Flots dans les graphes

Michel Habib
Algorithmique, Ens Cachan 2012

12 janvier 2012

Plan

Définitions

On considère un graphe orienté $G = (X, U)$, deux sommets $s, t \in X$. Les arcs sont munis de capacités.
$c : U \rightarrow R+$
un flot sur G est un vecteur indexé par U (donc de taille m) qui vérifie :
$\forall xy \in U$, $0 \leq \phi(x, y) \leq c(u)$.
$\forall x \in X$, $\Sigma_{yx \ entrant \ en \ x} \phi(x, y) = \Sigma_{xz \ sortant \ de \ x} \phi(x, z)$
Lois de Kirchoff de conservation aux sommets.
Etant donné deux sommets particuliers s, t de G, on cherche un flot qui maximise le flot sur l'arc ts.

On considère un graphe orienté $G = (X, U)$, deux sommets $s, t \in X$. Les arcs sont munis de capacités.
$c : U \to R+$

Prototype de théorème Min-Max

La valeur minimale d'une coupe séparant s et t est égale à la valeur du flot maximum entre s et t dans G

Recherche d'une chaîne améliorante

RechercheChaîne(G, ϕ);

Données: un graphe orienté $G = (X, U)$, une fonction de capacité $c : U \to R+$, deux sommets $s, t \in X$, un flot $\phi \geq 0$

Résultat: soit une chaîne améliorante de s à t, soit la maximalité de ϕ

$OUVERTS \leftarrow \{s\}$, $FERMES \leftarrow \emptyset$;

tant que $OUVERTS \neq \emptyset$ **faire**

  $x \leftarrow Choix(OUVERTS)$;

  Explorer(x);

  $Ajout(x, FERMES)$, $Retrait(x, OUVERTS)$;

"Le flot ϕ est maximum" ;

Explorer(x);

pour *Tous les successeurs y de x* **faire**
  si $y \notin FERMES \cup OUVERTS$ *et* $\phi(xy) < c(yx)$ **alors**
    si $y = t$ **alors**
      STOP : Réponse VRAI "On a trouvé une chaîne améliorante de s à t"
    sinon
      Ajout(y, OUVERTS)

pour *Tous les prédécesseurs y de x* **faire**
  si $y \notin FERMES \cup OUVERTS$ *et* $\phi(yx) > 0$ **alors**
    si $y = t$ **alors**
      STOP : Réponse Vrai "On a trouvé une chaîne améliorante de s à t"
    sinon
      Ajout(y, OUVERTS)

Algorithme de Ford-Fulkerson

Afin de calculer un flot maximum dans un graphe, on peut appliquer la procédure suivante :

$\phi \rightarrow 0$;
tant que *RechercheChaîne*(G, ϕ)=*Vrai* **faire**
  Améliorer le flot sur la chaîne ;
  mise à jour de ϕ ;

Une coupe S est un sous-ensemble de sommets du graphe tel que $s \in S$ et $t \notin S$. La capacité $c(S)$ d'une coupe est la somme des capacités des arcs sortants de S.

Lemme

Pour tout flot ϕ et toute coupe S :
$\phi(t,s) + \Sigma_{xy \neq ts \ entrant \ en \ S}\phi(x,y) = \Sigma_{ab \ sortant \ de \ S}\phi(a,b) \leq c(S)$
d'où $\phi(t,s) \leq c(S)$.

Une coupe S est un sous-ensemble de sommets du graphe tel que $s \in S$ et $t \notin S$. La capacité $c(S)$ d'une coupe est la somme des capacités des arcs sortants de S.

Lemme

Pour tout flot ϕ et toute coupe S :
$\phi(t,s) + \Sigma_{xy \neq ts \text{ entrant en } S} \phi(x,y) = \Sigma_{ab \text{ sortant de } S} \phi(a,b) \leq c(S)$
d'où $\phi(t,s) \leq c(S)$.

corollaire

Si l'on trouve un flot ϕ et une coupe S tels que $\phi(t,s) = c(S)$, ils sont optimaux.

Fin de la preuve du Théorème FF

Preuve

Il est facile de vérifier que chaque chaîne améliorante permet de construire à partir de ϕ un nouveau flot ϕ' : qui vérifie $\phi(t,s) < \phi'(t,s)$.
Lorsqu'il n'existe plus de chaîne améliorante pour un flot ϕ, soit S l'ensemble des sommets explorés à partir de s. Nécessairement (vu la procédure d'exploration) les arcs rentrant en S on un flot nul et les arcs sortants sont saturés (valeur de flot $=$ capacité de l'arc).
L'inégalité du lemme est donc une égalité et le flot est donc bien optimum.

corollaire

A partir d'un flot maximum, on peut calculer en $O(m)$ une coupe minimale.

corollaire

A partir d'un flot maximum, on peut calculer en $O(m)$ une coupe minimale.

corollaire

Si les capacités sont entières l'algorithme de FF converge vers une solution entière.

corollaire

A partir d'un flot maximum, on peut calculer en $O(m)$ une coupe minimale.

corollaire

Si les capacités sont entières l'algorithme de FF converge vers une solution entière.

Remarque

Sans information supplémentaire sur le parcours de graphe utilisé, l'algorithme de FF n'est pas polynomial.

Graphe des écarts ou graphe résiduel

Pour un flot ϕ donné, on construit un graphe $G_\phi = (X, V)$ muni d'une valuation $\omega : V \rightarrow R+$ comme suit :
si $xy \in U$ alors :

- si $\phi(x, y) < c(x, y)$ alors $xy \in V$ et
  $w(x, y) = c(x, y) - \phi(x, y)$

Graphe des écarts ou graphe résiduel

Pour un flot ϕ donné, on construit un graphe $G_\phi = (X, V)$ muni d'une valuation $\omega : V \rightarrow R+$ comme suit :
si $xy \in U$ alors :

- si $\phi(x, y) < c(x, y)$ alors $xy \in V$ et $w(x, y) = c(x, y) - \phi(x, y)$
- si $\phi(x, y) > 0$ alors $yx \in V$ et $w(y, x) = \phi(x, y)$

Graphe des écarts ou graphe résiduel

Pour un flot ϕ donné, on construit un graphe $G_\phi = (X, V)$ muni d'une valuation $\omega : V \to R+$ comme suit :
si $xy \in U$ alors :

- si $\phi(x, y) < c(x, y)$ alors $xy \in V$ et $w(x, y) = c(x, y) - \phi(x, y)$
- si $\phi(x, y) > 0$ alors $yx \in V$ et $w(y, x) = \phi(x, y)$

Propriété

Il existe un chemin de s à t dans G_ϕ ss'il existe une chaîne améliorante dans G pour ϕ

Nous sommes donc ramenés à un problème d'existence de chemin dans un graphe.
On peut calculer :

1. Un plus court en nombre d'arcs

Nous sommes donc ramenés à un problème d'existence de chemin dans un graphe.
On peut calculer :

1. Un plus court en nombre d'arcs
2. Celui qui augmente le plus le flot

Nous sommes donc ramenés à un problème d'existence de chemin dans un graphe.
On peut calculer :

1. Un plus court en nombre d'arcs
2. Celui qui augmente le plus le flot
3. ...

Un algorithme polynomial simple

Soit Δ la puissance de 2 maximale telle que $\Delta \leq max_{sx \in U}(c(s,x))$. On notera $G_\phi(\Delta)$ le graphe partiel résiduel, constitué des arcs de capacités $\geq \Delta$.

$\phi \leftarrow 0$;
tant que $\Delta \geq 1$ **faire**
  tant que *il existe un chemin de s à t dans* $G_\phi(\Delta)$ **faire**
    améliorer ϕ;
    Mise à jour de $G_\phi(\Delta)$
  $\Delta \leftarrow \Delta/2$

Théorème

L'algorithme précédent calcul un flot maximum entre s et t en $O(m^2 log_2 C)$, avec C la capacité maximale d'un arc sortant de s.

1. Le nombre de passages dans la première boucle Tant que est borné par $log_2 C$.

1. Le nombre de passages dans la première boucle Tant que est borné par $log_2 C$.
2. Chaque chemin trouvé dans $G_\phi(\Delta)$ permet d'augmenter le flot d'au moins Δ.

1. Le nombre de passages dans la première boucle Tant que est borné par $log_2 C$.
2. Chaque chemin trouvé dans $G_\phi(\Delta)$ permet d'augmenter le flot d'au moins Δ.
3. A la fin de la 2ème boucle Tant que pour une valeur de Δ.
   Reprenons l'égalité du lemme sur la dernière coupe :
   $\phi(t, s) = \Sigma_{ab\ sortant\ de\ S}\phi(a, b) \leq$
   $c(S) - \Sigma_{xy \neq ts\ entrant\ en\ S}\phi(x, y)$
   Vu le marquage :
   $\phi(t, s) \geq c(S) - \Sigma_{ab\ sortant\ de\ S}\Delta - \Sigma_{xy \neq ts\ entrant\ en\ S}\Delta$
   d'où$\phi(t, s) \geq c(S) - m\Delta$ (majoration grossière), ou encore
   $c(S) \leq \phi(t, s) + m\Delta$.

1. Le nombre de passages dans la première boucle Tant que est borné par $log_2 C$.
2. Chaque chemin trouvé dans $G_\phi(\Delta)$ permet d'augmenter le flot d'au moins Δ.
3. A la fin de la 2ème boucle Tant que pour une valeur de Δ.
   Reprenons l'égalité du lemme sur la dernière coupe :
   $\phi(t,s) = \Sigma_{ab \ sortant \ de \ S}\phi(a,b) \leq$
   $c(S) - \Sigma_{xy \neq ts \ entrant \ en \ S}\phi(x,y)$
   Vu le marquage :
   $\phi(t,s) \geq c(S) - \Sigma_{ab \ sortant \ de \ S}\Delta - \Sigma_{xy \neq ts \ entrant \ en \ S}\Delta$
   d'où$\phi(t,s) \geq c(S) - m\Delta$ (majoration grossière), ou encore
   $c(S) \leq \phi(t,s) + m\Delta$.
4. Soit $\phi_p(t,s)$ la valeur du flot obtenu à l'étape Δ.
   $\phi^*(t,s) \leq c(S) \leq \phi(t,s) + m\Delta$. Donc à la phase suivante avec la valeur $\Delta/2$, le flot ne peut donc augmenter que de $m\Delta$, et il y a au plus $2m$ améliorations du flot, à l'étape $\Delta/2$.

Chaque amélioration nécessite un parcours en $O(m)$, d'où le résultat :
$O(m^2 log_2 C)$.
Notons que cet algorithme est polynomial.

Théorème

Tout flot se décompose comme une somme de chemins positifs de s à t.

1. Capacités sur les noeuds (on se ramène à un problème de flot en dédoublant les sommets).

Théorème

Tout flot se décompose comme une somme de chemins positifs de s à t.

1. Capacités sur les noeuds (on se ramène à un problème de flot en dédoublant les sommets).
2. Théorèmes de Menger (conséquences directes du théorème de Ford-Fulkerson)

Théorème

Tout flot se décompose comme une somme de chemins positifs de s à t.

1. Capacités sur les noeuds (on se ramène à un problème de flot en dédoublant les sommets).
2. Théorèmes de Menger (conséquences directes du théorème de Ford-Fulkerson)
3. Flots compatibles (on peut se ramener à un problème de flot maximum, dans un graphe construit à partir du premier).

Propriété

La capacité d'une coupe est une fonction submodulaire :
$\forall A, B \subseteq X - \{t\}$ with $s \in A \cap B$
$c(A) + c(B) \geq c(A \cap B) + c(A \cup B)$

Propriété

La capacité d'une coupe est une fonction submodulaire :
$\forall A, B \subseteq X - \{t\}$ with $s \in A \cap B$
$c(A) + c(B) \geq c(A \cap B) + c(A \cup B)$

Preuve

Une étude exhaustive de tous les cas possibles montre que la seule différence provient des capacités des arcs de $B - A$ vers $A - B$ ainsi que celles des arcs allant $A - B$ vers $B - A$.
Ces capacités sont présentes dans le terme de gauche, les capacités étant positives ou nulles, d'où la conclusion.

Considérons une fonction $\theta : X \rightarrow R+$.
Ceci nous permet de définir pour un vecteur ψ_θ de R^m comme suit :
$\forall xy \in U$, $\psi_\theta(xy) = \theta(y) - \theta(x)$

Théorème

$\forall \theta$ and $\forall \phi$ flot sur G, nous avons : $\psi_\theta.{}^t\phi = 0$

Considérons une fonction $\theta : X \to R+$.
Ceci nous permet de définir pour un vecteur ψ_θ de R^m comme suit :
$\forall xy \in U$, $\psi_\theta(xy) = \theta(y) - \theta(x)$

Théorème

$\forall \theta$ and $\forall \phi$ flot sur G, nous avons : $\psi_\theta.^t\phi = 0$

Preuve

$\forall \theta$ and $\forall \phi$ flot sur G, nous avons :
$\psi_\theta.^t\phi = \Sigma_{xy \in U}\ \phi(xy).(\theta(y) - \theta(x))$
Si A est la matrice d'incidence sommets-arcs, il suffit de remarquer que :
$\psi_\theta.^t\phi = T.A.^t\phi$ où T est le vecteur à n composantes tel que $T_i = \theta(x_i)$
Et comme $A.^t\phi = 0$, ceci donne le résultat.

Application

Soit G le réseau routier de France, et ϕ le flot des véhicules circulant à un instant ϵ.
On considère θ la fonction de température au même instant en chaque sommet du réseau.
$\forall \epsilon \ \psi_\theta . {}^t\phi = 0$

Certificats pour un flot ϕ

- Vérification des conservations aux nœuds : $O(n+m)$

Certificats pour un flot ϕ

- Vérification des conservations aux nœuds : $O(n+m)$
- Vérification de la maximalité de $\phi(s,t)$:
  À l'aide du calcul d'une coupe minimale.
  On peut procéder à un parcours sur (G,ϕ) en $O(n+m)$

Certificats pour une coupe minimale S

Il faudrait calculer un flot maximum, ce qui ne se fait pas en un simple parcours !

Malgré le théorème de Ford-Fulkerson flot max et coupe min ne sont pas algorithmiquement équivalents.

Pour le calcul d'un flot max :

1. Algorithme avec seuil décroissants en $O(m^2 log_2 C)$.

Pour le calcul d'un flot max :

1. Algorithme avec seuil décroissants en $O(m^2 log_2 C)$.
2. Ford-Fulkerson + plus courtes chaînes améliorantes en $O(m^2 n)$.

Lemme 1

Soit $D = (X, U)$ un multigraphe orienté vérifiant :

Alors D contient k chemins arc-disjoints de s à t.

Lemme 1

Soit $D = (X, U)$ un multigraphe orienté vérifiant :

1. $\forall x \neq s, t \ d^-(x) = d^+(x)$

Alors D contient k chemins arc-disjoints de s à t.

Lemme 1

Soit $D = (X, U)$ un multigraphe orienté vérifiant :

1. $\forall x \neq s, t \; d^-(x) = d^+(x)$
2. $d^+(s) - d^-(s) = k > 0$

Alors D contient k chemins arc-disjoints de s à t.

Lemme 1

Soit $D = (X, U)$ un multigraphe orienté vérifiant :

1. $\forall x \neq s, t \; d^-(x) = d^+(x)$
2. $d^+(s) - d^-(s) = k > 0$

Alors D contient k chemins arc-disjoints de s à t.

Remarque

Pas de condition sur les degrés de t, ni sur la connexité !

Preuve

On construit un chemin en partant de s en suivant les arcs de D aléatoirement, mais en ne prenant un arc qu'une fois. Ce chemin peut passer au plus $d^{-}(s)$ fois en s, mais à chaque fois il peut ressortir en utilisant un arc non déjà parcouru.
Comme le graphe est fini et que l'on est jamais bloqué en un sommet nécessairement le chemin s'arrête en t.
On vire ce chemin et l'on procède par induction sur k.

Preuve

On construit un chemin en partant de s en suivant les arcs de D aléatoirement, mais en ne prenant un arc qu'une fois. Ce chemin peut passer au plus $d^{-}(s)$ fois en s, mais à chaque fois il peut ressortir en utilisant un arc non déjà parcouru.
Comme le graphe est fini et que l'on est jamais bloqué en un sommet nécessairement le chemin s'arrête en t.
On vire ce chemin et l'on procède par induction sur k.

Remarque

Le lemme est faux pour un graphe infini, il suffit de considérer un graphe constitué de 3 chemins infinis issus de s.

Lemme 2

Considérons une suite de flots $f_1, \ldots$ telle que f_{i+1} s'obtient à partir de f_i en augmentant sur une chaîne améliorante de longueur minimale P_i, alors :

Lemme 2

Considérons une suite de flots $f_1, \ldots$ telle que f_{i+1} s'obtient à partir de f_i en augmentant sur une chaîne améliorante de longueur minimale P_i, alors :

1. $\forall k, |P_k| \leq |P_{k+1}|$

Lemme 2

Considérons une suite de flots $f_1, \dots$ telle que f_{i+1} s'obtient à partir de f_i en augmentant sur une chaîne améliorante de longueur minimale P_i, alors :

1. $\forall k, |P_k| \leq |P_{k+1}|$
2. $|P_k| + 2 \leq |P_l|, \forall l$ tel que $P_k \cup P_l$ contienne une paire d'arcs inverses

Preuve

On considère le graphe D construit sur $P_k \cup P_{k+1}$ en orientant les arcs de s vers t et en ôtant les arcs utilisés dans les deux sens (par contre D peut contenir des arcs parallèles).
D est construit à partir d'arcs du graphe G initial. On peut appliquer le lemme 1 précédent sur D avec $k = 2$. Il existe donc deux chemins arc-disjoints Q_1 et Q_2 allant de s à t, qui sont aussi des f_k chaînes améliorantes.
Par définition : $|P_k| \leq |Q_1|$ et $|P_k| \leq |Q_2|$.
Donc $2|P_k| \leq |Q_1| + |Q_2| \leq |P_k| + |P_{k+1}|$
D'où $|P_k| \leq |P_{k+1}|$.

Preuve (suite)

Pour la deuxième partie du lemme, il suffit de considérer deux chaînes ayant deux arcs inverses et d'appliquer le même raisonnement en remarquant que l'on gagne 2 dans les inégalités.

Théorème

L'algorithme de Ford-Fulkerson calcule un flot maximum en $O(m^2 n)$

Théorème

L'algorithme de Ford-Fulkerson calcule un flot maximum en $O(m^2 n)$

Preuve

Chaque chaîne améliorante sature (resp. met à 0) au moins un arc. Appelons cet arc goulot d'étranglement. Un arc de G est goulot d'étranglement au plus $n/2$ fois (cf. lemme 2.2).
Il y a donc au plus $m.n/2$ chaînes améliorantes.
Si on utilise un BFS pour les calculer, la complexité de l'algorithme est $O(m^2 n)$.

La majoration du résultat précédent est assez grossière, et en pratique l'algorithme converge très rapidement.

Notion de préflot

On définit pour chaque $f \in R^m$:

- $\forall x \in X,\ f_{in}(x) = \sum_{yx \in E} f(yx)$

Notion de préflot

On définit pour chaque $f \in R^m$:

- $\forall x \in X,\ f_{in}(x) = \sum_{yx \in E} f(yx)$
- $\forall x \in X,\ f_{out}(x) = \sum_{\{xz\} \in E} f(xz)$

Notion de préflot

On définit pour chaque $f \in R^m$:

- $\forall x \in X,\ f_{in}(x) = \sum_{yx \in E} f(yx)$
- $\forall x \in X,\ f_{out}(x) = \sum_{\{xz\} \in E} f(xz)$
- $exces(x) = f_{in}(x) - f_{out}(x)$

Notion de préflot

On définit pour chaque $f \in R^m$:

- $\forall x \in X,\ f_{in}(x) = \sum_{yx \in E} f(yx)$
- $\forall x \in X,\ f_{out}(x) = \sum_{\{xz\} \in E} f(xz)$
- $exces(x) = f_{in}(x) - f_{out}(x)$

Définition

Un préflot sur $G = (X, U)$ avec deux sommets s, t.
$f \in R^m$ tel que :
$\forall u \in U,\ 0 \leq f(u) \leq c(u)$
$\forall x \neq s,\ exces(x) \geq 0.$
Remarque : si pour un préflot f, $\forall x \neq s, t,\ exces(x) = 0$ alors f est un flot.

Sommets Actifs, étiquetage de distance

$\delta^+(x) = \{xy \in U\}$ = l'ensemble des arcs sortant de x.
Un sommet x tel que $exces(x) > 0$ est appelé actif.
Un étiquetage de distance pour un flot f est une fonction :
$\psi : X \to N$, telle que $\psi(t) = 0$, $\psi(s) = n$, et $\forall xy \in G_f$,
$\psi(x) \leq \psi(y) + 1$.

Data: un graphe $G = (X, U, s, t)$;

Result: Un flot de valeur max f compatible avec G;

"Initialisations :"

$a(s) = n$ **foreach** $x \in X \backslash \{s\}$ **do**

  $a(x) = 0$

foreach $u \in \delta^+(s)$ **do**

  $f(u) = c(u)$

foreach $u \notin \delta^+(s)$ **do**

  $f(u) = 0$

Calcul du graphe des écarts G_f et des sommets actifs.

"On a saturé tous les arcs sortant de s."

"Le corps de la procédure :"

while $\exists x \in X$ *avec* $exces(x) > 0$ **do**

  Choisir $x \in X - \{s, t\}$ tq $exces(x) > 0$ et $a(x)$ est maximale parmi les sommets actifs

  if $\exists y \in X$, *tq* $xy \in U_f$ *avec* $a(y) < a(x)$ **then**

    push(x,y)

  else

    relabel(x)

Les deux fonctions.

push(x, y) :

if xy *est un arc retour de* G_f ;

then

$\quad \delta = min(f(xy), exces(x));$

$\quad f(xy) \leftarrow f(xy) - \delta;$

else

$\quad \delta = min(c(xy) - f(x, y), exces(x));$

$\quad f(xy) \leftarrow f(xy) + \delta;$

Mettre à jour : $exces(x)$, $exces(y)$, le graphe des écarts $G_f = (X, U_f)$;

relabel (x) : $a(x) \leftarrow a(x) + 1$

Lemme

Si a est un étiquetage de distance pour f, il n'existe pas de chemin de s à t dans le graphe résiduel G_f.

Lemme

Si a est un étiquetage de distance pour f, il n'existe pas de chemin de s à t dans le graphe résiduel G_f.

Preuve

Considérons un chemin de s à t de longueur $k < n$. En considérant la fonction a le long du chemin on arrive à une contradiction : $a(t) \geq n - k$.

Invariant : f est un préflot sur G et a est un étiquetage de distance pour f.
Si l'algorithme s'arrête, il n'y a plus de sommet strictement en excès et nous avons donc un flot et vu le lemme précédent, nous avons un flot maximum.

Complexité

1. Si $exces(x) > 0$, il existe un chemin de s à x.

Complexité

1. Si $exces(x) > 0$, il existe un chemin de s à x.
2. $\forall x \in X,\ a(x) \leq 2n - 1$

Complexité

1. Si $exces(x) > 0$, il existe un chemin de s à x.
2. $\forall x \in X$, $a(x) \leq 2n - 1$
3. Chaque sommet est renuméroté au plus $2n - 1$ fois. Donc en tout au plus $2n^2$.

Complexité

1. Si $exces(x) > 0$, il existe un chemin de s à x.
2. $\forall x \in X,\ a(x) \leq 2n - 1$
3. Chaque sommet est renuméroté au plus $2n - 1$ fois. Donc en tout au plus $2n^2$.
4. Le nombre de push opérations qui sature un arc est au plus de 2nm

Complexité

1. Si $exces(x) > 0$, il existe un chemin de s à x.
2. $\forall x \in X$, $a(x) \leq 2n - 1$
3. Chaque sommet est renuméroté au plus $2n - 1$ fois. Donc en tout au plus $2n^2$.
4. Le nombre de push opérations qui sature un arc est au plus de 2nm
5. les autres push sont au plus en nombre $2n^2m$

L'algorithme des préflots nécessite $O(n^2 m)$.
On peut améliorer la complexité avec la variante de : Golberg, Tarjan (1988) en $O(n^2 m^{1/2})$