Un algorithme d'approximation pour le sous-digraphe fortement connexe minimal.

Stéphane Bessy

Laboratoire LaPCS, Université Claude Bernard Lyon 1, 50 Avenue Tony Garnier, 69007 Lyon, France.

Dans cet article, nous étudions le problème du sous-digraphe couvrant minimum d'un digraphe D. Ce problème est NP-difficile et le meilleur algorithme polynomial d'approximation, du à A. Vetta [5], a un rapport d'approximation de 3/2. Nous proposons ici un algorithme d'approximation en $O(|V(D)|^4)$ avec un rapport $1+2\alpha(D)/|V(D)|$, où $\alpha(D)$ est la stabilité de D. Cet algorithme est donc meilleur lorsque le digraphe considéré est dense ($\alpha(D) < |V(D)|/4$) .

Keywords: algorithme d'approximation, problème MSSS, sous-digraphe couvrant minimum.

1 Introduction

Dans un réseau, lorsque l'on veut établir une communication de groupe, on recherche une sous-structure de coût minimum qui sera dédiée à cette communication. Dans une telle structure, un message doit pouvoir être acheminé de tout membre du groupe vers tout autre. Si on modélise le réseau par un graphe non orienté, cela revient à trouver un sous-graphe connexe de coût minimum, couvrant tous les membres du groupe. C'est le *problème de l'arbre de Steiner* qui est NP-difficile (cf [4]). Cependant lorsque le groupe est l'ensemble des nœuds du réseau, c'est le *problème de l'arbre couvrant de poids minimum* qui peut être résolu polynomialement avec les algorithmes très connus de Kruskal et Prim.

Malheureusement, beaucoup de réseaux ne peuvent pas être modélisés par un graphe non orienté. Par exemple, dans un réseau ad-hoc, un nœud a peut disposer de suffisamment de puissance pour émettre jusqu'à un nœud b alors que le nœud b ne puisse pas faire l'inverse. Dans cet article, nous étudions donc le cas où le réseau est modélisé par un graphe orienté ou *digraphe*. Un réseau dans lequel la communication entre tout couple de nœuds est possible correspond à un digraphe *D fortement connexe*, c'est-à-dire dans lequel, pour tout couple de sommets (x,y), il existe un chemin de x à y dans D. De plus, nous considérerons le cas dans lequel le groupe considéré est l'ensemble des nœuds du réseau et où les arêtes sont toutes de même coût 1. Le problème est donc de trouver un sous-digraphe fortement connexe couvrant minimum :

Problème MSSS (Minimum Strong Spanning Sub-digraph):
Instance : D un digraphe fortement connexe.
Question : Trouver un sous-digraphe fortement connexe couvrant D et ayant un nombre minimum d'arcs.

Contrairement au cas non orienté, ce problème est NP-difficile car il contient le problème de l'existence d'un circuit hamiltonien. Le meilleur algorithme d'approximation connu, dû à A. Vetta [5], a un rapport d'approximation de $3/2$, i.e. il fournit un sous-digraphe couvrant D fortement connexe ayant au plus $\frac{3}{2}.msss(D)$ arcs avec $msss(D)$ le nombre d'arcs minimum d'un sous-digraphe couvrant fortement connexe.

Un *stable* de D est un ensemble de sommet de D deux-à-deux non reliés dans D. La *stabilité* de D, notée $\alpha(D)$, est la taille d'un plus grand stable de D. Dans cet article, nous exposons un algorithme en $O(|V(D)|^4)$, correspondant au théorème suivant :

Théorème 1 *Tout digraphe D fortement connexe possède un sous-digraphe couvrant fortement connexe ayant au plus* $|V(D)|+2.\alpha(D)-2$ *arcs.*

Comme $msss(D) \geq |V(D)|$, celui-ci possède au plus $(1+2\alpha(D)/|V(D)|).msss(D)$ arcs. Le rapport d'approximation de cet algorithme est donc au plus 3 car $\alpha(D) \leq |V(D)|$ et est proche de 1 lorsque $\alpha(D)$ est petit, c'est-à-dire que le graphe est dense. En particulier, lorsque $\alpha(D) < |V(D)|/4$ notre algorithme est meilleur que celui de A. Vetta.
Seuls certains points de l'algorithme sont détaillés, une étude plus précise peut se trouver dans [1] ou [2].

2 Cas d'un sommet dominant.

Le coeur de l'algorithme repose sur le cas particulier où un sommet domine tout le digraphe. Pour un sommet w, on note $N_D^+(w)$ (ou w^+) l'ensemble des voisins sortants de w, et $N_D^-(w)$ (ou w^-) l'ensemble des voisins entrants de w. On suppose donc dans cette partie que $w \cup w^+ \cup w^-$ couvre D. On suppose de plus que D ne contient pas de circuits de longueur 2, ceux-ci s'éliminant facilement grâce à un pré-traitement.

2.1 Structures de données.

Pour un chemin P de D, on note $d(P)$ son *début*, $f(P)$ sa *fin* et $l(P)$ sa *longueur*. L'*intérieur* de P, c'est-à-dire $P \setminus \{d(P), f(P)\}$ est noté $P^\star$. Si x et y sont des sommets de P, on note xPy le sous-chemin de P (possiblement vide) de début x et de fin y. On note plus simplement Py le chemin $d(P)Py$ et xP le chemin $xPf(P)$. Si P a pour sommets $x_1, \dots, x_k$, on note $P = x_1 \dots x_k$. Enfin, si A et B sont deux ensembles de sommets de D disjoints, P est un (A,B)-chemin si $d(P) \in A$, $f(P) \in B$ et $P^\star \cap (A \cup B) = \emptyset$.

Une *anse* est un chemin P pour lequel on permet éventuellement que $d(P) = f(P)$.

Une *arborescence sortante* est une orientation d'un arbre dans laquelle tout sommet a degré entrant au plus 1. Le seul sommet de degré entrant 0 est appelé la *racine*. Les sommets de degré sortant 0 sont les *feuilles*. De façon duale, on définit une *arborescence entrante*.
Une *bi-arborescence* A est l'union d'une arborescence entrante A_- et d´une arborescence sortante A_+, A_+ et A_- ayant uniquement leur racine en commun. Les feuilles de A_- (resp. A_+) sont appelées les *feuilles entrantes* de A et notées $fe(A)$ (resp. *feuilles sortantes* de A et notées $fs(A)$). La racine commune est appelée *centre* de A et notée $c(A)$. Notons que le centre de A peut être une feuille de A, si A_+ ou A_- est constituée d'un seul sommet. Les sommets de A qui ne sont pas des feuilles sont appelés *sommets internes* de A.

Fixons un sommet w de D. Un *w-système* de D est un ensemble $S = \{W, A_1, \dots, A_k \mid P_1, \dots, P_l\}$, où W et A_i, pour $1 \leq i \leq k$, sont des bi-arborescences dont les centres sont respectivement w et a_i, et où P_j, pour $1 \leq j \leq l$, sont des anses, le tout vérifiant les propriétés suivantes :

i) Les ensembles $V(W), V(A_1), \dots, V(A_k), V(P_1^\star), \dots, V(P_l^\star)$ sont deux-à-deux disjoints.

ii) Le digraphe R_S formé par $\bigcup\{W, A_1, \dots, A_k, P_1, \dots, P_l\}$ est un sous-digraphe couvrant de D, on le nomme *réalisation* de S.

iii) Le début (resp. la fin) de P_j, pour $1 \leq j \leq l$, est un sommet sortant (resp. entrant) d'une A_i ou de W.

iv) Tout sommet x de D, excepté possiblement w, vérifie $d_{R_S}^+(x) \geq 1$ et $d_{R_S}^-(x) \geq 1$.

v) Pour $1 \leq i \leq k$, les voisins sortants (resp. les voisins entrants) de a_i dans R_S sont des voisins entrants (resp. des voisins sortants) de w dans D.

On nomme l le *nombre d'anses* de S et k le *nombre d'arbres* de S. La réalisation d'un w-système n'est pas forcément fortement connexe cependant on peut contrôler le nombre d'arcs de cette réalisation grâce au résultat suivant (voir [1]).

Lemme 1 *La réalisation d'un w-système couvrant D et ayant l anses et k arbres possède au plus* $|V(D)| + l - k$ *arcs.*

Pour S un w-système de D, on note A_S l'ensemble des bi-arborescences $\{A_1, \dots A_k\}$, W_S la bi-arborescence W et CH_S l'ensemble des anses de S. Pour un sommet x de S, $arb_S(x)$ désigne la bi-arborescence de S contenant x si elle existe et $ch_S(x)$ l'anse de S contenant x si elle existe et est unique.

2.2 w-système initial.

Il s'agit maintenant de construire un w-système de D. Un *étoile entrante* est une union de chemins de longueur 1 ayant la même fin, c'est-à-dire une arborescence entrante de hauteur 1. On définit de manière similaire une *étoile sortante*. On note X les voisins sortants de w^+ dans w^- et Y les voisins entrants de w^- dans w^+. On recouvre $X \cup Y$ par un ensemble Et d'étoiles entrantes et sortantes disjointes tel que toute étoile entrante (resp. sortante) ait ses feuilles entrantes (resp. sortantes) dans w^+ (resp. w^-) et sa feuille sortante (resp. entrante) dans w^- (resp. w^+). Cette construction se fait simplement de manière gloutonne sur les sommets de $X \cup Y$. En prolongeant Et par des arborescences sortantes sur w^- et des arborescences entrantes sur w^+, on peut alors construire les bi-arborescences A_i du w-système initial de D.

SYSTEME-INITIAL$(D, Et; S)$

1 $CH_S \leftarrow \emptyset$; $A_S \leftarrow \{w\} \cup Et$;
2 $V' \leftarrow V \setminus (X \cup Y \cup \{w\})$;
3 **pour** $y \in V'$ **faire**
4   **si** $y \in w^+$ **alors** on choisit P un $(y, V(S))$-chemin; $arb_S(f(P)) \leftarrow arb_S(f(P)) \cup P$;
5   **si** $y \in w^-$ **alors** on choisit P un $(V(S), y)$-chemin; $arb_S(d(P)) \leftarrow arb_S(d(P)) \cup P$;
6 **pour** $A \in A_S$ **faire**
7   **si** $|fe(A)| = 1$ **alors** $A_S \leftarrow A_S \setminus A$ et $W_S \leftarrow W_S \cup A \cup \{wf\}$; (où $f = fe(A)$)
8   **si** $|fs(A)| = 1$ **alors** $A_S \leftarrow A_S \setminus A$ et $W_S \leftarrow W_S \cup A \cup \{fw\}$; (où $f = fs(A)$)
9 **pour** $A \in A_S \cup \{W_S\}$ **faire**
10   **pour** $x \in fe(A)$ **faire** $CH_S \leftarrow CH_S \cup \{wx\}$;
11   **pour** $x \in fs(A)$ **faire** $CH_S \leftarrow CH_S \cup \{xw\}$;

La construction des bi-arborescences couvrant $w^+ \cup w^-$, effectuée aux lignes 3 à 5, demande un temps en $O(|V(D)|.|E(D)|)$. Les lignes 6 à 8 servent à rattacher les arborescences entrantes ou sortantes à W_S et consomment un temps en $O(|V(D)|)$. Enfin, la création des chemins de S des lignes 9 à 11 consomme un temps en $O(|V(D)|)$. Finalement, SYSTEME-INITIAL fonctionne en $O(|V(D)|.|E(D)|)$.

2.3 Réduction du nombre d'anses.

L'algorithme suivant permet de réduire la taille d'un ensemble de chemins disjoints $\mathcal{P}$ couvrant un sous-ensemble V' de sommets de D tant que $|\mathcal{P}| > \alpha(D[V'])$. Cet algorithme s'appuie sur la preuve du Théorème de Gallai-Milgam donnée par J. A. Bondy [3]. Pour un ensemble $\mathcal{P} = \{P_1, \ldots, P_p\}$ de p chemins, l'algorithme renvoie $s = 0$ et $Q = \emptyset$ si la réduction n'est pas possible (ce qui arrive si $\alpha(D[V']) = p$), et $s = 1$ si la réduction est possible. Dans ce dernier cas, Q contient les $p-1$ chemins résultants. De plus les débuts (resp. fins) des chemins de Q forment un sous-ensemble des débuts (resp. fins) des chemins de $\mathcal{P}$.

ECHANGE-CHEMINS$(D, \mathcal{P}; s, Q)$

1 **si** il existe i et j tels que $d(P_i)d(P_j) \in E$ **alors**
2     $P_1 \leftrightarrow P_i$; $P_2 \leftrightarrow P_j$;
3   **si** $l(P_1) = 0$ **alors**
4         $s \leftarrow 1$; $Q_1 \leftarrow d(P_1)P_2$; **pour** $i = 2, \ldots, p-1$ **faire** $Q_i \leftarrow P_{i+1}$;
5   **sinon** $x \leftarrow d(P_1)$; $y \leftarrow d(P_2)$; $z \leftarrow d(P_1 \setminus x)$;
6   ECHANGE-CHEMIN$(D \setminus x, P_1 \setminus x, P_2, \ldots, P_p; s, Q)$;
7   **si** $s = 1$ **alors si** on trouve i tel que $d(Q_i) = z$; $Q_i \leftarrow xQ_i$;
8   **sinon** on trouve i tel que $d(Q_i) = y$; $Q_i \leftarrow xQ_i$;
9 **sinon** $s \leftarrow 0$; $Q \leftarrow \emptyset$;

En omettant l'appel récursif de la ligne 6, les instructions des autres lignes consomment un temps en $O(p^2)$, ceci à cause des recherches effectuées lignes 1, 7 et 8. On effectue récursivement des appels à ECHANGE-CHEMINS tant que $l(P_1) > 0$ et à chaque fois D perd un sommet. Donc, au pire on effectue $|V'| - p + 1$ appels (en comptant l'appel initial). Donc, ECHANGE-CHEMINS a un temps d'exécution total en $O(p^2(|V'|-p))$, majorable par $O(|V'|^3)$.

L'algorithme suivant permet de réduire un w-système S de D en un w-système S' dont les feuilles entrantes (resp. sortantes) des bi-arborescences ont toutes degré entrant (resp. sortant) supérieur ou égal à 2. On pourra ainsi appliquer un échange de chemins sur l'intérieur des anses de S' en conservant la propriété **iv)** de la définition des w-systèmes.

REDUCTION($D,S;S'$)

1 $FS \leftarrow \emptyset$; **pour** $A \in A_S \cup W_S$ **faire** $FS \leftarrow FS \cup fs(A) \setminus \{w\}$;
2 $FE \leftarrow \emptyset$; **pour** $A \in A_S \cup W_S$ **faire** $FE \leftarrow FE \cup fe(A) \setminus \{w\}$;
3 **tant que** $FS \neq \emptyset$ **faire pour** $x \in FS$
4 | **cas 1:** $d^+_{R_S}(x) \geq 2$ **alors** $FS \leftarrow FS \setminus x$;
5 | **cas 2:** $d^+_{R_S}(x) = 1$ et x n'est pas centre de $arb_S(x)$ **alors**
6 | | on trouve y le voisin entrant de x dans R_S;
7 | | $arb_S(x) \leftarrow arb_S(x) \setminus x$; $ch_S(x) \leftarrow yx \cup ch_S(x)$; $FS \leftarrow (FS \setminus x) \cup \{y\}$:
8 | **cas 3:** $d^+_{R_S}(x) = 1$, x est centre de $arb_S(x)$ et $f(ch_S(x)) \notin arb_S(x)$ **alors**
9 | | $arb_S(f(ch_S(x))) \leftarrow arb_S(x) \cup ch_S(x) \cup arb_S(f(ch_S(x)))$;
10 | | $FS \leftarrow FS \setminus x$; $FE \leftarrow FE \setminus f(ch_S(x))$;
11 | | $A_S \leftarrow A_S \setminus arb_S(x)$; $CH_S \leftarrow CH_S \setminus ch_S(x)$;
12 | **cas 4:** $d^+_{R_S}(x) = 1$, x est centre de $arb_S(x)$ et $f(ch_S(x)) \in arb_S(x)$ **alors**
13 | | On construit le circuit C, union de $ch_S(x)$ et du chemin de $f(ch_S(x))$ à x dans $arb_S(x)$;
14 | | on trouve $y \in w^- \cap C$ et $z \in w^+$ successeur de y sur C;
15 | | $FS \leftarrow FS \setminus x$; $FE \leftarrow (FE \cup \{z\}) \setminus f(ch(x;S))$;
16 | | $W_S \leftarrow (W_S \cup arb(x;S) \cup ch(x;S) \cup yw) - yz$; $CH_S \leftarrow (CH_S \cup \{wz\}) \setminus ch_S(x)$;
17 Symétriquement, on traite FE, l'ensemble des feuilles entrantes des bi-arborescences de S;

Quatre cas possibles peuvent apparaître et il n'est, a priori, pas clair que l'algorithme termine. Une preuve de cela est donnée dans [1], il est, de plus, montré que les cas **3** et **4** se produisent au plus $O(|V(D)|)$ fois par appel de l'algorithme REDUCTION. A cause des tests des lignes 8 et 12 et de la recherche à la ligne 14, les cas **3** et **4** ont chacun une exécution en $O(|V(D)|)$. Les cas **1** et **2** ont un temps d'exécution constant et s'exécutent au plus $|V(D)|$ fois. Finalement, REDUCTION consomme un temps en $O(|V(D)|^2)$.

Enfin, l'algorithme SYSTEME permet d'obtenir un w-système S de D ayant au plus $\alpha(D[w^+ \cup w^-])$ anses.

SYSTEME($D,Et;S$)

1 SYSTEME-INITIAL($D,Et;S_1$);
2 $s \leftarrow 1$;
3 **tant que** $s = 1$ **faire**
4 | REDUCTION($D,S_1;S_2$);
5 | **si** il existe $P \in P_{S_2}$ tel que $l(P) = 1$ **alors** $P_{S_2} \leftarrow P_{S_2} \setminus P$;
6 | **sinon** ECHANGE-CHEMINS($D,\{P^\star : P \in P_{S_2}\}; s, \mathcal{P}$);
7 | | **pour** $P \in \mathcal{P}$ **faire**
8 | | $y \leftarrow N^-_{R_{S_1}}(d(P))$; $z \leftarrow N^+_{R_{S_1}}(f(P))$; $P \leftarrow yPz$;
9 | | $CH_{S_2} \leftarrow \mathcal{P}$;
10 | $S_1 \leftarrow S_2$;
11 $S \leftarrow S_1$;

Pour calculer le temps d'exécution de SYSTEME, il suffit de s'intéresser à la boucle des lignes 3 à 10, celle-ci s'exécutant tant qu'il est possible de réduire le nombre d'anses de S, c'est-à-dire au plus $|V(D)|-1$ fois. L'appel à ECHANGE-CHEMINS demande le plus de temps parmi les instructions des lignes 4 à 10, c'est-à-dire un temps en $O(|V(D)|^3)$, et finalement, SYSTEME s'exécute en $O(|V(D)|^4)$.

2.4 Recouvrement fortement connexe.

Le nombre d'anses du w-système S étant maintenant majoré par $\alpha(D[w^+ \cup w^-])$, il ne reste plus qu'à rendre sa réalisation fortement connexe sans trop augmenter le nombre d'arcs. Un tel sous-digraphe R de D est fourni par l'algorithme suivant.

RECOUVREMENT($D,S;R$)

1 $R \leftarrow R_S$;
2 **tant que** R n'est pas fortement connexe **faire**
3 | choisir X une composante fortement connexe de R qui ne contient pas w avec $|X| \geq 2$;
4 | choisir C un circuit de X et trouver dans C un arc xy tel que $x \in w^-$ et $y \in w^+$;
5 | $R \leftarrow (R \cup xwy) - xy$;

Il est prouvé dans [1] que RECOUVREMENT retourne un sous-digraphe fortement connexe couvrant D et ayant au plus $|V(D)| + \alpha(D) - 1$ arcs. La recherche des composantes fortement connexe de R_S aux lignes 2 et 3 demande un temps en $O(|E(D)|)$. Les autres instructions demandant un temps négligeable, RECOUVREMENT consomme un temps en $O(|V(D)|^2)$.

3 Algorithme principal.

Pour un digraphe D fortement connexe quelconque, il est alors possible de fournir en temps polynomial une $(1 + 2\alpha(D)/|V(D)|)$-approximation pour le problème MSSS en utilisant les algorithmes précédents. En résumé, un tel algorithme choisit un sommet w_1 de D, construit un digraphe auxiliaire D_{w_1} contenant w_1, w_1^+ et w_1^- tel que tout sous-digraphe fortement connexe couvrant D_{w_1} corresponde à un sous-digraphe fortement connexe D'_{w_1} de D contenant w_1, w_1^+ et w_1^-. Les algorithmes SYSTEME et RECOUVREMENT permettent de construire un tel D'_{w_1} contenant au plus $|D'_{w_1}| + \alpha(D[w_1^+ \cup w_1^-]) - 1$ arcs. L'algorithme contracte alors D'_{w_1} en un sommet w_2 et poursuit récursivement la construction. Le fait que les structures D'_{w_i} soient disjointes nous assure que le temps d'exécution total sera au plus en $O(|V(D)|^4)$. Un calcul précis de la stabilité du digraphe D vis-à-vis de la stabilité des digraphes D_{w_i} (voir [1]) permet d'affirmer que l'ensemble d'arcs retourné par l'algorithme correspond aux arcs d'un sous-digraphe fortement connexe couvrant D et a taille au plus $|V(D)| + 2\alpha(D) - 2$.

References

[1] S. Bessy. Stabilité et décomposition en circuits d'un digraphe, *Thèse de doctorat*, 2003.

[2] S. Bessy and S. Thomassé. Every strong digraph has a spanning strong subgraph with at most $n + 2\alpha - 2$ arcs. *Journal of Combinatorial Theory Ser. B*, **87** (2): 289-299, 2003.

[3] J. A. Bondy. A short proof of the Chen-Manalastas theorem. *Discrete Math.*, **146** (1-3): 289-292, 1995.

[4] F. K. Hwang, D. S. Richards and P. Winter, The Steiner tree problem, *Ann. Discrete Math.*, **53**, North-Holland, Amsterdam, 1992.

[5] A. Vetta. Approximating the minimum strongly connected subgraph via matching lower bound. *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (Washington DC, 2001*: 417-426, Philadelphia, PA, 2001. SIAM.