

1. Théorie et algorithmes

Flots maximum

Considérons un graphe orienté $R=(V,E)$ contenant un sommet s sans prédécesseur, appelé **source**, et un sommet t sans successeur, appelé **puits**. Un **flot** dans R est une fonction f qui attribue un entier $f(e)$ à tout arc de R de telle sorte qu'on ait **conservation du flot** en chaque sommet v de R autre que la source et le puits, c'est-à-dire que la somme des valeurs sur les arcs entrant en v est égale à la somme des valeurs sur les arcs sortant de v . La somme des valeurs sur les arcs sortant de la source s est donc égale à la somme des valeurs sur les arcs entrant dans le puits t , et cette somme, notée $f_{s \rightarrow t}$ est la **quantité de flot circulant de s vers t** .

Une **capacité** c_e est associée à chaque arc e de R , et il est souhaité que la quantité de flot qui circule sur chaque arc e soit positive et au plus égale à c_e . Un flot qui respecte ces contraintes est dit **compatible**. Le problème du flot maximum consiste à déterminer un flot compatible de s à t qui maximise $f_{s \rightarrow t}$. On dit que **le flot est maximum**. Une formulation en programmation linéaire est donnée ci-dessous, où $W^+(v)$ représente l'ensemble des arcs sortant de v , alors que $W^-(v)$ représente l'ensemble des arcs entrant en v .

$$\begin{array}{ll}
 \text{Max} & \sum_{e \in W^+(s)} f(e) \\
 \text{s. c.} & \sum_{e \in W^+(v)} f(e) - \sum_{e \in W^-(v)} f(e) = 0 \quad \text{pour tout } v \neq s, t \\
 & 0 \leq f(e) \leq c_e \quad \text{pour tout arc } e \\
 & f(e) \text{ entier} \quad \text{pour tout arc } e
 \end{array}$$

La matrice des contraintes de ce problème est **totalelement unimodulaire** (concept qui sort du cadre de ce cours), ce qui a pour effet que si on ôte les contraintes qui imposent que les valeurs $f(e)$ doivent être entières, il existe une solution optimale entière.

Soit A un ensemble de sommets contenant la **source** mais pas le puits. L'ensemble des arcs (u,v) tels que u est dans A et v est hors de A est appelé une **coupe**. La capacité d'une coupe est la somme des capacités des arcs de cette coupe.

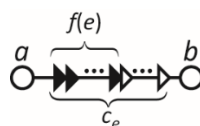
Propriété

1. Quel que soit le flot compatible f et quelle que soit la coupe dans R , la valeur $f_{s \rightarrow t}$ est inférieure ou égale à la capacité de cette coupe.
2. Il existe un flot compatible f et une coupe dans R telle que la valeur $f_{s \rightarrow t}$ est égale à la capacité de la coupe.

Corollaire

Si on est capable d'exhiber un flot f et une coupe telle que la valeur $f_{s \rightarrow t}$ est égale à la capacité de la coupe, alors f est un flot maximum et la coupe est de capacité minimale.

Dans les prochaines illustrations, nous représenterons la capacité d'un arc à l'aide d'une suite de triangles qu'on remplit de l'origine de l'arc vers sa destination. Le nombre de triangles remplis correspond au nombre d'unités $f(e)$ de flot circulant sur l'arc e . Par exemple, pour $e=(a,b)$, on a



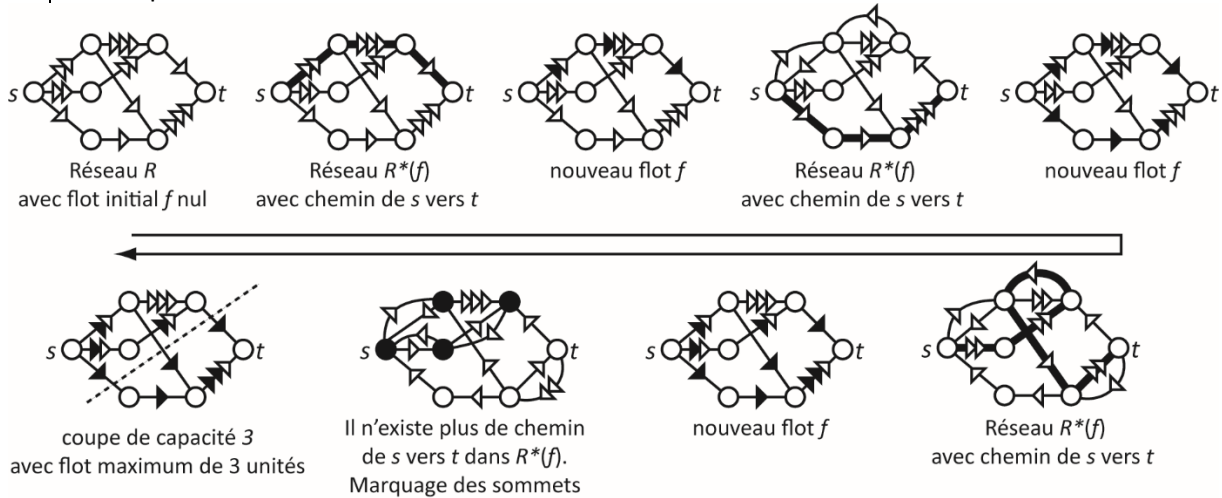
L'algorithme de Ford et Fulkerson permet de déterminer à la fois un flot maximum et une coupe de capacité minimum. Il utilise la notion de graphe auxiliaire, noté $R^*(f)$, et associé à tout flot f dans G . Ce graphe auxiliaire contient les mêmes sommets que R et ses arcs sont les suivants :

- Si un arc $e=(u,v)$ est tel que $f(e)<c_e$ alors on crée un arc (u,v) dans $R^*(f)$ de capacité $c^*(u,v)=c_e-f(e)$;
- Si un arc $e=(u,v)$ est tel que $f(e)>0$ alors on crée un arc (v,u) dans $R^*(f)$ de capacité $c^*(v,u)=f(e)$.

En d'autres termes, le graphe auxiliaire indique les modifications qu'il est possible d'apporter à f tout en conservant la compatibilité.

Algorithme de Ford et Fulkerson

1. Choisir un flot compatible initial f (par exemple le flot nul).
2. Tant qu'il existe un chemin P de s vers t dans $R^*(f)$ faire
 - Déterminer $\Delta = \min_{e \in P} c^*(e)$
 - Pour chaque arc (u,v) de P faire les modifications suivantes à f dans R :
rajouter Δ unités de flot sur (u,v) si (u,v) est dans R ;
ôter Δ unités de flot sur (v,u) si (v,u) est dans R .
3. Déterminer l'ensemble A des sommets atteignables depuis s dans $R^*(f)$. Le flot obtenu est maximum et la coupe qui relie les sommets de A aux sommets hors de A est de capacité minimum.



Flot avec borne inférieure de de flot

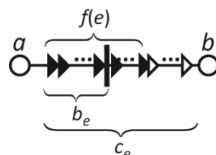
Supposons maintenant qu'une **borne inférieure** b_e est associée à chaque arc e de R , et il est souhaité que la quantité de flot qui circule sur chaque arc e soit au plus égale à b_e . La contrainte de compatibilité devient donc $b_e \leq f(e) \leq c_e$. Le réseau auxiliaire $R^*(f)$ est alors construit de manière similaire : il contient les mêmes sommets que R et ses arcs sont les suivants :

- Si un arc $e=(u,v)$ est tel que $f(e)<c_e$ alors on crée un arc (u,v) dans $R^*(f)$ de capacité $c^*(u,v)=c_e-f(e)$;
- Si un arc $e=(u,v)$ est tel que $f(e)>b_e$ alors on crée un arc (v,u) dans $R^*(f)$ de capacité $c^*(v,u)=f(e)-b_e$.

Le problème avec l'algorithme de Ford Fulkerson est l'étape 1 car il se peut qu'un flot nul ne soit pas compatible. Étant donné un flot f qui respecte les capacités, mais pas nécessairement les bornes inférieures de flot, on définit la **déficiance** $D(f)$ de f comme suit :

$$D(f) = \sum_e \max\{0, b_e - f(e)\}$$

Un flot de déficiance $D(f)=0$ satisfait donc les bornes inférieures de flot. Dans les illustrations, nous utilisons les conventions suivantes :



2. Quelques problèmes classiques

Problème d'affectation – couplage de coût minimum dans un graphe biparti

Considérons un graphe biparti $G=(V,E)$ non orienté, avec une partition V_1, V_2 des sommets telle que chaque arête de G a une extrémité dans V_1 et l'autre dans V_2 . Chaque arête e a un coût d_e . On recherche un **couplage maximum de coût minimum**. Il s'agit d'un **problème d'affectation** dans lequel chaque sommet de V_1 correspond à une tâche, et chaque sommet de V_2 correspond à une ressource. On désire affecter une unique ressource à chaque tâche, et chaque ressource ne peut être affectée qu'à une seule tâche.

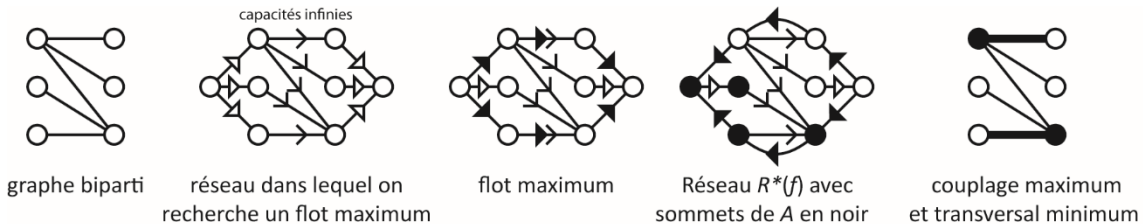
Pour résoudre ce problème, il suffit d'orienter chaque arête de V_1 vers V_2 , de rajouter une source s avec un arc (s,v) pour tout sommet v de V_1 , un puits t avec un arc (v,t) pour tout sommet v de V_2 . Les arcs incidents à s et t sont de capacité 1 et de coût 0 alors que les arcs correspondant aux arêtes du graphe original ont des capacités infinies et leur coût original. Un flot maximum à coût minimum correspond à la solution du problème d'affectation.

La taille minimale d'un transversal dans un graphe G quelconque est au moins égale à la taille maximale d'un couplage dans G puisqu'il faut prendre au moins un sommet par arête d'un couplage pour obtenir un transversal. Ces deux valeurs peuvent être différentes. Par exemple, dans un pentagone, le plus petit transversal contient 3 sommets alors que le plus grand couplage ne contient que 2 arêtes. Cependant, dans un graphe biparti, les deux valeurs sont toujours égales

Propriété

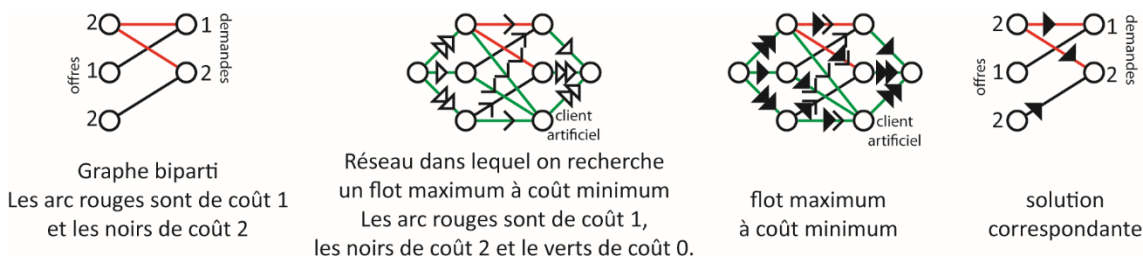
Dans un graphe biparti, la taille minimale d'un transversal est égale à la taille maximale d'un couplage

Il est facile d'obtenir un transversal de taille minimum. Il suffit pour cela de déterminer un couplage de cardinalité maximale, et lorsqu'il n'existe plus de chemin de s vers t dans le graphe auxiliaire $R^*(f)$, on détermine l'ensemble A des sommets atteignables depuis s dans $R^*(f)$: le transversal minimum est constitué des sommets de V_1 qui ne sont pas dans A et des sommets de V_2 qui sont dans A .



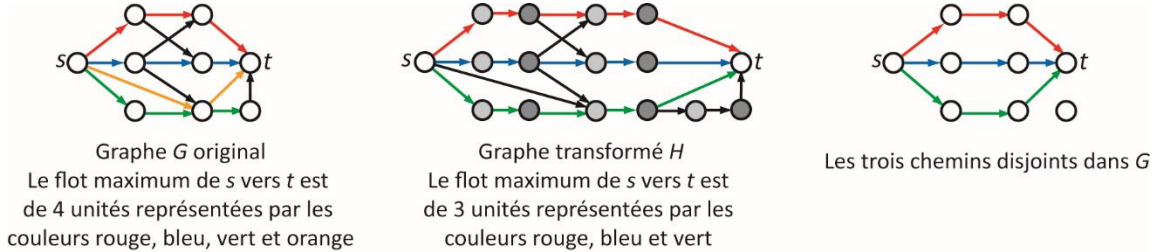
Problème de transbordement

Considérons un graphe biparti $G=(V,E)$ non orienté, avec une partition V_1, V_2 des sommets telle que chaque arête de G a une extrémité dans V_1 et l'autre dans V_2 . Supposons que chaque sommet (entrepôt) v de V_1 a une offre O_v et que chaque sommet (client) v de V_2 a une demande D_v . On connaît le coût unitaire de livraison de tout sommet de V_1 vers tout sommet de V_2 . On désire satisfaire la demande à coût minimum. Il s'agit d'un **problème de transbordement**. Pour le résoudre, on construit un graphe similaire à celui décrit ci-dessus, à la différence près que les arcs de s vers un sommet v de V_1 sont de capacité O_v , et les arcs reliant un sommet v de V_2 à t sont de capacité D_v . Si l'offre est plus grande que la demande, on rajoute un client artificiel ayant la demande résiduelle. Si l'offre est plus petite que la demande, il n'existe pas de solution.



Problème des chemins disjoints

Considérons un graphe orienté $G=(V,E)$. On désire déterminer le nombre maximum de chemins disjoints reliant deux sommets s et t . Pour résoudre ce problème, il faut construire un graphe H obtenu à partir de G en remplaçant chaque sommet v autre que s et t par deux sommets v_1 et v_2 . Chaque arc entrant en v dans G est remplacé par un arc entrant en v_1 dans H , et chaque arc sortant de v dans G est remplacé par un arc sortant de v_2 dans H . On rajoute finalement un arc (v_1,v_2) , et tous les arcs de H sont de capacité 1. On recherche alors un flot maximum de s vers t .



Indice chromatique d'un graphe biparti.

L'indice chromatique $q(G)$ d'un graphe G quelconque est toujours au moins égal au degré maximum $\Delta(G)$ dans G . On a vu qu'il peut être strictement supérieur (par exemple si G est un triangle). Par contre, ces deux valeurs sont toujours égales dans un graphe biparti.

Propriété

Dans un graphe biparti G , l'indice chromatique est égal au degré maximum, c'est-à-dire $q(G)=\Delta(G)$.

L'algorithme suivant détermine une coloration des arêtes de G biparti en $\Delta(G)$ couleurs. Considérons la partition V_1, V_2 des sommets de G telle que chaque arête de G a une extrémité dans V_1 et l'autre dans V_2 .

1. Construire un graphe H en faisant deux copies de G et en reliant chaque sommet v à sa copie à l'aide de $\Delta(G)$ -degré(v) arêtes parallèles.
2. Tant que H contient des arêtes faire
 - a. Déterminer un couplage maximum dans H (à l'aide d'un flot maximum)
 - b. Colorer les arêtes du couplage qui sont dans la première copie de H avec une nouvelle couleur et ôter de H toutes les arêtes du couplage.

