Algorithmique des Réseaux Sociaux

5 Novembre

Algorithme de Goldberg Tarjan pour les flots maximums et extensions

Soit $G = (V, E)$ un graphe orienté. On note $n = |V|$ et $m = |E|$. Pour chaque arête $(v, w) \in E$, on définit une capacité $c(v, w) > 0$ et par convention, on prend $c(v, w) = 0$ si $(v, w) \notin E$. Étant donné une source s et un puit t dans V, un flot dans G est une fonction f des pairs de sommets à valeur dans $\mathbb{R}$ telle que
(i) capacité : $f(v, w) \leq c(v, w)$ pour tout $(v, w) \in V \times V$,
(ii) antisymmétrie : $f(v, w) = -f(w, v)$ pour tout $(v, w) \in V \times V$,
(iii) conservation : $\sum_{u \in V} f(u, v) = 0$ pour tout $v \in V \backslash \{s, t\}$.
La valeur du flot f est $\sum_{v \in V} f(v, t)$ et un flot maximum est un flot de valeur maximum.

Une coupe $(X, \overline{X})$ est une partition de l'ensemble des sommets ($X \cup \overline{X} = V$ et $X \cap \overline{X} = \emptyset$) telle que $s \in X$ et $t \in \overline{X}$. La capacité de la coupe est $c(X, \overline{X}) = \sum_{v \in X, w \in \overline{X}} c(v, w)$. Si f est un flot, le flot au travers de la coupe est $f(X, \overline{X}) = \sum_{v \in X, w \in \overline{X}} f(v, w)$. Le principe de conservation implique que cette valeur est égale à la valeur du flot. De plus la contrainte de capacité implique que $f(X, \overline{X}) \leq c(X, \overline{X})$. Donc la valeur d'un flot maximum est inférieure ou égale à la capacité d'une coupe minimale. Le théorème de Ford et Fulkerson montre que ces deux quantités sont égales.

Nous allons définir un algorithme calculant une coupe minimale et un flot maximum. Pour cela, nous introduisons la notion de préflot f : une fonction des pairs de sommets à valeur dans $\mathbb{R}$ satisfaisant (i) et (ii) mais une forme affaiblie de (iii) :
(iii') $\sum_{u \in V} f(u, v) \geq 0$ pour tout $v \in V \backslash \{s\}$.
Le flot en excès au sommet $v \neq s$ est alors $e(v) = \sum_{u \in V} f(u, v) \geq 0$. Un préflot tel que $e(v) = 0$ pour tout $v \in V - \{s, t\}$ est un flot.

Pour un préflot f, nous définissons la capacité résiduelle de (v, w) notée $r_f(v, w) = c(v, w) - f(v, w)$. Si $r_f(v, w) > 0$ alors l'arc dirigé (v, w) est un arc résiduel et le graph résiduel du préflot est $G_f = (V, E_f)$ le graphe ayant pour ensemble de sommets V et pour ensemble d'arêtes E_f, les arcs résiduels. Un chemin f-augmentant est un chemin de s à t dans G_f.

1. (rappel) Montrer qu'un flot est maximum si et seulement si il n'existe pas de chemin f-augmentant (on pourra considérer l'ensemble $S_f = \{v \in V$ il existe un chemin de s à v dans $G_f\} \cup \{s\}$)
2. (rappel) En déduire le théorème de Ford et Fulkerson. Comment à partir d'un flot maximum trouver une (s, t)-coupe minimale ?

Pour définir l'algorithme, nous avons besoin d'une fonction de l'ensemble des sommets dans $\mathbb{N}$ telle que : $h(s) = n$, $h(t) = 0$ et $h(v) \leq h(w) + 1$ pour toute arête résiduelle (v, w). Nous appellerons une fonction satisfaisant ces conditions une fonction hauteur pour f. Un sommet v sera dit actif si $v \in V \backslash \{s, t\}$, $h(v) < \infty$ et $e(v) > 0$.

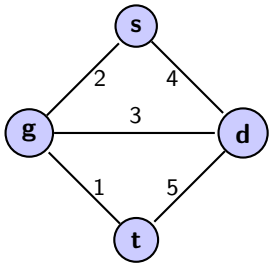
L'algorithme commence avec le préflot f qui est égal à la capacité des arêtes pour chaque arête sortant de la source et zéro sur toutes les autres arêtes. La fonction de hauteur initiale est $h(s) = n$ et $h(v) = 0$ pour tout $v \in V \backslash \{s\}$. L'algorithme réalise ensuite les deux opérations *push* et *relabel* tant que possible et dans un ordre quelconque. Lorsqu'il n'y a plus de sommets actifs, l'algorithme a terminé.

push(v, w) :
- Conditions : v est actif, $r_f(v, w) > 0$ et $h(v) = h(w) + 1$.
- Action : envois $\delta = \min(e(v), r_f(v, w))$ de flot de v vers w comme suit :

$$f(v, w) \leftarrow f(v, w) + \delta,\ f(w, v) \leftarrow f(w, v) - \delta,\ e(v) \leftarrow e(v) - \delta,\ e(w) \leftarrow e(w) + \delta.$$

relabel(v)
- Conditions : v est actif et $\forall w \in V$, si $r_f(v, w) > 0$ alors $h(v) \leq h(w)$.
- Action : $h(v) \leftarrow \min(h(w) + 1,\ (v, w) \in E_f)$ (avec le minimum sur l'ensemble vide égal à ∞).

3. Vérifier que l'algorithme est correct sur l'exemple suivant (graphe non-orienté, i.e. $c(u, v) = c(v, u)$) :



Nous allons montrer que l'algorithme est correct et étudier sa complexité. Nous dirons qu'une opération *push*(v, w) est saturante si $r_f(v, w) = 0$ après le *push* et non-saturante sinon.

4. Si f est un préflot, h une fonction de hauteur pour f et v un sommet actif, montrer qu'alors une opération *push* ou *relabel* est applicable sur v.
5. Montrer qu'à tout moment, la fonction h est une fonction de hauteur pour f.
6. Montrer que si f est un préflot et h une fonction de hauteur pour f alors le puits t n'est pas atteignable depuis la source s dans G_f.
7. En utilisant le résultat de la question 1, en déduire que si l'algorithme termine et que toutes les hauteurs sont finies alors le préflot calculé est un flot maximum.
8. Montrer que pour tout sommet v, la hauteur $h(v)$ ne décroit jamais. De plus, une opération *relabel*(v) augmente $h(v)$ strictement.
9. Soit f un préflot et S l'ensemble des sommets atteignables depuis v dans G_f. Montrer que si $s \notin S$ alors $\sum_{w \in S} e(w) \leq 0$. En déduire que si v est tel que $e(v) > 0$ alors la source s est atteignable depuis v dans G_f.
10. En déduire qu'à tout moment de l'execution de l'algorithme et pour tout sommet v, on a $h(v) \leq 2n - 1$.
11. En déduire que le nombre total d'opération *relabel* est borné par $2n^2$. En déduire également que le nombre d'opérations *push* saturante est bornée par $2nm$.
12. Soit $\Phi = \sum_{v,:v \text{ actif}} h(v)$. Montrer qu'une opération *push* non-saturante fait décroître Φ d'au moins 1. En déduire que le nombre d'opérations *push* non-saturante est au plus $4n^2m$.
13. En déduire que l'algorithme finit après $O(n^2m)$ opérations élémentaires.

Si le but est uniquement de calculer une coupe minimale, une modification simple de l'algorithme permet de calculer une telle coupe sans construire un flot maximum. Pour cela, le seul changement est de considérer un sommet v comme actif si $v \in V \backslash \{s, t\}$, $e(v) > 0$ et $h(v) < n$. Quand l'algorithme modifié termine, on note la coupe $S, \overline{S}$ avec $\overline{S}$ qui contient exactement tous les sommets depuis lesquels t est atteignable dans G_f.

14. Montrer que l'algorithme modifié termine en $O(n^2m)$ opérations élémentaires, que $S, \overline{S}$ est une coupe séparant s de t et que pour $v \in S$ et $w \in \overline{S}$, on a $f(v, w) = c(v, w)$.
15. Montrer que si $h(v) < n$ alors, $h(v)$ est une borne inférieure sur la distance de v à t dans G_f ; que si $h(v) \geq n$ alors $h(v) - n$ est une borne inférieure sur la distance de v à s dans G_f et que dans ce cas, t n'est pas atteignable depuis v dans G_f.
16. Montrer que lorsque l'algorithme modifié termine, $e(t)$ est la valeur d'un flot minimum et $S, \overline{S}$ est une coupe minimale.