

Cours 3: Flots et couplages

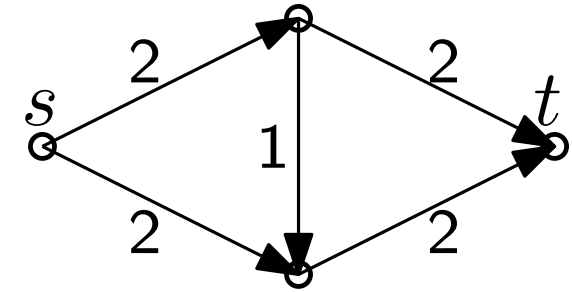
- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Cours 3: Flots et couplages

- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Réseau de transport et flots

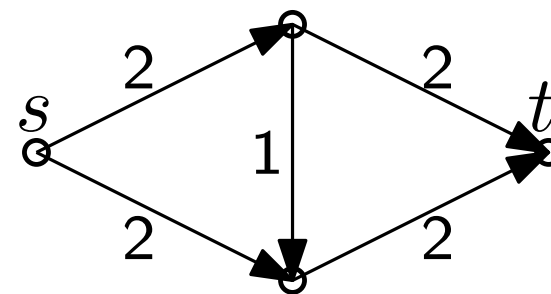
Données: Un graphe orienté $G = (X, A)$, une valuation $c : \mathcal{A} \rightarrow \mathbb{N}$, et 2 sommets s et t avec $d_{\text{in}}(s) = 0$ et $d_{\text{out}}(t) = 0$.



- Les sommets s et t sont la **source** et **puits**, $c(a)$ la **capacité** de l'arc a .

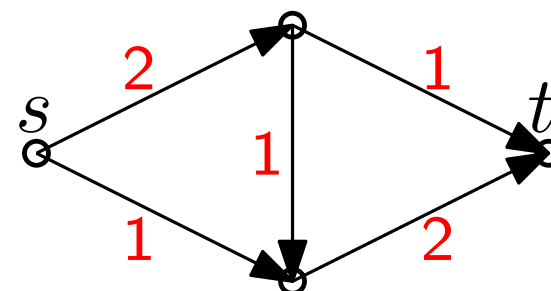
Réseau de transport et flots

Données: Un graphe orienté $G = (X, A)$, une valuation $c : A \rightarrow \mathbb{N}$, et 2 sommets s et t avec $d_{\text{in}}(s) = 0$ et $d_{\text{out}}(t) = 0$.



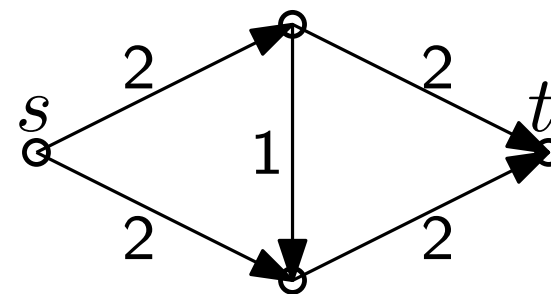
- Les sommets s et t sont la **source** et **puits**, $c(a)$ la **capacité** de l'arc a .
- Un **flot** est une application ϕ de A dans \mathbb{R}^+ telle que
 - le flot en chaque arc a est inférieur à la capacité: $\phi(a) \leq c(a)$
 - pour tout sommet de $X \setminus \{s, t\}$, flot entrant = flot sortant

La **valeur** du flot ϕ est le flot sortant de s (égal au flot entrant en t).



Réseau de transport et flots

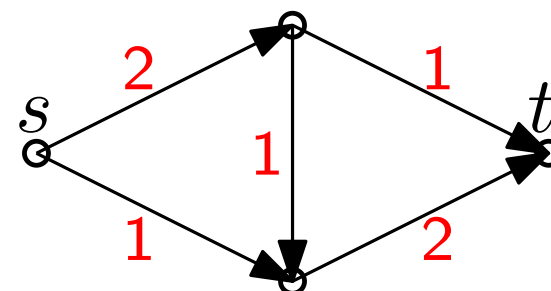
Données: Un graphe orienté $G = (X, A)$, une valuation $c : A \rightarrow \mathbb{N}$, et 2 sommets s et t avec $d_{\text{in}}(s) = 0$ et $d_{\text{out}}(t) = 0$.



- Les sommets s et t sont la **source** et **puits**, $c(a)$ la **capacité** de l'arc a .
- Un **flot** est une application ϕ de A dans \mathbb{R}^+ telle que
 - le flot en chaque arc a est inférieur à la capacité: $\phi(a) \leq c(a)$
 - pour tout sommet de $X \setminus \{s, t\}$, flot entrant = flot sortant

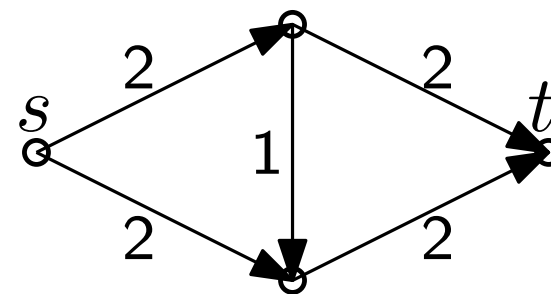
La **valeur** du flot ϕ est le flot sortant de s (égal au flot entrant en t).

Problème: Trouver un flot de valeur maximale.



Réseau de transport et flots

Données: Un graphe orienté $G = (X, A)$, une valuation $c : A \rightarrow \mathbb{N}$, et 2 sommets s et t avec $d_{\text{in}}(s) = 0$ et $d_{\text{out}}(t) = 0$.

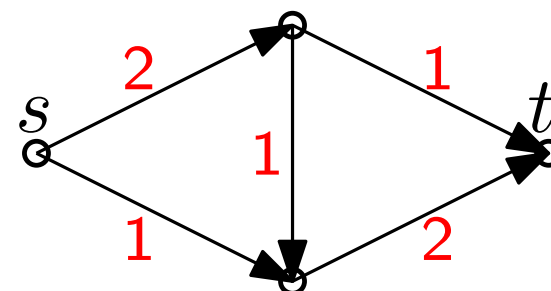


- Les sommets s et t sont la **source** et **puits**, $c(a)$ la **capacité** de l'arc a .
- Un **flot** est une application ϕ de A dans \mathbb{R}^+ telle que
 - le flot en chaque arc a est inférieur à la capacité: $\phi(a) \leq c(a)$
 - pour tout sommet de $X \setminus \{s, t\}$, flot entrant = flot sortant

La **valeur** du flot ϕ est le flot sortant de s (égal au flot entrant en t).

Problème: Trouver un flot de valeur maximale.

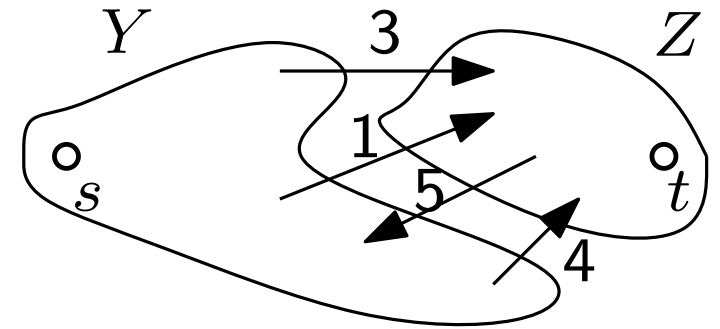
- Un flot ϕ est **saturé** si sur tout chemin de s à t il existe un arc a tel que $\phi(a) = c(a)$.



Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

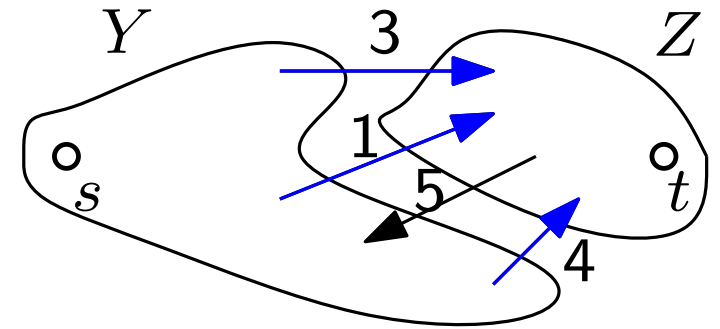


Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .

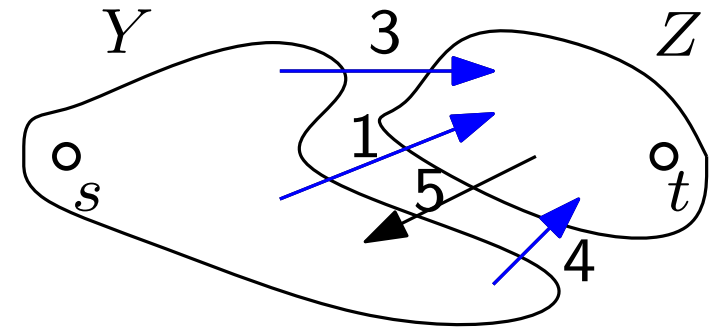


Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .



$$C(Y, Z) = 8$$

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

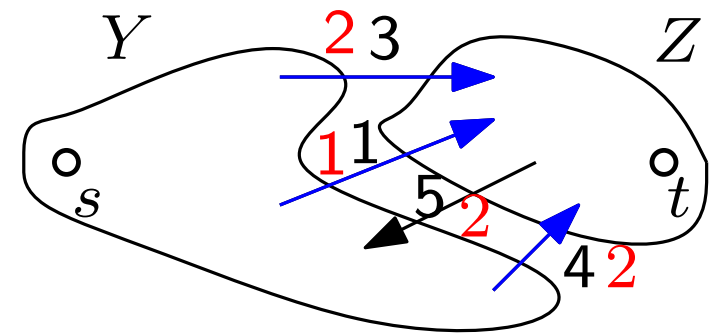
$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$



$$C(Y, Z) = 8$$

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

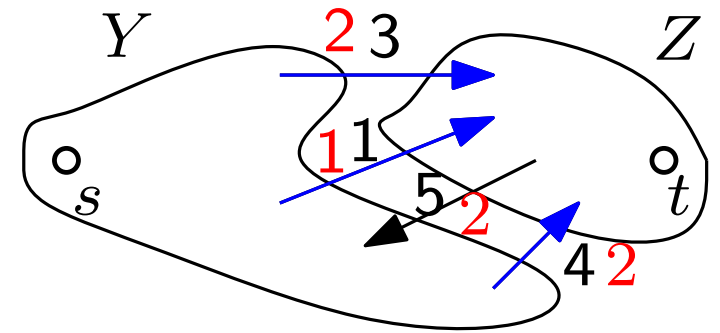
$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$



$$C(Y, Z) = 8$$

$$\phi(Y, Z) = 5$$

$$\Delta(Y, Z) = 3$$

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

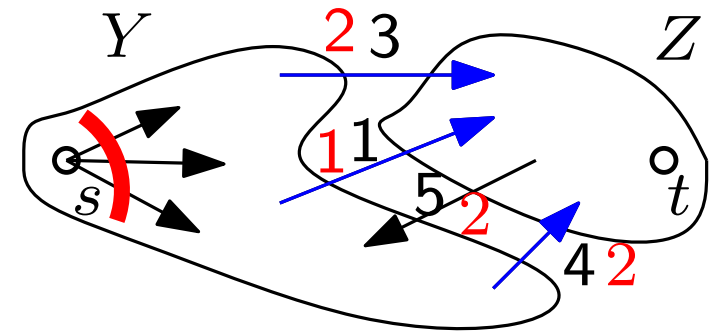
- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$

Tout flot a pour valeur $V_\phi = \phi(\{s\}, X \setminus \{s\}) = \Delta(\{s\}, X \setminus \{s\})$.



$$C(Y, Z) = 8$$

$$\phi(Y, Z) = 5$$

$$\Delta(Y, Z) = 3$$

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

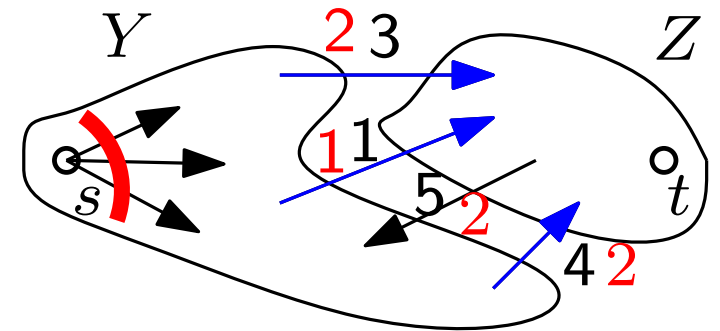
$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$

Tout flot a pour valeur $V_\phi = \phi(\{s\}, X \setminus \{s\}) = \Delta(\{s\}, X \setminus \{s\})$.

Lemme. Plus généralement $V_\phi = \Delta(Y, Z)$ pour toute coupe.



$$C(Y, Z) = 8$$

$$\phi(Y, Z) = 5$$

$$\Delta(Y, Z) = 3$$

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

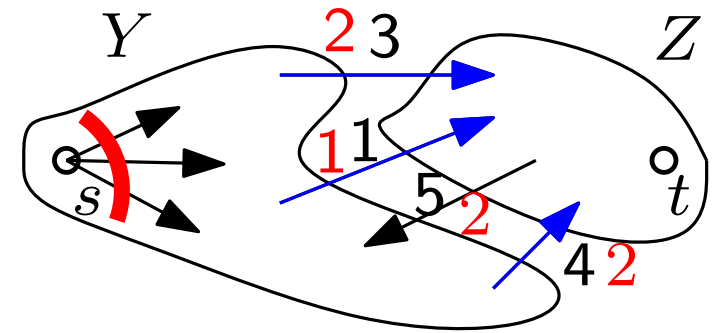
$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$



$$C(Y, Z) = 8$$

$$\phi(Y, Z) = 5$$

$$\Delta(Y, Z) = 3$$

Tout flot a pour valeur $V_\phi = \phi(\{s\}, X \setminus \{s\}) = \Delta(\{s\}, X \setminus \{s\})$.

Lemme. Plus généralement $V_\phi = \Delta(Y, Z)$ pour toute coupe.

Preuve par récurrence sur $|Y|$: $Y' = Y \cup \{y\}$, $i + i' = o + o'$.

Les coupes d'un réseau de transport

- Une **coupe** est une partition de l'ensemble des sommets en 2 parties disjointes, l'une contenant la source et l'autre le puit:

$$Y \cup Z = X, \quad Y \cap Z = \emptyset, \quad s \in Y, \quad t \in Z$$

- La **capacité** $C(Y, Z)$ d'une coupe est la somme des capacités des arcs de Y à Z .
- Le **flux** de Y à Z dans un flot ϕ est

$$\phi(Y, Z) = \sum_{u \in Y \times Z} \phi(u)$$

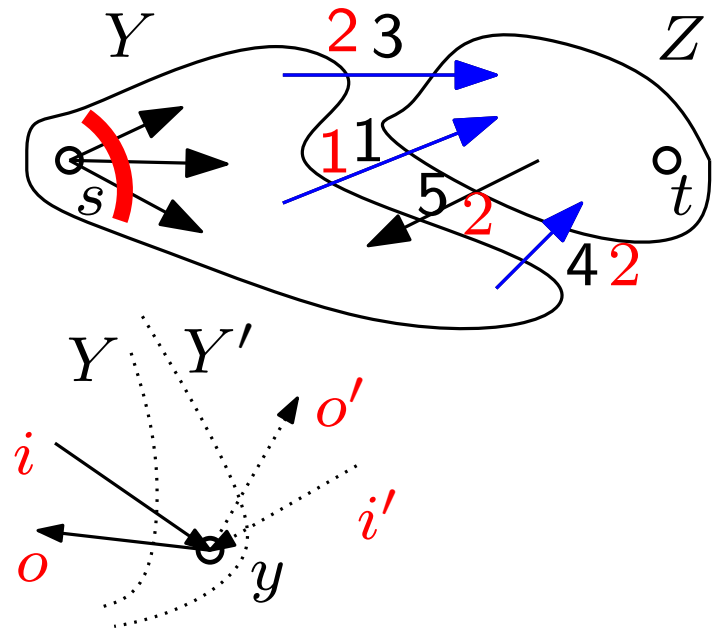
et le **flux orienté** de la coupe (Y, Z) est

$$\Delta(Y, Z) = \phi(Y, Z) - \phi(Z, Y).$$

Tout flot a pour valeur $V_\phi = \phi(\{s\}, X \setminus \{s\}) = \Delta(\{s\}, X \setminus \{s\})$.

Lemme. Plus généralement $V_\phi = \Delta(Y, Z)$ pour toute coupe.

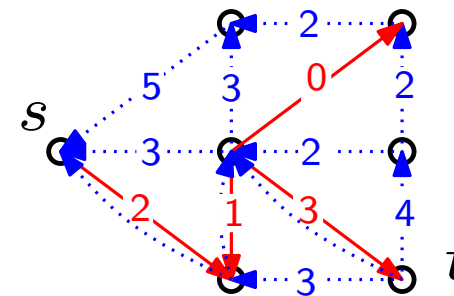
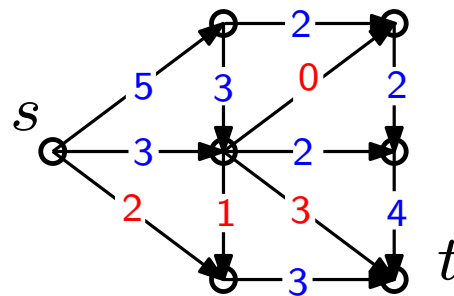
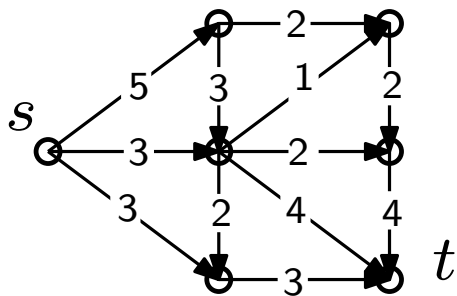
Preuve par récurrence sur $|Y|$: $Y' = Y \cup \{y\}$, $i + i' = o + o'$.



Le graphe des augmentations d'un flot

Le graphe des augmentations G_ϕ :

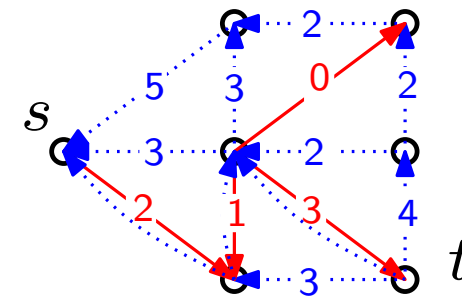
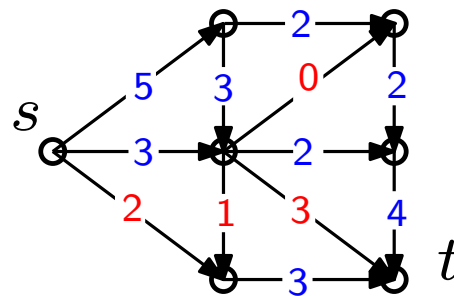
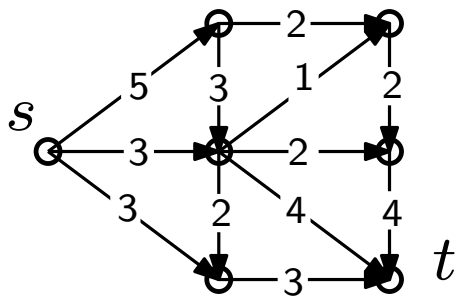
- Les sommets de G_ϕ sont ceux de G ,
- Tout arc $a = (x, y)$ de G reste dans G_ϕ si $\phi(a) < c(a)$.
- L'opposé (y, z) de l'arc $a = (x, y)$ de G est dans G_ϕ si $\phi(a) > 0$.



Le graphe des augmentations d'un flot

Le graphe des augmentations G_ϕ :

- Les sommets de G_ϕ sont ceux de G ,
- Tout arc $a = (x, y)$ de G reste dans G_ϕ si $\phi(a) < c(a)$.
- L'opposé (y, z) de l'arc $a = (x, y)$ de G est dans G_ϕ si $\phi(a) > 0$.

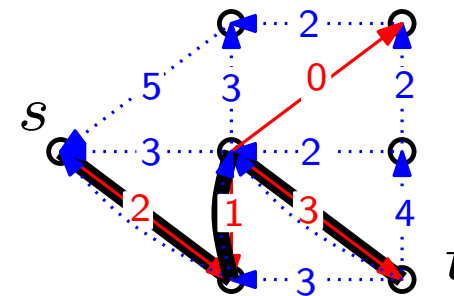
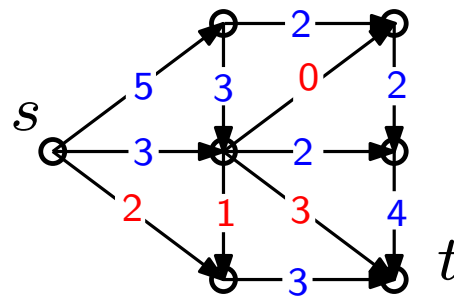
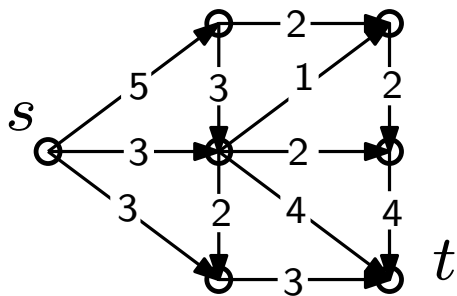


Lemme. S'il existe un chemin de s à t dans G_ϕ , le flot peut être amélioré.

Le graphe des augmentations d'un flot

Le graphe des augmentations G_ϕ :

- Les sommets de G_ϕ sont ceux de G ,
- Tout arc $a = (x, y)$ de G reste dans G_ϕ si $\phi(a) < c(a)$.
- L'opposé (y, z) de l'arc $a = (x, y)$ de G est dans G_ϕ si $\phi(a) > 0$.

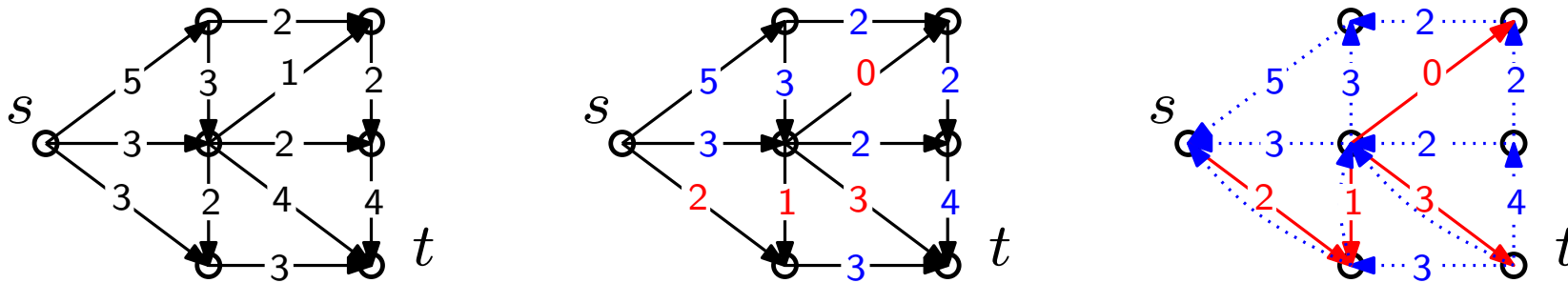


Lemme. S'il existe un chemin de s à t dans G_ϕ , le flot peut être amélioré.

Le graphe des augmentations d'un flot

Le graphe des augmentations G_ϕ :

- Les sommets de G_ϕ sont ceux de G ,
- Tout arc $a = (x, y)$ de G reste dans G_ϕ si $\phi(a) < c(a)$.
- L'opposé (y, z) de l'arc $a = (x, y)$ de G est dans G_ϕ si $\phi(a) > 0$.



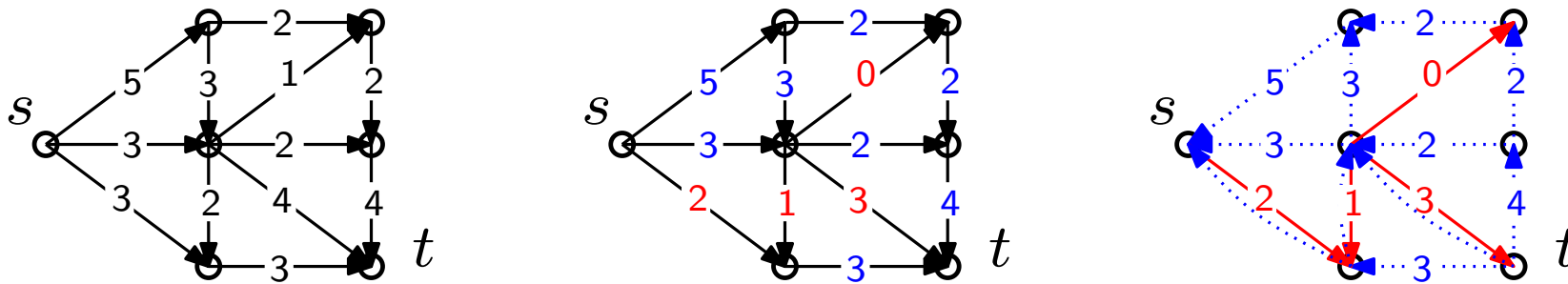
Lemme. S'il existe un chemin de s à t dans G_ϕ , le flot peut être amélioré.

Lemme. S'il n'existe pas de chemin de s à t dans G_ϕ , alors il existe une coupe (Y, Z) telle que $c(Y, Z) = V_\phi$.

Le graphe des augmentations d'un flot

Le graphe des augmentations G_ϕ :

- Les sommets de G_ϕ sont ceux de G ,
- Tout arc $a = (x, y)$ de G reste dans G_ϕ si $\phi(a) < c(a)$.
- L'opposé (y, z) de l'arc $a = (x, y)$ de G est dans G_ϕ si $\phi(a) > 0$.



Lemme. S'il existe un chemin de s à t dans G_ϕ , le flot peut être amélioré.

Lemme. S'il n'existe pas de chemin de s à t dans G_ϕ , alors il existe une coupe (Y, Z) telle que $c(Y, Z) = V_\phi$.

Théorème. Valeur du flot maximal = Capacité de la coupe minimale.

Cours 3: Flots et couplages

- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale.

Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale.

Remarques. ● Le flot maximal obtenu est à valeur entière.

Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale. **Complexité $O(m\Phi)$**

Remarques.

- Le flot maximal obtenu est à valeur entière.
- On a pas spécifié quel chemin on choisit à chaque étape.

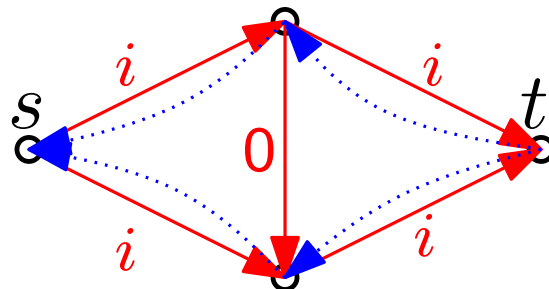
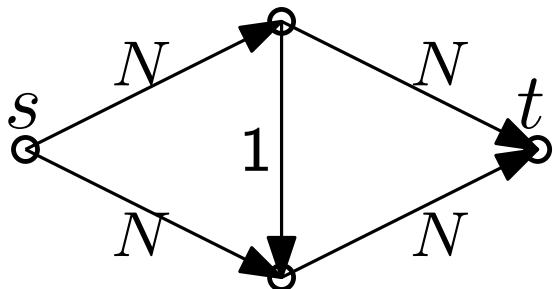
Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale. **Complexité $O(m\Phi)$**

Remarques. • Le flot maximal obtenu est à valeur entière.

- On a pas spécifié quel chemin on choisit à chaque étape.



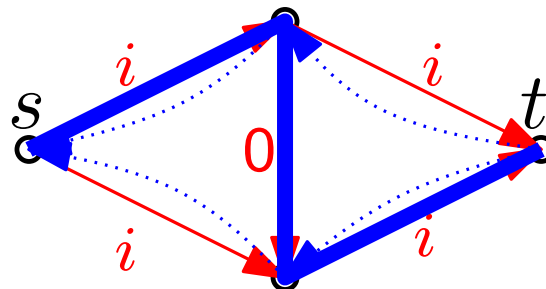
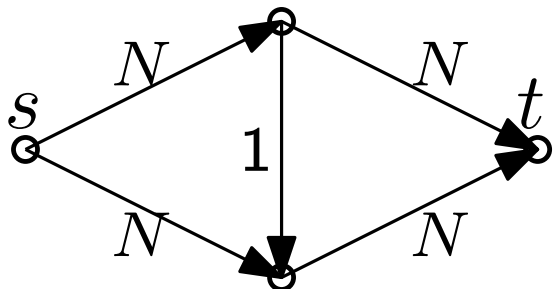
Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale. **Complexité $O(m\Phi)$**

Remarques. • Le flot maximal obtenu est à valeur entière.

- On a pas spécifié quel chemin on choisit à chaque étape.



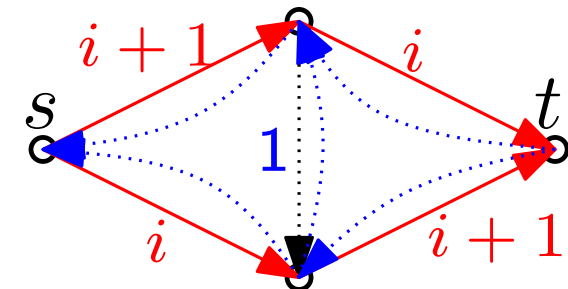
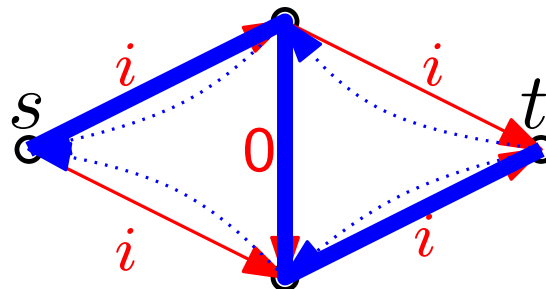
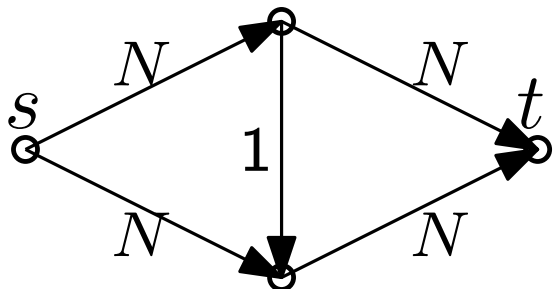
Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale. **Complexité $O(m\Phi)$**

Remarques. • Le flot maximal obtenu est à valeur entière.

- On a pas spécifié quel chemin on choisit à chaque étape.



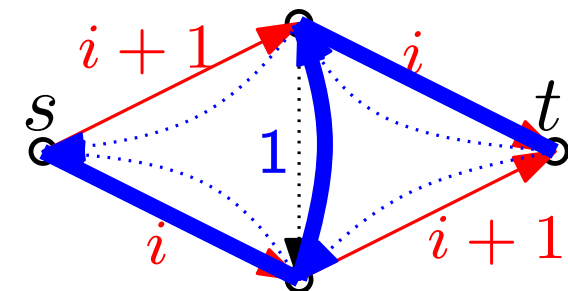
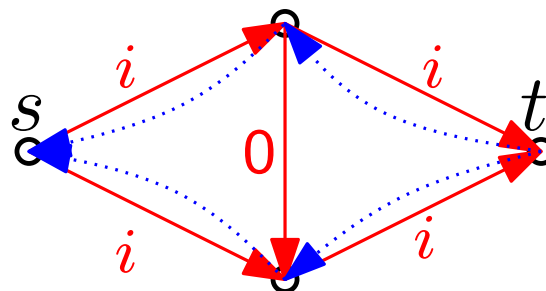
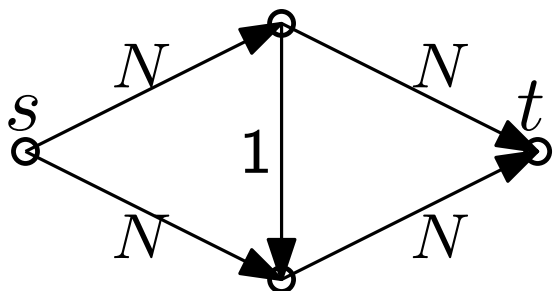
Algorithme de Ford Fulkerson

- Initialiser avec un flot quelconque (nul par exemple).
- Tant qu'il existe un chemin dans le graphe des augmentations, augmenter le flot.

Théorème. L'algorithme de Ford Fulkerson constitue un flot maximal. Les sommets atteignables depuis s dans le graphe des augmentations forment une coupe minimale. **Complexité $O(m\Phi)$**

Remarques. • Le flot maximal obtenu est à valeur entière.

- On a pas spécifié quel chemin on choisit à chaque étape.

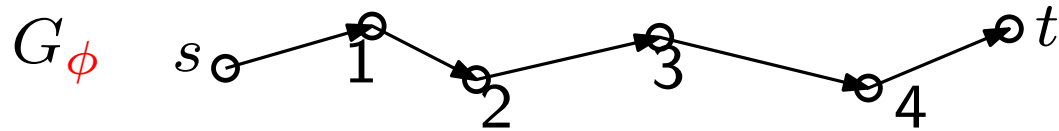


Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .

Algorithme de Dinic, Edmonds et Karp

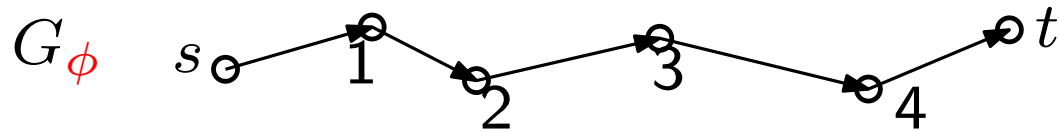
- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



autrement dit on choisit un plus court chemin d'augmentation.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .

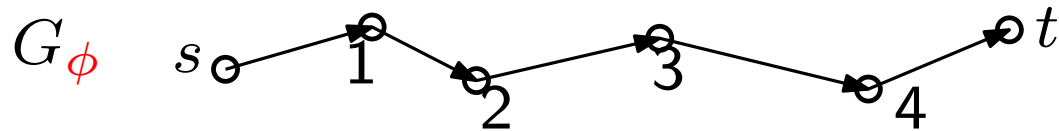


autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



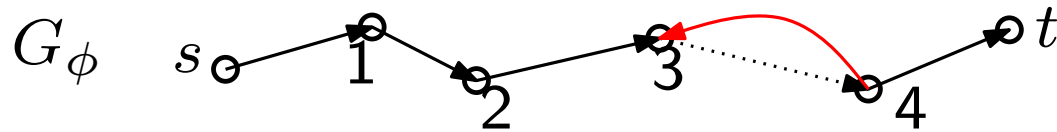
autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

Preuve. ● Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



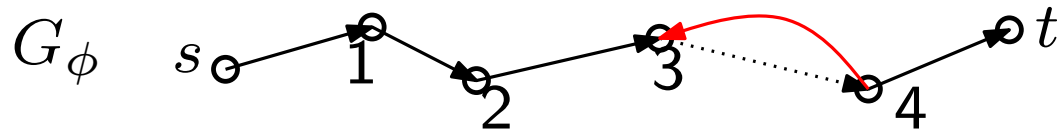
autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

Preuve. ● Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



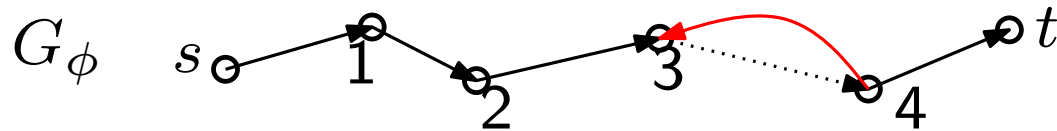
autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

- Preuve.**
- Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.
 - Quand l'arc (x, y) du chemin se retourne, les distances à s ne peuvent qu'augmenter. Si (y, x) revient sur le chemin ce sera plus loin.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



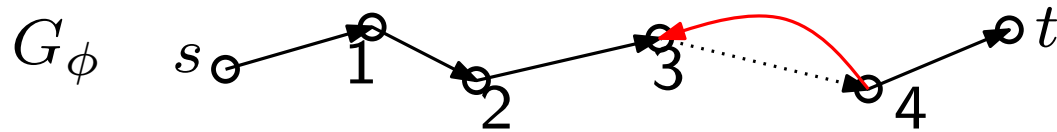
autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

- Preuve.**
- Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.
 - Quand l'arc (x, y) du chemin se retourne, les distances à s ne peuvent qu'augmenter. Si (y, x) revient sur le chemin ce sera plus loin.
 - Donc chacun des m arcs peut être retourné au plus n fois.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



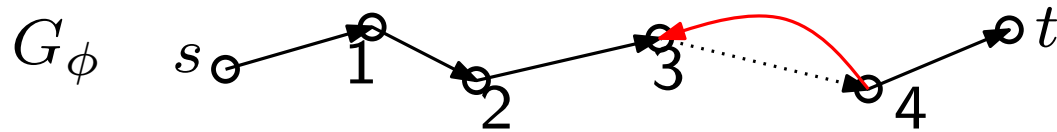
autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

- Preuve.**
- Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.
 - Quand l'arc (x, y) du chemin se retourne, les distances à s ne peuvent qu'augmenter. Si (y, x) revient sur le chemin ce sera plus loin.
 - Donc chacun des m arcs peut être retourné au plus n fois.
 - Le nombre de chemins d'augmentation utilisés est donc $O(nm)$.

Algorithme de Dinic, Edmonds et Karp

- Effectuer Ford-Fulkerson en choisissant à chaque étape le premier chemin d'augmentation obtenu par recherche en largeur dans G_ϕ .



autrement dit on choisit un plus court chemin d'augmentation.

Théorème. L'algorithme DEK effectue au plus $O(nm^2)$ opérations si le graphe G à n sommets et m arcs.

- Preuve.**
- Lorsqu'un chemin d'augmentation est utilisé, le flot devient nul ou maximal sur au moins l'une des arêtes: celle-ci est retournée.
 - Quand l'arc (x, y) du chemin se retourne, les distances à s ne peuvent qu'augmenter. Si (y, x) revient sur le chemin ce sera plus loin.
 - Donc chacun des m arcs peut être retourné au plus n fois.
 - Le nombre de chemins d'augmentation utilisés est donc $O(nm)$.
 - Le coût de la recherche d'un chemin d'augmentation est $O(m)$.

Cours 3: Flots et couplages

- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Répartition d'approvisionnement

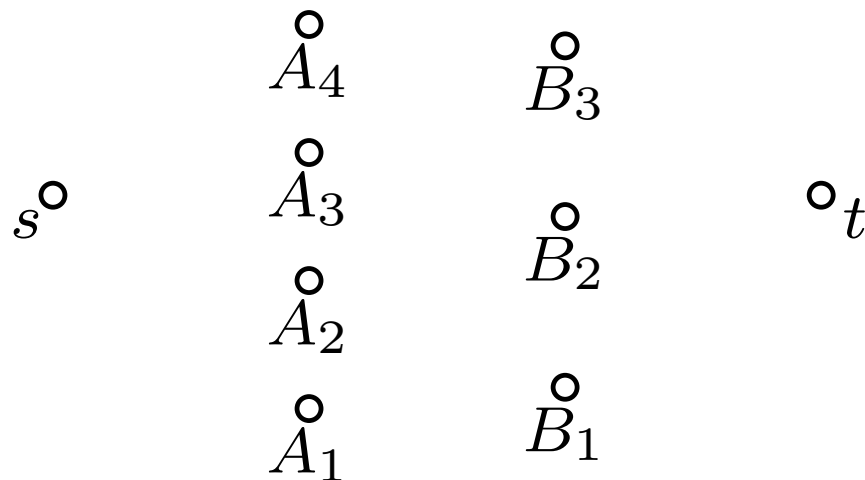
- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.
- Problème:** maximiser l'approvisionnement mutuel.

Répartition d'approvisionnement

- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.

Problème: maximiser l'approvisionnement mutuel.

Modélisation en flot: sommets $X = \{s, t, A_1, \dots, A_p, B_1, \dots, B_q\}$

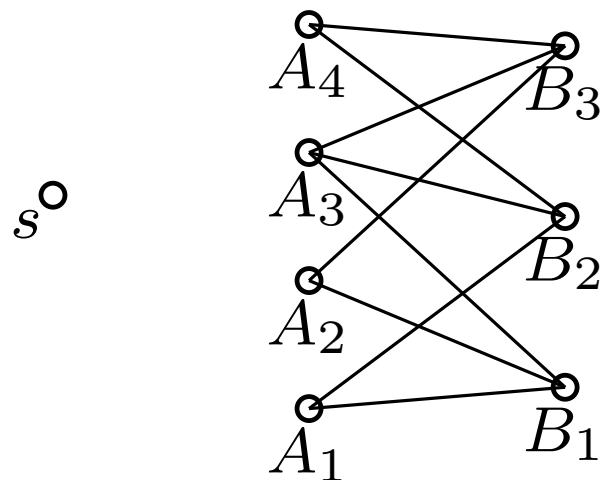


Répartition d'approvisionnement

- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.

Problème: maximiser l'approvisionnement mutuel.

Modélisation en flot: sommets $X = \{s, t, A_1, \dots, A_p, B_1, \dots, B_q\}$



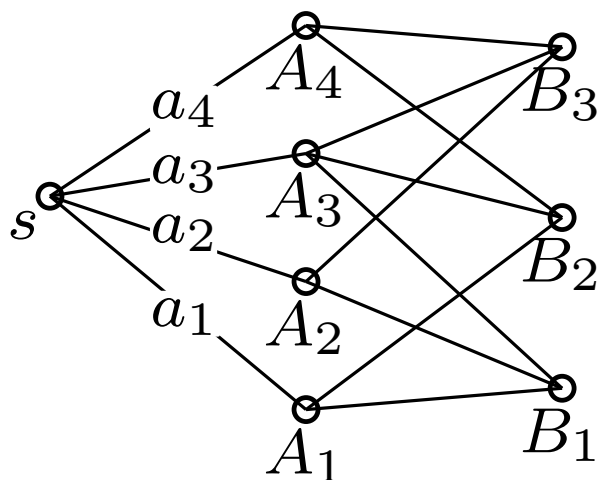
arête (A_i, B_j) si $i \in C_j$
de capacité ∞ .

Répartition d'approvisionnement

- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.

Problème: maximiser l'approvisionnement mutuel.

Modélisation en flot: sommets $X = \{s, t, A_1, \dots, A_p, B_1, \dots, B_q\}$



arête (A_i, B_j) si $i \in C_j$
de capacité ∞ .

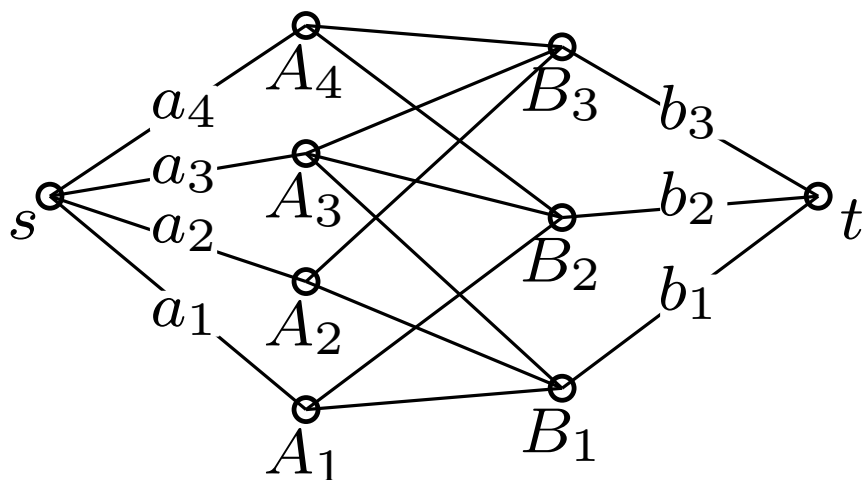
arête (s, A_i) de capacité a_i

Répartition d'approvisionnement

- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.

Problème: maximiser l'approvisionnement mutuel.

Modélisation en flot: sommets $X = \{s, t, A_1, \dots, A_p, B_1, \dots, B_q\}$



arête (A_i, B_j) si $i \in C_j$
de capacité ∞ .

arête (s, A_i) de capacité a_i

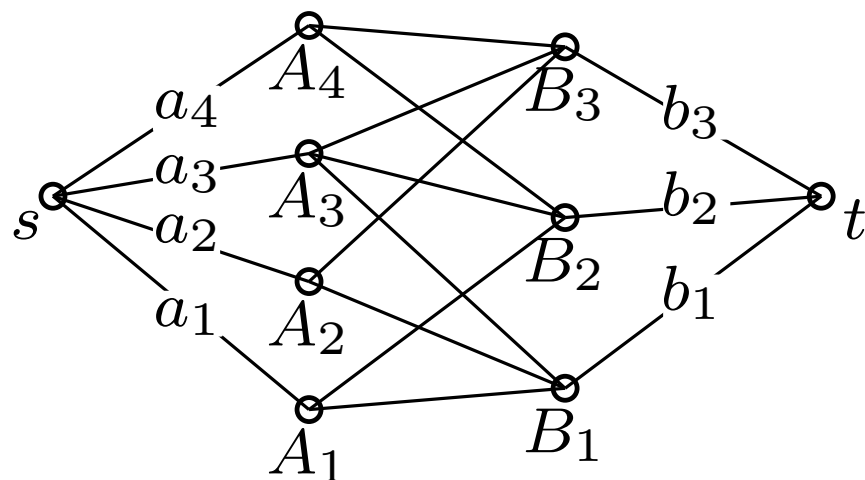
arête (B_j, t) de capacité b_j

Répartition d'approvisionnement

- Données:**
- les fournisseurs A_1, \dots, A_p , avec A_i de capacité a_i .
 - les clients B_1, \dots, B_q , le client B_i demande une quantité b_i .
 - les contraintes: B_j ne peut être livré que par les $\{A_i\}_{i \in C_j}$.

Problème: maximiser l'approvisionnement mutuel.

Modélisation en flot: sommets $X = \{s, t, A_1, \dots, A_p, B_1, \dots, B_q\}$



arête (A_i, B_j) si $i \in C_j$
de capacité ∞ .

arête (s, A_i) de capacité a_i

arête (B_j, t) de capacité b_j

Approvisionnement mutuel maximal = flot maximal dans ce graphe.

Problème de l'expédition idéale

- Données:**
- les instruments A_1, \dots, A_p , avec A_i de coût c_i .
 - des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .
 - pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Problème de l'expédition idéale

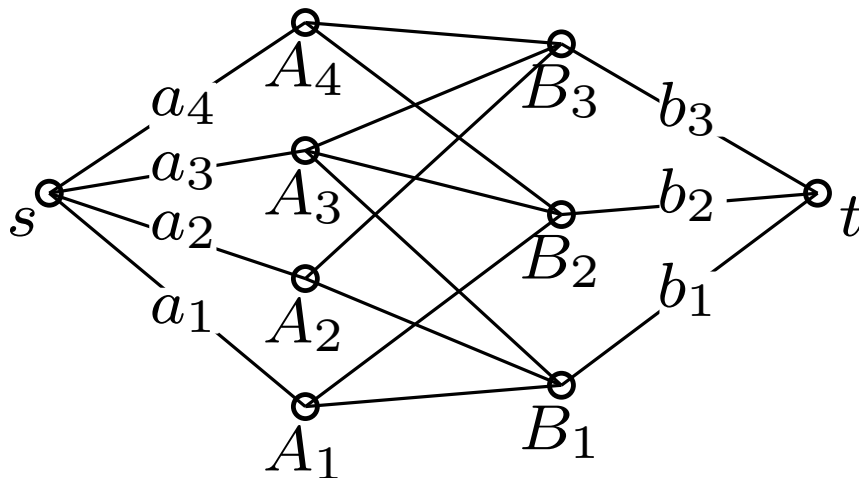
Données: • les instruments A_1, \dots, A_p , avec A_i de coût c_i .

• des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .

• pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Modélisation en flot: on utilise le même graphe



Problème de l'expédition idéale

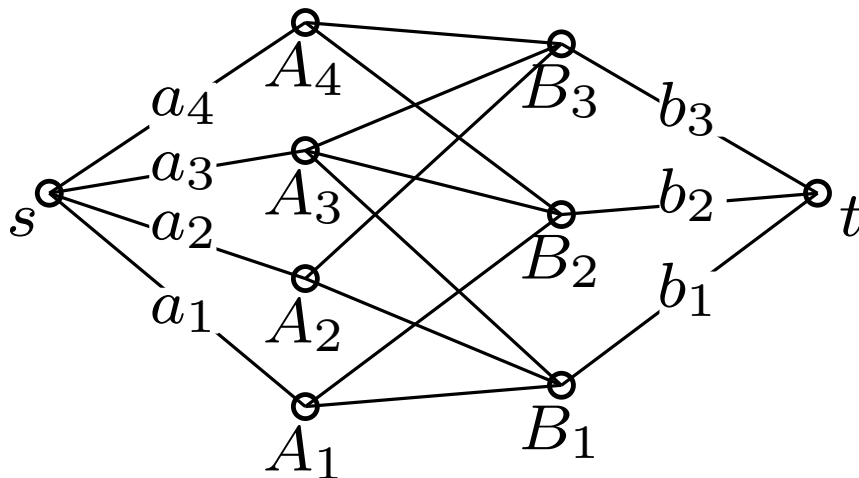
Données: • les instruments A_1, \dots, A_p , avec A_i de coût c_i .

• des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .

• pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Modélisation en flot: on utilise le même graphe



Gain maximal = $\sum b_j - c(Y, Z)$ pour la coupe de capacité minimale.

Problème de l'expédition idéale

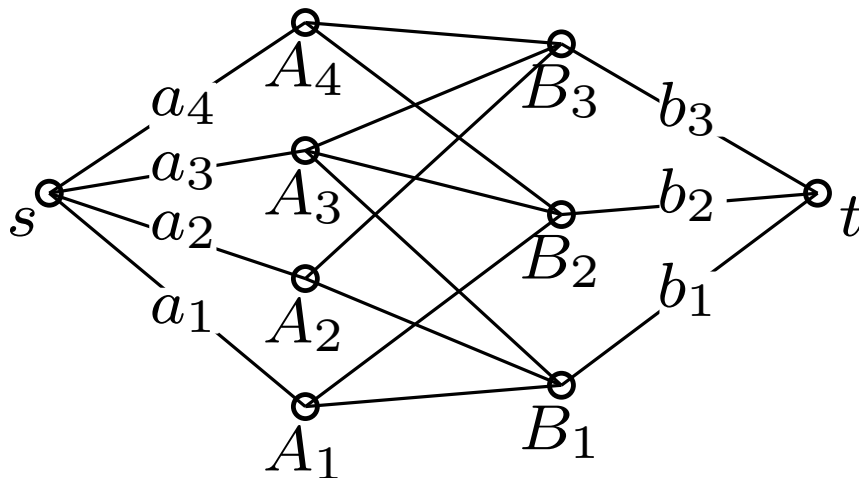
Données: • les instruments A_1, \dots, A_p , avec A_i de coût c_i .

• des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .

• pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Modélisation en flot: on utilise le même graphe



une coupe minimale doit éviter les arcs (A_i, B_j)

Gain maximal = $\sum b_j - c(Y, Z)$ pour la coupe de capacité minimale.

Problème de l'expédition idéale

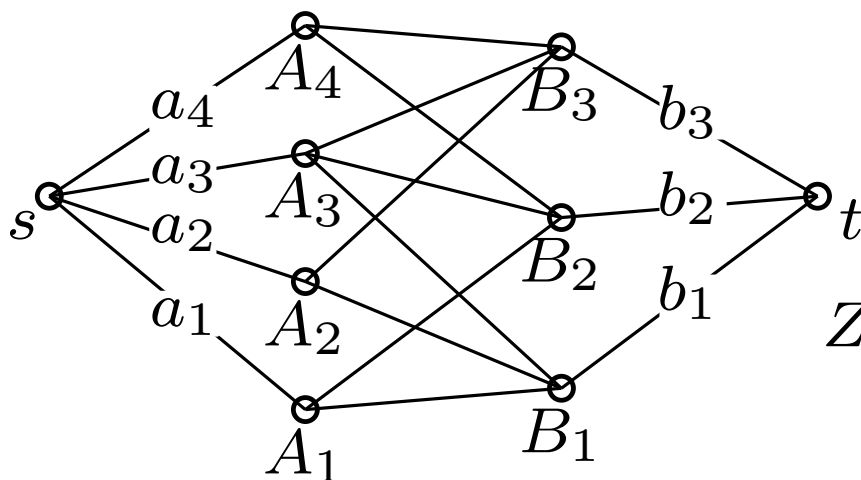
Données: • les instruments A_1, \dots, A_p , avec A_i de coût c_i .

• des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .

• pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Modélisation en flot: on utilise le même graphe



une coupe minimale doit éviter les arcs (A_i, B_j)

une expédition \Leftrightarrow une coupe (Y, Z) :
 $Z = \{A_i \text{ emportés}\} \cup \{B_i \text{ effectués}\} \cup \{z\}$

Gain maximal = $\sum b_j - c(Y, Z)$ pour la coupe de capacité minimale.

Problème de l'expédition idéale

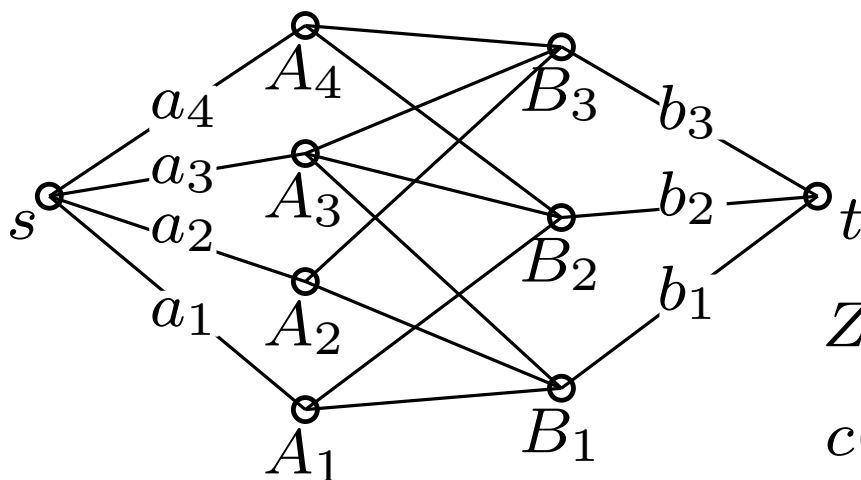
Données: • les instruments A_1, \dots, A_p , avec A_i de coût c_i .

• des expériences B_1, \dots, B_q à faire, B_i rapporte b_i .

• pour pouvoir effectuer B_j il faut avoir $\{A_i\}_{i \in C_j}$.

Problème: maximiser le gain de l'expédition.

Modélisation en flot: on utilise le même graphe



une coupe minimale doit éviter les arcs (A_i, B_j)

une expédition \Leftrightarrow une coupe (Y, Z) :

$$Z = \{A_i \text{ emportés}\} \cup \{B_i \text{ effectués}\} \cup \{z\}$$

$$c(Y, Z) = \sum_{A_i \in Z} a_i + \sum_{B_i \notin Z} b_j$$

Gain maximal = $\sum b_j - c(Y, Z)$ pour la coupe de capacité minimale.

Cours 3: Flots et couplages

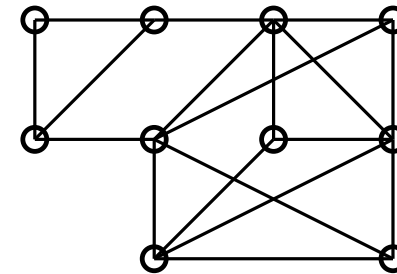
- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

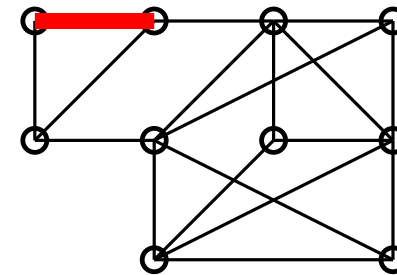


Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

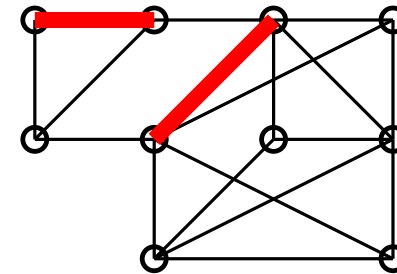


Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

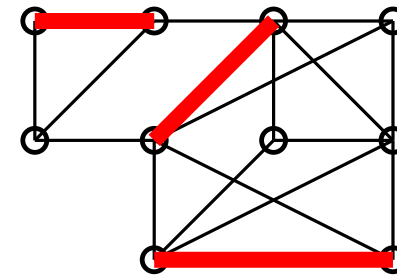


Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

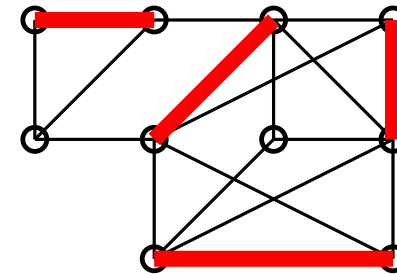


Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

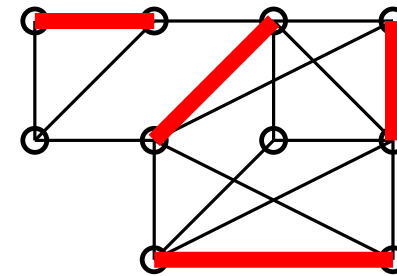


Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.



maximal pour l'inclusion

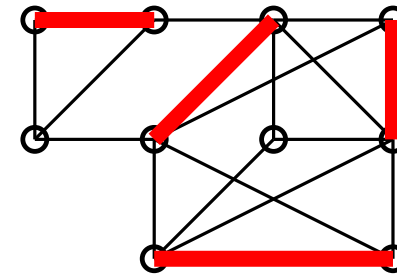
Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.



maximal pour l'inclusion

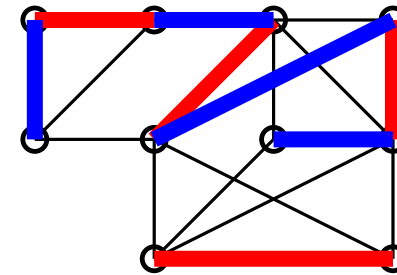
Couplages dans un graphe non orienté

Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.



maximal pour l'inclusion

Couplages dans un graphe non orienté

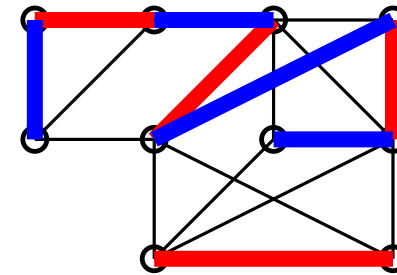
Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.

Théorème. Couplage maximum \Leftrightarrow pas de chemins alternants entre 2 sommets libres.



maximal pour l'inclusion

Couplages dans un graphe non orienté

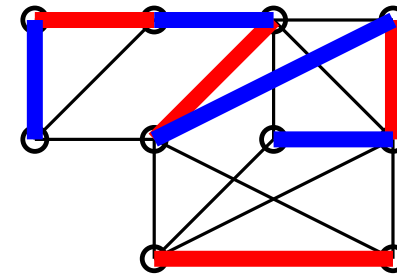
Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.

Théorème. Couplage maximum \Leftrightarrow pas de chemins alternants entre 2 sommets libres.



maximal pour l'inclusion

On peut trouver les chemins alternants en temps polynomial (Edmonds).

Couplages dans un graphe non orienté

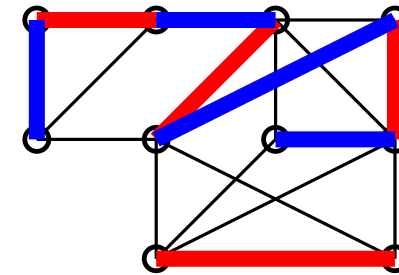
Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.

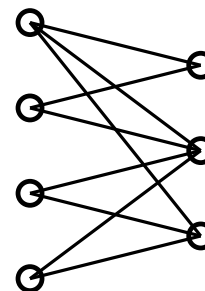
Théorème. Couplage maximum \Leftrightarrow pas de chemins alternants entre 2 sommets libres.



maximal pour l'inclusion

On peut trouver les chemins alternants en temps polynomial (Edmonds).

Pour les graphes bipartis, on se ramène à un problème de flot avec des arcs de capacité 1.



Couplages dans un graphe non orienté

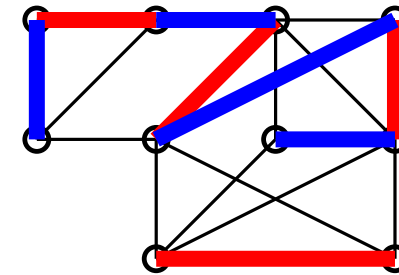
Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.

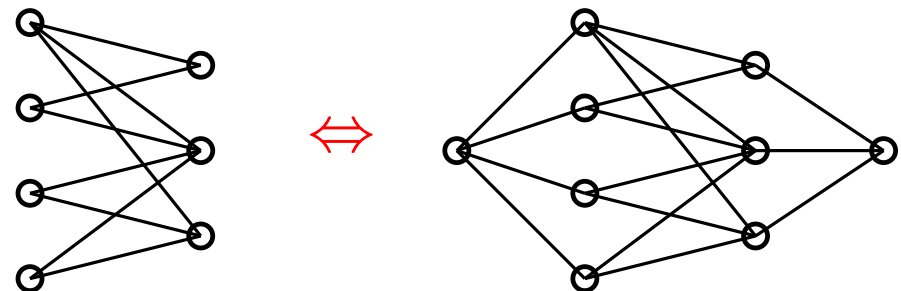
Théorème. Couplage maximum \Leftrightarrow pas de chemins alternants entre 2 sommets libres.



maximal pour l'inclusion

On peut trouver les chemins alternants en temps polynomial (Edmonds).

Pour les graphes bipartis, on se ramène à un problème de flot avec des arcs de capacité 1.



Couplages dans un graphe non orienté

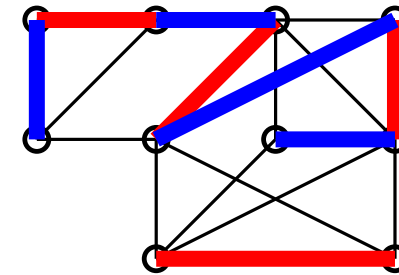
Données: un graphe non orienté (ou symétrique)

Un **couplage** est un ensemble d'arêtes sans extrémités communes

Problème: déterminer un couplage de cardinalité maximale.

Un **chemin alternant** alterne les arêtes libres et couplées.

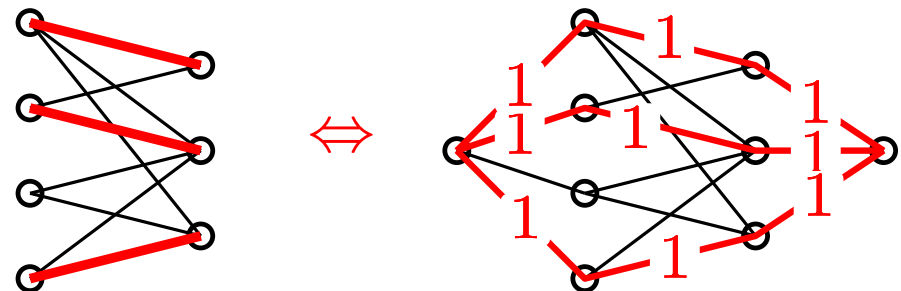
Théorème. Couplage maximum \Leftrightarrow pas de chemins alternants entre 2 sommets libres.



maximal pour l'inclusion

On peut trouver les chemins alternants en temps polynomial (Edmonds).

Pour les graphes bipartis, on se ramène à un problème de flot avec des arcs de capacité 1.



INF-550-3: Flots et couplages

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Lemme. \mathcal{F} est un matroïde.

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Lemme. \mathcal{F} est un matroïde.

Preuve. Si $|Y''| > |Y'|$ on superpose les couplages associés à Y' et Y'' pour trouver une chaîne alternante qui permet d'étendre Y' .

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Lemme. \mathcal{F} est un matroïde.

Preuve. Si $|Y''| > |Y'|$ on superpose les couplages associés à Y' et Y'' pour trouver une chaîne alternante qui permet d'étendre Y' .

Corollaire (Cours 1). Les transversaux d'un ensemble de parties forment un matroïde.

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Lemme. \mathcal{F} est un matroïde.

Preuve. Si $|Y''| > |Y'|$ on superpose les couplages associés à Y' et Y'' pour trouver une chaîne alternante qui permet d'étendre Y' .

Corollaire (Cours 1). Les transversaux d'un ensemble de parties forment un matroïde.

Remarques. • La famille des couplages ne forme pas un matroïde.
mais c'est l'intersection de 2 matroïdes: les ensembles d'arêtes sans extrémité commune dans Y et dans Z respectivement.

Couplages et matroïdes

Soit $G = (Y \cup Z, E)$ un graphe biparti.

La famille \mathcal{F} des **parties couplables de Y** est l'ensemble des $Y' \subset Y$ tels qu'il existe un couplage parfait entre Y' et un $Z' \subset Z$.

Lemme. \mathcal{F} est un matroïde.

Preuve. Si $|Y''| > |Y'|$ on superpose les couplages associés à Y' et Y'' pour trouver une chaîne alternante qui permet d'étendre Y' .

Corollaire (Cours 1). Les transversaux d'un ensemble de parties forment un matroïde.

Remarques.

- La famille des couplages ne forme pas un matroïde.
mais c'est l'intersection de 2 matroïdes: les ensembles d'arêtes sans extrémité commune dans Y et dans Z respectivement.
- Par augmentation on a trouvé un élément max de cette intersection.

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme glouton: $T := \emptyset$; tant qu'il existe une arête e d'une couleur n'apparaissant pas dans T , et ne créant pas de cycle, faire $T := T \cup \{e\}$.

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme glouton: $T := \emptyset$; tant qu'il existe une arête e d'une couleur n'apparaissant pas dans T , et ne créant pas de cycle, faire $T := T \cup \{e\}$.

Ça ne donne pas l'optimum !

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme Améliorant. Initialiser $T := \emptyset$ et itérer:

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme Améliorant. Initialiser $T := \emptyset$ et itérer:

- Soit $D(T)$ le graphe d'amélioration (E, X) biparti avec arêtes:
 - de $a \in T$ à $b \in E \setminus T$ si $T - a + b$ est sans cycle
 - de $b \in E \setminus T$ à $a \in T$ si $T - a + b$ est multicolore

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme Améliorant. Initialiser $T := \emptyset$ et itérer:

- Soit $D(T)$ le graphe d'amélioration (E, X) biparti avec arêtes:
 - de $a \in T$ à $b \in E \setminus T$ si $T - a + b$ est sans cycle
 - de $b \in E \setminus T$ à $a \in T$ si $T - a + b$ est multicolore
- Soit $E_1 = \{b \mid T + b \text{ sans cycle}\}$, $E_2 = \{b \mid T + b \text{ multicolore}\}$

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme Améliorant. Initialiser $T := \emptyset$ et itérer:

- Soit $D(T)$ le graphe d'amélioration (E, X) biparti avec arêtes:
 - de $a \in T$ à $b \in E \setminus T$ si $T - a + b$ est sans cycle
 - de $b \in E \setminus T$ à $a \in T$ si $T - a + b$ est multicolore
- Soit $E_1 = \{b \mid T + b \text{ sans cycle}\}$, $E_2 = \{b \mid T + b \text{ multicolore}\}$
- Si $D(T)$ a un chemin C de $E_1 \setminus T$ à $E_2 \setminus T$
faire $T := T \Delta C_{\min}$ où C_{\min} est un plus court tel chemin.

Intersection de matroïdes

Données: un graphe à n sommets dont les arêtes sont partitionnées en E_1, \dots, E_k sous-ensembles disjoints (couleurs).

Problème: Trouver une forêt multicolore maximale.

Remarque: ces forêts sont dans l'intersection de 2 matroïdes.

Algorithme Améliorant. Initialiser $T := \emptyset$ et itérer:

- Soit $D(T)$ le graphe d'amélioration (E, X) biparti avec arêtes:
 - de $a \in T$ à $b \in E \setminus T$ si $T - a + b$ est sans cycle
 - de $b \in E \setminus T$ à $a \in T$ si $T - a + b$ est multicolore
- Soit $E_1 = \{b \mid T + b \text{ sans cycle}\}$, $E_2 = \{b \mid T + b \text{ multicolore}\}$
- Si $D(T)$ a un chemin C de $E_1 \setminus T$ à $E_2 \setminus T$
faire $T := T \Delta C_{\min}$ où C_{\min} est un plus court tel chemin.

Théorème. L'algorithme Améliorant trouve l'indépendant maximal dans l'intersection de 2 matroïdes.

Cours 3: Flots et couplages

- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Mariages stables

Données: • Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$

Mariages stables

- Données:
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .

Mariages stables

- Données:
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.

Mariages stables

- Données:
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.

Mariages stables

- Données:**
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.
 - Un couplage dans ce graphe décrit un (multi)mariage acceptable. Si x est couplé à y on note $\alpha(x) = y$ et $\beta(y) = x$.

Mariages stables

- Données:**
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.
 - Un couplage dans ce graphe décrit un (multi)mariage acceptable. Si x est couplé à y on note $\alpha(x) = y$ et $\beta(y) = x$.

exemple: $X = \{x_1, x_2, x_3\}$ et $Y = \{y_1, y_2, y_3\}$

F_1	y_2	y_1	y_3	G_1	x_1	x_3	x_2
F_2	y_1	y_3	y_2	G_2	x_3	x_1	x_2
F_3	y_1	y_2	y_3	G_3	x_1	x_3	x_2

Mariages stables

- Données:**
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.
 - Un couplage dans ce graphe décrit un (multi)mariage acceptable. Si x est couplé à y on note $\alpha(x) = y$ et $\beta(y) = x$.

exemple: $X = \{x_1, x_2, x_3\}$ et $Y = \{y_1, y_2, y_3\}$

F_1	y_2	y_1	y_3	G_1	x_1	x_3	x_2
F_2	y_1	y_3	y_2	G_2	x_3	x_1	x_2
F_3	y_1	y_2	y_3	G_3	x_1	x_3	x_2

- Un mariage stable est un couplage tel que si x et y ne sont pas couplés alors x préfère $\alpha(x)$ à y ou y préfère $\beta(y)$ à x .

Mariages stables

- Données:**
- Deux ensembles $X = \{x_1, \dots, x_p\}$ et $Y = \{y_1, \dots, y_p\}$
 - Les listes de préférences F_i de x_i et G_j de y_j pour tous i, j .
 - Le graphe des mariages est le graphe biparti sur $X \cup Y$ avec une arête (x_i, y_j) si $x_i \in G_j$ et $y_j \in F_i$.
 - Un couplage dans ce graphe décrit un (multi)mariage acceptable. Si x est couplé à y on note $\alpha(x) = y$ et $\beta(y) = x$.

exemple: $X = \{x_1, x_2, x_3\}$ et $Y = \{y_1, y_2, y_3\}$

F_1	y_2	y_1	y_3
F_2	y_1	y_3	y_2
F_3	y_1	y_2	y_3

G_1	x_1	x_3	x_2
G_2	x_3	x_1	x_2
G_3	x_1	x_3	x_2

$(x_1, y_1), (x_2, y_2), (x_3, y_3)$
instable: $x_1 \leftrightarrow y_2$
 $(x_1, y_1), (x_2, y_3), (x_3, y_2)$
stable

- Un mariage stable est un couplage tel que si x et y ne sont pas couplés alors x préfère $\alpha(x)$ à y ou y préfère $\beta(y)$ à x .

Mariages stables, algorithme de Gale Shapley.

- Initialiser $\alpha(x_i)$ au premier élément de la liste F_i , et $\beta(y)$ indéfini.
- Itérer:
 1. pour chaque y_j choisir l'élément x_i le plus intéressant dans G_j parmi les x_i tels que $\alpha(x_i) = y_j$ et poser $\beta(y_j) = x_i$.
 2. pour chaque x_i
 - s'il existe y_j tel que $\beta(y_j) = x_i$, alors $\alpha(x_i) = y_j$,
 - sinon, si $\alpha(x_i)$ n'est pas le dernier dans F_i , descendre d'un rang
 - sinon laisser $\alpha(x)$ indéfini.

Mariages stables, algorithme de Gale Shapley.

- Initialiser $\alpha(x_i)$ au premier élément de la liste F_i , et $\beta(y)$ indéfini.
- Itérer:
 1. pour chaque y_j choisir l'élément x_i le plus intéressant dans G_j parmi les x_i tels que $\alpha(x_i) = y_j$ et poser $\beta(y_j) = x_i$.
 2. pour chaque x_i
 - s'il existe y_j tel que $\beta(y_j) = x_i$, alors $\alpha(x_i) = y_j$,
 - sinon, si $\alpha(x_i)$ n'est pas le dernier dans F_i , descendre d'un rang
 - sinon laisser $\alpha(x)$ indéfini.

Théorème. Si les toutes les listes de préférences sont complètes, l'algorithme de Gale-Shapley constitue un mariage stable.

Cours 3: Flots et couplages

- Flots et coupes
- Algorithmes de calcul du flot maximal
- Modélisation par flots
- Couplages et graphes des augmentations
- Mariages stables

Retenir: **Modélisation par flots et coupes, algorithmes de Ford-Fulkerson/Edmonds Karp, dualité MaxFlow=MinCut**