Exploiter les agrégats et les lois de puissances pour le pair-à-pair

Philippe Gauron[†]

LRI, Univ. Paris-Sud, 91405 Orsay cedex, France. courriel : gauron@lri.fr.

Il a été récemment montré que les centres d'intérêt des utilisateurs de réseaux pair-à-pair ont des propriétés *d'agrégat* que l'on peut utiliser afin de réduire le trafic induit par les méthodes de recherche à base d'inondation. Ces centres d'intérêt ont aussi des propriétés de *lois de puissance* utilisables pour la conception d'algorithmes de recherche basés sur le routage. Dans cet article, nous montrons que l'utilisation conjointe de ces deux propriétés permet la conception d'algorithmes de recherche décentralisés, simples et efficaces. En particulier, nous présentons un algorithme ne nécessitant pas de topologie logique structurée pour connecter les pairs. Des simulations effectuées sur des traces issues d'un réseau réel montrent que les recherches s'effectuent en un nombre d'étapes logarithmique en utilisant cet algorithme.

Keywords: Pair-à-pair, Peer-to-peer, P2P, réseaux sans échelle

Introduction

Cet article s'intéresse aux systèmes pair-à-pair *décentralisés*. Dans ces systèmes, deux méthodes de transfert de requêtes sont utilisables : l'inondation (comme dans Gnutella), et les tables de hachage réparties (THR ou DHT : Distributed HashTables, comme CAN et Chord). Chacune de ces approches a ses inconvénients : le trafic engendré par l'inondation encombre la bande passante, tandis que les protocoles à base de THR nécessitent des connexions inspirées de topologies statiques parfois difficiles à maintenir. Les THR ne permettent pas non plus de recherches complexes, par exemple utilisant des expressions régulières. C'est pourquoi la conception de protocoles de recherche simples renvoyant des réponses rapides et ayant un faible trafic de contrôle reste un problème ouvert.

Notre article propose un algorithme bénéficiant des atouts des deux méthodes : une recherche *simple* dans un système pair-à-pair *non structuré* permettant des recherches complexes. Ce procédé s'appuie sur les propriétés statistiques des requêtes dans les réseaux pair-à-pair réels. En effet, on peut vouloir rapprocher le réseau logique des centres d'intérêts des pairs : si deux pairs ont des intérêts communs, alors ils vont probablement beaucoup échanger, et gagneraient donc à être proches dans le réseau logique. En pratique, les pairs ont tendance à se regrouper en communautés, au sein desquelles ils font beaucoup d'échanges, alors qu'ils ne font que que très peu d'échanges avec l'extérieur. Sur ces bases ont été proposées des améliorations pour des systèmes existants (par exemple [4, 7, 8]). Notre objectif dans cet article est de montrer que ces travaux peuvent être poussés plus avant tout en conservant un système très simple.

1 Le graphe des intérêts des pairs

On peut représenter les intérêts des pairs sous forme d'un graphe où deux pairs sont connectés si et seulement si ils ont des intérêts communs. Donner une définition formelle d'« avoir des intérêts communs » est difficile. Plusieurs propositions ont été faites, basées

[†]Cette recherche a reçu le soutien du projet INRIA « Grand Large », et du projet « PairAPair » de l'ACI « Grandes Masses de Données »

sur des mots-clés, les objets en commun, ou des représentations vectorielles. On utilisera ici la définition simple suivante : deux pairs sont connectés dans le graphe des intérêts s'ils ont échangé un objet dans le passé. Remarquons que deux pairs connectés de cette façon peuvent avoir à un moment donné des intérêts très différents, mais leurs intérêts ont au moins été liés à un instant donné.

Il a été montré récemment (voir par exemple [3, 4]) que, comme beaucoup de réseaux sociaux et de systèmes complexes, le graphe des intérêts défini ci-dessus a des propriétés statistiques non-triviales le rendant très différent des graphes aléatoires, en particulier :

- il a une faible densité (le degré moyen est très faible par rapport au nombre de nœuds) ;
- sa distance moyenne est faible (logarithmique par rapport au nombre de nœuds) ;
- il est formé d'agrégats locaux (si la densité globale est faible, la densité locale, elle, est forte) ;
- il est sans échelle (les degrés sont très hétérogènes, la plupart des nœuds ont un degré faible, mais quelques uns ont de forts degrés).

2 Utilisation des propriétés des réseaux sans échelle et des propriétés d'agrégats

L'utilisation des lois de puissance des réseaux réels pour concevoir des méthodes de recherches efficaces a été proposée dans plusieurs travaux (comme [1, 2, 6]). Les auteurs de ces articles approximent les distributions de degrés hétérogènes par des lois de puissance, et étudient les propriétés des marches (aléatoires ou déterministes) dans des graphes aléatoires avec de telles distributions de degrés (voir aussi [5]). Ainsi, en passant par les voisins de plus fort degrés, une requête trouve sa cible en un temps linéaire [1]. Par ailleurs, [2] montre par simulation qu'une recherche passant par les nœuds de plus fort degré est plus efficace qu'une marche aléatoire. Elle reste toutefois polynomiale si l'on tient compte des boucles.

Tout comme la nature hétérogène des pairs est visible dans la distribution des degrés, des facteurs culturels et sociaux engendrent une structure d'agrégats sur le graphe des intérêts. Par exemple, si un pair p_1 est intéressé par un objet O possédé par un pair p_2, il va probablement être intéressé par d'autres objets possédés par p_2. En fait, p_1 sera probablement intéressé par les objets possédés par les pairs intéressés par O. Ainsi (1) les pairs s'organisent en communautés (sous-graphes denses), et (2) deux pairs qui échangent des données sont fortement susceptibles d'échanger à nouveau d'autres données. Différents travaux ont proposé d'utiliser ces propriétés pour améliorer des systèmes existants, et ont mis en évidence que la nature d'agrégats des intérêts peut être utilisée pour créer des systèmes pair-à-pair efficaces [4, 7, 8].

3 Notre contribution

La propriété de loi de puissance a été utilisée avec un routage vers le plus fort degré dans les réseaux sans échelle mais n'offrant pas de propriété d'agrégats spécifique. Les algorithmes de recherche obtenus ainsi sont polynomiaux (c'est-à-dire en n^{α}, $\alpha > 1$). Par ailleurs, la propriété d'agrégats a été utilisée pour améliorer des protocoles à base d'inondation donc sans routage. La méthode proposée ici consiste à utiliser le routage vers le voisin de plus fort degré directement dans le graphe des intérêts. En effet, la loi de puissance du graphe permet un routage de plus fort degré, tandis que la présence d'agrégats permet à ce routage d'atteindre les objets recherchés en temps logarithmique, par rapport à un temps polynomial permis par les réseaux avec agrégats.

Dans la méthode que nous proposons, *QRE*, le réseau logique est ainsi le graphe des intérêts (défini par les requêtes précédément exécutées) évoluant donc à chaque requête. Dans le réseau logique de *QRE*, un nœud est simplement connecté à tout nœud avec lequel il a déjà échangé des objets. Il tient donc compte de la structure d'agrégat observée dans les réseaux pair-à-pair (les communautés). De plus, il utilise une recherche par les voisins de plus fort degrés, bénéficiant ainsi des résultats observés sur les graphes en loi de puissance.

Une requête est de la forme $\langle @, O, k \rangle$ où @ est l'adresse de la source (par exemple l'adresse IP), O la description de l'objet recherché, et $k \geq 1$ le nombre de copies de O cherchées. Chaque nœud maintient à jour le degré de ses voisins ainsi que la listes des objets qu'ils possèdent. L'algorithme de recherche effectue une recherche en profondeur, guidée par le voisin de plus fort degré, qui s'arrête lorsque $k = 0$. Pour chaque nœud : s'il a O, il l'envoie au demandeur @. Si un de ses voisin a l'objet O, ce dernier est contacté pour envoyer O à @. Puis la requête est renvoyée au voisin de plus fort degré non visité préalablement. À chaque copie trouvée, le fournisseur et la source sont connectés, et k est décrémenté.

Pour éviter les boucles, les nœuds gardent trace des destinations des requêtes qu'ils ont transmises. Remarquons alors que si au moins k copies de l'objet demandé existent, la requête les trouve (il y a donc exhaustivité), mais s'il n'y a pas assez de copies, la requête va parcourir tout le graphe. Pour gérer cela, on peut borner le temps de vie des requêtes.

4 Simulations

Lorsqu'un nœud se connecte, c'est pour récupérer un objet ou en mettre à disposition. Il utilise une passerelle tirée aléatoirement uniformément et lance à travers celle-ci une recherche de l'objet O qu'il cherche ou qu'il possède. Il se connecte ensuite au nœud qui répond à sa requête (qui lui renvoie O). En attendant d'avoir de nouveaux voisins, le nœud se connecte temporairement à la passerelle. En pratique, la passerelle est souvent un nœud de la même communeauté que le nouveau nœud, ce qui raccourcit les délais des requêtes.

Afin d'évaluer les performances de notre méthode, nous avons effectué des simulations à partir de *traces réelles*. En effet, aucun modèle ne permet de représenter le comportement des pairs, donc d'évaluer notre méthode en tenant compte à la fois des propriétés d'agrégats et sans échelle, et de la probabilité que deux pairs ayant déjà échangé rééchangent. Les traces que nous avons utilisées sont issues d'*edonkey*, et sont analysées dans [3], et elles représentent 2h 53min de fonctionnement. Nos simulations ont utilisé 46,202 pairs.

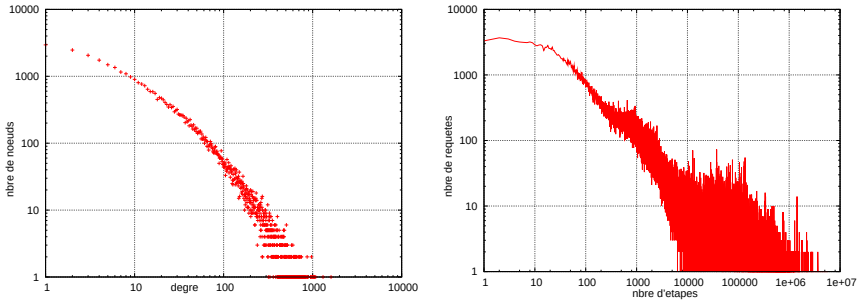

FIG. 1: Distribution des degrés (gauche) et nombre de sauts (droite) pour trouver une donnée

La figure 1 (gauche) montre une distribution des degrés à queue lourde, similaire à celles déjà observées pour ce type de réseau. La moyenne des degrés est de 47,9 et seuls 0.25% des nœuds ont un degré supérieur à 300. *QRE* peut donc gérer de grandes quantités de nœuds, ce qui montre qu'il passe à l'échelle.

La figure 1 (droite) représente le nombre de requêtes qui s'effectuent en k sauts. On peut l'approximer par une loi de puissance : la plupart des requêtes s'effectuent en très peu de sauts et très peu de requêtes nécessitent beaucoup de sauts. Les données très rares (ou inaccessibles) nécessitent le parcours complet du réseau.

Le figure 2 montre le nombre d'étapes nécessaires pour trouver un objet en fonction du nombre de nœuds dans le réseau. Une régression linéaire montre que le temps de recherche est linéaire en fonction du logarithme du nombre de nœuds. Les simulations montrent donc que QRE trouve les objets en un nombre de sauts moyen logarithmique. Il est ainsi au moins aussi efficace que toute méthode basée sur une THR (comme Chord).

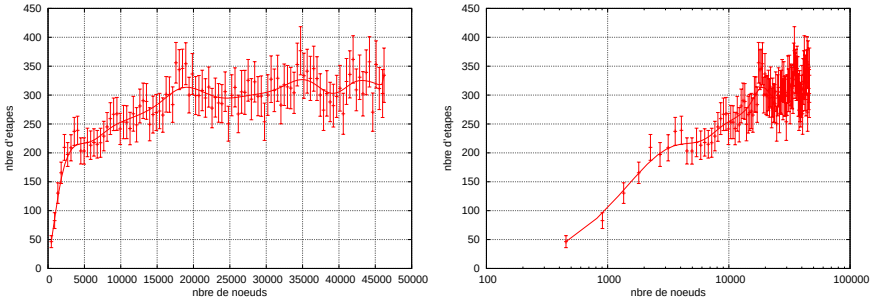

FIG. 2: Nombre d'étapes pour trouver un objet

Conclusion

Des méthodes de recherche plus subtiles et peut-être plus efficaces peuvent sans doute être créées à partir des graphes des intérêts, en utilisant d'autres propritétés du graphe encore à analyser, ou en combinant d'autres approches à celle-ci. En particulier, il n'est pas sûr que l'algorithme de recherche en profondeur d'abord par les voisins de plus fort degrés soit le plus approprié pour ce graphe. Des problèmes peuvent aussi apparaitre si les intérêts des utilisateurs changent au cours du temps, ou si plusieurs personnes utilisent le même nœud. Une solution possible à ce dernier problème serait de permettre plusieurs profils par nœuds, pour éviter l'utilisation de liens inappropriés. Une étude plus poussée serait aussi nécessaire afin de mesurer l'impact d'une limitation des degrés des pairs, ainsi que la robustesse et la sécurité.

Remerciements : Cet article est issu d'un article soumis à EuroPar'05 et écrit avec P. Fraigniaud et M. Latapy. Jean-Loup Guillaume a fourni des générateurs de graphe qui ont servi au travail préliminaire.

Références

[1] L. Adamic, R. Lukose, A. Puniyani, and B. Huberman. Search in power law networks. Physical Review E, vol. 64, 046135, 2001.

[2] B. Kim, C. Yoon, S. Han, and H. Jeong. Path finding strategies in scale-free networks. Physical Review E, vol. 65, 027103, 2002.

[3] S. Le-Blond, M. Latapy, and J.-L. Guillaume. Statistical analysis of a P2P query graph based on degrees and their time evolution. In LNCS vol. 3326, pages 126-137 Springer, proceedings of the 6-th Int. Workshop on Distributed Computing (IWDC), 2004.

[4] F. Le-Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulié. Clustering in Peer-to-Peer File Sharing Workloads. In 3rd Int. Workshop on Peer-to-Peer Systems (IPTPS), 2004.

[5] Q. Lv, P. Cao, E. Cohen, K. Li, S. Shenker. Search and replication in unstructured peer-to-peer networks. In 6th Int. Conference on Supercomputing, pages 84-95, 2002.

[6] N. Sarshar, P. Boykin, and V. Roychowdhury. Percolation search in power law networks : making unstructured peer-to-peer networks scalable. In 4th Int. Conference on Peer-to-Peer Computing (P2P'04), pages 2-9, 2004.

[7] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient content location using interest-based locality in peer-to-peer systems. In 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), IEEE, 2003.

[8] S. Voulgaris, A.-M. Kermarrec, L. Massoulié, and M. van Steen. Exploiting Semantic Proximity in Peer-to-peer Content Searching. In 10th IEEE Int. Workshop on Future Trends in Distributed Computing Systems (FTDCS), pages 238-243, 2004.