Complexité paramétrée (4)

Techniques algorithmiques

Christophe PAUL
(CNRS - LIRMM)

13 novembre 2014

Arbre de recherche bornée

- Vertex Cover
- Règles de branchement

Compression itérative

- Vertex Cover
- Feedback Vertex Set

Algorithmes randomisés

- Feedback Vertex Set
- Color Coding - Longest Path

Ensembles et séparateurs importants

- Multicoupe dans les graphes

Vertex Cover (1)

Règle de branchement : Soit (G, k) une instance de Vertex Cover telle que $k > 0$ et $\exists (u, v) \in E(G)$, alors brancher sur :

1. $(G - u, k - 1)$
2. $(G - v, k - 1)$

Vertex Cover (1)

Règle de branchement : Soit (G, k) une instance de Vertex Cover telle que $k > 0$ et $\exists (u, v) \in E(G)$, alors brancher sur :

1. $(G - u, k - 1)$
2. $(G - v, k - 1)$

Règle de réduction : Soit (G, k) une instance de Vertex Cover.

1. Si $k = 0$ et $E(G) \neq \emptyset$, retourner Faux
2. Si $E(G) = \emptyset$, retourner Vrai

Théorème : Vertex Cover (paramétré par la taille de la solution) admet un algorithme de recherche bornée de complexité $2^k \cdot (n + m)$.

Vertex Cover (1)

Règle de branchement : Soit (G, k) une instance de Vertex Cover telle que $k > 0$ et $\exists (u, v) \in E(G)$, alors brancher sur :

1. $(G - u, k - 1)$
2. $(G - v, k - 1)$

Règle de réduction : Soit (G, k) une instance de Vertex Cover.

1. Si $k = 0$ et $E(G) \neq \emptyset$, retourner Faux
2. Si $E(G) = \emptyset$, retourner Vrai

Théorème : Vertex Cover (paramétré par la taille de la solution) admet un algorithme de recherche bornée de complexité $2^k \cdot (n + m)$.

Peut-on faire mieux ? (Diminuer la taille de l'arbre de branchement)

Vertex Cover (2)

Règle de réduction [Sunflower] : Soient (G, k) une instance de Vertex Cover et x un sommet de degré $d(x) > k$, alors réduire vers $(G - x, k - 1)$

Vertex Cover (2)

Règle de réduction [Sunflower] : Soient (G, k) une instance de Vertex Cover et x un sommet de degré $d(x) > k$, alors réduire vers $(G - x, k - 1)$

Règle de branchement (2) : Soient (G, k) une instance de Vertex Cover et x un sommet de degré $d(x) \geqslant 1$, alors brancher sur

1. $(G - x, k - 1)$ – x est sélectionné dans la solution
2. $(G - N[x], k - d(x))$ – $N(x)$ est sélectionné dans la solution

Vertex Cover (2)

Règle de réduction [Sunflower] : Soient (G, k) une instance de Vertex Cover et x un sommet de degré $d(x) > k$, alors réduire vers $(G - x, k - 1)$

Règle de branchement (2) : Soient (G, k) une instance de Vertex Cover et x un sommet de degré $d(x) \geqslant 1$, alors brancher sur

1. $(G - x, k - 1)$ – x est sélectionné dans la solution
2. $(G - N[x], k - d(x))$ – $N(x)$ est sélectionné dans la solution

Observation :

- Si on peut garantir l'existence d'un sommet de "grand" degré, alors une branche de l'arbre sera plus courte.

Vertex Cover (3)

Observation : Le problème Vertex Cover est polynomial sur les graphes de degré maximum 2.

Vertex Cover (3)

Observation : Le problème Vertex Cover est polynomial sur les graphes de degré maximum 2.

Algorithme Vertex Cover(G, k)

- Si G contient un sommet x tel que $d(x) \geqslant 3$, alors appliquer la règle de branchement (2)
- Sinon résoudre Vertex Cover en temps polynomial.

Vertex Cover (3)

Observation : Le problème Vertex Cover est polynomial sur les graphes de degré maximum 2.

Algorithme Vertex Cover(G, k)

- Si G contient un sommet x tel que $d(x) \geqslant 3$, alors appliquer la règle de branchement (2)
- Sinon résoudre Vertex Cover en temps polynomial.

Quelle est la taille de l'arbre de branchement ?

Vertex Cover (4) – analyse de l'arbre de branchement

$$T_{vc}(k+3) \geqslant T_{vc}(k+2) + T_{vc}(k) + 1 \quad (1)$$

$$T_{vc}(1) = T_{vc}(2) = 1 \qquad T_{vc}(0) = 0$$

Vertex Cover (4) – analyse de l'arbre de branchement

$$T_{vc}(k+3) \geqslant T_{vc}(k+2) + T_{vc}(k) + 1 \quad (1)$$

$$T_{vc}(1) = T_{vc}(2) = 1 \qquad T_{vc}(0) = 0$$

En fixant $T_{vc}(k) = c^k - 1$, nous devons résoudre

$$c^3 = c^2 + 1 \quad (2)$$

Vertex Cover (4) – analyse de l'arbre de branchement

$$T_{vc}(k+3) \geqslant T_{vc}(k+2) + T_{vc}(k) + 1 \quad (1)$$

$$T_{vc}(1) = T_{vc}(2) = 1 \qquad T_{vc}(0) = 0$$

En fixant $T_{vc}(k) = c^k - 1$, nous devons résoudre

$$c^3 = c^2 + 1 \quad (2)$$

On prend la plus petite racine positive, ici $c = 5^{\frac{1}{4}} \leqslant 1,47$

Théorème : Vertex Cover (paramétré par la taille de la solution) admet un algorithme de recherche bornée de complexité $1,47^k \cdot (n+m)$.

Règles de branchement

Une règle de branchement pour un problème paramétré (P, κ) est un algorithme polynomial qui étant donnée une instance $(x, \kappa(x))$, $(k > 1)$ retourne un ensemble $\mathcal{I} = \{(x_1, \kappa(x_1)), \ldots, (x_c, \kappa(x_\ell))\}$ de $\ell > 0$ instances telles que

1. $\forall i \in [\ell]$, $|x_i| \leqslant |x|$ et $k_i < k$ et
2. $(x, \kappa(x)) \in (P, \kappa) \iff \exists i \in [\ell],\ (x_i, \kappa(x_i)) \in (P, \kappa)$

Règles de branchement

Une règle de branchement pour un problème paramétré (P, κ) est un algorithme polynomial qui étant donnée une instance $(x, \kappa(x))$, $(k > 1)$ retourne un ensemble $\mathcal{I} = \{(x_1, \kappa(x_1)), \ldots, (x_c, \kappa(x_\ell))\}$ de $\ell > 0$ instances telles que

1. $\forall i \in [\ell]$, $|x_i| \leqslant |x|$ et $k_i < k$ et
2. $(x, \kappa(x)) \in (P, \kappa) \iff \exists i \in [\ell],\ (x_i, \kappa(x_i)) \in (P, \kappa)$

A chaque règle de branchement est associé un vecteur de branchement (ordonné par valeurs décroissantes)

$$(v_1 = \kappa(x) - \kappa(x_1), \ldots, v_\ell = \kappa(x) - \kappa(x_\ell))$$

Lemme : Une règle de branchement de vecteur $(v_1, \ldots, v_\ell)$ développe un arbre de recherche de taille c^k, où c est la plus petite racine positive de

$$a^k = a^{k-v_1} + \cdots + a^{k-v_\ell}$$

(Étrange) Vertex Cover

Etant donné un graphe G paramétré par $k \in \mathbb{N}$, G admet-il un vertex cover de taille 2^k ?

(Étrange) Vertex Cover

Etant donné un graphe G paramétré par $k \in \mathbb{N}$, G admet-il un vertex cover de taille 2^k ?

Observation : Si le problème (Étrange) Vertex Cover peut être résolu par une règle de branchement, alors **P** = **NP**

(Étrange) Vertex Cover

Etant donné un graphe G paramétré par $k \in \mathbb{N}$, G admet-il un vertex cover de taille 2^k ?

Observation : Si le problème (Étrange) Vertex Cover peut être résolu par une règle de branchement, alors **P** = **NP**

- chaque application de la règle de branchement décroît strictement le paramètre donc divise la taille de la solution recherchée S par au moins 2

(Étrange) Vertex Cover

Etant donné un graphe G paramétré par $k \in \mathbb{N}$, G admet-il un vertex cover de taille 2^k ?

Observation : Si le problème (Étrange) Vertex Cover peut être résolu par une règle de branchement, alors **P** = **NP**

- chaque application de la règle de branchement décroît strictement le paramètre donc divise la taille de la solution recherchée S par au moins 2
- La hauteur de l'arbre de recherche est donc $O(\log |S|)$
- et la taille de l'arbre de recherche est polynomial en $|S|$! ! !

(Étrange) Vertex Cover

Etant donné un graphe G paramétré par $k \in \mathbb{N}$, G admet-il un vertex cover de taille 2^k ?

Observation : Si le problème (Étrange) Vertex Cover peut être résolu par une règle de branchement, alors **P** = **NP**

- chaque application de la règle de branchement décroît strictement le paramètre donc divise la taille de la solution recherchée S par au moins 2
- La hauteur de l'arbre de recherche est donc $O(\log |S|)$
- et la taille de l'arbre de recherche est polynomial en $|S|$! ! !

Remarque : Si (G, ℓ) est une instance de Vertex Cover telle que $\ell = 2^k - c$, en ajoutant un couplage de taille c on obtient une instance (G', k) de étrange Vertex Cover.

Arbre de recherche bornée

- Vertex Cover
- Règles de branchement

Compression itérative

- Vertex Cover
- Feedback Vertex Set

Algorithmes randomisés

- Feedback Vertex Set
- Color Coding - Longest Path

Ensembles et séparateurs importants

- Multicoupe dans les graphes

Compression itérative - Principe

Utilisée pour les problèmes de **minimisation** dont le paramètre est la taille k de la solution.

1. **Etape de compression :**
   Etant donnée une solution de taille $k+1$, trouver un algo **FPT** qui
   - soit on construit une solution de taille k
   - ou prouve qu'il n'y a pas de solution de taille k

Compression itérative - Principe

Utilisée pour les problèmes de **minimisation** dont le paramètre est la taille k de la solution.

1. **Etape de compression :**
   Etant donnée une solution de taille $k+1$, trouver un algo **FPT** qui
   - soit on construit une solution de taille k
   - ou prouve qu'il n'y a pas de solution de taille k
2. **Itération :**
   L'algorithme considère les sous-instances $X_i = X[x_1, \ldots x_i]$ les unes après les autres. Et à partir d'une solution S_i pour X_i :

Compression itérative - Principe

Utilisée pour les problèmes de **minimisation** dont le paramètre est la taille k de la solution.

1. **Etape de compression :**
   Etant donnée une solution de taille $k+1$, trouver un algo **FPT** qui
   - soit on construit une solution de taille k
   - ou prouve qu'il n'y a pas de solution de taille k
2. **Itération :**
   L'algorithme considère les sous-instances $X_i = X[x_1, \ldots x_i]$ les unes après les autres. Et à partir d'une solution S_i pour X_i :
   - construit une solution triviale pour G_{i+1} de taille $|S_{i+1}|$
   - applique l'étape de compression pour essayer d'améliorer cette solution

Compression itérative pour Vertex Cover

Lemme : Soient $V = \{v_1, \dots v_n\}$ les sommets d'un graphe G et S_i un Vertex Cover de taille k pour $G_i = G[v_1, \dots v_n]$.
On peut vérifier s'il existe un Vertex Cover de taille k pour G_{i+1} et si c'est le cas le trouver en temps $O(2^{k+1}.m)$.

Compression itérative pour Vertex Cover

Lemme : Soient $V = \{v_1, \dots v_n\}$ les sommets d'un graphe G et S_i un Vertex Cover de taille k pour $G_i = G[v_1, \dots v_n]$.
On peut vérifier s'il existe un Vertex Cover de taille k pour G_{i+1} et si c'est le cas le trouver en temps $O(2^{k+1}.m)$.

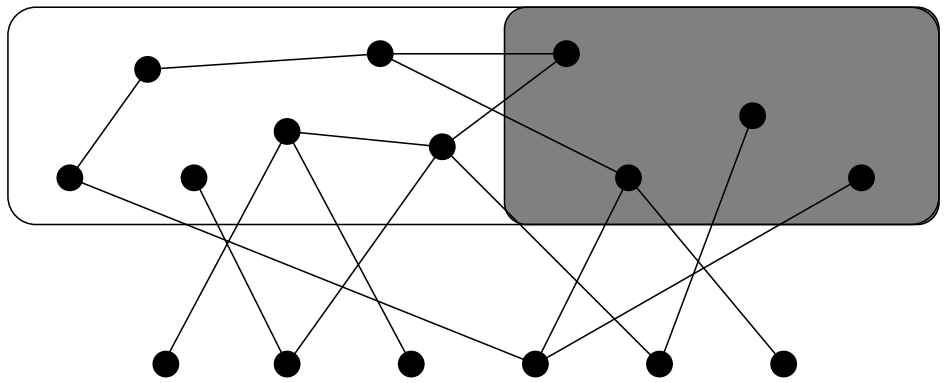
Conséquence

(Vertex Cover, k) peut être résolu en temps $O(2^{k+1}.m.n)$ par la méthode de la compression itérative.

Etape de compression pour Vertex Cover

- Soit C un Vertex Cover de taille k pour G_i, alors $C' = C \cup \{v_{i+1}\}$ est un Vertex Cover de taille $k+1$ pour G_{i+1}

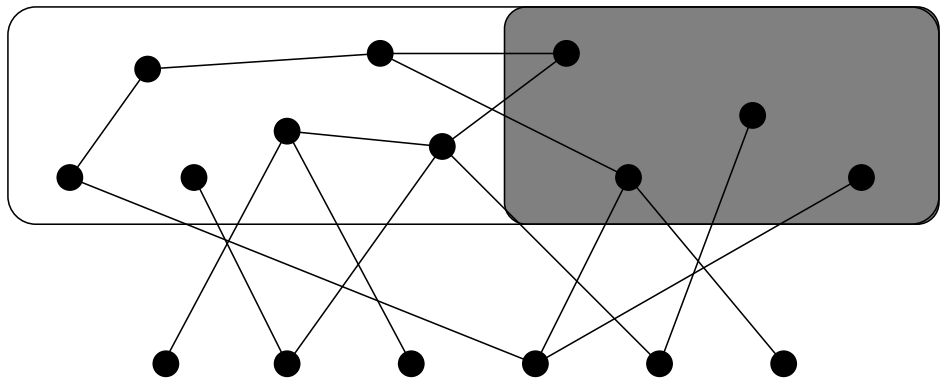
Etape de compression pour Vertex Cover



- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C"
  C'_1 : l'ensemble des sommets qui seront gardés dans C"

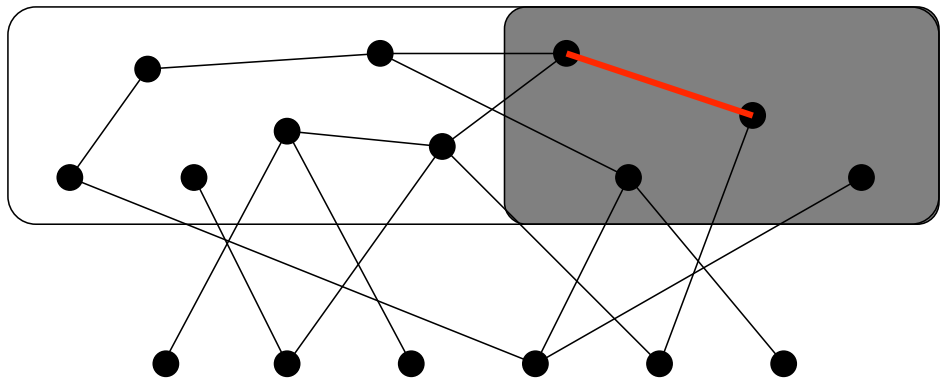
- Soit C un Vertex Cover de taille k pour G_i, alors
  $C' = C \cup \{v_{i+1}\}$ est un Vertex Cover de taille $k+1$ pour G_{i+1}

Etape de compression pour Vertex Cover



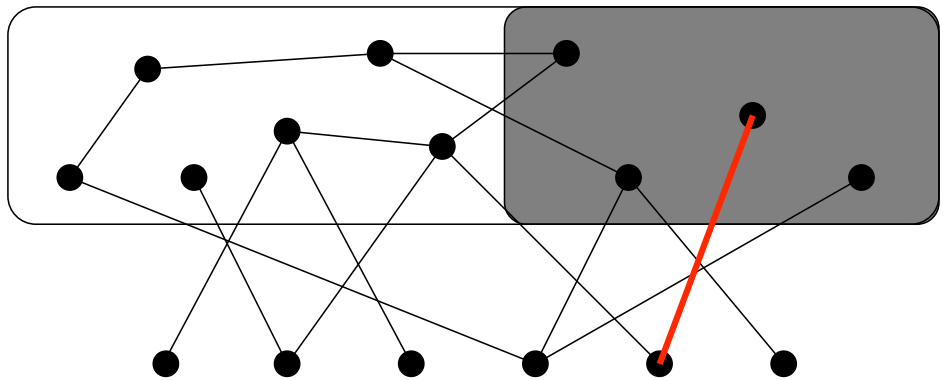
- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C"
  C'_1 : l'ensemble des sommets qui seront gardés dans C"
  1. $C"_0 = \emptyset$
  2. $\forall (u, v) \in E_{i+1}$ tq $u \notin C'_1$ et $v \notin C'_1$:

Etape de compression pour Vertex Cover



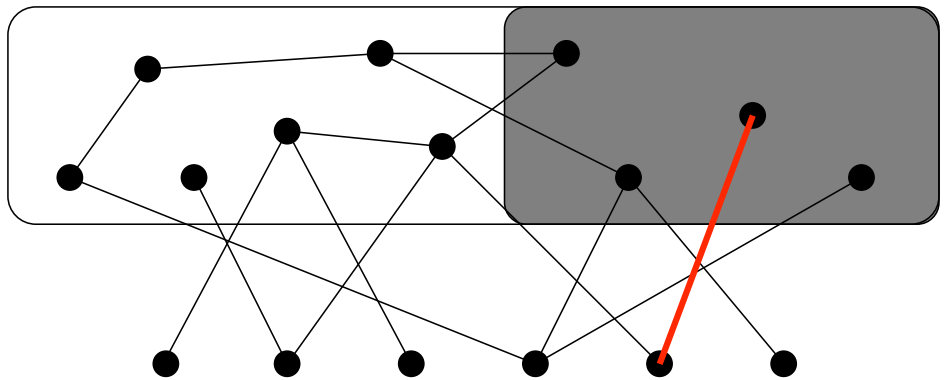
- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C''
  C'_1 : l'ensemble des sommets qui seront gardés dans C''
  1. $C''_0 = \emptyset$
  2. $\forall (u, v) \in E_{i+1}$ tq $u \notin C'_1$ et $v \notin C'_1$:
     2.1 Si $u \in C'_0$ et $v \in C'_0$, alors compression impossible

Etape de compression pour Vertex Cover



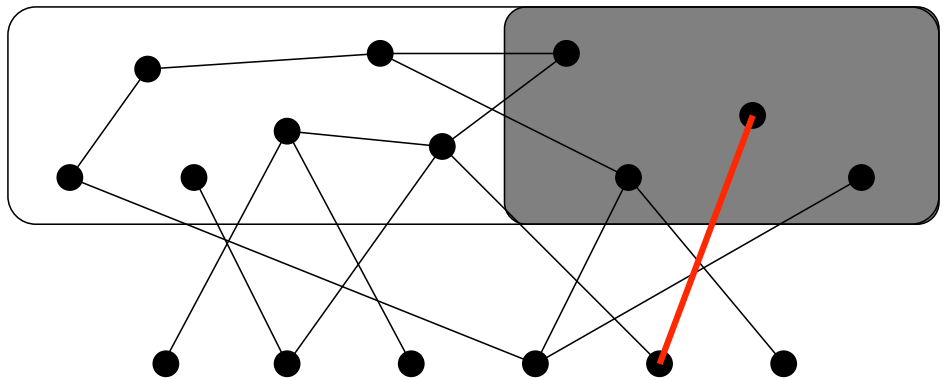
- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C"
  C'_1 : l'ensemble des sommets qui seront gardés dans C"
  1. $C"_0 = \emptyset$
  2. $\forall (u, v) \in E_{i+1}$ tq $u \notin C'_1$ et $v \notin C'_1$:
     - 2.1 Si $u \in C'_0$ et $v \in C'_0$, alors compression impossible
     - 2.2 Si $u \in C'_0$ et $v \in V_{i+1} \setminus C'$, alors $C''_0 = C''_0 \cup \{v\}$
     - 2.3 Si $u \in V_{i+1} \setminus C'$ et $v \in C'_0$, alors $C''_0 = C''_0 \cup \{u\}$

Etape de compression pour Vertex Cover



- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C"
  C'_1 : l'ensemble des sommets qui seront gardés dans C"
  1. $C''_0 = \emptyset$
  2. $\forall (u, v) \in E_{i+1}$ tq $u \notin C'_1$ et $v \notin C'_1$:
     - 2.1 Si $u \in C'_0$ et $v \in C'_0$, alors compression impossible
     - 2.2 Si $u \in C'_0$ et $v \in V_{i+1} \setminus C'$, alors $C''_0 = C''_0 \cup \{v\}$
     - 2.3 Si $u \in V_{i+1} \setminus C'$ et $v \in C'_0$, alors $C''_0 = C''_0 \cup \{u\}$
  3. Si $|C'_1 \cup C''_0| \leqslant k$, alors retourner $C'' = C'_1 \cup C''_0$

Etape de compression pour Vertex Cover



- Soit (C'_0, C'_1) une partition de C' avec
  C'_0 : l'ensemble des sommets qui seront absents dans C"
  C'_1 : l'ensemble des sommets qui seront gardés dans C"
  1. $C"_0 = \emptyset$
  2. $\forall (u, v) \in E_{i+1}$ tq $u \notin C'_1$ et $v \notin C'_1$:
     - 2.1 Si $u \in C'_0$ et $v \in C'_0$, alors compression impossible
     - 2.2 Si $u \in C'_0$ et $v \in V_{i+1} \setminus C'$, alors $C''_0 = C''_0 \cup \{v\}$
     - 2.3 Si $u \in V_{i+1} \setminus C'$ et $v \in C'_0$, alors $C''_0 = C''_0 \cup \{u\}$
  3. Si $|C'_1 \cup C''_0| \leqslant k$, alors retourner $C'' = C'_1 \cup C''_0$
- Il suffit donc de tester les 2^{k+1} partitions possibles de C'

Compression itérative - Feedback Vertex Set

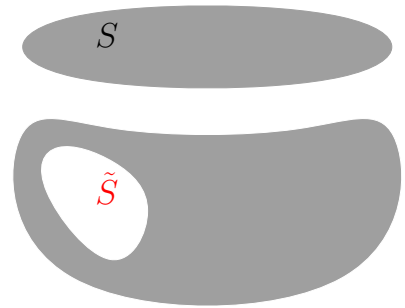
Feedback Vertex Set

- Etant donnés un graphe G et un entier k. Existe-t-il un ensemble S d'au plus k sommets tel que $G - S$ soit acyclique ?

Compression itérative - Feedback Vertex Set

Feedback Vertex Set

- Etant donnés un graphe G et un entier k. Existe-t-il un ensemble S d'au plus k sommets tel que $G - S$ soit acyclique ?



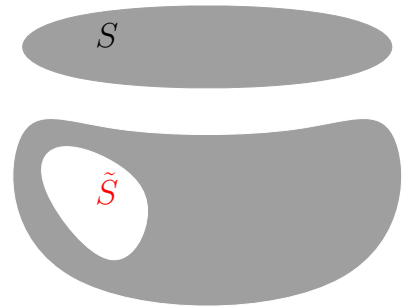
Disjoint Feedback Vertex Set

- Etant donnés un graphe G et un ensemble S de taille k tel que $G - S$ soit acyclique. Existe-t-il un ensemble $\tilde{S}$ de sommets tel que $G - \tilde{S}$ soit acyclique et $S \cap \tilde{S} = \emptyset$ et $|\tilde{S}| < |S|$?

Compression itérative - Feedback Vertex Set

Feedback Vertex Set

- Etant donnés un graphe G et un entier k. Existe-t-il un ensemble S d'au plus k sommets tel que $G - S$ soit acyclique ?



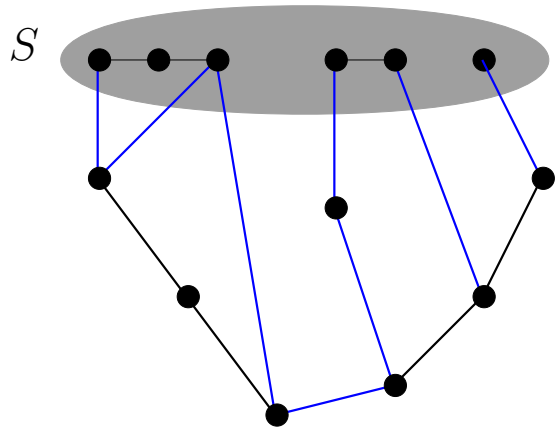
Disjoint Feedback Vertex Set

- Etant donnés un graphe G et un ensemble S de taille k tel que $G - S$ soit acyclique. Existe-t-il un ensemble $\tilde{S}$ de sommets tel que $G - \tilde{S}$ soit acyclique et $S \cap \tilde{S} = \emptyset$ et $|\tilde{S}| < |S|$?

Lemme : Si l'on peut résoudre Disjoint Feedback Vertex Set en temps $O^*(c^k)$ (avec $c \in \mathbb{N}^+$), alors on peut résoudre Feedback Vertex Set en temps $O^*((c+1)^k)$.

Compression itérative - Feedback Vertex Set

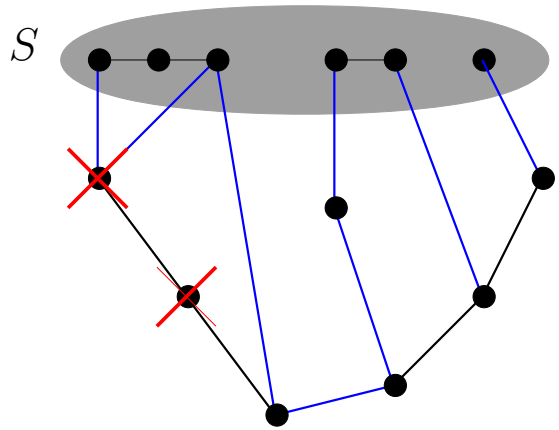
Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



Règle réduction 1 : Si x possède deux voisins dans une composante connexe de $G[S]$, alors retourner $(G - \{x\}, S, k - 1)$

Compression itérative - Feedback Vertex Set

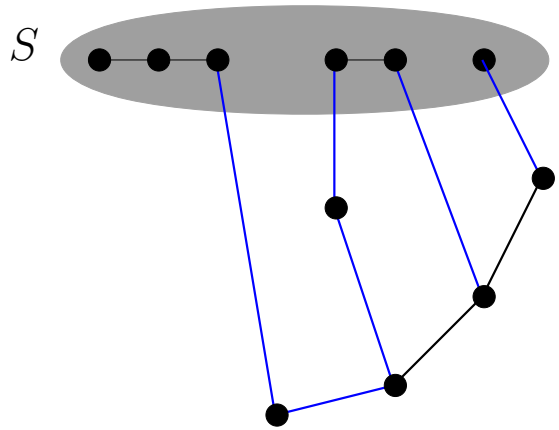
Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



Règle réduction 1 : Si x possède deux voisins dans une composante connexe de $G[S]$, alors retourner $(G - \{x\}, S, k - 1)$

Compression itérative - Feedback Vertex Set

Soit (G, S, k) une instance de Disjoint Feedback Vertex Set

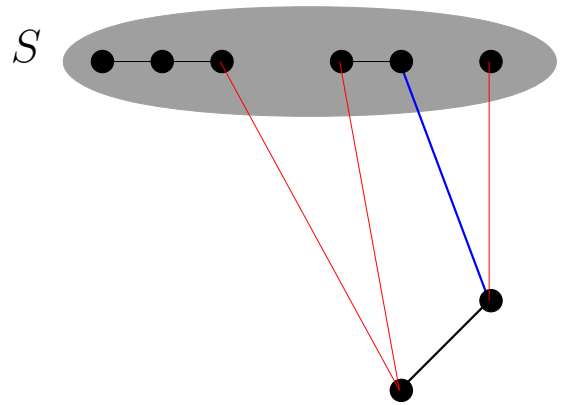


Règle réduction 1 : Si x possède deux voisins dans une composante connexe de $G[S]$, alors retourner $(G - \{x\}, S, k - 1)$

Règle réduction 2 : Si x possède un seul voisin y dans S et un seul voisin z dans $G - S$, alors retourner $(G - x \cup \{yz\}, S, k)$

Compression itérative - Feedback Vertex Set

Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



Règle réduction 1 : Si x possède deux voisins dans une composante connexe de $G[S]$, alors retourner $(G - \{x\}, S, k - 1)$

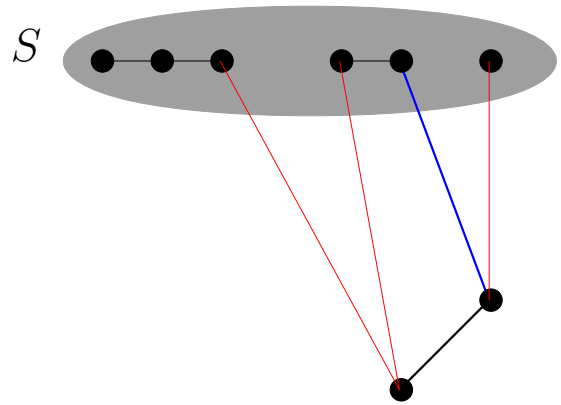
Règle réduction 2 : Si x possède un seul voisin y dans S et un seul voisin z dans $G - S$, alors retourner $(G - x \cup \{yz\}, S, k)$

Règle de branchement : Si x possède deux voisins dans des composantes connexes différentes de $G - S$, alors brancher sur

$(G - x, S, k - 1)$ et $(G, S \cup \{x\}, k)$

Compression itérative - Feedback Vertex Set

Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



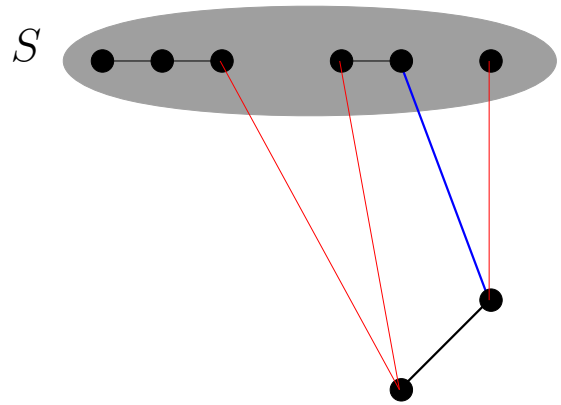
Théorème : L'alogrithme de réduction et branchement résoud Disjoint Feedback Vertex Set en temps $O^*(4^k)$.

- Pour analyser la taille de l'arbre de branchement, on utilise la mesure

$$\mu = k + \#cc(G[S]) \leqslant 2k$$

Compression itérative - Feedback Vertex Set

Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



Théorème : L'alogrithme de réduction et branchement résoud Disjoint Feedback Vertex Set en temps $O^*(4^k)$.

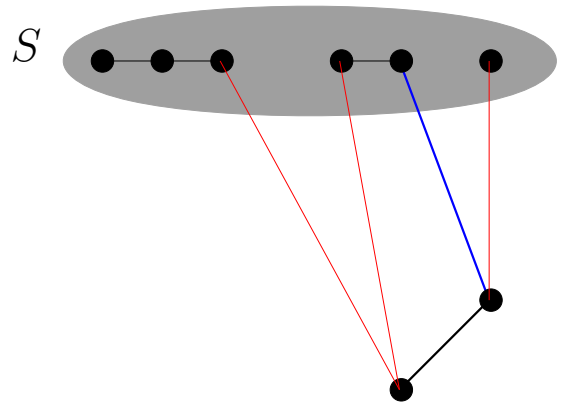
- Pour analyser la taille de l'arbre de branchement, on utilise la mesure

$$\mu = k + \#cc(G[S]) \leqslant 2k$$

- Si x est supprimé (ou ajouté dans $\tilde{S}$), μ décroit
  Notons que l'ajout de x dans $G[S]$ ne crée pas de cycle

Compression itérative - Feedback Vertex Set

Soit (G, S, k) une instance de Disjoint Feedback Vertex Set



Théorème : L'alogrithme de réduction et branchement résoud Disjoint Feedback Vertex Set en temps $O^*(4^k)$.

- Pour analyser la taille de l'arbre de branchement, on utilise la mesure

$$\mu = k + \#cc(G[S]) \leqslant 2k$$

- Si x est supprimé (ou ajouté dans $\tilde{S}$), μ décroit
  Notons que l'ajout de x dans $G[S]$ ne crée pas de cycle
- Sinon x est ajouté à S et $\#cc(G[S]$ décroit

Arbre de recherche bornée

- Vertex Cover
- Règles de branchement

Compression itérative

- Vertex Cover
- Feedback Vertex Set

Algorithmes randomisés

- Feedback Vertex Set
- Color Coding - Longest Path

Ensembles et séparateurs importants

- Multicoupe dans les graphes

Feedback Vertex Set randomisé

Soit G un graphe sans boucle, ni sommet de degré 0, 1 ou 2.

- les sommets avec boucles sont supprimés car, ils appartiennent à tout FVS ;
- les sommets de degré 0 ou 1 n'apparaîssent dans aucun cycle, ils peuvent être supprimés ;
- il exsite toujours un FVS ne contenant pas de sommet de degré 2 : s'il en existe un, on peut contracter une arête incidente.

Feedback Vertex Set randomisé

Soit G un graphe sans boucle, ni sommet de degré 0, 1 ou 2.

Lemme : Soit S un FVS de G. Notons $X = V(G) \setminus S$ et $E_X = E(G) \cap (X \times X)$. Alors

$$|E_X| \leqslant |E(G)|/2$$

Notons $E_{S,X} = E(G) \cap (S \times X)$

Feedback Vertex Set randomisé

Soit G un graphe sans boucle, ni sommet de degré 0, 1 ou 2.

Lemme : Soit S un FVS de G. Notons $X = V(G) \setminus S$ et $E_X = E(G) \cap (X \times X)$. Alors

$$|E_X| \leqslant |E(G)|/2$$

Notons $E_{S,X} = E(G) \cap (S \times X)$

- Puisque G est de degré minimum 3 :

$$3 \cdot |X| \leqslant \Sigma_{x \in X}\, d(x) = 2 \cdot |E_X| + |E_{S,X}|$$

Feedback Vertex Set randomisé

Soit G un graphe sans boucle, ni sommet de degré 0, 1 ou 2.

Lemme : Soit S un FVS de G. Notons $X = V(G) \setminus S$ et $E_X = E(G) \cap (X \times X)$. Alors

$$|E_X| \leqslant |E(G)|/2$$

Notons $E_{S,X} = E(G) \cap (S \times X)$

- Puisque G est de degré minimum 3 :

$$3 \cdot |X| \ \leqslant \ \Sigma_{x \in X} \ d(x) \ = \ 2 \cdot |E_X| + |E_{S,X}|$$

- Puisque S est un FVS : $|E_X| < |X|$ donc

$$3 \cdot |E_X| < 2 \cdot |E_X| + |E_{S,X}|$$

Feedback Vertex Set randomisé (2)

Théorème [Becker, Bar-Yehuda, Geiger 2000]
Le problème Feedback Vertex Set admet un algorithme de complexité $O(c4^k \cdot kn)$ (avec $c > 0$) qui retourne un FVS de taille k d'un graphe G (s'il en existe un) avec probabilité au moins $1 - (1 - \frac{1}{4^k})^{c4^k}$.

Feedback Vertex Set randomisé (2)

Théorème [Becker, Bar-Yehuda, Geiger 2000]
Le problème Feedback Vertex Set admet un algorithme de complexité $O(c4^k \cdot kn)$ (avec $c > 0$) qui retourne un FVS de taille k d'un graphe G (s'il en existe un) avec probabilité au moins $1 - (1 - \frac{1}{4^k})^{c4^k}$.

- Soit S un ensemble de sommets obtenu en tirant k fois uniformément une arête puis un sommet de l'arête

$$\mathbb{P}(S \text{ n'est pas un FVS}) = 1 - \frac{1}{4^k}$$

Feedback Vertex Set randomisé (2)

Théorème [Becker, Bar-Yehuda, Geiger 2000]
Le problème Feedback Vertex Set admet un algorithme de complexité $O(c4^k \cdot kn)$ (avec $c > 0$) qui retourne un FVS de taille k d'un graphe G (s'il en existe un) avec probabilité au moins $1 - (1 - \frac{1}{4^k})^{c4^k}$.

- Soit S un ensemble de sommets obtenu en tirant k fois uniformément une arête puis un sommet de l'arête

$$\mathbb{P}(S \text{ n'est pas un FVS}) = 1 - \frac{1}{4^k}$$

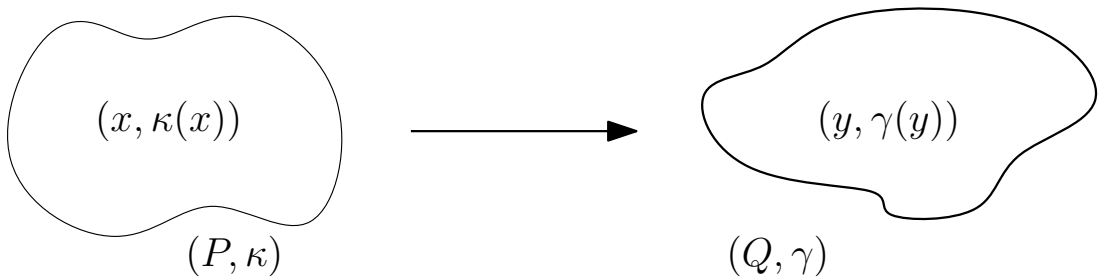
- Soient $S_1 \dots S_\ell$ avec $\ell \in [c4^k]$

$$\mathbb{P}(\forall i \in [c4^k],\ S_i \text{ n'est pas un FVS}) = (1 - \frac{1}{4^k})^{c4^k}$$

$$\mathbb{P}(\exists i \in [c4^k],\ S_i \text{ est un FVS}) = 1 - (1 - \frac{1}{4^k})^{c4^k}$$

Color Coding : Idée – Principe [Alon, Yuster, Zwick]

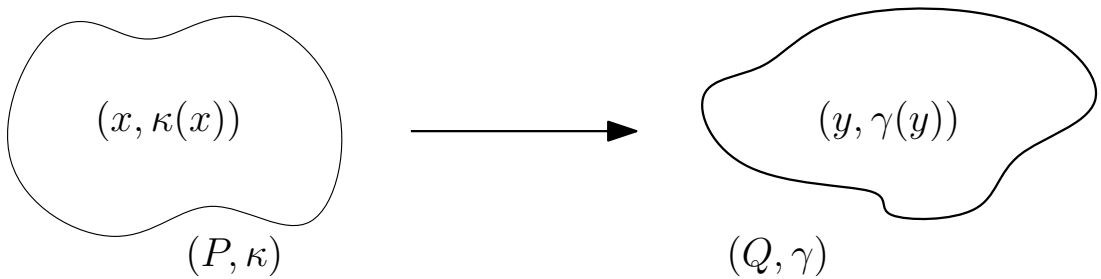
Color Coding : Transformer un problème (P, κ) en un problème "plus facile" (Q, γ) à l'aide d'une coloration aléatoire



- (Q, γ), la version colorée de (P, κ), teste l'existence d'une solution multicolorée, i.e. utilisant une et une seule fois chaque couleur.

Color Coding : Idée – Principe [Alon, Yuster, Zwick]

Color Coding : Transformer un problème (P, κ) en un problème "plus facile" (Q, γ) à l'aide d'une coloration aléatoire



- (Q, γ), la version colorée de (P, κ), teste l'existence d'une solution multicolorée, i.e. utilisant une et une seule fois chaque couleur.

Analyse

- Estimer la probabilité qu'une solution à (P, κ) soit multicolorée $\rightsquigarrow$ en déduire le nombre d'itérations (colorations aléatoires) nécessaires pour obtenir sûrement une solution multicolorée
- Test efficace de l'existence d'une solution multicolorée ?

Longest Path (1)

Tester l'existence d'un chemin mulitcoloré de longueur k dans graphe G

Lemme : Soient P_k de longueur k dans un graphe G et $\omega : V(G) \to [k]$ une coloration aléatoire de G. Alors

$$\mathbb{P}(P_k^\omega) = \frac{k!}{k^k} > \frac{1}{e^k}$$

avec P_k^ω le chemin multicoloré

Longest Path (1)

Tester l'existence d'un chemin mulitcoloré de longueur k dans graphe G

Lemme : Soient P_k de longueur k dans un graphe G et $\omega : V(G) \to [k]$ une coloration aléatoire de G. Alors

$$\mathbb{P}(P_k^\omega) = \frac{k!}{k^k} > \frac{1}{e^k}$$

avec P_k^ω le chemin multicoloré

- k^k colorations possibles pour P_k
- $k!$ d'entre elles sont multicolorées

Longest Path (1)

Tester l'existence d'un chemin mulitcoloré de longueur k dans graphe G

Lemme : Soient P_k de longueur k dans un graphe G et $\omega : V(G) \to [k]$ une coloration aléatoire de G. Alors

$$\mathbb{P}(P_k^\omega) = \frac{k!}{k^k} > \frac{1}{e^k}$$

avec P_k^ω le chemin multicoloré

Lemme : Un algorithme testant l'existence d'un chemin P_k multicoloré en temps $f(k)$ implique un algorithme de complexité $e^k \cdot f(k) \cdot n^{O(1)}$ pour Longest Path.

Longest Path (1)

Tester l'existence d'un chemin mulitcoloré de longueur k dans graphe G

Lemme : Soient P_k de longueur k dans un graphe G et $\omega : V(G) \to [k]$ une coloration aléatoire de G. Alors

$$\mathbb{P}(P_k^\omega) = \frac{k!}{k^k} > \frac{1}{e^k}$$

avec P_k^ω le chemin multicoloré

Lemme : Un algorithme testant l'existence d'un chemin P_k multicoloré en temps $f(k)$ implique un algorithme de complexité $e^k \cdot f(k) \cdot n^{O(1)}$ pour Longest Path.

- Soit G un graphe contenant un chemin de longeur k
- Soit G^ω une coloration de G $\qquad \mathbb{P}(P_k^\omega \notin G^\omega) < 1 - \frac{1}{e^k}$

Longest Path (1)

Tester l'existence d'un chemin mulitcoloré de longueur k dans graphe G

Lemme : Soient P_k de longueur k dans un graphe G et $\omega : V(G) \to [k]$ une coloration aléatoire de G. Alors

$$\mathbb{P}(P_k^\omega) = \frac{k!}{k^k} > \frac{1}{e^k}$$

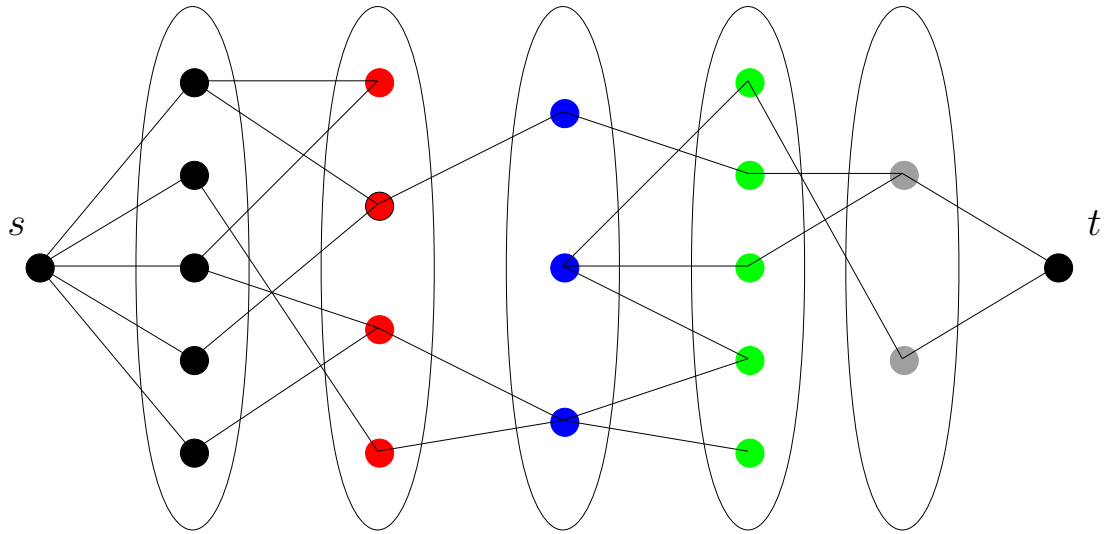
avec P_k^ω le chemin multicoloré

Lemme : Un algorithme testant l'existence d'un chemin P_k multicoloré en temps $f(k)$ implique un algorithme de complexité $e^k \cdot f(k) \cdot n^{O(1)}$ pour Longest Path.

- Soit G un graphe contenant un chemin de longeur k
- Soit G^ω une coloration de G $\qquad \mathbb{P}(P_k^\omega \notin G^\omega) < 1 - \frac{1}{e^k}$
- Après e^k colorations G^{ω_i}, $\qquad \mathbb{P}(\forall i, P_k^\omega \notin G^{\omega_i}) < (1 - \frac{1}{e^k})^{e^k} = \frac{1}{e}$

Longest Path (2)

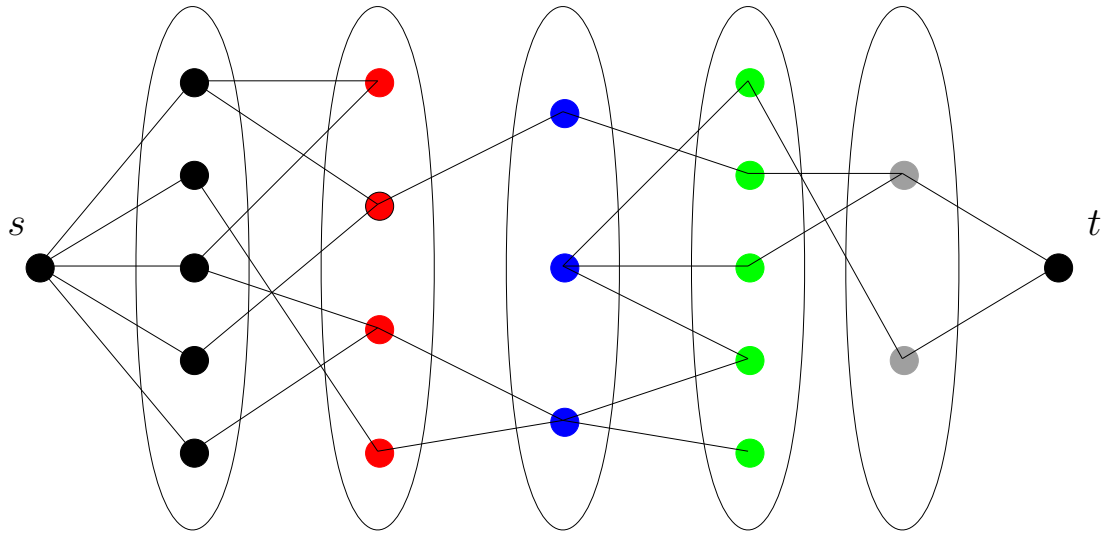
- Tester l'existence d'un chemin multicoloré à l'aide de "flots"



- Tester les $k!$ permutations de couleurs
- Supprimer les arêtes monochromatiques et entre deux couleurs non-consécutives
- Tester l'existence d'un chemin (orienté) entre s et t

Longest Path (2)

- Tester l'existence d'un chemin multicoloré à l'aide de "flots"



- Tester les $k!$ permutations de couleurs
- Supprimer les arêtes monochromatiques et entre deux couleurs non-consécutives
- Tester l'existence d'un chemin (orienté) entre s et t

Complexité : $O(k! \cdot |E(G)|)$

Longest Path (2)

- Tester l'existence d'un chemin multicoloré à l'aide de programmation dynamique

On définit $2^k \cdot |V(G)|$ variables booléennes telles que

$$T(x, C) = \text{VRAI}$$

ssi il existe un chemin mulitcoloré de s à x sur les couleurs de C.

Longest Path (2)

- Tester l'existence d'un chemin multicoloré à l'aide de programmation dynamique

On définit $2^k \cdot |V(G)|$ variables booléennes telles que

$$T(x, C) = \text{VRAI}$$

ssi il existe un chemin mulitcoloré de s à x sur les couleurs de C.

- $T(s, \emptyset) = \text{VRAI}$
- $T(x, C) = \bigvee_{xy \in E} T(y, C \setminus \{\omega(x)\})$

Longest Path (2)

- Tester l'existence d'un chemin multicoloré à l'aide de programmation dynamique

On définit $2^k \cdot |V(G)|$ variables booléennes telles que

$$T(x, C) = \text{VRAI}$$

ssi il existe un chemin mulitcoloré de s à x sur les couleurs de C.

- $T(s, \emptyset) = \text{VRAI}$
- $T(x, C) = \bigvee_{xy \in E} T(y, C \setminus \{\omega(x)\})$

Complexité pour remplir la table : $O(2^k \cdot (|V(G)| + |E(G)|)$

Longest Path (3)

Observation : il est possible de dérandomiser l'algorithme

Théorème [Alon, Yuster, Zwick]
Longest Path admet un algorithme FPT (randomisé) de complexité $2^{O(k)} \cdot (|V(G)| + |E(G)|)$.

Exercice :

1. Proposer un algorithme de color-coding pour k-Disjoint Triangle
2. Proposer un algorithme de color-coding pour le problème Vertex Cover.
3. Pourquoi color-coding ne s'applique pas pour Clique ?

Arbre de recherche bornée

- Vertex Cover
- Règles de branchement

Compression itérative

- Vertex Cover
- Feedback Vertex Set

Algorithmes randomisés

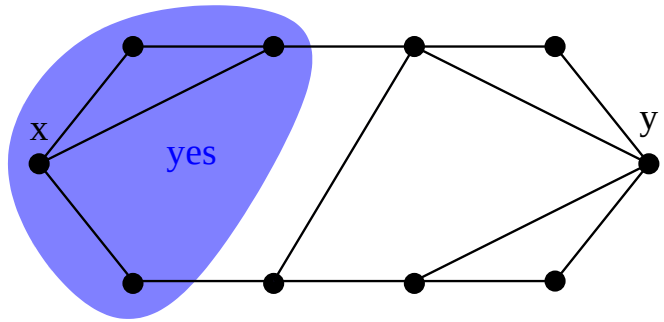
- Feedback Vertex Set
- Color Coding - Longest Path

Ensembles et séparateurs importants

- Multicoupe dans les graphes

Ensembles et séparateurs Importants

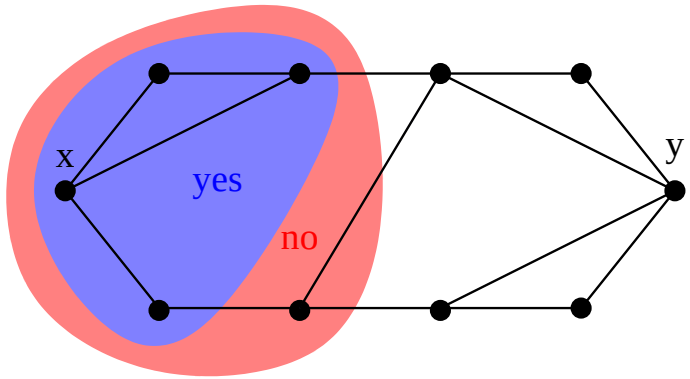
Soit x, y deux sommets.
Un ensemble S est (x, y)-important si



- $x \in S$ et $y \notin S$
- $G[S]$ est connexe
- $\nexists\ Y$ tq. $S \subset Y$, $d_G(Y) \leqslant d_G(S)$ et $G[Y]$ est connexe

Ensembles et séparateurs Importants

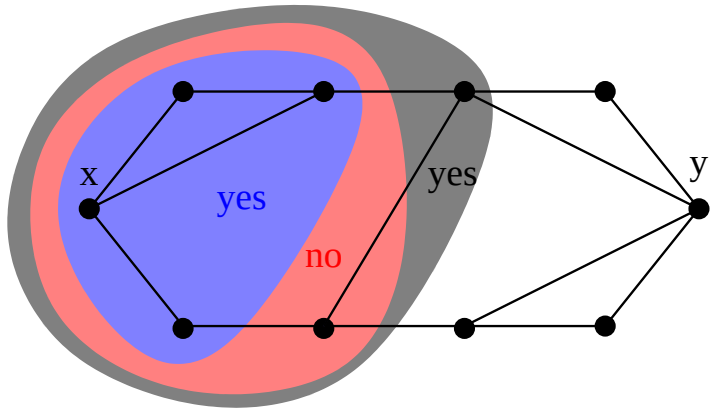
Soit x, y deux sommets.
Un ensemble S est (x, y)-important si



- $x \in S$ et $y \notin S$
- $G[S]$ est connexe
- $\nexists\ Y$ tq. $S \subset Y$, $d_G(Y) \leqslant d_G(S)$ et $G[Y]$ est connexe

Ensembles et séparateurs Importants

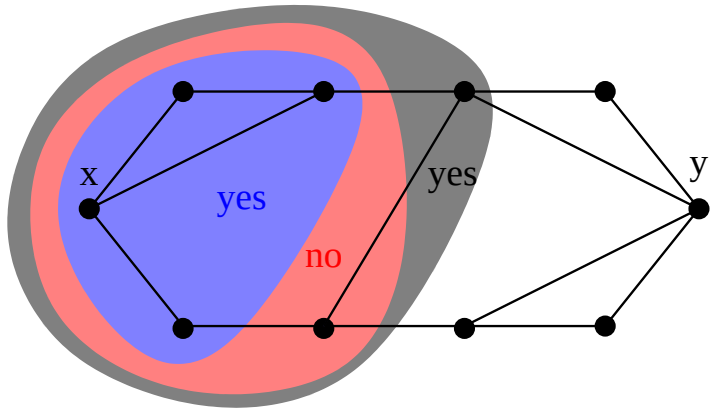
Soit x, y deux sommets.
Un ensemble S est (x, y)-important si



- $x \in S$ et $y \notin S$
- $G[S]$ est connexe
- $\nexists\ Y$ tq. $S \subset Y$, $d_G(Y) \leqslant d_G(S)$ et $G[Y]$ est connexe

Ensembles et séparateurs Importants

Soit x, y deux sommets.
Un ensemble S est (x, y)-important si



- $x \in S$ et $y \notin S$
- $G[S]$ est connexe
- $\nexists$ Y tq. $S \subset Y$, $d_G(Y) \leqslant d_G(S)$ et $G[Y]$ est connexe

$$\partial(S) = \{(u, v) \in E(G) \mid u \in S \vee v \notin S\}$$

Si S est un (x, y)-ensemble important, alors $\partial(S)$ est un (x, y)-séparateur (coupe) important

Remarque : pour des ensembles de sommets X et Y

- (X, Y)-ensembles et (X, Y)-séparateurs importants

Remarque : pour des ensembles de sommets X et Y

- (X, Y)-ensembles et (X, Y)-séparateurs importants

Théorème [Chen et al., Lokshtanov et Marx]
Pour tout $k \geqslant 0$, il existe au plus 4^k ensembles importants S tel que $|\partial(S)| \leqslant k$. ($\longrightarrow$ énumeration en temps $4^k.n^{O(1)}$)

Remarque : pour des ensembles de sommets X et Y

- (X, Y)-ensembles et (X, Y)-séparateurs importants

Théorème [Chen et al., Lokshtanov et Marx]
Pour tout $k \geqslant 0$, il existe au plus 4^k ensembles importants S tel que $|\partial(S)| \leqslant k$. ($\longrightarrow$ énumeration en temps $4^k.n^{O(1)}$)

Lemme : Si $\mathcal{S}$ est l'ensemble des (x, y)-ensembles importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Observons que (1) $\Rightarrow$ Théorème car

- on s'intéresse à $\mathcal{S}'$, le sous-ensemble des (x, y)-ensembles important avec $d_G(S) \leqslant k$

Remarque : pour des ensembles de sommets X et Y

- (X, Y)-ensembles et (X, Y)-séparateurs importants

Théorème [Chen et al., Lokshtanov et Marx]
Pour tout $k \geqslant 0$, il existe au plus 4^k ensembles importants S tel que $|\partial(S)| \leqslant k$. ($\longrightarrow$ énumeration en temps $4^k.n^{O(1)}$)

Lemme : Si $\mathcal{S}$ est l'ensemble des (x, y)-ensembles importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Observons que (1) $\Rightarrow$ Théorème car

- on s'intéresse à $\mathcal{S}'$, le sous-ensemble des (x, y)-ensembles important avec $d_G(S) \leqslant k$
- donc $\sum_{S \in \mathcal{S}'} 4^{k - d_G(S)} \leqslant 4^k$
- et chaque terme de la somme est supérieur ou égal à 1

Lemme Si $\mathcal{S}$ est l'ensemble des (x, y)-ensemble importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Lemme Si $\mathcal{S}$ est l'ensemble des (x, y)-ensemble importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Soit λ le degré minumum d'un (x, y)-ensemble important.

Preuve par induction sur # arêtes que $\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 2^{-\lambda}}$ (2)

Lemme Si $\mathcal{S}$ est l'ensemble des (x, y)-ensemble importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Soit λ le degré minumum d'un (x, y)-ensemble important.

Preuve par induction sur # arêtes que $\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 2^{-\lambda}}$ (2)

Observation $\exists !$ (x, y)-ensemble important S^* tq.

$$\boxed{d_G(S^*) = \lambda \text{ et } S^* \text{ est maximal}}$$

Lemme Si $\mathcal{S}$ est l'ensemble des (x, y)-ensemble importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Soit λ le degré minumum d'un (x, y)-ensemble important.

Preuve par induction sur # arêtes que $\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 2^{-\lambda}}$ (2)

Observation $\exists !$ (x, y)-ensemble important S^* tq.

$$\boxed{d_G(S^*) = \lambda \text{ et } S^* \text{ est maximal}}$$

- Par la sous-modularité de la fonction de degré, si S' et S'' sont deux tels ensembles, alors

$$d_G(S') + d_G(S'') \geqslant d_G(S' \cup S'') + d_G(S' \cap S'')$$

Lemme Si $\mathcal{S}$ est l'ensemble des (x, y)-ensemble importants, alors

$$\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 1} \quad (1)$$

Soit λ le degré minumum d'un (x, y)-ensemble important.

Preuve par induction sur # arêtes que $\boxed{\sum_{S \in \mathcal{S}} 4^{-d_G(S)} \leqslant 2^{-\lambda}}$ (2)

Observation $\exists !$ (x, y)-ensemble important S^* tq.

$$\boxed{d_G(S^*) = \lambda \text{ et } S^* \text{ est maximal}}$$

- Par la sous-modularité de la fonction de degré, si S' et S'' sont deux tels ensembles, alors

$$d_G(S') + d_G(S'') \geqslant d_G(S' \cup S'') + d_G(S' \cap S'')$$

$\Rightarrow$ $d_G(S' \cup S'') \leqslant \lambda$: contradiction avec la maximalité de S' et S''

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)
  - $\sum_{S \in \mathcal{S}_e} 4^{-d_G(S)} \leqslant \sum_{S' \in \mathcal{S}'} 4^{-(d_{G'}(S')-1)} \leqslant 2^{-(\lambda-1)}/4 = 2^{-\lambda}/2$

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)
  - $\sum_{S \in \mathcal{S}_e} 4^{-d_G(S)} \leqslant \sum_{S' \in \mathcal{S}'} 4^{-(d_{G'}(S')-1)} \leqslant 2^{-(\lambda-1)}/4 = 2^{-\lambda}/2$
- $e \notin \partial(S)$: Remarquons que $S^* \cup \{v\} \subseteq S$

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)
  - $\sum_{S \in \mathcal{S}_e} 4^{-d_G(S)} \leqslant \sum_{S' \in \mathcal{S}'} 4^{-(d_{G'}(S')-1)} \leqslant 2^{-(\lambda-1)}/4 = 2^{-\lambda}/2$
- $e \notin \partial(S)$: Remarquons que $S^* \cup \{v\} \subseteq S$
  - Soit G^* résultant de la contraction de $S^* \cup v$ en s dans G

Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)
  - $\sum_{S \in \mathcal{S}_e} 4^{-d_G(S)} \leqslant \sum_{S' \in \mathcal{S}'} 4^{-(d_{G'}(S')-1)} \leqslant 2^{-(\lambda-1)}/4 = 2^{-\lambda}/2$
- $e \notin \partial(S)$: Remarquons que $S^* \cup \{v\} \subseteq S$
  - Soit G^* résultant de la contraction de $S^* \cup v$ en s dans G
  - $\lambda^* > \lambda$
  - $\mathcal{S}^* = \{\tilde{S} \mid \tilde{S}\ (s, y)\text{-ensemble important de } G'\} \longleftrightarrow \mathcal{S}_{\bar{e}}$

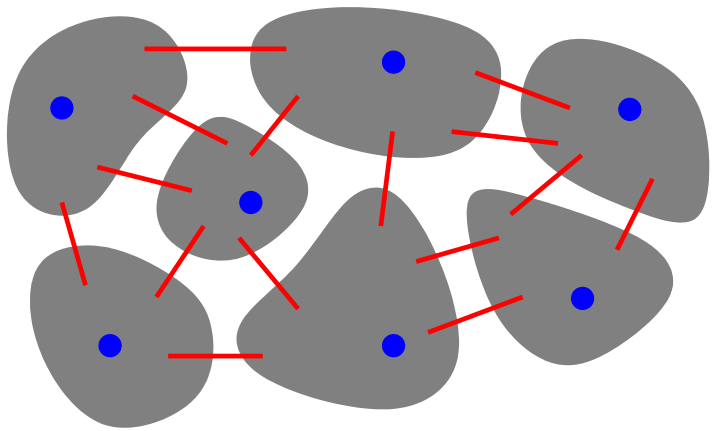
Observation. tout (x, y)-ensemble important S vérifie $S^* \subseteq S$.

Soit $e = uv$ avec $u \in S^*$ et $v \notin S^*$,
$\mathcal{S}_e$ les (x, y)-ensembles importants contenant u mais pas v
et $\mathcal{S}_{\bar{e}} = \mathcal{S} \setminus \mathcal{S}_e$

- $e \in \partial(S)$: S est un (x, y)-ensemble important dans $G' = G - e$
  - $\mathcal{S}'$ les (x, y)-ensembles importants de G' $\quad \lambda' \geqslant \lambda - 1$
  - $\sum_{S' \in \mathcal{S}'} 4^{-d_{G'}(S')} \leqslant 2^{-(\lambda-1)}$ (par hyp. d'induction)
  - $\sum_{S \in \mathcal{S}_e} 4^{-d_G(S)} \leqslant \sum_{S' \in \mathcal{S}'} 4^{-(d_{G'}(S')-1)} \leqslant 2^{-(\lambda-1)}/4 = 2^{-\lambda}/2$
- $e \notin \partial(S)$: Remarquons que $S^* \cup \{v\} \subseteq S$
  - Soit G^* résultant de la contraction de $S^* \cup v$ en s dans G
  - $\lambda^* > \lambda$
  - $\mathcal{S}^* = \{\tilde{S} \mid \tilde{S}\ (s, y)\text{-ensemble important de } G'\} \longleftrightarrow \mathcal{S}_{\bar{e}}$
  - Par hyp. d'induction, nous avons $\sum_{S \in \mathcal{S}_{\bar{e}}} 4^{-d_G(S)} \leqslant 2^{-\lambda}/2$

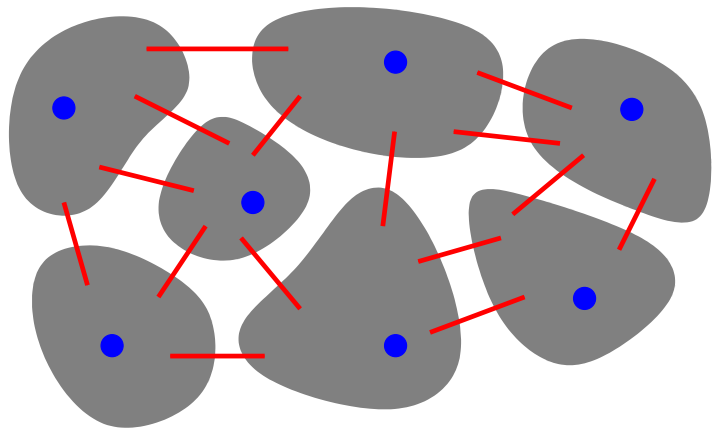
Multicoupe paramétrée (multiway cut)

- Un graphe G et un ensemble T de sommets terminaux
- Un paramètre $k \in \mathbb{N}$
- Existe-t-il un ensemble S de k arêtes tel que chaque composante de $G \setminus S$ contient au plus un terminal de T ?



Multicoupe paramétrée (multiway cut)

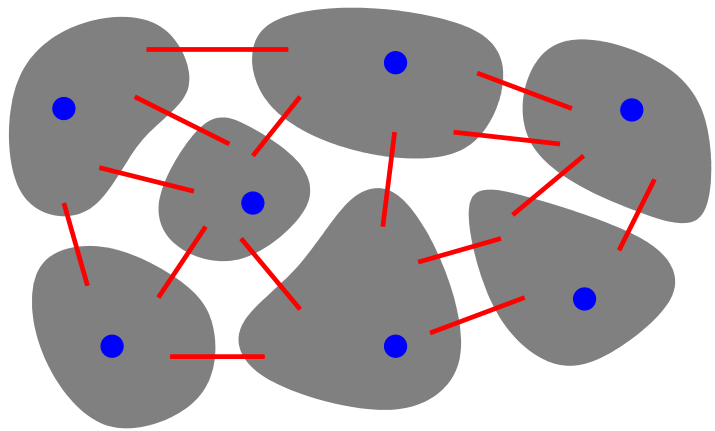
- Un graphe G et un ensemble T de sommets terminaux
- Un paramètre $k \in \mathbb{N}$
- Existe-t-il un ensemble S de k arêtes tel que chaque composante de $G \setminus S$ contient au plus un terminal de T ?



Théorème : Le problème Multicoupe paramétrée est FPT

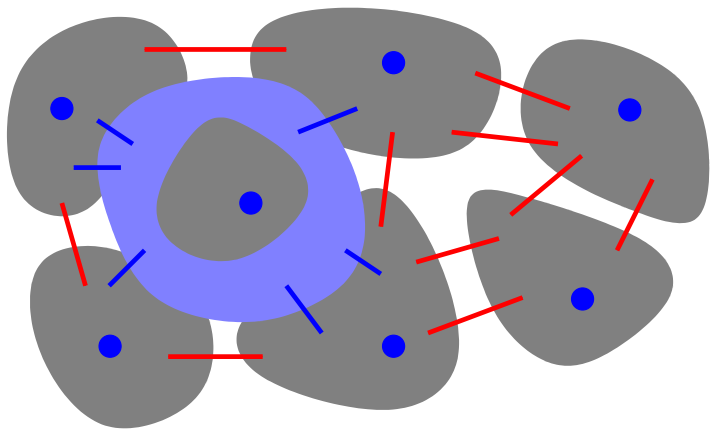
Idée :

- Si t est un terminal, alors toute solution contient un $(t, T \setminus \{t\})$-séparateur S,



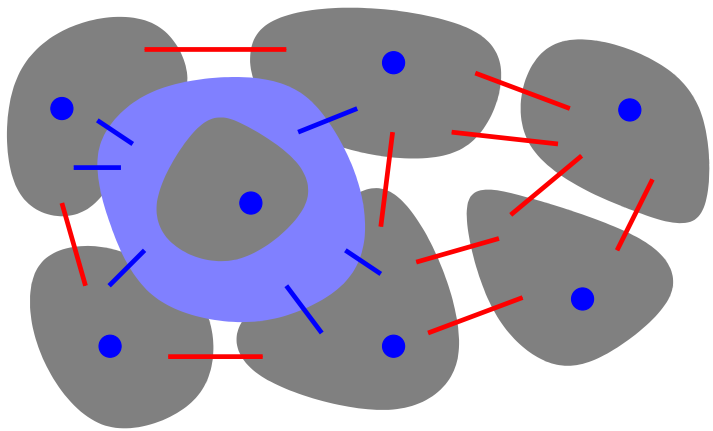
Idée :

- Si t est un terminal, alors toute solution contient un $(t, T \setminus \{t\})$-séparateur S,
- on peut choisir S de sorte que la composante connexe de $G \setminus S$ contenant t soit un $(t, T \setminus \{t\})$-ensemble important.

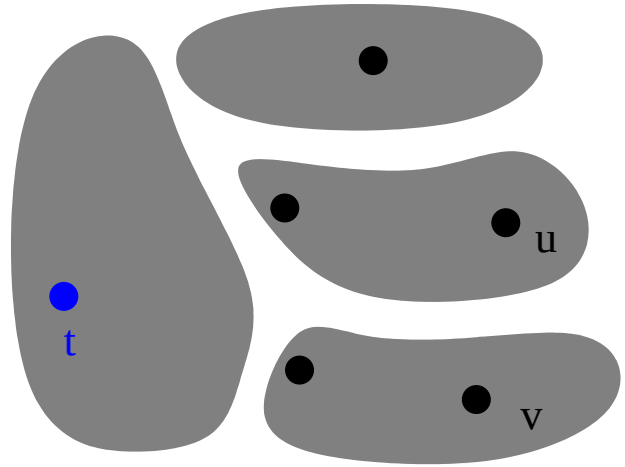


Idée :

- Si t est un terminal, alors toute solution contient un $(t, T \setminus \{t\})$-séparateur S,
- on peut choisir S de sorte que la composante connexe de $G \setminus S$ contenant t soit un $(t, T \setminus \{t\})$-ensemble important.

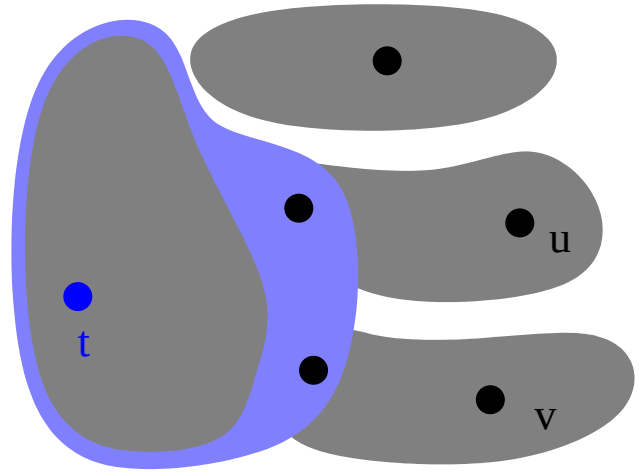


Lemme : $\forall t \in T$, Multicoupe paramétrée possède une solution contenant un $(t, T \setminus \{t\})$-séparateur important.



Soit X_t la composante connexe de $G \setminus S$ avec S une solution optimale

- si X_t n'est pas un $(t, T \setminus \{t\})$-ensemble important, alors il existe X tel que

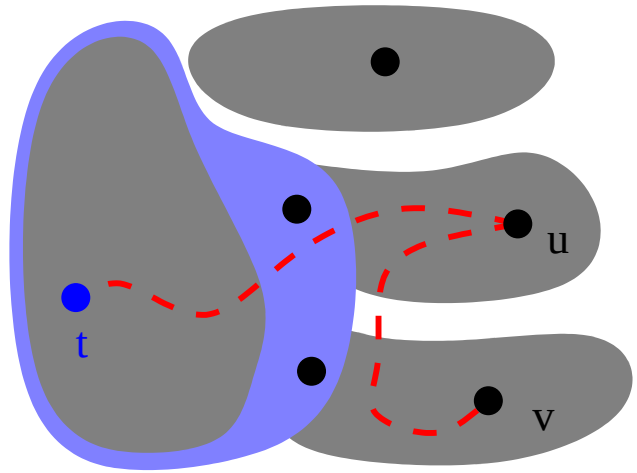


Soit X_t la composante connexe de $G \setminus S$ avec S une solution optimale

- si X_t n'est pas un $(t, T \setminus \{t\})$-ensemble important, alors il existe X tel que

  X soit un $(t, T \setminus \{t\})$-ensemble important et $X_t \subset X$

Observation : $S' = (S \setminus \partial(X_t)) \cup \partial(X)$ est une multicoupe optimale



Soit X_t la composante connexe de $G \setminus S$ avec S une solution optimale

- si X_t n'est pas un $(t, T \setminus \{t\})$-ensemble important, alors il existe X tel que

  X soit un $(t, T \setminus \{t\})$-ensemble important et $X_t \subset X$

Observation : $S' = (S \setminus \partial(X_t)) \cup \partial(X)$ est une multicoupe optimale

- par définition, $|S'| \leqslant |S|$
- $\forall u \in T \setminus \{t\}$, $G \setminus S'$ ne contient pas de chemin entre t et u
- Si $G \setminus S'$ contient un chemin entre u et v, $u, v \in T \setminus \{t\}$, alors $G \setminus S$ contient ce chemin

Algorithme pour Multicoupe paramétrée

On propose un algorithme de branchement sur les séparateurs importants :

Soit (G, k) une instance de Multicoupe paramétrée

Algorithme pour Multicoupe paramétrée

On propose un algorithme de branchement sur les séparateurs importants :

Soit (G, k) une instance de Multicoupe paramétrée

1. Si tout terminal de T appartient à une composante connexe différente de G ou $k \leqslant 0$, alors la solution est triviale

Algorithme pour Multicoupe paramétrée

On propose un algorithme de branchement sur les séparateurs importants :

Soit (G, k) une instance de Multicoupe paramétrée

1. Si tout terminal de T appartient à une composante connexe différente de G ou $k \leqslant 0$, alors la solution est triviale
2. Soit t un terminal non isolé de $T \setminus \{t\}$
3. Pour chaque $(t, T \setminus \{t\})$-séparateurs importants S de G, résoudre Multicoupe paramétrée sur $(G \setminus S, k - |S|)$

Algorithme pour Multicoupe paramétrée

On propose un algorithme de branchement sur les séparateurs importants :

Soit (G, k) une instance de Multicoupe paramétrée

1. Si tout terminal de T appartient à une composante connexe différente de G ou $k \leqslant 0$, alors la solution est triviale
2. Soit t un terminal non isolé de $T \setminus \{t\}$
3. Pour chaque $(t, T \setminus \{t\})$-séparateurs importants S de G, résoudre Multicoupe paramétrée sur $(G \setminus S, k - |S|)$

Théorème : Multicoupe paramétrée se résoud en $\boxed{4^k . n^{O(1)}}$

Algorithme pour Multicoupe paramétrée

On propose un algorithme de branchement sur les séparateurs importants :

Soit (G, k) une instance de Multicoupe paramétrée

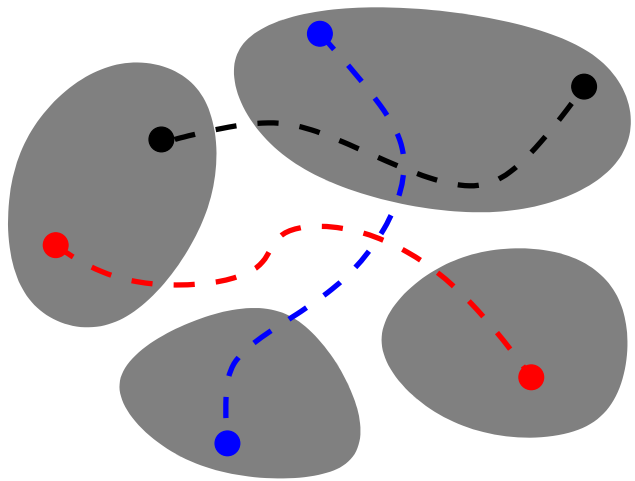
1. Si tout terminal de T appartient à une composante connexe différente de G ou $k \leqslant 0$, alors la solution est triviale
2. Soit t un terminal non isolé de $T \setminus \{t\}$
3. Pour chaque $(t, T \setminus \{t\})$-séparateurs importants S de G, résoudre Multicoupe paramétrée sur $(G \setminus S, k - |S|)$

Théorème : Multicoupe paramétrée se résoud en $\boxed{4^k . n^{O(1)}}$

- le degré de branchement est borné par 4^k, le nombre maximum de $(t, T \setminus \{t\})$-séparateurs importants
- au plus k étapes de branchement
- (une analyse plus fine donne la complexité annoncée)

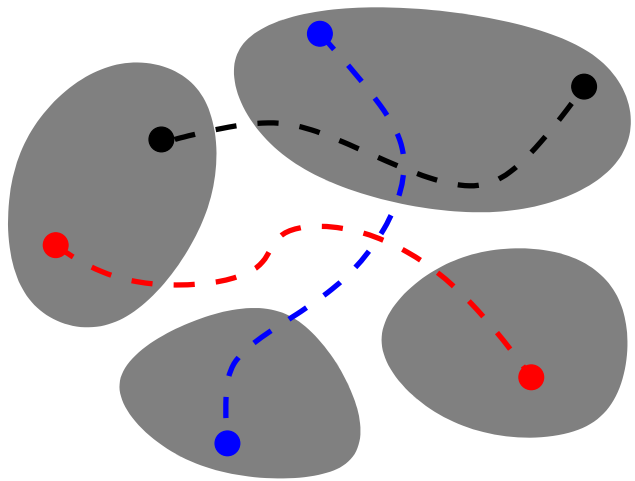
Multicoupe avec requêtes (Mutlicut)

- Un graphe G et un ensemble $\{(s_1, t_1), \dots (s_l, t_l)\}$ de requêtes entre des paires de terminaux
- Existe-t-il un ensemble S de k arêtes tel que $G \setminus S$ ne contient aucun chemin entre s_i et t_i $(\forall i \in [l])$?



Multicoupe avec requêtes (Mutlicut)

- Un graphe G et un ensemble $\{(s_1, t_1), \dots (s_l, t_l)\}$ de requêtes entre des paires de terminaux
- Existe-t-il un ensemble S de k arêtes tel que $G \setminus S$ ne contient aucun chemin entre s_i et t_i $(\forall i \in [l])$?

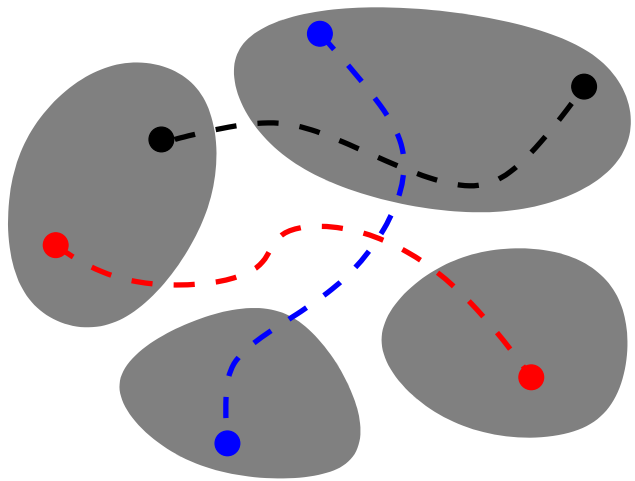


Théorème : Multicoupe avec requêtes est FPT paramétré par k et l

- Appliquer Multicoupe paramétrée sur toutes les partitions de $\{s_1, t_1, \dots s_l, t_l\}$ (les terminaux sont les parties)

Multicoupe avec requêtes (Mutlicut)

- Un graphe G et un ensemble $\{(s_1, t_1), \ldots (s_l, t_l)\}$ de requêtes entre des paires de terminaux
- Existe-t-il un ensemble S de k arêtes tel que $G \setminus S$ ne contient aucun chemin entre s_i et t_i $(\forall i \in [l])$?



Théorème [Bousquet, Daligault, Thomassé] [Marx,Razgon]
Multicoupe avec requêtes est FPT paramétré par k