Algoritmos Parametrizados

Compressão Iterativa

Lehilton Pedrosa

Segundo Semestre de 2016

Instituto de Computação – Unicamp

Roteiro

1. Compressão iterativa
2. Conjunto de retroalimentação em torneios
3. Conjunto de vértices de retroalimentação

Compressão iterativa

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Objetivo: minimizar a quantidade de vértices

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Objetivo: minimizar a quantidade de vértices

Observação 2

Em alguns problemas:

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Objetivo: minimizar a quantidade de vértices

Observação 2

Em alguns problemas:

- obter uma solução não ótima é mais fácil

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Objetivo: minimizar a quantidade de vértices

Observação 2

Em alguns problemas:

- obter uma solução não ótima é mais fácil
- uma solução aproximada indica a estrutura da solução ótima

Problemas de minimização de vértices

Observação 1

Em muitos problemas, queremos:

- remover alguns vértices
- satisfazer alguma propriedade global

Objetivo: minimizar a quantidade de vértices

Observação 2

Em alguns problemas:

- obter uma solução não ótima é mais fácil
- uma solução aproximada indica a estrutura da solução ótima

Objetivo: converter a solução não ótima em ótima

Ilustrando

2-aproximação para cobertura por vértices

1. Resolva o PL(G) e obtenha x
2. $Z \leftarrow V_1 \cup V_{1/2}$
3. Devolva Z

Ilustrando

2-aproximação para cobertura por vértices

1. Resolva o PL(G) e obtenha x
2. $Z \leftarrow V_1 \cup V_{1/2}$
3. Devolva Z

Fato: o algoritmo acima devolve uma solução com no máximo $2\mathrm{OPT}$ vértices

Ilustrando

2-aproximação para cobertura por vértices

1. Resolva o PL(G) e obtenha x
2. $Z \leftarrow V_1 \cup V_{1/2}$
3. Devolva Z

Fato: o algoritmo acima devolve uma solução com no máximo $2\mathrm{OPT}$ vértices

Como é uma solução ótima?

Comprimindo a solução

COMPRIMIR-COBERTURA(G, k, Z):

Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não

Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$

Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:

Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:
   2.1 $A \leftarrow Z \setminus Z_X$

Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:
   - 2.1 $A \leftarrow Z \setminus Z_X$
   - 2.2 Se $G[A]$ tem arestas, tente próximo

Comprimindo a solução

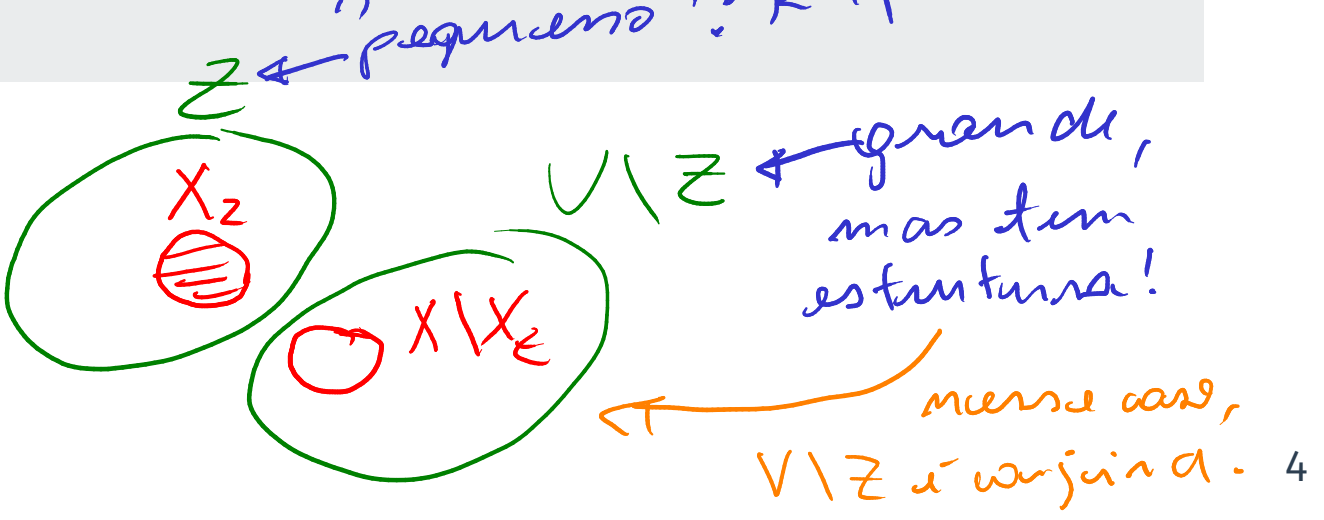
Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:
   - 2.1 $A \leftarrow Z \setminus Z_X$
   - 2.2 Se $G[A]$ tem arestas, tente próximo
   - 2.3 Se $|N(A)| + |Z_X| \leq k$, devolva $N(A) \cup Z_X$

Comprimindo a solução

COMPRIMIR-COBERTURA(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:
   - 2.1 $A \leftarrow Z \setminus Z_X$
   - 2.2 Se $G[A]$ tem arestas, tente próximo
   - 2.3 Se $|N(A)| + |Z_X| \leq k$, devolva $N(A) \cup Z_X$
3. Devolva não



Comprimindo a solução

Comprimir-Cobertura(G, k, Z):

1. Se $|Z| \geq k$, devolva não
2. Para cada $X_Z \subseteq Z$, com $|X_Z| \leq k$:
   - 2.1 $A \leftarrow Z \setminus Z_X$
   - 2.2 Se $G[A]$ tem arestas, tente próximo
   - 2.3 Se $|N(A)| + |Z_X| \leq k$, devolva $N(A) \cup Z_X$
3. Devolva não

Tempo de execução: $2^{|Z|} n^{\mathcal{O}(1)} = 4^k n^{\mathcal{O}(1)}$

Subproblema importante

Problema

Dados $k \geq 0$, um grafo G, uma cobertura por vértices Z e um subconjunto $X_Z \subseteq Z$, o grafo $G - X_Z$ contém uma cobertura por vértices de tamanho $k - |X_Z|$ que não contém vértices de $Z - X_Z$?

Subproblema importante

Problema

Dados $k \geq 0$, um grafo G, uma cobertura por vértices Z e um subconjunto $X_Z \subseteq Z$, o grafo $G - X_Z$ contém uma cobertura por vértices de tamanho $k - |X_Z|$ que não contém vértices de $Z - X_Z$?

Ou reescrevendo:

Subproblema importante

Problema

Dados $k \geq 0$, um grafo G, uma cobertura por vértices Z e um subconjunto $X_Z \subseteq Z$, o grafo $G - X_Z$ contém uma cobertura por vértices de tamanho $k - |X_Z|$ que não contém vértices de $Z - X_Z$?

Ou reescrevendo:

Problema da Cobertura por vértices Disjunto

Dados $k \geq 0$, um grafo G e um conjunto W, existe uma cobertura por vértices de G disjunta de W com no máximo k vértices?

Dificuldades

A estratégia tem algumas Dificuldades:

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida
2. Como obter uma solução aproximada melhor?

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida
2. Como obter uma solução aproximada melhor?

   $\Rightarrow$ nenhuma aproximação polinomial melhor que 1.3606 para Cobertura

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida
2. Como obter uma solução aproximada melhor?

   $\Rightarrow$ nenhuma aproximação polinomial melhor que 1.3606 para Cobertura

Ideia para melhorar:

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida
2. Como obter uma solução aproximada melhor?

   $\Rightarrow$ nenhuma aproximação polinomial melhor que 1.3606 para Cobertura

Ideia para melhorar:

- obter uma solução de custo $k + 1$

Dificuldades

A estratégia tem algumas Dificuldades:

1. O tempo de execução depende fortemente da aproximação obtida
2. Como obter uma solução aproximada melhor?

   $\Rightarrow$ nenhuma aproximação polinomial melhor que 1.3606 para Cobertura

Ideia para melhorar:

- obter uma solução de custo $k + 1$
- construir solução iterativamente

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, \ldots, i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, ..., i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
    - $X_i \leftarrow V_i$

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, ..., i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
   - $X_i \leftarrow V_i$
4. Para cada $i = k + 1, \ldots, n$:
   - $X'_i \leftarrow X_{i-1} \cup \{v_i\}$

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, ..., i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
   - $X_i \leftarrow V_i$
4. Para cada $i = k + 1, \ldots, n$:
   - $X'_i \leftarrow X_{i-1} \cup \{v_i\}$
   - Comprima X'_i e obtenha X_i

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, ..., i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
   - $X_i \leftarrow V_i$
4. Para cada $i = k + 1, \ldots, n$:
   - $X'_i \leftarrow X_{i-1} \cup \{v_i\}$
   - Comprima X'_i e obtenha X_i
   - Se a compressão falhar, devolva não

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, \ldots, i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
   - $X_i \leftarrow V_i$
4. Para cada $i = k+1, \ldots, n$:
   - $X'_i \leftarrow X_{i-1} \cup \{v_i\}$
   - Comprima X'_i e obtenha X_i
   - Se a compressão falhar, devolva não
5. Devolva X_n

é fácil obter uma sol de tamanho $\leq k+1$ para G_i a partir de uma sol de tamanho k para G_{i-1}

Compressão iterativa

Cobertura-Iterativa(G, k):

1. Ordene arbitrariamente $V = \{v_1, \ldots, v_n\}$
2. Defina $V_i = \{1, ..., i\}$ e $G_i = G[V_i]$, para $1 \leq i \leq n$
3. Para cada $i = 1, \ldots, k$:
   - $X_i \leftarrow V_i$
4. Para cada $i = k + 1, \ldots, n$:
   - $X'_i \leftarrow X_{i-1} \cup \{v_i\}$
   - Comprima X'_i e obtenha X_i
   - Se a compressão falhar, devolva não
5. Devolva X_n

O algoritmo mantém a invariante:

- X_i é cobertura por vértice de G_i

Framework: Problema de Compressão

Consideramos um problema $(*)$ cuja instância é (G, k)

Compressão-$(*)$

Dada uma instância (G, k) de $(*)$ e uma solução $Z \subseteq V(G)$ com $|Z| \leq k + 1$:

- encontre uma solução de tamanho k para (G, k), ou
- dê um certificado de que não há solução.

Framework: Problema de Compressão

Consideramos um problema $(*)$ cuja instância é (G, k)

Compressão-$(*)$

Dada uma instância (G, k) de $(*)$ e uma solução $Z \subseteq V(G)$ com $|Z| \leq k + 1$:

- encontre uma solução de tamanho k para (G, k), ou
- dê um certificado de que não há solução.

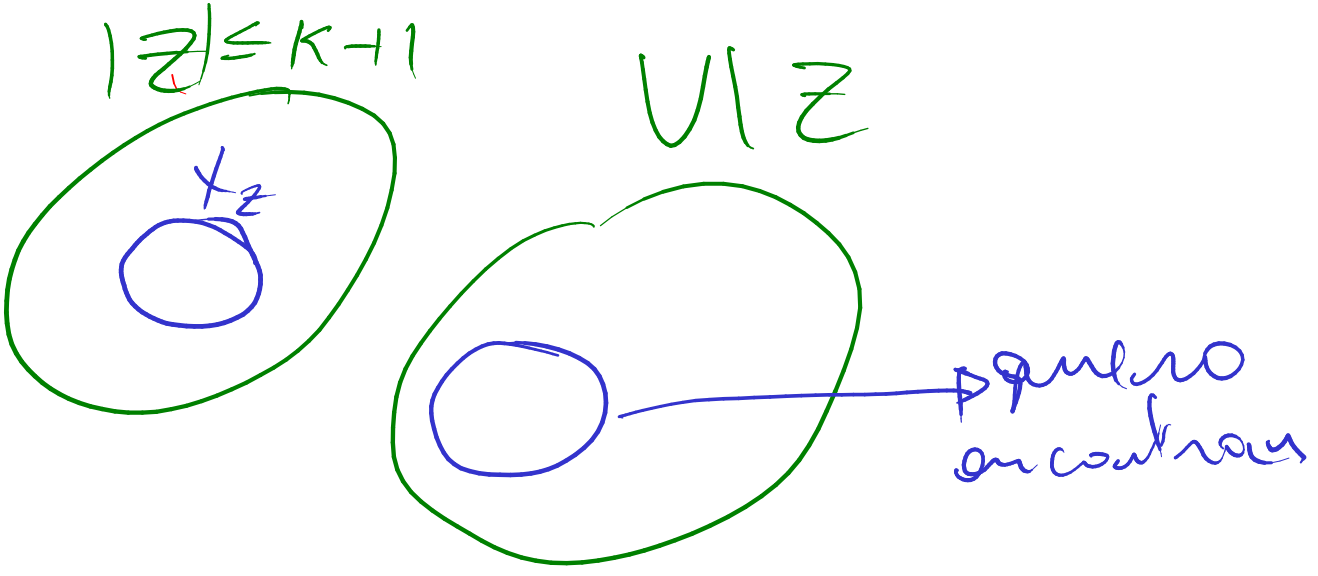
$\Rightarrow$ Se Compressão-$(*)$ tem algoritmo com tempo $f(k)n^c$, então $(*)$ tem algoritmo com tempo $f(k)n^{c+1}$.

Framework: Problema de Solução Disjunta

$(*)$-Disjunto

Considere uma instância (G, k) de $(*)$, uma solução $Z \subseteq V(G)$ com $|Z| \leq k + 1$ e $X_Z \subseteq Z$:

- encontre uma solução X com $|X| \leq k - |X_Z|$ para $G - X_Z$ tal que X e $W := Z \setminus X_Z$ sejam disjuntos;
- dê um certificado de que não uma tal solução.



Framework: Problema de Solução Disjunta

$(*)$-Disjunto

Considere uma instância (G, k) de $(*)$, uma solução $Z \subseteq V(G)$ com $|Z| \leq k+1$ e $X_Z \subseteq Z$:

- encontre uma solução X com $|X| \leq k - |X_Z|$ para $G - X_Z$ tal que X e $W := Z \setminus X_Z$ sejam disjuntos;
- dê um certificado de que não uma tal solução.

$\Rightarrow$ Se $(*)$-Disjunto tem algoritmo com tempo $g(k)n^{\mathcal{O}(1)}$, então Compressão-$(*)$ tem algoritmo com tempo

$$\sum_{i=0}^{k} \binom{k+1}{i} g(k-1)n^{\mathcal{O}(1)}.$$

Framework: Problema de Solução Disjunta

$(*)$-Disjunto

Considere uma instância (G, k) de $(*)$, uma solução $Z \subseteq V(G)$ com $|Z| \leq k+1$ e $X_Z \subseteq Z$:

- encontre uma solução X com $|X| \leq k - |X_Z|$ para $G - X_Z$ tal que X e $W := Z \setminus X_Z$ sejam disjuntos;
- dê um certificado de que não uma tal solução.

$\Rightarrow$ Se $(*)$-Disjunto tem algoritmo com tempo $g(k)n^{\mathcal{O}(1)}$, então Compressão-$(*)$ tem algoritmo com tempo

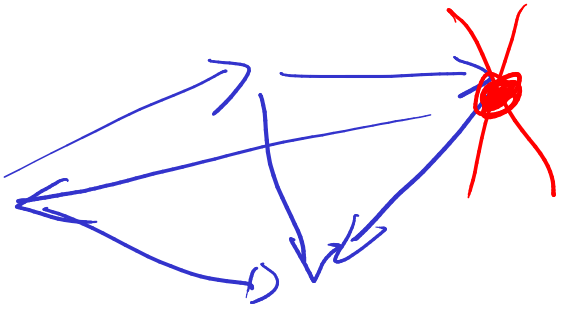
$$\sum_{i=0}^{k} \binom{k+1}{i} g(k-1)n^{\mathcal{O}(1)}.$$

$\Rightarrow$ Se $g(k) = \alpha^k$, então o tempo total é $(1+\alpha)^k n^{\mathcal{O}(1)}$.

$1^k \Rightarrow 1$ $\quad (1+1)^k n^{O(1)}$

Conjunto de retroalimentação em torneios

Conjunto de retroalimentação em torneios



Problema de vértices de retroalimentação em torneios (FVST)

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

Aplicando framework: usando compressão

Lema

Se existe um algoritmo para Compressão do Conjunto de retroalimentação em torneios em tempo $f(k)n^c$, então existe algoritmo para Conjunto de retroalimentação em torneios em tempo $f(k)n^{c+1}$.

Comprimir

Prova: Algoritmo:

1) Obtenha uma ordenação de $v_1 v_2 v_3 \dots v_n$

2) Seja $G_i = G[\{v_1, v_2, \dots, v_i\}]$

3) Defina $X_i = V(G_i)$, $i \leq k$

4) Para cada $i = k+1 \dots n$:

a) Seja $X'_i = X_{i-1} \cup \{v_i\}$ // $|X'_i| = k+1$ e cobre G_i

b) $X_i =$ Comprimir(G_i, k, X'_i)

Recurso Comprimir $\leq n$ vezes.

Aplicando framework: usando compressão disjunta

Conjunto de retroalimentação em torneios disjunto

Dados T, $W \subseteq V(T)$ um conjunto de retroalimentação de T e $k \geq 0$, existe um conjunto de retroalimentação $X \subseteq A$, com tamanho até k, onde $A := V(T) \setminus W$?

Lema

Se existe um algoritmo para Conjunto de retroalimentação em torneios disjunto em tempo $g(k)n^c$, então existe algoritmo para Compressão de Conjunto de retroalimentação em torneios em tempo $\sum_{i=0}^{k} \binom{k+1}{i} g(k-i)n^{\mathcal{O}(1)}$.

Prove: Algoritmo (T, K, Z)

$|Z \setminus X_Z| = |Z| - |X_Z| = K+1-|X_Z|$

1) Para cada $X_Z \subseteq Z$:

a) $R = AlgDisj(T - X_Z, Z \setminus X_Z, K-|X_Z|)$

executa no máximo $g(K-|X_Z|)$

b) Se R for não, continuo

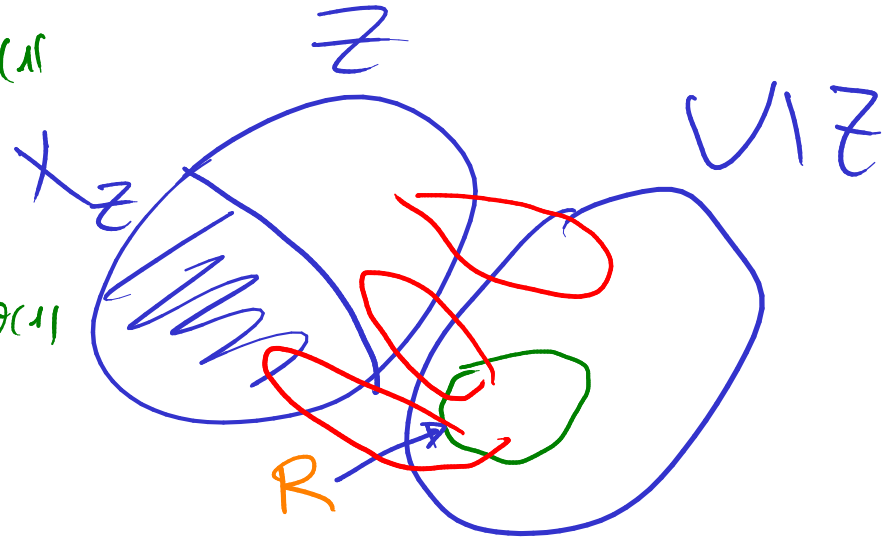
c) Se R for solução, devolvo $X_Z \cup R$.

2) Responda Não.

□

$$Tempo = \sum_{X_Z \subseteq Z} g(K-|X_Z|) \cdot n^{O(1)}$$

$$= \sum_{i=0}^{K+1} \binom{K+1}{i} g(K-i) \cdot n^{O(1)}$$



Conjunto de retroalimentação em torneios disjunto

Lema

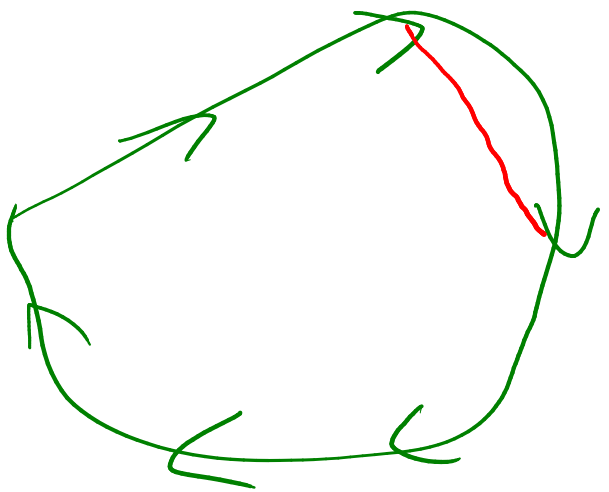
Seja T um torneio. Então

Conjunto de retroalimentação em torneios disjunto

Lema

Seja T um torneio. Então

1. *T é cíclico sss T tem um triângulo;*



Conjunto de retroalimentação em torneios disjunto

Lema

Seja T um torneio. Então

1. *T é cíclico sss T tem um triângulo;*
2. *Existe uma única ordenação topológica de T,* ↝ não existe.

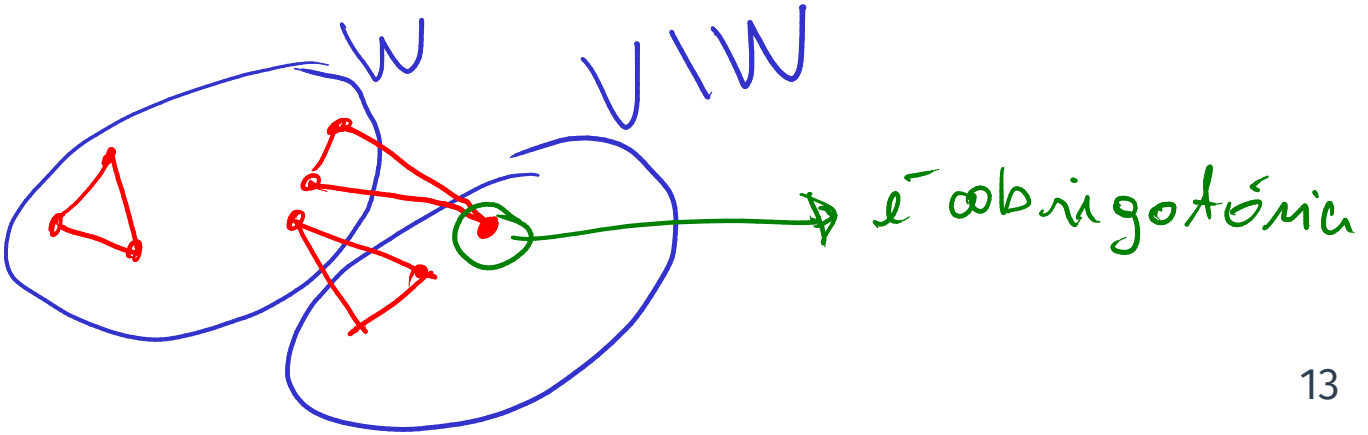
Conjunto de retroalimentação em torneios disjunto

Lema

Seja T um torneio. Então

1. *T é cíclico sss T tem um triângulo;*
2. *Existe uma única ordenação topológica de T.*

Observação: dizemos que um torneio T é transitivo se para todo u, v, w, $(u, w), (w, v) \in E(T)$, então $(u, v) \in E(T)$.



Conjunto de retroalimentação em torneios disjunto

$A = V \setminus W$

Lema

Seja T um torneio. Então

1. *T é cíclico sss T tem um triângulo;*
2. *Existe uma única ordenação topológica de T.*

Observação: dizemos que um torneio T é transitivo se para todo u, v, w, $(u, w), (w, v) \in E(T)$, então $(u, v) \in E(T)$.

Redução FVST.1: Se T contém um triângulo direcionado x, y, z, e somente $z \in A$, então devolva $(T - z, W, k - 1)$ e indica z na solução.

Ordenações

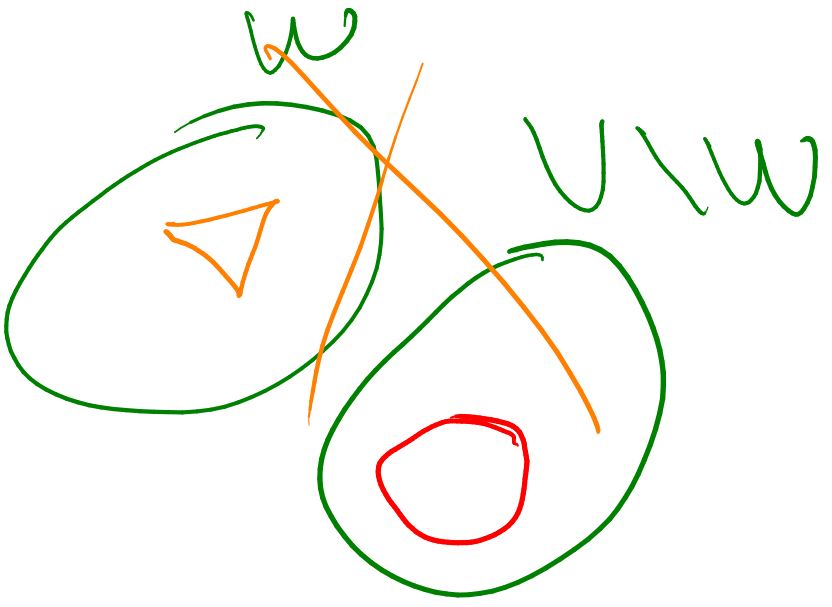
Definimos as seguintes ordenações:

Ordenações

Definimos as seguintes ordenações:

Suponhamos q W não tem triângulo, do contrário NÃO.

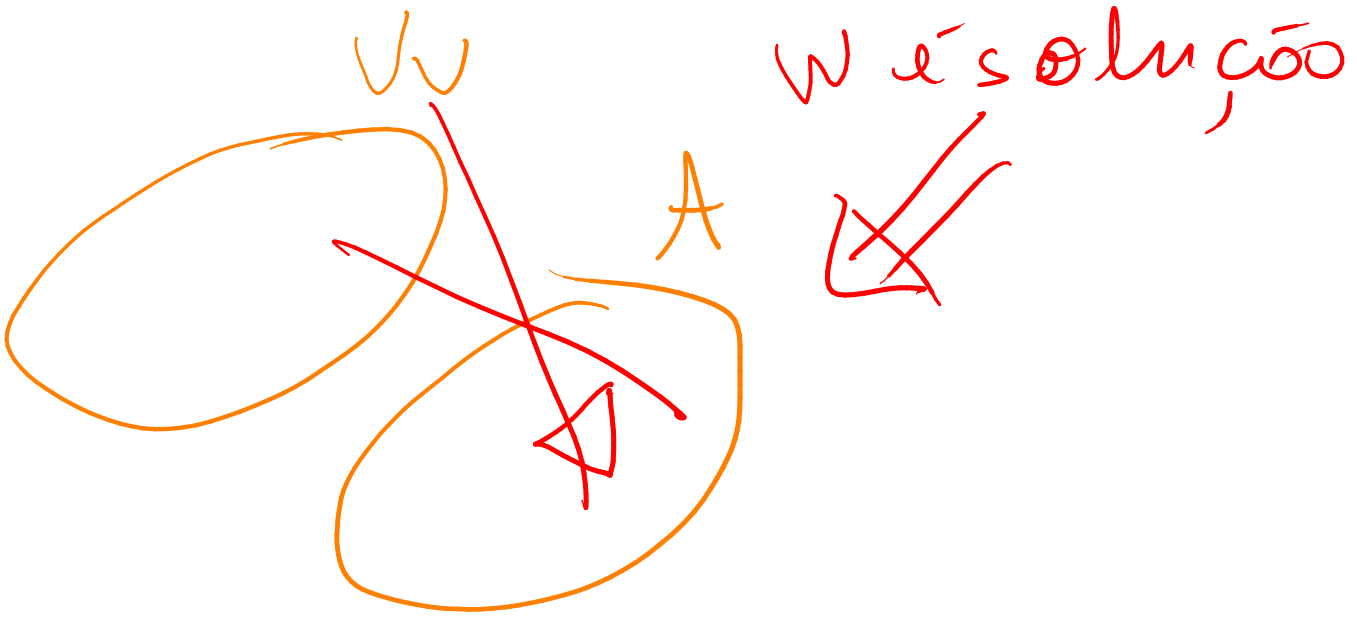
- $\sigma = (w_1, \ldots, w_q)$ é a ordenação de W, onde $q = |W|$



Ordenações

Definimos as seguintes ordenações:

- $\sigma = (w_1, \ldots, w_q)$ é a ordenação de W, onde $q = |W|$
- $\rho = (u_1 \ldots, u_p)$ é a ordenação de A, onde $p = |A|$



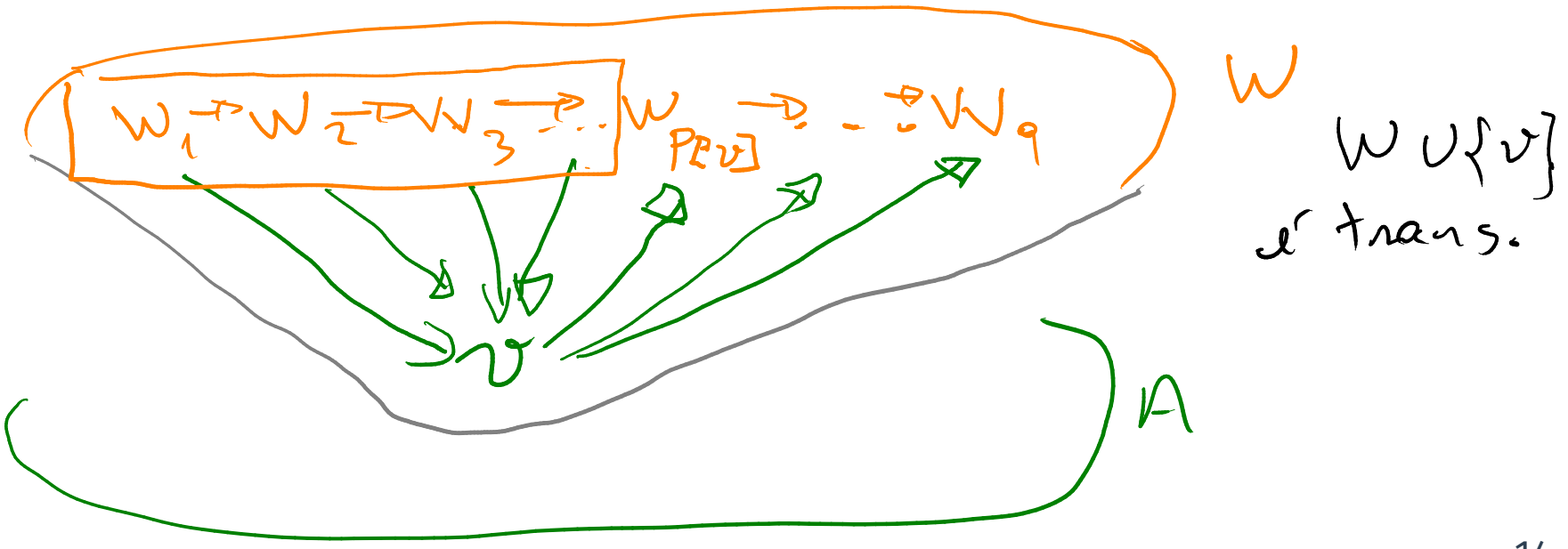
Ordenações

Definimos as seguintes ordenações:

- $\sigma = (w_1, \ldots, w_q)$ é a ordenação de W, onde $q = |W|$
- $\rho = (u_1 \ldots, u_p)$ é a ordenação de A, onde $p = |A|$

Seja $v \in A$, a posição natural de v em σ é $p[v]$ definido por

$$(v, w_i) \in E(T) \Leftrightarrow i \geq p[v]$$



Ordenações

Definimos as seguintes ordenações:

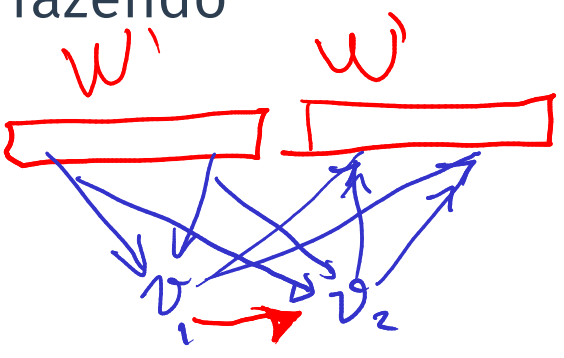
- $\sigma = (w_1, \ldots, w_q)$ é a ordenação de W, onde $q = |W|$
- $\rho = (u_1 \ldots, u_p)$ é a ordenação de A, onde $p = |A|$

Seja $v \in A$, a posição natural de v em σ é $p[v]$ definido por

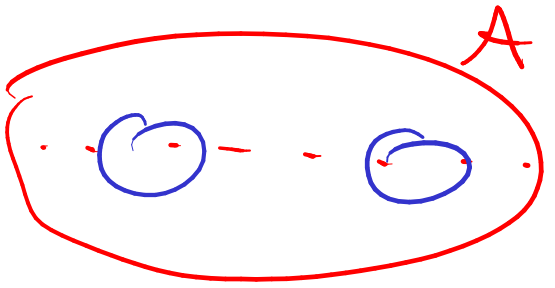
$$(v, w_i) \in E(T) \Leftrightarrow i \geq p[v]$$

Definimos π como uma outra ordenação de A fazendo seguinte:

- ordene os vértices v de A por $p[v]$;
- se houver empate, ordene como em ρ.

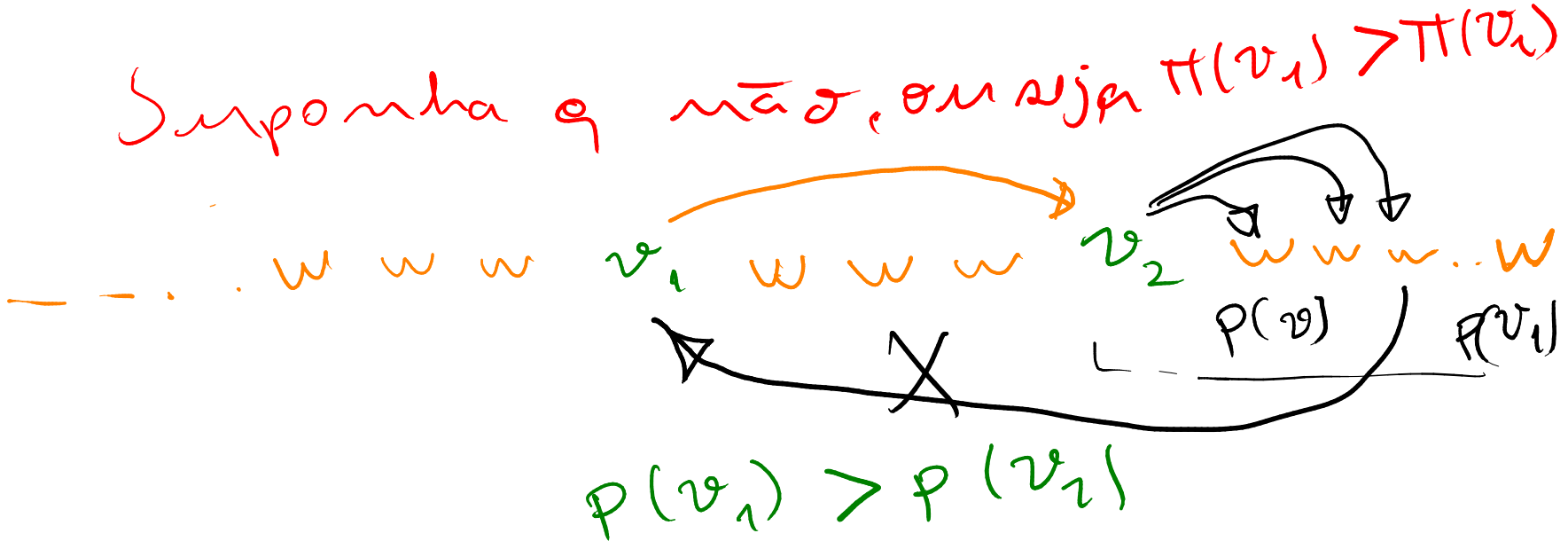


Reduzindo para subsequência comum máxima



Observações:

1. Se $T - X$ é transitivo para $X \subseteq A$, então uma ordenação topológica de $T[A \setminus X]$ respeita π
2. Qualquer ordenação $T[A \setminus X]$ deve respeitar ρ.

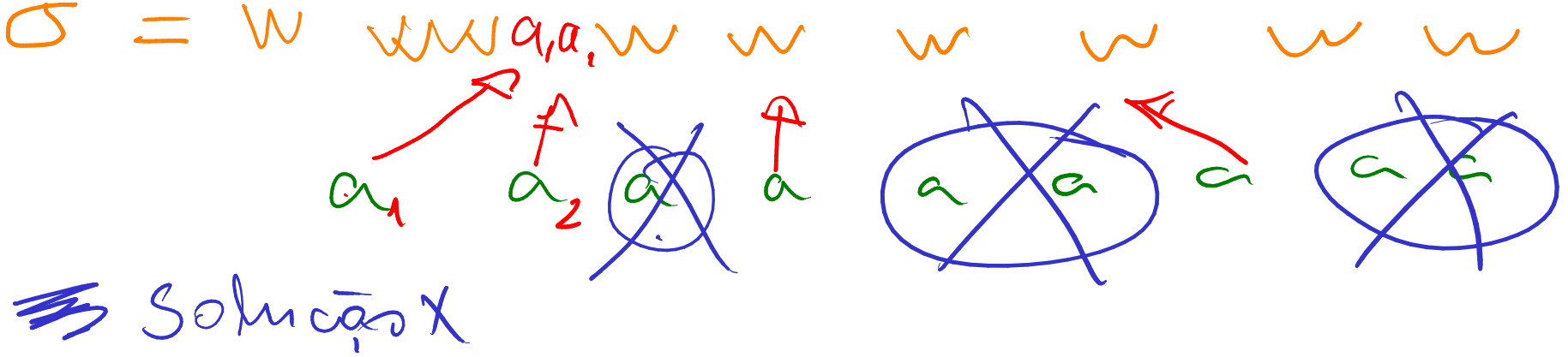


Reduzindo para subsequência comum máxima

Observações:

1. Se $T - X$ é transitivo para $X \subseteq A$, então uma ordenação topológica de $T[A \setminus X]$ respeita π
2. Qualquer ordenação $T[A \setminus X]$ deve respeitar ρ.

De outra forma: Quantos vértices de A conseguimos adicionar a σ sem criar ciclos?



Reduzindo para subsequência comum máxima

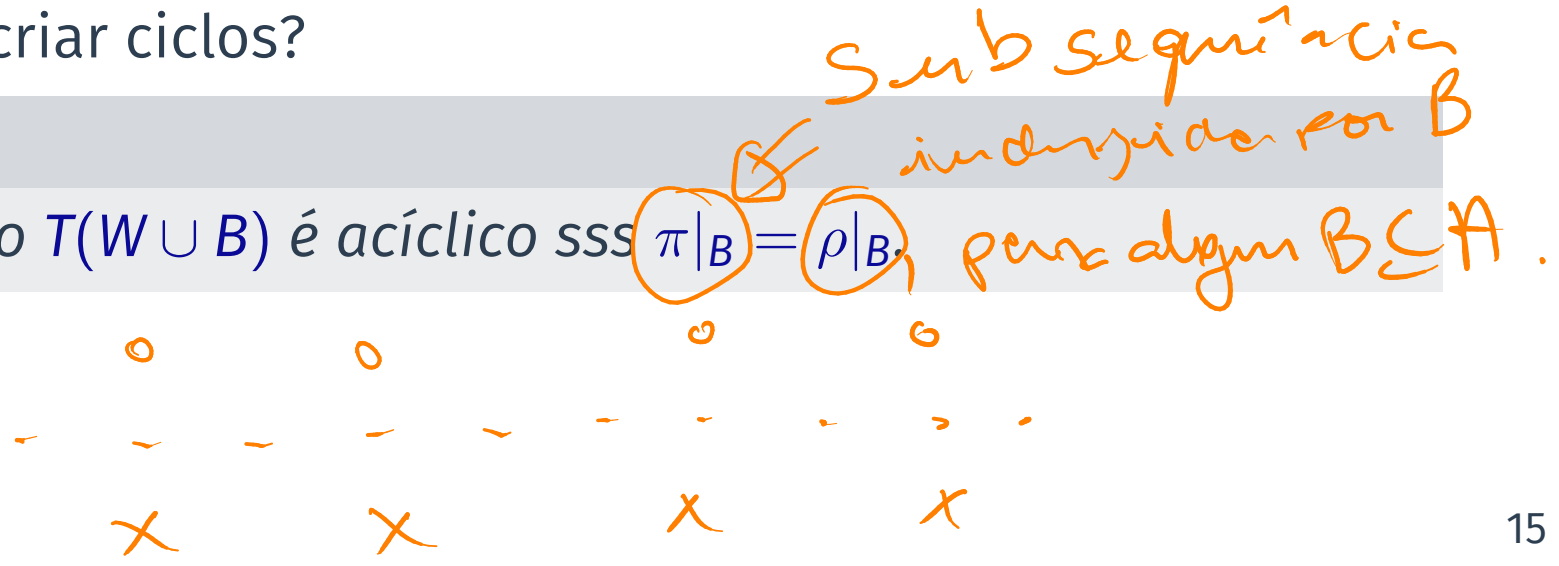
Observações:

1. Se $T - X$ é transitivo para $X \subseteq A$, então uma ordenação topológica de $T[A \setminus X]$ respeita π
2. Qualquer ordenação $T[A \setminus X]$ deve respeitar ρ.

De outra forma: Quantos vértices de A conseguimos adicionar a σ sem criar ciclos?

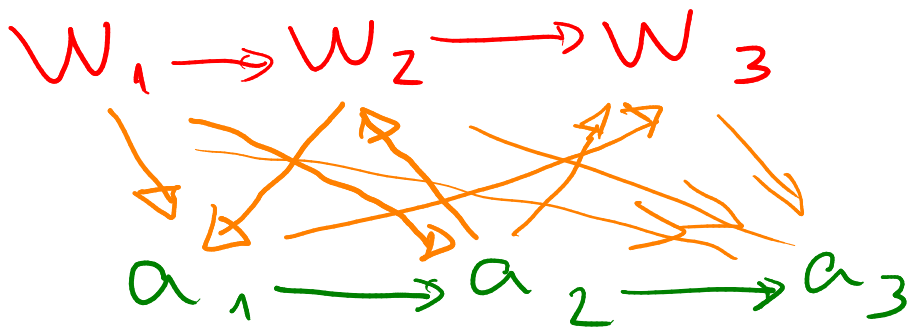
Lema

O torneio $T(W \cup B)$ é acíclico sss $\pi|_B = \rho|_B$.



- Reduzimos o problema em encontrar um conjunto B de tamanho máximo tq. $\pi|_B = \rho|_B$.

$\pi = a_2\ a_1\ a_3$
$\rho = a_1\ a_2\ a_3$
$\Rightarrow a_1 a_3,\ a_2 a_3$



$\rho(a_1) = 3$
$\rho(a_2) = 2$
$\rho(a_3) = 4$
$\Rightarrow$ não podemos inserir a_1 e a_2 em C pois não respeitaria π.

Combinando tudo

Lema

Subsequência comum máxima pode ser resolvida em tempo polinomial.

Combinando tudo

Lema

Subsequência comum máxima pode ser resolvida em tempo polinomial.

Lema

Conjunto de retroalimentação em torneios disjunto é polinomial.

Combinando tudo

Lema

Subsequência comum máxima pode ser resolvida em tempo polinomial.

Lema

Conjunto de retroalimentação em torneios disjunto é polinomial.

Teorema

Conjunto de retroalimentação em torneios pode ser resolvido em tempo $2^k n^{\mathcal{O}(1)}$.

Conjunto de vértices de retroalimentação

Problema

Conjunto de vértices de retroalimentação

de retroalimentação

Dado G e $k \geq 0$, existe um conjunto de vértices $X \subseteq V(G)$ tal que $|X| \leq k$?

Problema

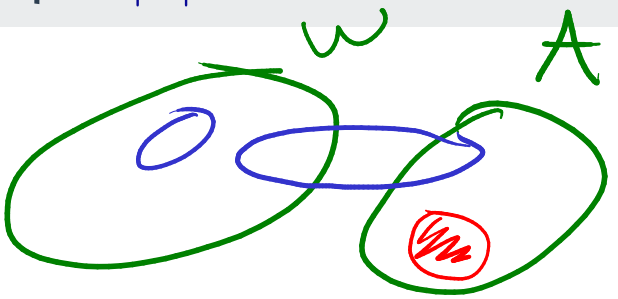
Conjunto de vértices de retroalimentação

Dado G e $k \geq 0$, existe um conjunto de vértices $X \subseteq V(G)$ tal que $|X| \leq k$?

Framework de compressão iterativa: podemos nos concentrar na versão disjunta.

Conjunto de vértices de retroalimentação disjunto

Dado G e $k \geq 0$ e $W \subseteq V(G)$, um conjunto de retroalimentação de de tamanho $k+1$, existe um conjunto de vértices $X \subseteq V(G) \setminus W$ tal que $|X| \leq k$?



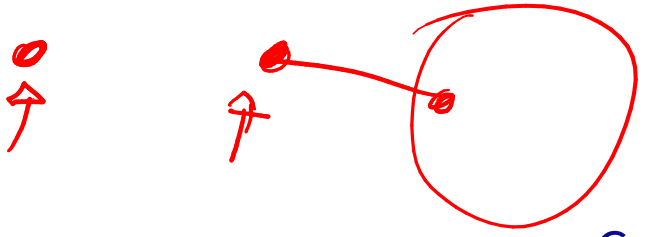
Reduções

Reduções

Seja $H = G - W$:

Reduções

Seja $H = G - W$:



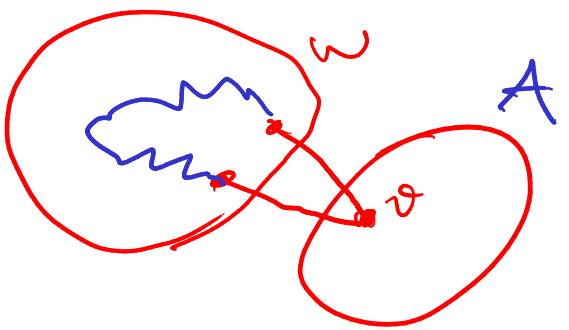
Redução FVS.1: Remova vértices com grau zero ou um em G.

Reduções

Seja $H = G - W$:

Redução FVS.1: Remova vértices com grau zero ou um em G.

Redução FVS.2: Se existe um vértice $v \in V(H)$ tal que $G[W \cup \{v\}]$ é cíclico, então:



Reduções

Seja $H = G - W$:

Redução FVS.1: Remova vértices com grau zero ou um em G.

Redução FVS.2: Se existe um vértice $v \in V(H)$ talq eu $G[W \cup \{v\}]$ é cíclico, então:

- inclua v na solução,

Reduções

Seja $H = G - W$:

Redução FVS.1: Remova vértices com grau zero ou um em G.

Redução FVS.2: Se existe um vértice $v \in V(H)$ talq eu $G[W \cup \{v\}]$ é cíclico, então:

- inclua v na solução,
- remova v de H e

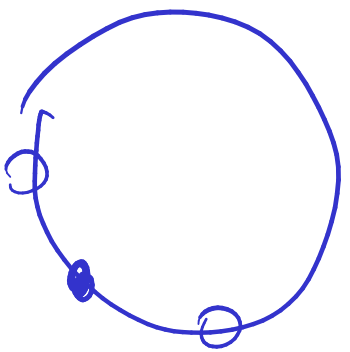
Reduções

Seja $H = G - W$:

Redução FVS.1: Remova vértices com grau zero ou um em G.

Redução FVS.2: Se existe um vértice $v \in V(H)$ talq eu $G[W \cup \{v\}]$ é cíclico, então:

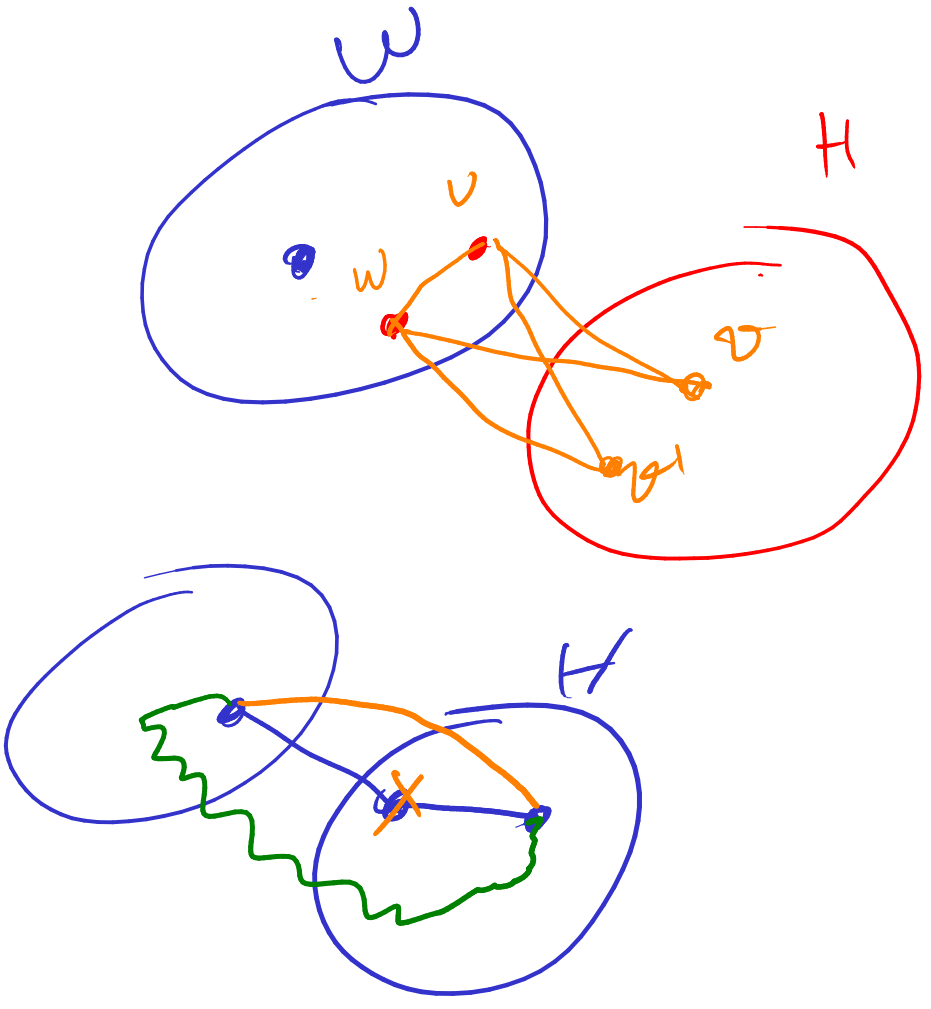
- inclua v na solução,
- remova v de H e
- diminua o parâmetro k.



Devolva a instância $(G - v, W, k - 1)$.

Mais uma redução

Redução FVS.3: Seja um vértice v com grau 2 em G, tal que $N(v) = \{u, w\}$.



Mais uma redução

Redução FVS.3: Seja um vértice v com grau 2 em G, tal que $N(v) = \{u, w\}$. Se $u \in V(H)$ ou $w \in V(H)$, então:

Mais uma redução

Redução FVS.3: Seja um vértice v com grau 2 em G, tal que $N(v) = \{u, w\}$. Se $u \in V(H)$ ou $w \in V(H)$, então:

- remova v

Mais uma redução

Redução FVS.3: Seja um vértice v com grau 2 em G, tal que $N(v) = \{u, w\}$. Se $u \in V(H)$ ou $w \in V(H)$, então:

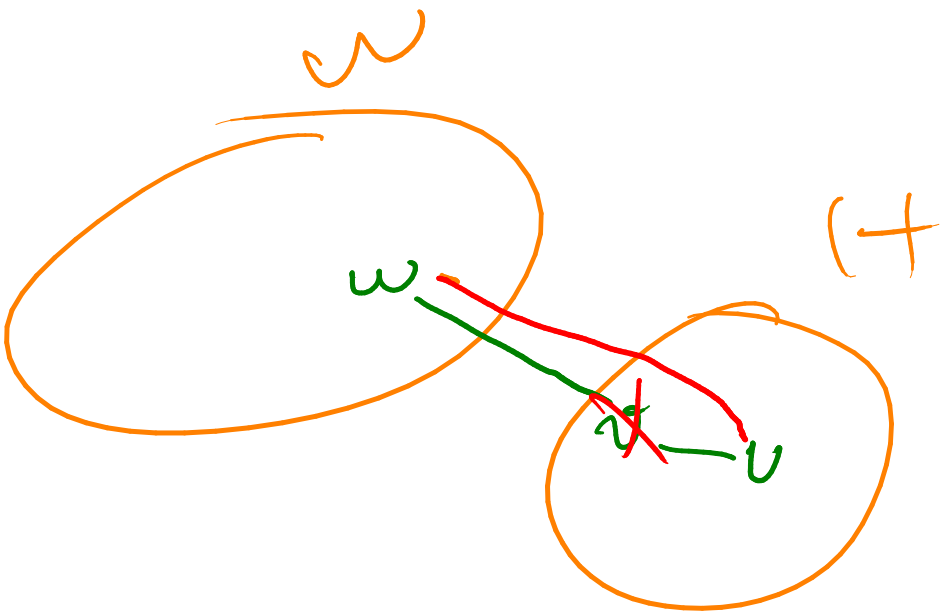
- remova v
- adicione uma aresta conectando u a v.

Mais uma redução

Redução FVS.3: Seja um vértice v com grau 2 em G, tal que $N(v) = \{u, w\}$. Se $u \in V(H)$ ou $w \in V(H)$, então:

- remova v
- adicione uma aresta conectando u a w

Atenção: o grafo pode passar a ser um multigrafo.



Resolvendo a versão disjunta

Lema

Conjunto de vértices de retroalimentação disjunto pode ser resolvido em tempo $4^k n^{\mathcal{O}(1)}$.

1) Se $G[W]$ contém ciclo, responde NÃO!

Agora presumimos que $G[W]$ é acíclico

a) aplicamos FVS. $\{1, 2, 3\}$, e obtemos (G, W, k)

Se $k < 0$, responde não.

→ Como W é ger. direto.

⇒ H é floresta

⇒ Seja $x \in V(H)$ com grau 1 ou 0.

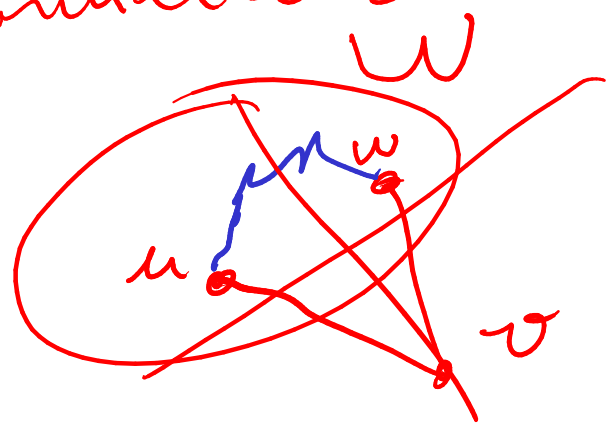
⇒ Como aplicamos FVS.1,

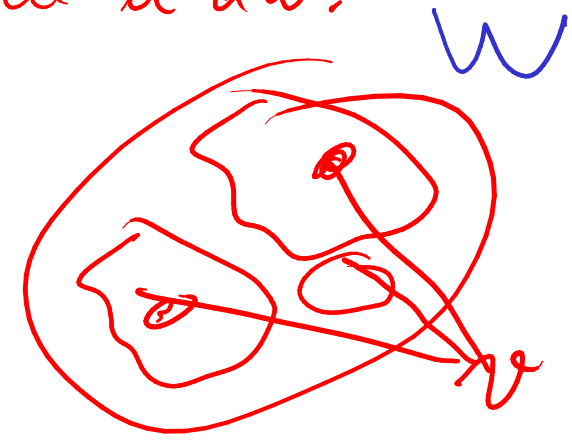
sabemos que x tem vizinho em W.

⇐ não pode acontecer

⇒ Como aplicamos FVS.3, x tem 2 vizinhos em W.

⇒ Mas aplicamos FVS.2, então não existe caminho em W de u a w.

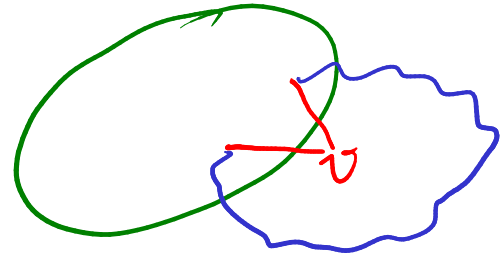




Algoritmo de ramificação:

1) Resolvemos $(G-x, W, K-1)$ ←

2) Resolvemos $(G, W \cup \{x\}, K)$ ←



Tempo:

ramos: 2

altura: usamos uma medida.

$$\mu(I) = K + \gamma(W)$$

1) $\mu(I') = K' + \gamma(I') = (K-1) + \gamma(I)$

$= \mu(I) - 1$

2) $\mu(I') = K' + \gamma(I') \leq K + \gamma(I) - 1$

$= \mu(I) - 1$

$\Rightarrow$ altura $\leq \mu(I) \Rightarrow$ Tempo: $2^{\mu(I)}$

Tempo de execução: $2^{M(I)} n^{\Theta(1)} \leq$

$2^{K+|W|} n^{\Theta(1)} \leq$

$2^{K+K-1} n^{\Theta(1)} =$

$2^{2K} n^{\Theta(1)} = 4^K n^{\Theta(1)}$.

Resolvendo a versão disjunta

Lema

Conjunto de vértices de retroalimentação disjunto pode ser resolvido em tempo $4^k n^{\mathcal{O}(1)}$.

Utilizando o framework de compressão iterativa:

Teorema

Conjunto de vértices de retroalimentação pode ser resolvido em tempo $4^k n^{\mathcal{O}(1)}$.

$5^k n^{\theta(1)}$