

Algorithme distribué tricritère pour la construction de structures de connexions

Fabien Baille**, Lélia Blin*, et Christian Laforest*

*IBISC, CNRS / Université d'Evry, Tour Evry 2, 523 place des terrasses, 91000 EVRY, France
{lbin,laforest}@lami.univ-evry.fr

**LIAFA, CNRS / Université Paris VII, 2 Place Jussieu, 75251, Paris Cedex 05, France
fbaille@liafa.jussieu.fr

Nous proposons dans cet article un algorithme distribué permettant la réservation de ressources (liens d'un réseau) pour interconnecter des machines dispersées dans un réseau. Par la suite la structure formé par les liens réservés pour connecter les machines, sera appelé *structure de connexion*. L'innovation majeure de ce papier réside dans le fait que nous prouvons que la structure possède simultanément de bonnes qualités en termes de distances (minimisation du diamètre et de la somme des distances entre les membres du groupe dans la structure finale) et de poids (problème de l'arbre de Steiner) induits. Nous proposons donc un algorithme d'approximation multicritère distribué asynchrone dont nous évaluons le nombre de messages échangés.

Keywords: Algorithme distribué, approximation tricritère, arbre de Steiner

1 Introduction

Dans des applications telles que les visio-conférences, des utilisateurs dispersés dans un réseau (appelés *membres d'un groupe* par la suite) désirent communiquer et échanger des données. Ces utilisateurs attendent des échanges de qualité pour un prix le plus bas possible. Pour assurer cette qualité les fournisseurs d'accès qui proposent ce service, peuvent réserver (ou louer) des ressources (ici des liens du réseau) pour l'usage exclusif du groupe. Nous appelons *structure de connexion* l'ensemble des liens réservés. Les paramètres de qualité de cette structure sur lesquels nous nous focalisons ici sont liés à la latence. Le coût du service est alors fonction du nombre de ressources mises en jeu. Nous allons traiter ces deux paramètres en même temps.

1.1 Le modèle

Nous modélisons le réseau par un graphe connexe, non orienté, et non pondéré $G = (V, E)$ où V est l'ensemble des sommets (représentant les nœuds du réseau) et E l'ensemble des arêtes (représentant les liens).

Soit $M \subseteq V$ le groupe de $m = |M|$ membres à connecter. On appelle *structure de connexion* un sous-graphe connexe de G couvrant M , c'est à dire un graphe $S = (V_S, E_S)$ tel que $M \subseteq V_S \subseteq V$ et $E_S \subseteq E$.

Soit $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ un graphe quelconque. On note $d_{\mathcal{G}}(u, v)$ la *distance* entre deux sommets u et v , qui correspond au nombre minimum d'arêtes qu'il faut franchir pour aller de u à v dans \mathcal{G} .

1.2 les critères

Comme nous l'avons mentionné précédemment, nous cherchons à optimiser les aspects de la QoS liés à la latence. Pour cela, nous désirons construire une structure S où les distances sont minimisées. Naturellement, pour chaque paire de sommets u et v de M , on a $d_G(u, v) \leq d_S(u, v)$. On espère donc trouver une structure pour laquelle les distances entres deux membres soit la plus proche possible de celles du graphe original. Pour capturer cette propriété nous nous intéressons à la fois à la minimisation du *diamètre* et de la *distance*

moyenne entre deux membres. Cependant, il est clair que minimiser ce critère revient à minimiser la *somme des distances* entre membre, qui sera le paramètre traité ici.

Pour des questions de rentabilité, le fournisseur à intérêt à réserver le minimum de liens pour chaque groupe pour pouvoir fournir d'autres services sur les ressources restantes. C'est pour cela que nous cherchons ici à minimiser le nombre d'arêtes contenues dans la structure de connexion finale. De plus il est clair que ce problème de minimisation du nombre d'arêtes d'une structure de connexion est le problème NP-complet de l'arbre de *Steiner*. Par la suite nous appelons *poids* d'une structure de connexion le nombre d'arêtes qu'elle comporte.

La définition de ces trois critères, en utilisant les notations définies plus haut, est la suivante :

Notation 1 Soit $G = (V, E)$ un graphe et $M \subseteq V$ un groupe à connecter.

- Le **diamètre** de M dans G est : $D_G(M) = \max\{d_G(u, v) : u \in M, v \in M\}$.
- La **somme des distances** de M dans G est $C_G(M) = \sum_{u, v \in M} d_G(u, v)$.
- Le **poids** de $G = (V, E)$ est son nombre d'arêtes : $W(G) = |E|$.

Dans [?], il a été montré que minimiser *simultanément* le diamètre et le poids est impossible. Ce caractère conflictuel des critères s'applique aussi dans le cas étudié ici. Pour résoudre ce conflit nous proposons de ne pas optimiser strictement ces critères mais de les approcher *simultanément*. Ainsi, nous souhaitons construire une structure S telle que son diamètre $D_S(M)$, sa somme des distances $C_S(M)$ et son poids $W(S)$ ne sont respectivement pas plus élevés que $\rho_D D_G(M)$, $\rho_a C_G(M)$ et $\rho_W W(T^*(M))$ où $T^*(M)$ est l'arbre de Steiner optimal. ρ_D , ρ_a et ρ_W sont appelés *rapports d'approximation* pour respectivement le diamètre, la somme des distance et le poids. Une telle structure est dite (ρ_D, ρ_a, ρ_W) -*approché*. Il en est de même pour un algorithme qui construit de telles structures.

1.3 Algorithme distribué

Les approches proposées pour résoudre ce genre de problème utilisent généralement des approches centralisées, ce qui induit une connaissance globale du réseau sous-jacent. Comme il est difficile d'avoir une telle connaissance, une manière de contourner ce problème est de construire cette structure de manière distribuée, c'est à dire en échangeant des messages entre les nœuds du réseau sans qu'aucun nœud n'ait tout le travail à effectuer à un moment de l'algorithme. Dans notre modèle, nous faisons des hypothèses raisonnables sur la connaissance locale des nœuds que nous détaillons maintenant.

Comme dans la plupart des réseaux actuels, chaque nœud possède une *fonction de routage*. Chaque nœuds u peut l'utiliser pour déterminer la distance $d_G(u, v)$ qui le sépare d'un autre nœud v . Cette fonction de routage peut aussi être utilisée pour envoyer des messages entre u et v par le plus court chemin entre u et v ou encore pour allouer un plus court chemin entre u et v (i.e. pour réserver un lien entre u et v). Nous supposons que ces opérations nécessitent un nombre de messages égal au nombre de liens franchis (i.e. $d_G(u, v)$). De plus, chaque membre est réveillé et connaît les m identités des membres du groupe M .

Un paramètre important à analyser lors de la conception d'algorithmes distribués est le nombre de messages générés. Nous proposons ici une analyse du nombre maximal de messages échangés.

Dans cet article, nous allons tout d'abord décrire dans la section ?? notre algorithme et dans la section ??, nous allons montrer son caractère tricité et analyser la complexité en nombre de messages.

2 L'algorithme

Notre algorithme se déroule en deux étapes. Dans la première, nous déterminons un arbre de Steiner T_P approché. Dans la seconde, nous réduisons les distances entre les membres dans cet arbre en ajoutant des plus courts chemins entre certains membres et un membre particulier (le *médian*); nous montrons que ces plus courts chemins (bien choisis) permettent de réduire les distances mais n'augmentent pas de manière très significative le poids de l'arbre initial.

2.1 Phase 1 : Construction de l'arbre de Steiner approché

Comme tous les membres se connaissent, on peut supposer que leurs identifiants sont deux à deux distincts et que chaque membre peut créer un ordre des identifiants identique (l'ordre alphabétique des identi-

fians, par exemple). Sans perte de généralité, soit $M = \{u_1, \dots, u_m\}$ cet ordre.

Chaque nœud u_i se connecte (en utilisant sa table de routage locale) au membre le plus proche parmi $\{u_1, \dots, u_{i-1}\}$. Chacune de ces connexions peuvent se faire en parallèle et dans n'importe quel ordre. Au final, nous obtenons une structure qui est connexe mais qui n'est pas nécessairement un arbre. Pour éliminer les cycles potentiels et élaguer la structure, nous exécutons un parcours en profondeur deux fois en partant de u_1 et nous obtenons l'arbre T_P désiré.

Ensuite, nous déterminons un médian pour M . Un médian pour un groupe M dans un graphe G est un nœud r de M tel que la valeur $\sum_{v \in M} d_G(r, v)$ est minimale. Pour déterminer qui parmi les membre du groupe est médian, nous utilisons l'algorithme suivant.

u_1 envoie à u_2 la paire (S_1, u_1) où $S_1 = \sum_{v \in M} d_G(u_1, v)$. A la réception de ce message, u_2 calcule $S_2 = \sum_{v \in M} d_G(u_2, v)$ et si $S_1 < S_2$, u_2 envoie (S_1, u_1) à u_3 et il envoie (S_2, u_2) dans le cas contraire. Ceci continue en suivant l'ordre des u_i croissants. Au final, u_m connaît l'identité d'un médian qu'il fait connaître aux autres membres par une diffusion.

Une fois cette première étape effectuée, nous disposons d'une structure de connexion qui est un arbre couvrant M approché par rapport à l'arbre de Steiner optimal. Dans la phase 2, nous allons modifier cet arbre pour qu'en plus des garanties sur le poids, il dispose de garanties sur le diamètre et la somme des distances.

2.2 Phase 2 : Réduction des distances induites dans T_P par ajouts de plus courts chemins du graphe.

La phase 2 est une version modifiée et distribuée de l'algorithme *Médian-contrôle* qui a été présenté par un des auteurs dans [?]. L'idée de cet algorithme consiste à faire un parcours en profondeur de T_P en partant du médian r ; chaque fois qu'un membre $v_i \in M$ est atteint pour la première fois, si $d_G(r, v_i) + d_{T_P}(v_i, u) > \alpha d_G(r, u)$ (pour un $\alpha > 1$ décidé au début entre les membres ou fixé a priori) cela indique que ce membre v_i est trop éloigné du "centre de gravité" qu'est le médian. Il convient alors de le connecter directement au médian r par un plus court chemin du graphe, qui est ajouté à la structure initialement composée des liens de l'arbre T_P ; l'arbre T_P et ces chemins supplémentaires donnent alors la structure finale G_f .

Notons que par manque de place nous ne pouvons pas décrire le pseudo-code de notre méthode mais que les données nécessaires au membre v_i pour savoir s'il doit ou pas se connecter au médian peuvent être transportés dans le message de notre protocole (tout en gardant des messages de faibles tailles).

3 Approximation et nombre de messages

Faute de place, nous ne pouvons exposer dans leur intégralité toutes les preuves nécessaires à l'analyse de cet algorithme. Nous proposons toutefois un aperçu des arguments utilisés.

3.1 Approximation

Le théorème suivant donne les principales propriétés de notre méthode.

Théorème 1 *Notre algorithme est $(2\alpha, 2\alpha, O(\log(m)))$ -approché pour la minimisation simultanée de la somme des distances, du diamètre et du poids.*

Notons ici que le paramètre α peut permettre au fournisseur d'accès de donner plus d'importance aux critères de latence qu'au critère de poids (ou inversement) en le faisant varier.

Pour montrer ce résultat, nous avons eu besoin des résultats intermédiaires suivants.

Le prochain lemme dit que dans la structure G_f construite par notre algorithme, les distances des membres au médian sont éloignées d'un facteur au plus α des distances dans le graphe de départ.

Lemme 1 *Pour tout $u \in M$ on a $d_{G_f}(r, u) \leq \alpha d_G(r, u)$.*

Le résultat suivant dit que cette même structure a un poids qui est proche du poids de l'arbre de Steiner approché calculé à la phase 1 de notre algorithme (le poids des arêtes ajoutées par les plus courts chemins a un impact contrôlé sur le poids de l'arbre de départ).

Lemme 2 $W(G_f) \leq \left(1 + \frac{2}{\alpha-1}\right) W(T_P)$

Enfin, on peut remarquer que l'algorithme utilisé pour construire l'arbre de départ est en fait une exécution particulière de l'algorithme de Imase et Waxman de [?]. On montre ainsi que l'arbre T_P construit en phase 1 est une $O(\log m)$ -approximation de l'arbre de Steiner du groupe.

Ces résultats (plus quelques autres non présentés ici provenant par exemple de [?, ?, ?]) nous permettent de montrer à la fois pour les rapports d'approximation du théorème ?? pour les distances et pour le poids.

3.2 Nombre de messages

Nous donnons ici une idée de l'analyse de la complexité en nombre de messages. Il est important de voir ici que la complexité globale de notre algorithme est dominée par la phase 1 qui construit un arbre de Steiner approché et non par la phase 2 qui rend se dernier performant pour les critères de latence.

Lemme 3 *La phase 1 génère au plus $O(m \log(m) D_G(M))$ messages.*

Proof. La première phase de construction coûte au plus $(m-1)D_G(M)$ messages. L'étape d'élagage génère un nombre de messages proportionnel au nombre d'arêtes de T_P dont on montre qu'elle vaut $O(\log(m)W(T^*(M)))$. D'autre part, comme $W(T^*(M)) \leq (m-1)D_G(M)$, on en déduit que la construction de T_P coûte au plus $O(m \log(m) D_G(M))$ messages. Pour finir, il est facile de voir que notre algorithme qui trouve un médian génère au plus $O(m D_G(M))$ messages. \square

Lemme 4 *La phase 2 génère au plus $2\rho \left(1 + \frac{2}{\alpha-1}\right) W(T^*(M))$ où ρ est le rapport d'approximation de l'arbre de Steiner construit à la phase 1, et $W(T^*(M))$ est le poids de l'arbre de Steiner optimal.*

Proof. La phase 2 est un parcours en profondeur de l'arbre T_P . Chaque arête de cet arbre sont donc traversées deux fois par un message. De plus quelques connections directes sont établies entre certains membres et le médian. Toutes les arêtes traversées par au moins un message forment la structure G_f . Donc au plus $2|E(G_f)|$ messages sont échangés. Comme $|E(G_f)| = W(G_f)$, le lemme ?? permet de conclure. \square

Si α est une constante, et comme $\rho = O(\log(m))$ et $W(T^*(M)) \leq m D_G(M)$ on a :

Théorème 2 *Notre algorithme génère au plus $O(m \log(m) D_G(M))$ messages.*

4 Conclusion

Dans cet article nous avons proposé un algorithme distribué pour construire une structure de connexion présentant simultanément des garanties de performances induits par des impératifs de minimisation de la latence (distance moyenne et diamètre) et des impératifs de coût. Nous avons aussi évalué le nombre de messages échangés pour la construction. Ces garanties étant en pires cas, il serait intéressant de voir si elle peuvent être raffinées par des simulations.