Sur la complexité du routage OSPF

Frédéric Giroire, Stéphane Pérennes, Issam Tahiri

Sur la complexité du routage OSPF†

Frédéric Giroire and Stéphane Pérennes and Issam Tahiri

COATI, joint team INRIA, I3S (CNRS/University of Nice-Sophia Antipolis), Sophia Antipolis, France
Emails: firstname.lastname@inria.fr

Ce travail montre que dans un réseau (général) où le protocole de routage est OSPF avec la stratégie d'équilibrage de charge ECMP, le problème qui consiste à maximiser un flot simple d'une source vers un puits ne peut être approché à une constante près.

Keywords: routage, OSPF, ECMP, max flot, complexité.

1 Introduction & Etat de l'art

Le but d'un protocole de routage est de rendre chaque routeur capable de décider, quand un paquet est reçu, le routeur qui correspond au prochain saut. La décision doit être prise localement et rapidement, en permettant une utilisation efficace des ressources réseau. Quand le protocole *"Open Shortest Path First" (*OSPF*)* [osp98] est le protocole de routage utilisé dans le réseau, tous les paquets suivent des plus courts chemins, en prenant en compte les longueurs des liens fixées par l'administrateur réseau.

S'il y a plusieurs plus courts chemins entre deux noeuds u et v, le routage dépend de la règle qui est utilisée pour équilibrer le trafic entre les différents plus courts chemins. Il y a plusieurs règles et l'une des plus utilisées est ECMP ("Equal Cost Multiple Path"). Selon cette règle un routeur qui a plusieurs liens sortant sur des plus courts chemins vers une destination v repartit de façon uniforme le trafic qui lui arrive sur ces liens.

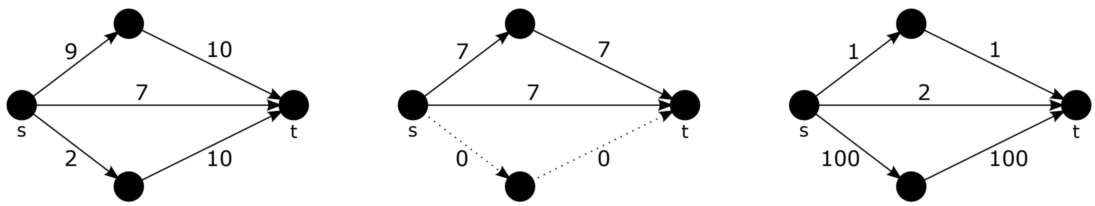
Pour mieux comprendre la difficulté algorithmique du routage OSPF, nous nous sommes concentré sur un problème essentiel qui a comme but la maximisation du débit quand il n'y a qu'*une seule paire* de noeuds communiquant sur le réseau. Nous appelons ce problème *Max-Ospf-Flow* et nous en donnons un exemple dans la figure 1 :

– INSTANCE : un graphe orienté valué $D = (V, A, c)$, où V est l'ensemble des noeuds, A est l'ensemble des arcs, et c est la fonction capacité qui associe à chaque arc (u, v) la capacité $c(u, v)$; et deux noeuds du graphe $s, t \in V$ qui représentent respectivement la source et le puits du flot.
– QUESTION : trouver le *flot maximum* allant de s à t avec l'affectation correspondante des longueurs pour les arcs de A, de sorte que le protocole "Open Shortest Path First" utilisant la stratégie "Equal Cost Multiple Path" atteigne ce flot max.

Des travaux précédents [FT04, ISI01, Ble05b] ont montré que trouver un routage OSPF qui optimise certains critères (charge, capacité, ...) de la performance du réseau est une tâche difficile ; et ceci est vrai en cas de mono-routage ou de multi-routage (avec la règle ECMP). En mono-routage, les longueurs doivent être choisies de manière à ce que, pour chaque paire de sommets, il y ait un unique plus court chemin. Dans [FT04], les auteurs ont étudié le problème d'optimisation des longueurs OSPF-ECMP afin de minimiser le coût total du réseau quand la fonction coût sur les arcs est convexe et croissante avec la congestion. Ils ont montré que ce problème est **NP**-Difficile. En particulier, ils définissent l'utilisation maximum du réseau comme $\max_{a \in A} \frac{l(a)}{c(a)}$ où $l(a)$ est la charge de l'arc a, et ils prouvent que la minimisation de l'utilisation maximum est **NP**-Difficile. Ils fournissent aussi des résultats sur le plus grand écart entre la performance du routage OSPF-ECMP et celle d'un routage optimal d'un flot à plusieurs commodités (des paires sources - puits) en terme de congestion. Dans [Ble05b], les auteurs ont étudié le multi-routage OSPF qui minimise l'utilisation maximum du réseau. Ils montrent que ce problème est difficile à approcher à un facteur $O(|V|^{1-\varepsilon})$.

Comme ces problèmes sont difficiles, pour les résoudre en pratique, on utilise souvent une approche à deux phases [BK02, HC04, DP+07]. Dans la première phase, nous trouvons un routage dont les chemins ne sont pas nécessairement réalisables avec le protocole OSPF. Dans la seconde phase, sont calculées, quand cela est possible, des longueurs qui réalisent ce routage en suivant le protocole OSPF. Cette seconde phase tente de résoudre ce que l'on appelle *le problème inverse du plus court*

† A long version of this paper can be found in [GPT13]



(a) capacitiés (b) Max-Ospf-Flow (c) longueurs correspondantes

FIGURE 1: Exemple d'un flot-ospf maximum de valeur 14.

chemin (ISP). Ce problème peut être formulé comme un programme linéaire et donc peut être résolu en temps polynomial. Même si les longueurs sont contraintes à prendre des valeurs entières, ISP garde une complexité polynomiale grâce au schéma d'arrondi présenté dans [BAG04]. Cependant, trouver une solution à ISP qui minimise la longueur maximum sur tous les arcs est **NP**-Difficile [Ble05a].

Nous considérons le problème d'approximation du flot-ospf maximum d'un réseau avec une commodité unique. Ce problème a été prouvé **NP**-Difficile dans [PSH$^+$02], et aucune approximation, avec garantie non triviale, n'a été proposée jusqu'à présent. Dans cet article, nous montrons que Max-Ospf-Flow ne peut pas être approché à un facteur constant près quand les capacités du graphe ne sont pas bornées. Nous avons aussi établi des résultats positifs (voir [GPT12]) que nous ne citons pas dans cet article.

2 Définition et simplification du problème

Dans cette section, nous montrons que, quand il y a une unique commodité, le problème inverse du plus court chemin est garanti d'avoir une solution, dès que le routage vérifie quelques conditions, et que cette solution peut être facilement trouvée. Ainsi, même si Max-Ospf-Flow reste difficile, nous pouvons nous concentrer sur la recherche d'une fonction de routage adéquate en oubliant l'affectation des longueurs aux arcs.

Dans le cas d'une simple commodité, résoudre le problème inverse du plus court chemin est trivial pour n'importe quel flot f allant de s à t. Pour cela nous définissons $A' = \{a \in A \mid f(a) > 0\}$, et nous posons $\forall a \in A', l(a) = 0$ et $\forall a \notin A', l(a) = 1$. De cette façon, tous les chemins de la source au puits utilisant uniquement des arcs de A' sont des plus courts chemins. Il est préférable parfois d'avoir une meilleure fonction longueur, telle que $\forall a \in A,\, l(a) > 0$, mais celle-ci n'existe que si A' est acyclique. Premièrement, si A' contient un cycle alors $l(a)$ doit forcément être nulle sur tous les arcs du cycle. Deuxièmement, si A' est acyclique, alors nous pouvons trouver une fonction longueur en utilisant l'existence d'une fonction potentielle $p : V \to \mathbb{R}$ telle que $p(u) < p(v)$ s'il y a un chemin allant de u vers v (cette fonction potentielle est donnée par le problème dual, voir [CCPS98]). Ensuite, nous posons $\forall a = (u,v) \in A, l(u,v) = p(v) - p(u)$. Remarquez que tous les chemins allant de s à t ont une longueur $p(t)$ - $p(s)$. En combinant ces deux faits, nous pouvons alors toujours définir des longueurs telles que, $\forall (u,v) \in V^2$, tous les chemins allant de u à v avec des arêtes dans A' aient la même longueur, et avec $l(a) > 0$ sur tous les arcs a qui ne sont pas contenus dans un cycle. Notez que, même si cela peut paraître étrange, il existe des graphes orientés pour lesquels toute solution optimale de Max-Ospf-Flow contient un cycle.

Donc pour toute fonction de flot, il existe toujours des longueurs pour lesquelles tout les chemins utilisés par le flot sont des plus courts chemins, et ces longueurs sont toutes strictement positives si le flot est acyclique. Cependant, cela ne veut pas dire que chaque fonction de flot peut être réalisée sous OSPF-ECMP. En effet f doit remplir la "condition d'équilibrage égal". Cela veut dire que nous devons avoir : pour chaque noeud $a \in V$ et $\forall (u,v) \in A$, soit $f(u,v) = 0$ ou bien $f(u,v) = out(u)$ où $out(u)$ dépend uniquement de u. Ceci nous permettra de reformuler Max-Ospf-Flow d'une manière plus simple, sans utiliser explicitement la fonction longueur. Mais nous devons tout d'abord introduire quelques définitions.

Définition 1 (graphe de flot) *Etant donnés un graphe orienté avec capacités $D = (V,A,c)$ et deux noeuds particuliers s et t (la source et le puits), le graphe de flot correspondant est défini par le 5-uplet $\mathcal{D} = (V,A,c,s,t)$.*

Définition 2 (fonction/valeur de flot) *Etant donné un graphe de flot $\mathcal{D} = (V,A,c,s,t)$, une fonction $f : A \to \mathbb{R}^+$ est une fonction de flot si $\forall a \in A, f(a) \leq c(a)$ et $\forall v \in V \setminus \{s,t\}, \sum_{(u,v)\in A} f(u,v) = \sum_{(v,u)\in A} f(v,u)$. La* valeur du flot *est $val(f) = \sum_{(s,v)\in A} f(s,v) - \sum_{(v,s)\in A} f(v,s)$.*

Définition 3 (fonction régulière - équilibrée) *Une fonction sur un ensemble S est dite régulière si elle prend uniquement deux valeurs dont* 0*. Etant donné un graphe $G = (V,A)$, une fonction f est dite équilibrée sur V ssi, pour chaque $u \in V$ la* *fonction f_u, définie par $\forall(u,v) \in A, f_u(v) = f(u,v)$, est régulière.*

Définition 4 (fonction de flot-ospf) *Pour un graphe de flot $\mathcal{D} = (V,A,c,s,t)$, une fonction $f : A \to \mathbb{R}^+$ est un flot-ospf si elle est une fonction de flot et si elle est équilibrée sur V.*

Maintenant, nous sommes prêts pour donner une formulation équivalente du problème **Max-Ospf-Flow** :

- **Instance : un graphe de flot $\mathcal{D} = (V,A,c,s,t)$.**
- **Question : trouver la valeur de flot maximum que peut avoir une fonction de flot-ospf dans A.**

Le problème de trouver la bonne fonction longueur a donc disparu de notre définition, puisque le fait de choisir les arcs qui auront un flot nul permet de déterminer la fonction longueur cachée. Nous notons dans le reste de l'article l'ensemble des entiers $\{1,\dots,N\}$ par $[N]$.

3 Résultats d'inapproximabilité

Nous utilisons une réduction au problème MAX-3-SAT (qui ne peut pas être approché à un facteur près de $\frac{7}{8}+\varepsilon$ pour tout $\varepsilon > 0$) pour prouver d'abord que le problème Max-Ospf-Flow est difficile à approcher, en temps polynomial, avec un facteur d'approximation de $\frac{31}{32}$. Ensuite, nous utilisons une méthode auto-amplifiante pour prouver que ce problème n'est pas dans APX. Ce rapport de recherche [GPT12] contient les preuves détaillées des résultats suivants.

Proposition 1 *Pour un entier $l > 3$, il est* **NP**-*Difficile d'approcher Max-Ospf-Flow avec un facteur $1 - \frac{l-3}{8l} + \varepsilon, \forall\varepsilon > 0$, même si $\forall a \in A, c(a) \in \{1,l\}$. Par example, il est* **NP**-*Difficile d'approcher Max-Ospf-Flow avec un facteur $\frac{31}{32} + \varepsilon, \forall\varepsilon > 0$, même si $\forall a \in A, c(a) \in \{1,4\}$.*

Esquisse de preuve Par souci de simplicité, notre réduction utilise un multi-graphe. Pour avoir un graphe à partir de ce dernier, il est possible d'insérer simplement un noeud subdivisant chaque arc en deux. Ceci permet d'avoir un graphe simple qui double seulement la taille du multi-graphe.

Etant donnée maintenant une instance I de MAX-3-SAT, définie par un ensemble de variables $\{x_i\}_{i\in[n]}$ et un ensemble de clauses $\{B_j\}_{j\in[m]}$, nous construisons le graphe de flot suivant $D_I = (V,A,c,s,t)$: Pour chaque $i \in [n]$, nous créons un sommet a_i représentant la variable x_i ; pour chaque $j \in [m]$, nous créons un sommet b_j représentant la clause B_j ; et pour chaque $i \in [n]$, nous ajoutons k_i arcs parallèles avec capacité 1 allant de s à a_i, où k_i est le nombre de fois où une variable x_i apparait dans les clauses. Nous ajoutons alors un arc avec capacité 1 depuis chaque b_j vers t, avec $j \in [m]$. En plus de ces arcs, nous ajoutons des arcs qui dépendent de la structure des clauses : $\forall(i,j) \in [n]\times[m]$, si B_j contient le littéral positif x_i nous ajoutons un arc avec capacité 1 de a_i vers b_j ; et $\forall(i,j) \in [n]\times[m]$, si B_j contient le littéral négatif $\bar{x}_i$ nous ajoutons l arcs ($l > 3$) parallèles avec capacité $\frac{1}{l}$ de a_i vers b_j. Un exemple de cette construction est donné en figure 2. Notez que la taille de $\mathcal{D}_I$ est linéaire en le nombre de clauses.

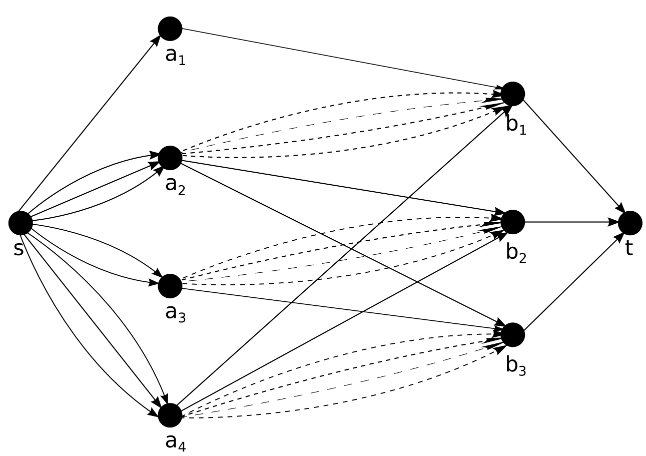


FIGURE 2: Le graph de flot $\mathcal{D}_I$ construit à patir d'un ensemble F contenant les clauses $B_1 = (x_1 \vee \bar{x}_2 \vee x_4)$, $B_2 = (x_2 \vee \bar{x}_3 \vee x_4)$ et $B_3 = (x_2 \vee x_3 \vee \bar{x}_4)$. [L'exemple suppose que $l = 4$. Les arcs en pointillé ont une capacité $\frac{1}{4}$. Les autres ont une capacité de 1].

L'idée de la réduction est d'émuler les variables binaires comme suit : pour $(i,j) \in [n]\times[m]$, nous définissons un lien virtuel $vl(a_i,b_j)$ comme l'ensemble des arcs connectant a_i à b_j, et nous disons que $vl(a_i,b_j)$ est un *lien virtuel positif* (resp. *négatif*) si x_i (resp. $\overline{x_i}$) est un littéral de B_j. Ensuite, il faut choisir au niveau de a_i d'envoyer le flot soit sur les liens virtuels positifs, soit sur ceux qui sont négatifs, le choix étant exclusif. □

Theorème 1 *Il est* **NP**-*Difficile d'approcher Max-Ospf-Flow à n'importe quel facteur constant près.*

Esquisse de preuve Pour prouver le résultat, nous utilisons une technique d'auto-amplification de l'erreur. En effet, le graphe de flot $\mathcal{D}_I$, associé à l'instance I de 3-SAT, et qui connecte s à t est semblable à un arc virtuel (s,t) avec une capacité m, mais pour lequel il est **NP**-Difficile d'utiliser une capacité plus grande que $\frac{31m}{32}$. Considérons un graphe de flot $\mathcal{D}$, nous allons remplacer chacun de ces arcs par une copie du graphe $\mathcal{D}_I$. Deux facteurs d'erreurs sont alors multipliés : un est dû au fait qu'il n'existe pas d'algorithme polynomial qui peut utiliser pleinement les capacités de ces arcs virtuels, et l'autre est dû à l'erreur sur le graphe original lui-même (que nous avons montré dans la proposition 1). □

Références

[BAG04] Walid Ben-Ameur and Eric Gourdin. Internet routing and related topology issues. *SIAM J. Discret. Math.*, 17(1) :18–49, January 2004.

[BK02] Andreas Bley and Thorsten Koch. Integer programming approaches to access and backbone IP-network planning. Technical Report ZR 02-41, Zuse Institute Berlin, 2002.

[Ble05a] Andreas Bley. Inapproximability results for the inverse shortest paths problem with integer lengths and unique shortest paths. In *Proceedings of the Second International Network Optimization Conference*, 2005.

[Ble05b] Andreas Bley. On the approximability of the minimum congestion unsplittable shortest path routing problem. In *Proceedings of the 11th international conference on Integer Programming and Combinatorial Optimization*, pages 97–110, Berlin, Heidelberg, 2005. Springer-Verlag.

[CCPS98] William J. Cook, William H. Cunningham, William R. Pulleyblank, and Alexander Schrijver. *Combinatorial optimization*. Wiley, 1998.

[DP+07] Mariusz Dzida, Mateusz Mycek, , Michał Pióro, Artur Tomaszewski, and Michał Zagozdzon. Valid inequalities for a shortest-path routing optimization problem. In *International Network Optimization Conference*, 2007.

[FT04] Bernard Fortz and Mikkel Thorup. Increasing internet capacity using local search. *Computational Optimization and Applications*, 29 :13–48, 2004.

[GPT12] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. Rapport de recherche RR-8175, INRIA, December 2012.

[GPT13] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. In *Proceedings of the International Network Optimization Conference (INOC'13)*, May 2013.

[HC04] Kaj Holmberg and Yves Carbonneaux. Optimization of internet protocol network design and routing. *Networks*, 43(1) :39–53, January 2004.

[ISI01] Rauf Izmailov, Bhaskar Sengupta, and Atsushi Iwata. Administrative weight allocation for pnni routing algorithms. In *IEEE Workshop on High Performance Switching and Routing*, pages 347–352, 2001.

[osp98] Ospf version 2, rfc2328. `http://www.ietf.org/rfc/rfc2328.txt`, 1998.

[PSH+02] Michał Pióro, Áron Szentesi, János Harmatos, Alpár Jüttner, Piotr Gajowniczek, and Stanislaw Kozdorowski. On open shortest path first related network optimisation problems. *Journal of Performance Evaluation*, 48(1-4) :201–223, May 2002.