Vers un routage compact distribué

Christian Glacet, Verdonk Lucas

Vers un routage compact distribué

Glacet Christian[1] and Verdonk Lucas[1]

[1] *INRIA & LaBRi, Université de Bordeaux 1, France (Supporté par le projet européen EULER – STREP 7)*

Dans cet article, nous proposons plusieurs schémas distribués de routage compact produisant des tables de routage d'au plus $O(\sqrt{n}\log n)$ entrées pour un réseau de n nœud, m arêtes et de diamètre D. La complexité de communication de ces algorithmes est de $O(nm)$ et $O(nm+n^2\cdot\log n\cdot\min[(\sqrt{n}\cdot\log n),D])$.

Keywords: Routage compact, Algorithmique distribué, Expérimentations

Contexte Selon les estimations actuelles, les tables des routeurs d'Internet (tables de routage - *Forwarding Information Base*) comportent de l'ordre de 400 000 entrées et croissent de 10% par an. Cet accroissement a pour effet l'augmentation du délai de transmission des messages (temps d'accès à la mémoire, mises à jour plus fréquentes des entrées, induisant des périodes d'instabilité). Or les projections sur l'amélioration technologique des routeurs sont inférieures au facteur d'expansion, si bien qu'une augmentation des délais de transmission semble inévitable sans une réorganisation structurelle des tables (cf. [KFB$^+$07]).

Dans un schéma de routage, on peut distinguer deux types de structures de données pour chaque nœud : une **connaissance partielle du réseau** permettant de construire les **tables de routage**. Le premier objectif est de réduire la taille des tables de routage. Minimiser l'espace mémoire requis pour la connaissance du réseau est un objectif secondaire.

Le protocole actuel BGP utilise des routes de coût minimum (plus courts chemins) qui nécessite des tables de taille linéaire en n, le nombre de routeurs. Sur le plan théorique, il est cependant possible d'organiser l'information de routage selon des tables de routage de taille sous-linéaire en relaxant la contrainte d'un routage de plus courts chemins. Ainsi, Thorup et Zwick [TZ01, CSTW09],... ont proposé des schémas de routage garantissant des tables de taille en $O(\sqrt{n}\log n)$ dont les routes ont un étirement (ratio de la longueur de la route suivie sur la longueur optimale) d'au plus 3. Ce compromis est le meilleur possible en terme d'étirement car on peut montrer que tout schéma garantissant des tables de taille $o(n)$ pour tous les graphes connexes à n nœuds possède un étirement $s \geq 3$. Malheureusement, le schéma précédent nécessite d'affecter une adresse virtuelle spécifique de $O(\log n)$ bits à chacun des routeurs. Ainsi, le routage vers une destination t est réalisé sur la base de cette adresse. Un protocole de routage réaliste a vocation à recalculer régulièrement ses tables, et non pas à modifier les adresses des routeurs fréquemment.

Un progrès a été réalisé par le schéma AGMNT [AGM$^+$08] permettant un routage *sans renommage* avec les mêmes performances. Cependant, ce schéma est construit sur un réseau statique de manière centralisée.

Modèle et contributions A notre connaissance, nous proposons le premier schéma sans renommage distribué dans un environnement synchrone ayant un étirement $s \leq 3$ (adapté de [AGM$^+$08]) : à chaque ronde de communication, les nœuds s'échangent des messages et mettent à jour leurs informations locales. Nous nous concentrons essentiellement sur la connaissance partielle du réseau et la quantité totale de messages. Ce schéma est également tolérant à une dynamique du réseau. Plus précisément, en partant d'une configuration légitime, nous garantissons qu'après plusieurs suppressions, si le graphe reste statique suffisamment longtemps alors notre algorithme permet de converger vers une configuration légitime. Dans ce modèle,

Schéma	info. topologie	Tables	étirement s	communication
BGP	$\Theta(n)$	$\Theta(n)$	1	$O(nm)$
AGMNT "enraciné"	$\Theta(m)$	$\tilde{\Theta}(\sqrt{n})$	≤ 3	$O(m+n^{3/2}D)$
Contribution avec BC	$O(n)$/$\tilde{\Theta}(\sqrt{n})$ en moy.	$\tilde{\Theta}(\sqrt{n})$	≤ 3	$O(nm+n^2 \cdot \log n \cdot \min[(\sqrt{n} \cdot \log n), D])$
Contribution sans BC	$\tilde{\Theta}(\sqrt{n})$	$\tilde{\Theta}(\sqrt{n})$	≤ 7 (moyen ≤ 5)	$O(nm)$

TABLE 1: Résumé des contributions, avec et sans Boules Contiguës ($\tilde{\Theta}(\cdot) = \Theta(\cdot \log n)$)

pour un graphe $G = (V, E)$ de $|V| = n$ nœuds, $|E| = m$ arêtes et de diamètre D, nous évaluons les différentes versions de schémas de routage en distribué dans le tableau 1. La connaissance partielle (info. topo.), taille de tables de routage et complexité de communication sont exprimés en nombre d'entrées. Chaque entrée prend en général $O(\log^{O(1)} n)$ bits de mémoire (sauf pour BGP qui peut prendre $\Theta(D \log n)$ bits). La complexité de communication représente la quantité totale de messages échangés pondéré par leur taille pour initialiser les informations locales. AGMNT "enraciné" correspond à la situation où un seul nœud possède une connaissance totale du réseau - m arêtes - et diffuse à tous les nœuds leur tables de routage.

Nous exhibons un schéma alternatif (sans boules contigues) moins coûteux en mémoire et en échanges de messages, mais ayant de moins bonnes garanties d'étirement en pire cas. Ce second algorithme permet néanmoins de conserver un étirement moyen très satisfaisant comme le montrent les résultats de nos expérimentations.

1 Résumé du schéma distribué

k est un paramètre compris entre 1 et n. On peut montrer que $k = 4\sqrt{n}$ minimise la taille des tables de routage. Tous les nœuds tirent une couleur de manière aléatoire uniforme dans $[1..k]$ à leur initialisation. De plus, tous les nœuds ont en commun une fonction de hachage h équilibrée dans le sens où $O(n/k)$ nœuds ont une valeur hachée identique.Tous les nœuds de couleur 1 sont appelés *landmarks* et cet ensemble est noté L. Chaque nœud u stocke :

1. **Sa boule de voisinage** $B(u)$: elle est composée des plus proches nœuds de u et respecte les propriétés suivantes : (i) *complétude*, la boule contient au moins une fois un nœud de chaque couleur, (ii) *minimalité*, la boule est minimale si en enlevant le nœud "le plus éloigné de u", suivant un ordre total sur le couple $(distance, identifiant)$ la boule n'est plus complète. u connaît le premier saut pour chaque entrée de $B(u)$.
2. L, son **landmark le plus proche** $L(u)$, ainsi que $\forall l \in L$ son père $pere_l(u)$ dans l'arbre T_l enraciné en l.
3. **Sa table de Couleur** $C(u)$: u doit apprendre un chemin (compressé) vers tout nœud $v \in V$ tel que $h(v) = c(u)$. Cette route ne constitue pas nécessairement un plus court chemin.

Routage Deux boules $B(u)$ et $B(v)$ sont dites *contiguës* si elles sont à distance ≤ 1, *ie.* il existe une arête (x, y), telle que $x \in B(u), y \in B(v)$. Le principe du schéma [AGM+08] est relativement simple, tout nœud u sait router vers les nœuds de sa boule $B(u)$ et les landmarks L en plus court chemin. Il connait également une route vers tout nœud v tel que $h(v) = c(u)$ avec un étirement borné. Pour router vers un nœud v n'appartenant à aucune de ces catégories, u délègue le routage à un nœud $w \in B(u)$ tel que $h(v) = c(w)$, w est le *représentant de u pour la couleur* $h(v)$. w connaît un chemin vers v, (a) via le landmark $L(v)$ (b) **ou** via une boule contiguë ($w \rightsquigarrow x \rightsquigarrow y \rightsquigarrow v$). Ces différents cas sont présentés dans la figure 1.

Construction et maintien à jour des tables Pour la suppression d'une arête (u, v), on considère que les nœuds u et v sont capables de détecter la disparition de l'arête. La partie (3b) n'est pas indispensable, elle permet potentiellement de trouver une alternative plus courte au routage (3a).

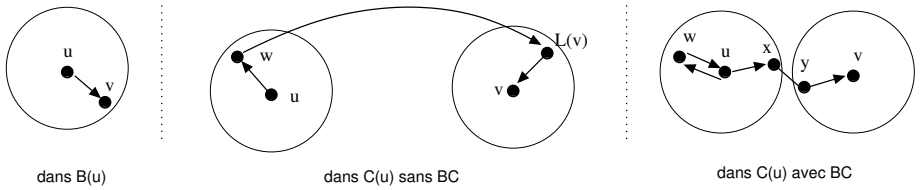

FIGURE 1: Cas de routage, si $h(v) = c(u)$ alors u sait router vers v et ne requiert pas de passer par un représentant w

1. **Boules de voisinage** Nous utilisons une version légèrement modifiée d'un BFS (*Bellman-Ford*) :
   - **initialisation** : tout nœud diffuse sa boule minimale à ses voisins dès que celle-ci est modifiée.
   - **mise à jour** Similairement à l'ajout d'entrées dans la boule, la suppression est propagée de proche en proche depuis u et v à tous les nœuds intéressés (nœuds ayant u ou v dans leur boule de voisinage).
2. **Landmarks**
   - **initialisation** De même que pour les boules de voisinage, l'algorithme utilisé est proche d'un *Bellman–Ford* distribué. De plus, dès qu'un nœud $u \in V$ a fini de construire sa boule de voisinage, il envoie un message à son plus proche landmark $L(u)$, calculant ainsi le chemin $u \rightsquigarrow L(u)$. Ce chemin sera utilisé lors de l'étape (3a).
   - **mise à jour** Pour tout $l \in L$, si $(u,v) \in T_l$ (on considère sans perte de généralité que $d(u,l) < d(v,l)$), tout $w \in T_l$ ayant u dans ses antécédents (au départ $w = u$) alors, w enlève le nœud l de sa connaissance et envoie un message à ses voisins pour qu'ils (1) suppriment également l de leur connaissance si leur père dans T_l est w (2) répondent à w en lui donnant leur distance au landmark l. w choisira alors le meilleur de ses voisins pour devenir son père si il en existe un.
3. **Couleur** le chemin peut prendre deux formes :
   (a) **Via un landmark - initialisation** Un landmark l diffuse les chemins calculés lors de l'étape (2), vers les nœuds ayant choisi l comme plus proche landmark. Ainsi, tout nœud $v \in V, h(v) = c(u)$ apprendra un chemin composé $u \rightsquigarrow l \rightsquigarrow v$. **Mise à jour**, identique à mise à jour de l'arbre (2).

   (b) **Via une Boule contiguë - initialisation**. Tout nœud u lance la construction d'un arbre de plus court chemin. Dès qu'un nœud $y \in V$ apprend qu'il est dans cet arbre alors que $y \notin B(u)$, il ajoute son identifiant au message de construction de l'arbre enraciné en u. Pour tout nœud v recevant un message de construction de l'arbre de u contenant l'identifiant d'un nœud y,si $y \in B(v)$, v prévient u que leurs boules sont contiguës et propage le message de construction de l'arbre de v vers ses voisins. Chaque nœud u diffuse dans $B(u)$ les chemins vers les nœuds v tels que $B(u)$ et $B(v)$ sont contigues, chaque nœud de $B(u)$ garde uniquement les informations relatives à sa couleur. **Mise à jour**, revient à un recalcul global.

2 Analyse et expérimentations

Convergence, nombre de messages Nous analysons le nombre d'entrées échangées au total à partir de la configuration vide (aucune information) puis à partir d'une configuration légitime en faisant une suppression d'arête. L'analyse de pire cas pour la construction revient quasiment à tout reconstruire si on considère les boules contigues. C'est toutefois très pessimiste comme le montre la colonne suppression dans le tableau 2. Les résultats expérimentaux portent sur un graphe aléatoire GLP ($n = 10000$,$m = 26000$ et $D = 9$), graphe avec une répartition des degrés suivant une loi en puissance et représentatif du graphe des AS d'Internet [Bol01]. Une analyse de complexité de communication moyenne serait intéressante.

Le temps de convergence est de $O(D)$, car après d rondes de communications tout nœud a pu récupérer les informations des nœuds à distance d. En pratique, on observe un temps de convergence de 18 rondes de

Algorithme	Initialisation		Suppression, en moyenne (expérimentation)
	analyse, pire cas	expérimentation	
Boules (1.1)	$O(m \cdot k \cdot \log k)$	$32 \cdot 10^6$	12 776
Landmarks (1.2)	$O(m \cdot \frac{n}{k})$	$8.5 \cdot 10^6$	437
Couleurs sans (1.3b)	$O(nm + n \cdot \min[(k \cdot \log k), D])$	$197 \cdot 10^6$	7 206
Couleurs avec (1.3b)	$O(nm + n^2 \cdot \log k \cdot \min[(k \cdot \log k), D])$	$652 \cdot 10^6$	$652 \cdot 10^6$

TABLE 2: quantité de messages échangés à la construction, puis après une suppression

communications sans les boules contiguës et 22 avec. Ainsi qu'un temps de convergence après une unique suppression d'arête de 5 rondes. On remarque également que les algorithmes proposés pour la gestion de la dynamique sont très satisfaisant car l'impact d'une unique suppression est très faible (de l'ordre de n entrées échangées au total) sauf pour les boules contiguës. :

Lemme 1. *Si $\forall u \in V$, les calculs $B(u)$ et T_u (si $u \in L$) sont finis, alors la table de couleur se construit en $O(nm + n^2 \cdot \log k \cdot \min[(k \cdot \log k), D])$ entrées échangées.*

Preuve résumée. L'algorithme le plus coûteux est celui de la construction des boules contiguës, nous ne présentons que cette partie de la preuve. Tout d'abord chaque nœud construit un arbre de plus court chemin pour un coût de communication total de $O(nm)$. Le plus coûteux est d'informer les nœuds w de la boule $B(u)$. Comme le nombre de nœud d'une couleur donnée est $O(n/k)$, le cout de communication pour w est de $O(n/k) * d(u,w)$. De plus, $|B(u)| = O(k \cdot \log k)$, $d(u,w) \leq \min[D, k \cdot \log k]$. Pour u, le cout de communication est donc $O(n/k \cdot k \cdot \log \cdot k \min[D, k \cdot \log k])$. ∎

Réduire le nombre de message de contrôle ? Sans boules de contiguës ? L'exécution montre que dans le cas du changement de topologie, peu de messages sont nécessaires pour mettre à jour la connaissance si on ne considère pas les boules contiguës. De plus, il y a très peu de différence pour l'étirement en moyenne : $s = 1.6$ avec boules contiguës contre $s = 1.7$ sans. Enfin, nous avons la garantie suivante :

Lemme 2. *Pour tout graphe, sans utiliser l'algorithme de Boules contiguës pour construire les tables des couleurs, l'étirement moyen est $\overline{s} \leq 5$ et au pire cas $s \leq 7$.*

Preuve La grande majorité des routages (probabilité $= 1 - \frac{1}{\sqrt{n}}$) s'effectue depuis une source u vers une destination v tels que $v \notin B(u)$, $v \notin L$ et $h(v) \neq c(u)$. Notons $p(u,v)$ la longueur du chemin utilisé lors du routage en passant par un représentant w puis par $L(v)$. On a donc $p(u,v) \leq d(u,w) + d(w,L(v)) + d(L(v),v)$ en utilisant des arguments d'inégalité triangulaire et le fait que $d(L(v),v) \leq d(L(u),v)$, on trouve :

- si $u \notin B(v)$, alors $p(u,v) \leq 5d(u,v)$ et $p(v,u) \leq 5d(u,v)$ on obtient un étirement aller/retour $\overline{s} \leq 5$.
- si $u \in B(v)$, alors $p(u,v) \leq 7d(u,v)$ et $p(v,u) = d(u,v)$ et donc l'étirement aller/retour est $\overline{s} \leq 4$. ∎

Cette solution est une bonne alternative par sa simplicité et par l'impact négligeable qu'elle a sur le étirement. Cependant, un algorithme de mise à jour des boules contiguës efficace est envisageable.

Références

[AGM+08] I. Abraham, C. Gavoille, D. Malkhi, N. Nisan, and M. Thorup. Compact name-independent routing with minimum stretch. *ACM Transactions on Algorithms (TALG)*, 4(3) :37, 2008.

[Bol01] B. Bollobás. *Random graphs*, volume 73. Cambridge Univ Pr, 2001.

[CSTW09] W. Chen, C. Sommer, S.H. Teng, and Y. Wang. Compact routing in power-law graphs. In *Proceedings of the 23rd international conference on Distributed computing*, pages 379–391. Springer-Verlag, 2009.

[KFB+07] D. Krioukov, K. Fall, A. Brady, et al. On compact routing for the internet. *ACM SIGCOMM Computer Communication Review*, 37(3) :41–52, 2007.

[TZ01] M. Thorup and U. Zwick. Compact routing schemes. In *Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–10. ACM, 2001.