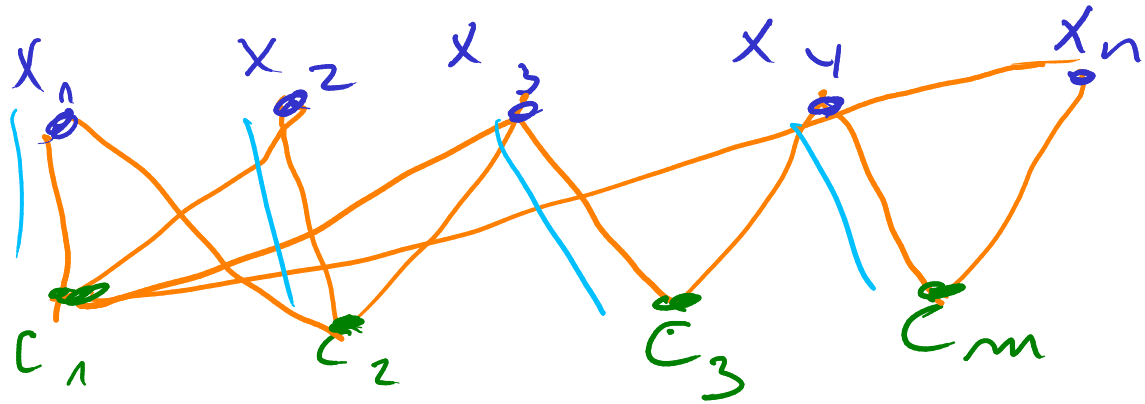
$\exists$ atrib. q satisfaz $\geq$

$|V| \geq |F| \Rightarrow |V| \geq \frac{m}{2}$

φ fórmula; n variáveis; m cláusulas; K

a) Se $m \geq 2K$, responda sim

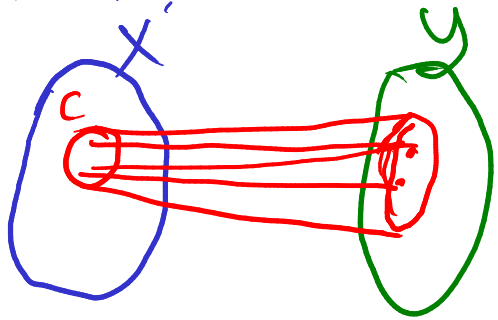
b) Sup. $K \leq m < 2K$



→ Calcule M emp. máx

a) Se $|M| = |X|$ ou se $|X| \geq K$

b)

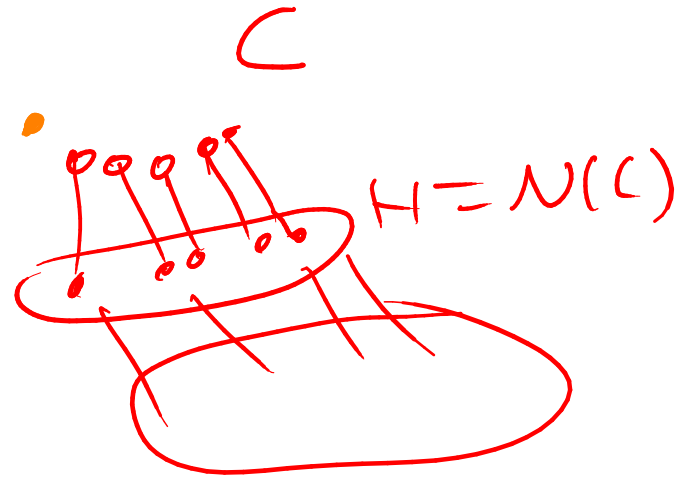


$\exists C \in X : |C| > N(L)$

$C := C$

$H := N(C)$

$R := V \setminus (N(C) \cup C)$

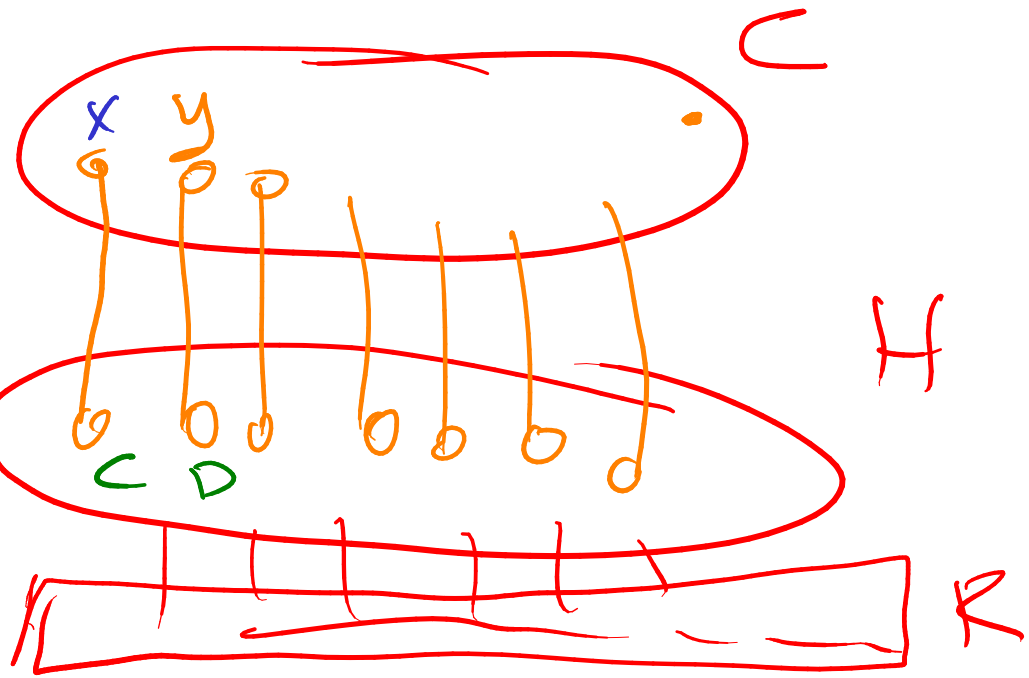


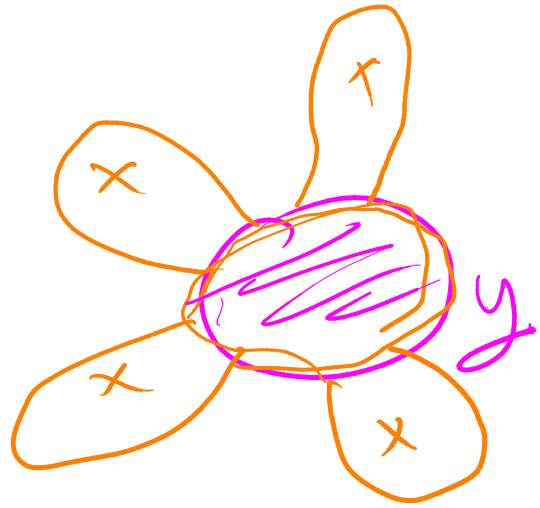
(C, H, R) é conexo

Prof:

Seja $G' = G - H - C$

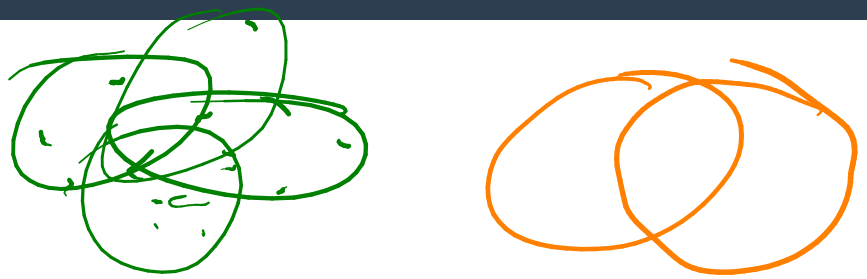
$K' = K - |H|$





Lema do Girassol

Girassol



Definição (Girassol)

Um girassol com k pétalas e centro (núcleo) Y é uma coleção de conjuntos $S_1, \ldots, S_k$, tais que:

Girassol

Definição (Girassol)

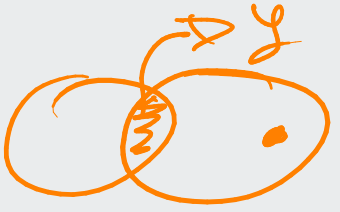
Um girassol com k pétalas e centro (núcleo) Y é uma coleção de conjuntos $S_1, \ldots, S_k$, tais que:

- $S_i \cap S_j = Y$ para todo $i \neq j$;

Girassol

Definição (Girassol)

Um girassol com k pétalas e centro (núcleo) Y é uma coleção de conjuntos $S_1, \ldots, S_k$, tais que:



- $S_i \cap S_j = Y$ para todo $i \neq j$;
- $S_i \setminus Y \neq \emptyset$ para todo i. (ou $S_i \neq Y$ p/ todo i)

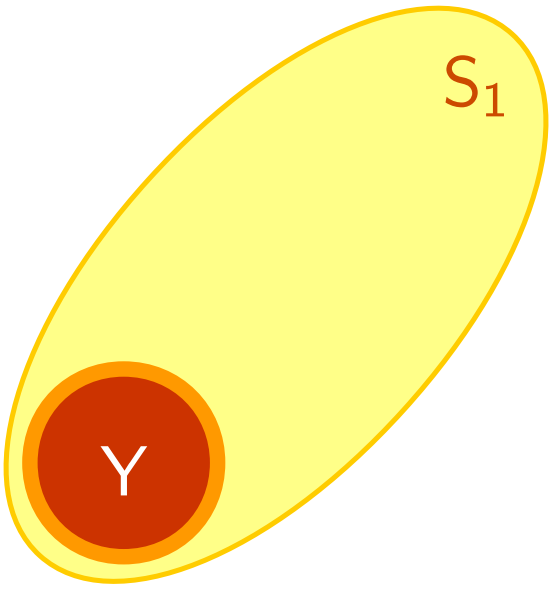
$S_i \neq Y$

Um girassol

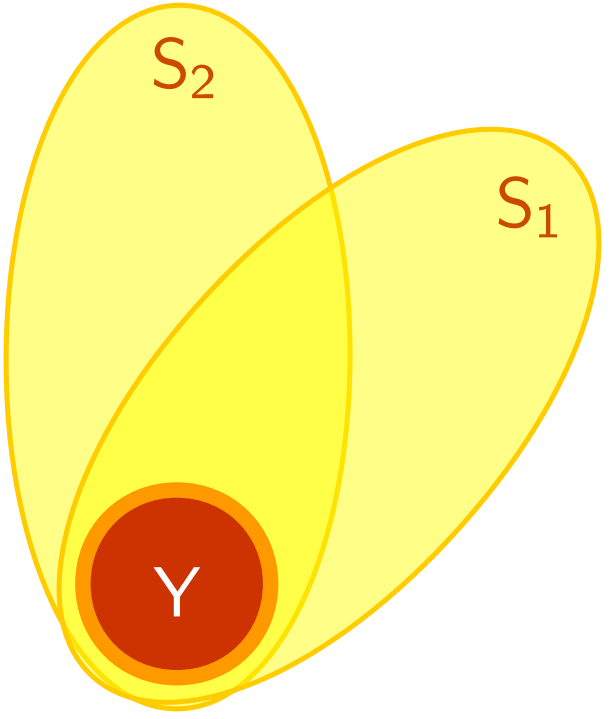
Um girassol

Um girassol

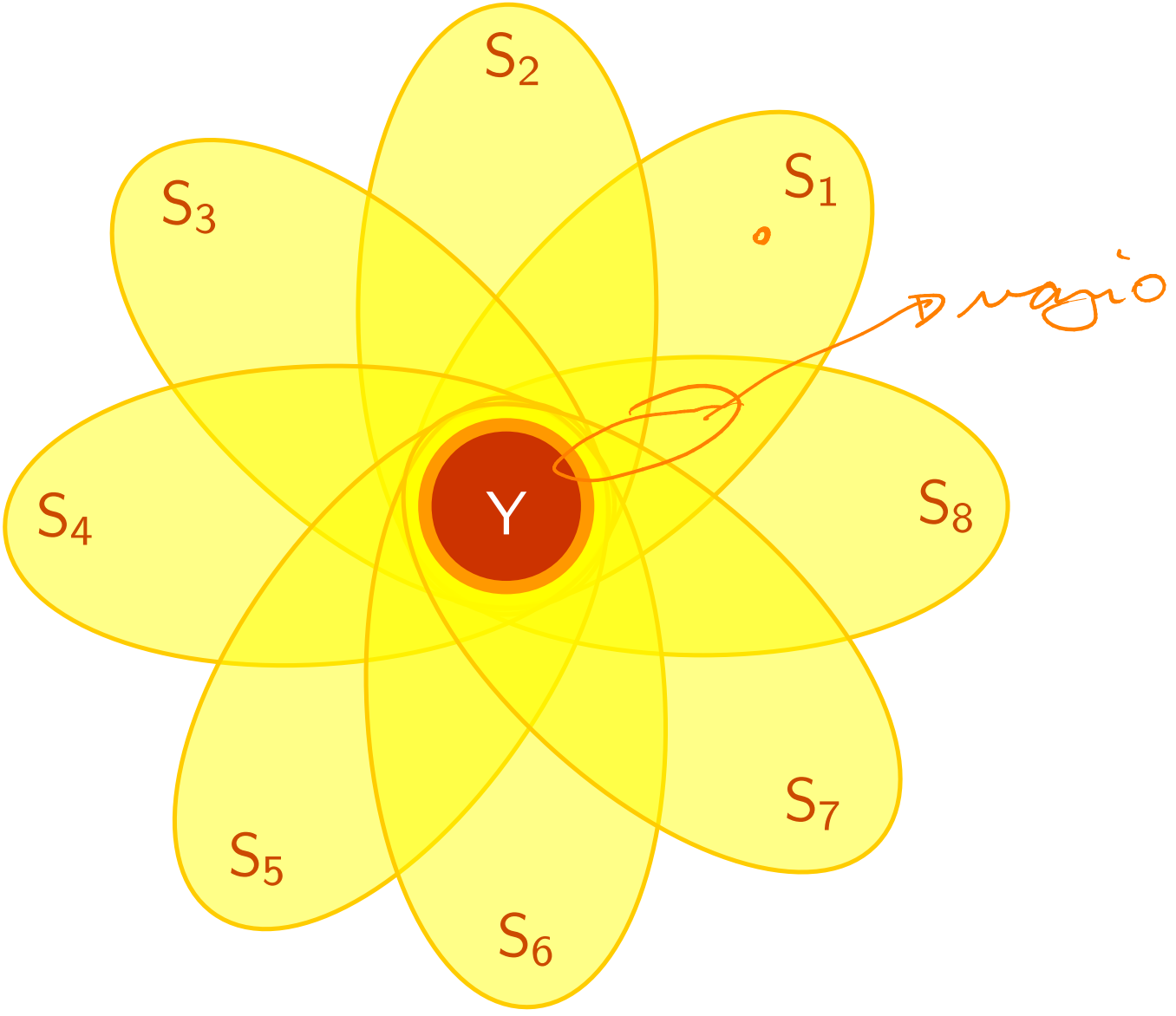
Um girassol



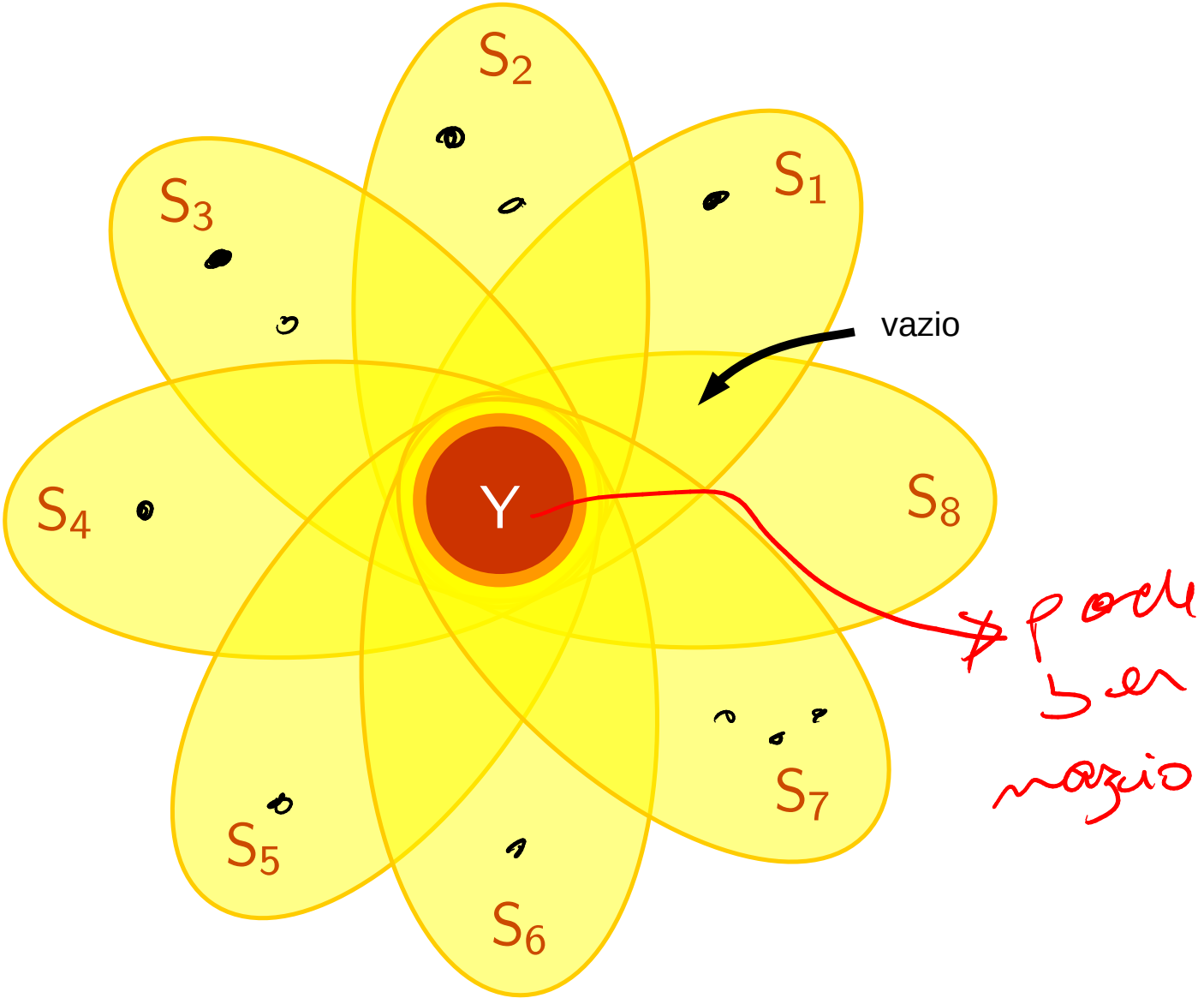
Um girassol



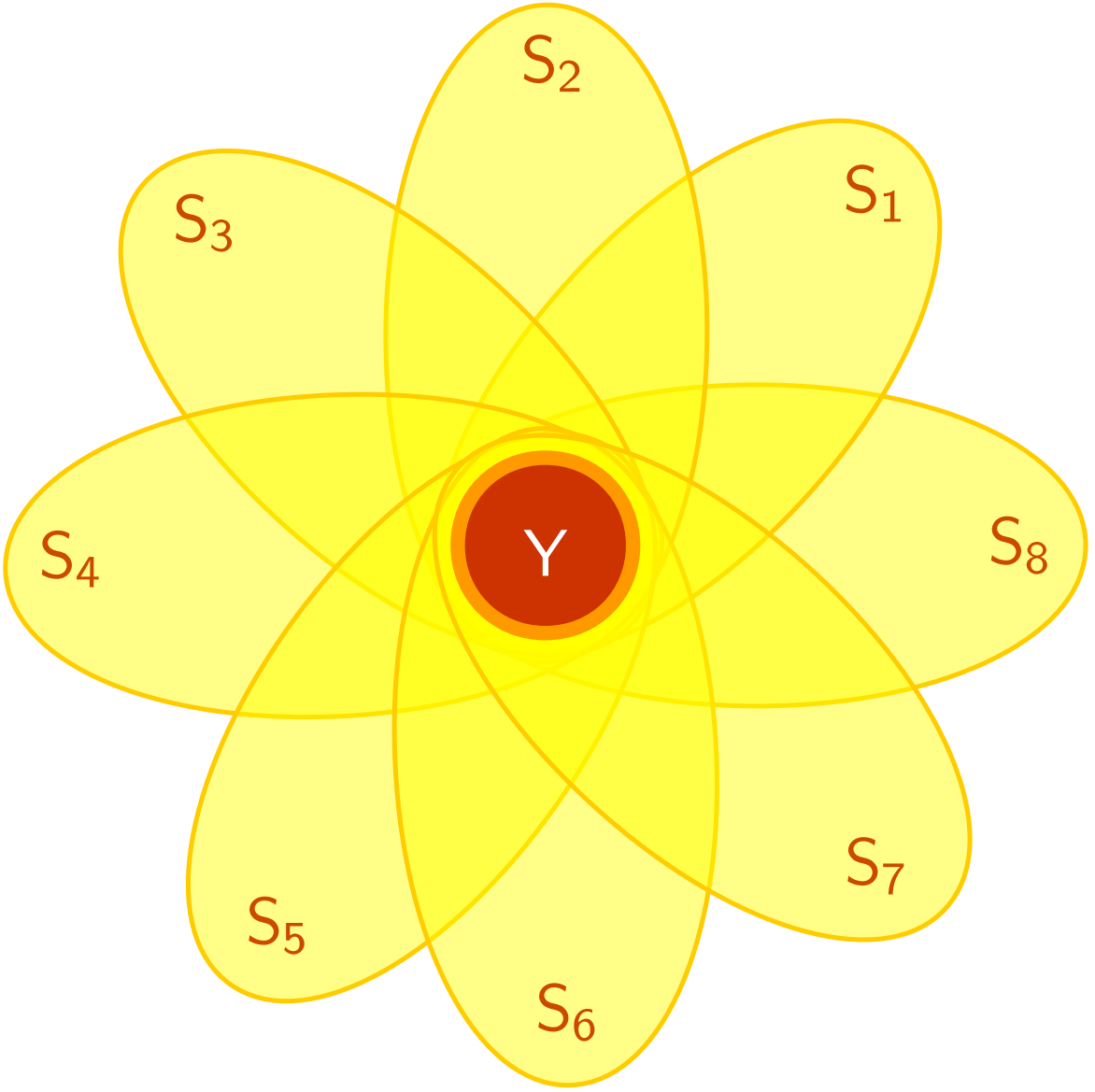
Um girassol



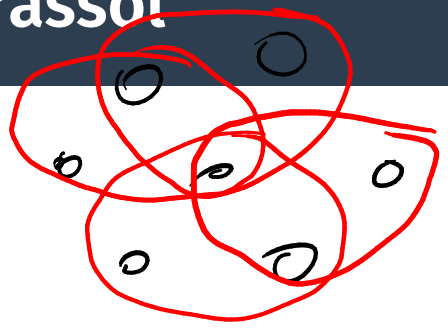
Um girassol



Um girassol



Lema do girassol



Lema

(Lema do girassol) Seja $\mathcal{A}$ uma família de conjuntos (possivelmente com duplicatas) sobre um universo U, tal que todo conjunto em $\mathcal{A}$ tem cardinalidade d.

Lema do girassol

Lema

Lema do girassol Seja $\mathcal{A}$ uma família de conjuntos (possivelmente com duplicatas) sobre um universo U, tal que todo conjunto em $\mathcal{A}$ tem cardinalidade d.

Se $|\mathcal{A}| > d!(k-1)^d$,

Lema do girassol

Lema

Lema do girassol Seja $\mathcal{A}$ uma família de conjuntos (possivelmente com duplicatas) sobre um universo U, tal que todo conjunto em $\mathcal{A}$ tem cardinalidade d.

Se $|\mathcal{A}| > d!(k-1)^d$, então $\mathcal{A}$ contém um girassol com k pétalas, que pode ser obtido em tempo polinomial em $|\mathcal{A}|$, $|U|$ e k.

$3! \cdot (11-1)^3 = 6 \cdot 1000 = 6000$

Preuve indurçãa em d $\quad |A_i| > d!(K-1)^d$

$= K-1$

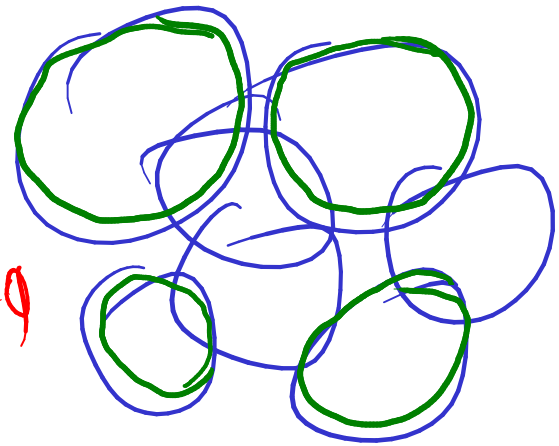
Se $d=1$:

OK

Se $d \geq 2$:

$\rightarrow S_{seq} = S_1, \ldots, S_\ell$

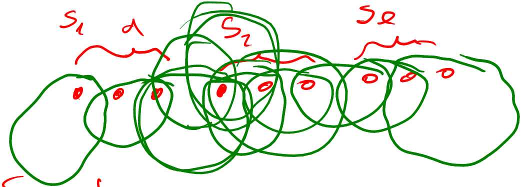
maximal tq $S_i \cap S_j = \emptyset$



a) $\ell \geq K$ OK

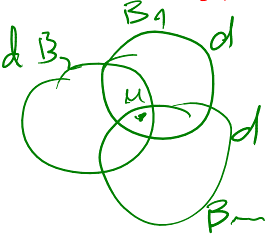
b) $\ell < K$

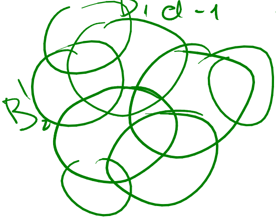
$$S = \bigcup_i S_i \qquad A > d!\,(k-1)^d$$



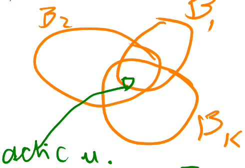
$\exists u \in S$ cob. por

$$\frac{|A|}{|S|} > \frac{d!\,(k-1)^d}{d \cdot l} \geq \frac{d!\,(k-1)^d}{d\,(k-1)} = (d-1)!\,(k-1)^{d-1}$$





Pela H.I.



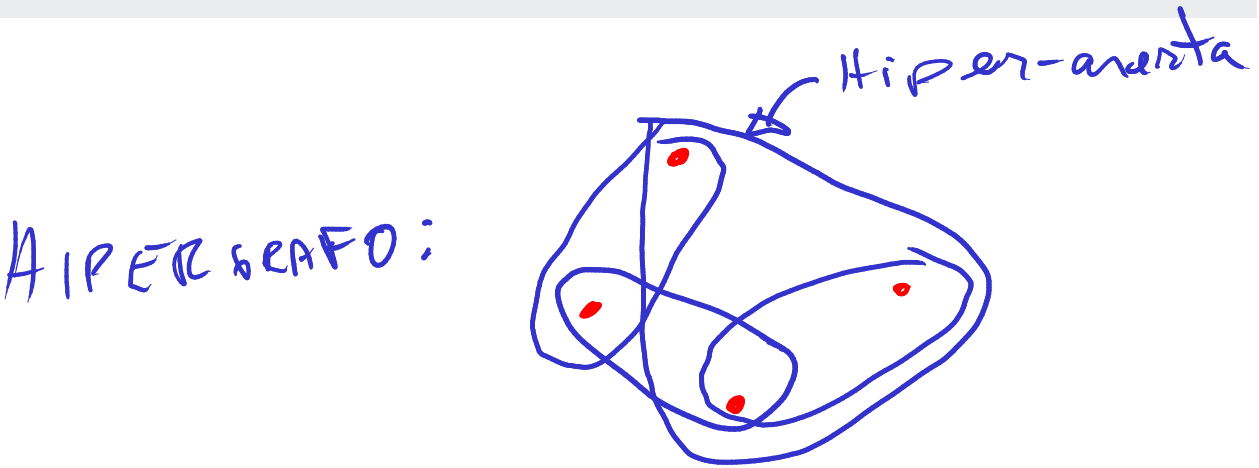
∎

Aplicação: Problema da transversal mínima

Problema da transversal mínima

Dada uma família de conjuntos $\mathcal{A}$ em universo U, tal que cada conjunto tem tamanho no máximo d e inteiro positivo k.

↳ 2p | grafo



Aplicação: Problema da transversal mínima

Problema da transversal mínima

Dada uma família de conjuntos $\mathcal{A}$ em universo U, tal que cada conjunto tem tamanho no máximo d e inteiro positivo k.

Existe um conjunto $H \subseteq U$ de tamanho no máximo k?

que intersecta todos conjuntos de $\mathcal{A}$

Aplicação: Problema da transversal mínima

Problema da transversal mínima

Dada uma família de conjuntos $\mathcal{A}$ em universo U, tal que cada conjunto tem tamanho no máximo d e inteiro positivo k.

Existe um conjunto $H \subseteq U$ de tamanho no máximo k?

Observação: O conjunto H é chamado de transversal.

Núcleo para transversal mínima

Teorema

O problema da transversal mínima admite um núcleo com no máximo $d!k^d$ conjuntos e $d!k^d \cdot d^2$ elementos.

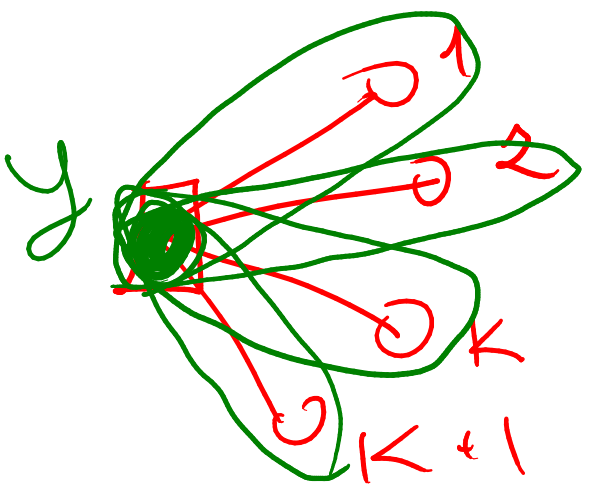
p/ 1 grafo: $2!\,k^2$ $\qquad$ $2!\,k^2 \cdot 2^2$

Núcleo para transversal mínima

Teorema

O problema da transversal mínima admite um núcleo com no máximo $d!k^d$ conjuntos e $d!k^d \cdot d^2$ elementos.

Redução HS.1: Se existe girassol $S = \{S_1, \ldots, S_{k+1}\}$ com centro Y, então devolva $(U', \mathcal{A}', k)$, onde



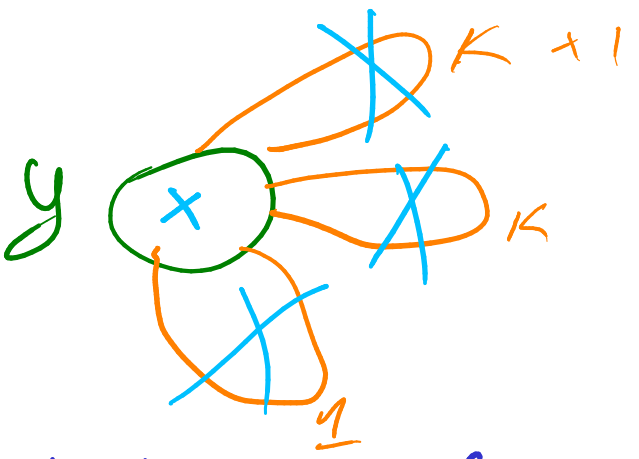
Núcleo para transversal mínima

Teorema

O problema da transversal mínima admite um núcleo com no máximo $d!k^d$ conjuntos e $d!k^d \cdot d^2$ elementos.

Redução HS.1: Se existe girassol $S = \{S_1, \ldots, S_{k+1}\}$ com centro Y, então devolva $(U', \mathcal{A}', k)$, onde

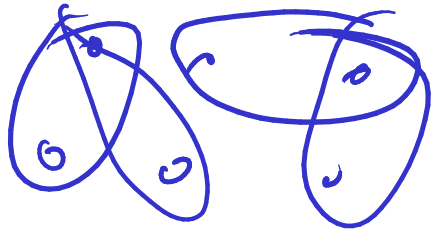
- $\mathcal{A}' := (\mathcal{A} \setminus S) \cup Y$;
- $U' := \bigcup_{X \in \mathcal{A}'} X$.



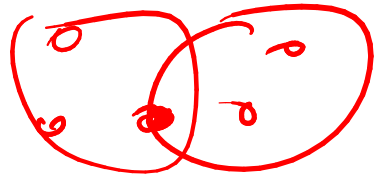
Red. HS.2. Se existe $K+1$ girassol com núcleo vazio, responda não!

• Seja $A_i = \{A \in \mathcal{A} : |A| = i\}$

A_2:



A_3:



$A = A_1 \cup A_2 \cup A_3 \cdots \cup A_d$

$\Downarrow$

$|A_i| \geq i!\,((k+1)-1)^i$

$\Downarrow$

Uso minha redução

$\Downarrow$

$|A_i| \leq i!\,((k+1)-1)^i$

Concluímos que: $|A_i| \leq i!(K)^i \quad \forall i$

$$\Rightarrow |A| = \sum_{i=1}^{d} A_i \leq \sum_{i=1}^{d} i!(K)^i$$

$$\leq \sum_{i=1}^{d} d!(K)^d = \underline{d \cdot d!(K)^d}$$

Conta o número de elementos:

$$\#\text{elementos} = \left| \bigcup_{A \in \mathcal{A}} A \right| \leq \sum_{A \in \mathcal{A}} |A| \leq \sum_{A \in \mathcal{A}} d \leq d^2 \cdot d! K^d$$