Projet : Ce que vous devez savoir avant de fabriquer un réseau de partage Peer to Peer

Julien Reynier encadré par Ahmed Serhrouchni

7 juin 2004

Introduction

Les réseaux Peer to Peer (d'égal à égal) apparaissent dans divers domaines sur Internet aussi variés que l'utilisation du cache utilisateur pour le téléchargement de page web, le partage de fichiers ou des calculs distribués. De nombreux protocoles P2P existent, nous présenterons les principaux d'entre eux de manière historique dans une première partie. Nous verrons ainsi comment la part des taches partagées augmente. Dans un second temps nous dresserons un état de l'art en essayant de dégager l'organisation d'un réseau P2P. Nous verrons aussi les défauts que l'approche P2P peut avoir par rapport au modèle client-serveur, ainsi les erreurs de design qui doivent être évitées feront l'objet de notre troisième partie. Enfin nous donnerons les pistes de l'organisation d'un P2P de partage de données distribuées.

Préambule : Qu'est-ce que Pair à Pair ?

Peer signifie semblable, égal en anglais, de sorte que Peer to Peer veut dire d'égal à égal. Dans le domaine d'Internet, le mot Pair à Pair (P2P) désigne l'approche qui s'oppose au modèle client-serveur. Dans une architecture P2P les tâches sont ainsi distribuées parmi les utilisateurs du service considéré, ceci ne signifie pas que tous les utilisateurs doivent nécessairement réaliser exactement le même travail. Beaucoup de tâches peuvent être distribuées, mais pour mériter le label P2P, il faut qu'il existe une version centralisée avec l'utilisation d'un serveur unique de cette même tâche. Nous verrons ici les algorithmes de hordes qui sont une version P2P de téléchargement. Ainsi que des P2P moteur de recherche P2P, mais dans un cas très particulier : pour trouver des données populaires que l'on va par la suite télécharger. L'idée de faire fonctionner google en P2P n'est pas à l'ordre du jour !

1 Histoire du P2P

1.1 Les échanges FTP

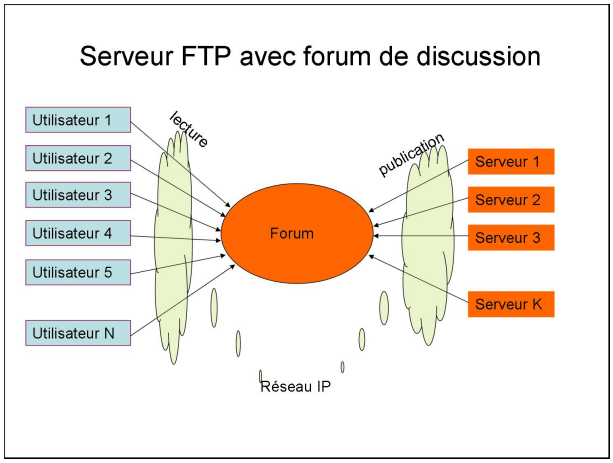

FIG. 1 – Organisation FTP

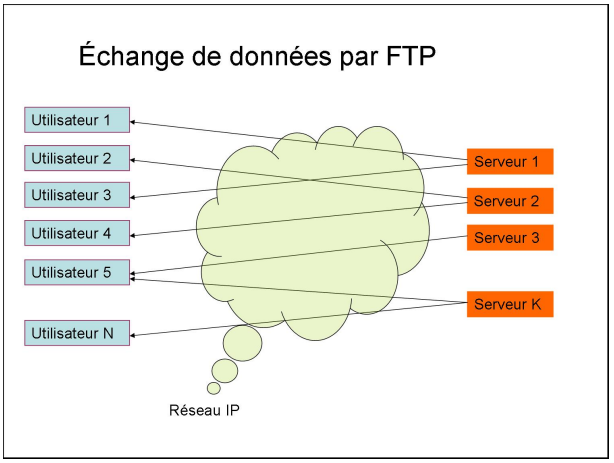

FIG. 2 – Téléchargement par FTP

La méthode historique et toujours très répandue pour échanger ou mettre à disposition des fichiers de toutes sortes est l'utilisation de serveurs FTP ou HTTP. Cette méthode est totalement décentralisée dans le sens où si deux personnes se mettent d'accord pour se transmettre un fichier elles n'ont pas besoin d'en avertir qui que ce soit. Elle nécessite cependant d'appartenir à un réseau de connaissances pour savoir quels fichiers sont disponibles et surtout où ils sont. Une solution à ce problème s'est développé dans le domaine informatique dès le milieu des années 90 avec l'émergence des forums de discussion. On pourra voir sur la figure 1 le principe de l'organisation. Il est nécessaire que certains utilisateurs se dévouent pour entretenir des forums de discussion (la tâche de recherche n'est pas P2P). De ce système oblige à avoir des serveurs (S)FTP qui doivent eux aussi être entretenus (le téléchargement est centralisé). De plus comme on peut le voir sur la figure 2, le transfert des données se fait par la suite des serveurs vers les clients. Il faut alors noter que plus il y a de clients, moins le système est efficace car les clients ne participent pas eux-même au service (les algorithmes de hordes résolvent en partie ce problème). Cette organisation a cependant de grandes qualités pour l'époque où elle a été utilisée, les clients qui n'ont pas des ordinateurs puissants et des accès à Internet très bas débit. Aujourd'hui si le problème du téléchargement proprement dit passe par les hordes, la recherche d'informations rares se fait toujours de manière avantageuse par des forums qui présentent de plus l'intérêt de brasser les idées nouvelles et d'assurer le contrôle de la qualité des informations fournies grâce à des modérateurs. Il existe des applications de partage de fichier qui fonctionnent sur ce principe. On pourra par exemple regarder Direct Connect qui a est toujours assez répandu.

1.2 Napster

Le premier réseau P2P à avoir su atteindre une échelle mondiale est Napster avec la généralisation du partage de musique en format compressé sur Internet. Son architecture générale est assez simple, un réseau de serveurs recensent les fichiers musicaux partagés par les utilisateurs connectés, et les utilisateurs interrogent ces serveurs pour localiser leurs musiques préférées. Du point de vue structure, la société commerciale Napster possédait les serveurs localisés à divers endroits dans le monde et les utilisateurs utilisaient un programme client leur permettant ensuite de faire les échanges de fichiers de manière totalement transparente. Par exemple, en 2001, lors de la première décision de justice qui a attaqué Napster le réseau comptait 160 ordinateurs de type *biPIII*700 et permettait couramment à plus de 5 millions d'utilisateurs de se connecter. Ce qui explique en grande partie le succès de Napster est sans doute sa facilité d'utilisation d'une part et d'autre part le fait qu'il n'est pas nécessaire de discuter avec les autres utilisateurs pour acquérir un fichier. Napster intègre une fonction de téléchargement de fichiers par ftp entre les utilisateurs (qui ne passe pas en général à travers les firewall et serveurs NAT), ceci est un point faible si on voulait généraliser l'utilisation de Napster à des fichiers plus volumineux.

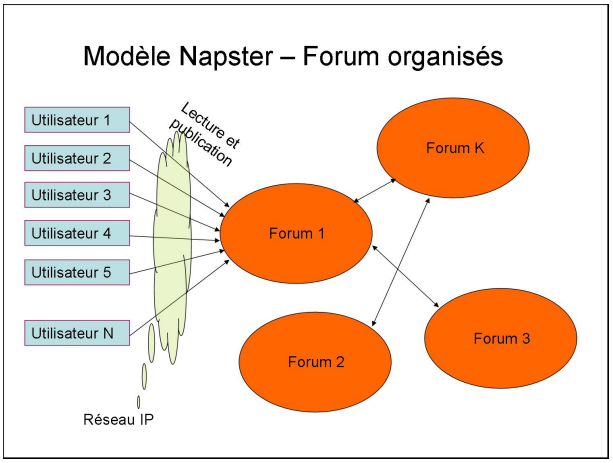

FIG. 3 – Organisation NAPSTER

1.3 Les Gnutella-like

1.3.1 Raison de l'apparition

Gnutella est le premier réseau totalement P2P puisque mis à part l'indispensable phase d'initialisation pour trouver le réseau, toutes les taches sont distribuées. Si c'est principalement pour résister aux attaques de la justice américaine que des protocoles véritablement décentralisés sont apparus, ils permettent de créer des réseaux extrêmement résistants aux pannes et aux attaques. Dans Gnutella (ou Kazaa) non seulement le transfert se fait de manière décentralisée, mais aussi la recherche de fichiers. Cependant il reste une

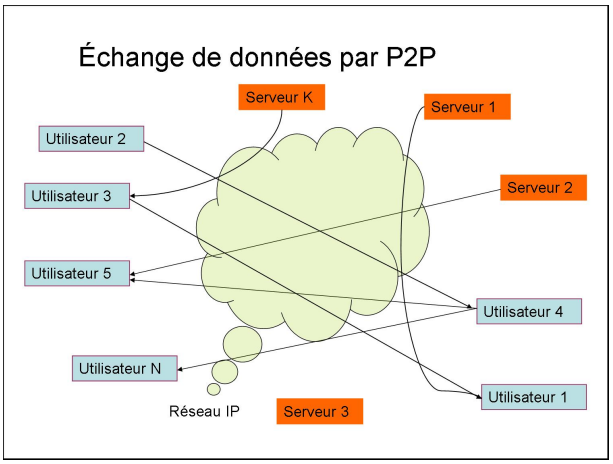

FIG. 4 – Téléchargements P2P

structure hiérarchisée du réseau, la plupart des utilisateurs se comportent comme des clients du point de vue de la recherche, mais certains peuvent devenir des serveurs locaux et participer au recensement des informations. Du point de vue pratique chaque utilisateur utilise un logiciel pour accéder au réseau et il se connecte pour commencer à une adresse connue (c'est le seul moyen connu pour se mettre en relation mis à part la diffusion d'un message en broadcast).

1.3.2 L'architecture

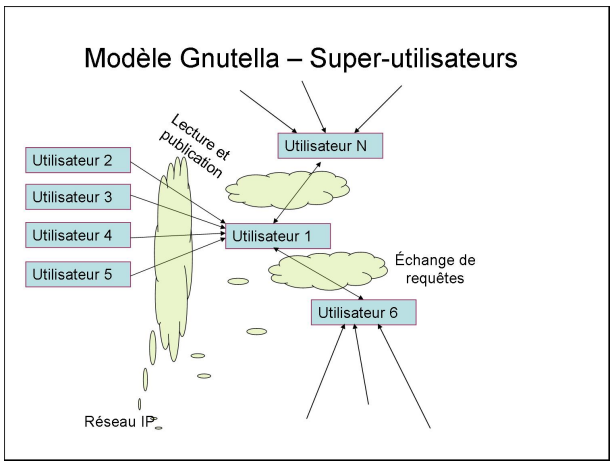

FIG. 5 – Organisation Gnutella

Cette architecture (voir figure 5) est en constante évolution, mais les principes de base restent toujours les mêmes. Un utilisateur se connecte à un serveur qui lui donne l'adresse IP d'un utilisateur proche de lui et il tente de s'y connecter. Il envoie ensuite la liste des fichiers qu'il partage à cet utilisateur. Lorsqu'une recherche est lancée le serveur proche est d'abord interrogé s'il n'a pas de réponse il envoie la requête en broadcast aux serveurs qu'il connaît avec un *TTL* (Time To Live) fixé au bout duquel la requête n'est plus forwardée. Quand un des serveurs est capable de donner une réponse favorable il envoie directement sa réponse à l'utilisateur initial (si le serveur disparaît du réseau il pourra être remplacé pour chacun des utilisateurs clients). Certains des utilisateurs deviennent eux-mêmes serveurs. Ils entrent alors en connaissance avec de nombreux autres serveurs et recensent les informations des utilisateurs qui lui sont directement connectés.

Il faut aussi noter que Gnutella est un des premier logiciel a avoir lancé le hording. C'est à dire que les utilisateurs qui téléchargent un même fichier peuvent en prendre des bouts de différentes sources. Ceci n'est pas simplement comme la fonction "resume" de certains bons logiciels de téléchargement qui peuvent changer de site au milieu d'un téléchargement. Mais il s'agit de télécharger différents morceaux sur différents utilisateurs en même temps. On a vu apparaître ces fonctions vers 2000 avec des clients de téléchargement comme ReGet ou GetRight, dans le domaine P2P, il s'agit de permettre aux utilisateurs qui sont en train de télécharger un fichier d'uploader les morceaux qu'ils ont déjà pour désaturer les Peers qu'ils utilisent.

Le principe du hording est que les utilisateurs téléchargeant un fichier donné font connaissance au gré des nécessités. Le fichier original est divisé en blocks qui sont les unités qui s'échangent dans la horde. Une fois un block téléchargé, il pourra être uploadé. Un système de signature (MD4 dans Kazaa) permet d'authentifier les blocks en plus des CRC pour contrôler les erreurs. Les détails fins des algorithmes de hordes en sont toujours au stade de la recherche.

1.4 Freenet

Freenet est le premier réseau P2P à avoir utilisé la méthode des tables de hachage pour distribuer les informations, ceci permet de mettre ne place des protocoles de routage pour retrouver de manière à peu près impossible à retracer une information disponible dans le réseau. De plus la recherche devient exhaustive. Pour commencer on se donne une fonction H qui à n'importe quelle entrée attribue une sortie (typiquement 128 ou 160 bits) qui correspond à l'espace des adresses virtuelles du réseau P2P. Le principe de base est que les clients se connectent avec une identité aléatoire (un $H(...)$), le point d'entrée du réseau leur désigne des voisins auxquels il se connecte directement par IP. Chaque nom de fichier est haché et envoyé par le protocole de routage à l'utilisateur dont l'adresse est la plus proche selon une distance définie dans le protocole. La recherche se fait ensuite de même en en prenant H, ce qui fournit l'adresse du tiers qui connaît le possesseur du fichier recherché. Ce tiers renvoie l'adresse virtuelle du possesseur du fichier au demandeur par l'intermédiaire du tiers.

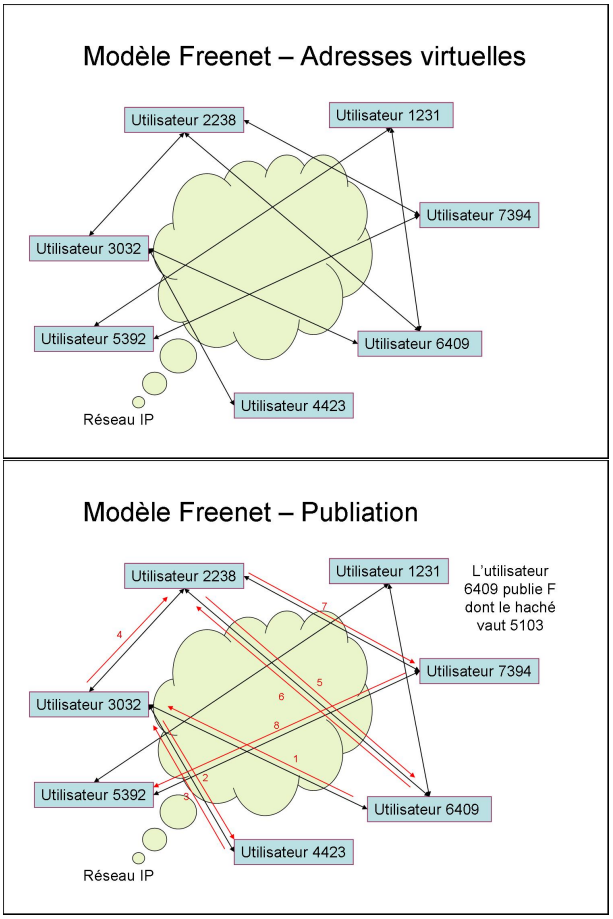

FIG. 6 – Organisation de Freenet

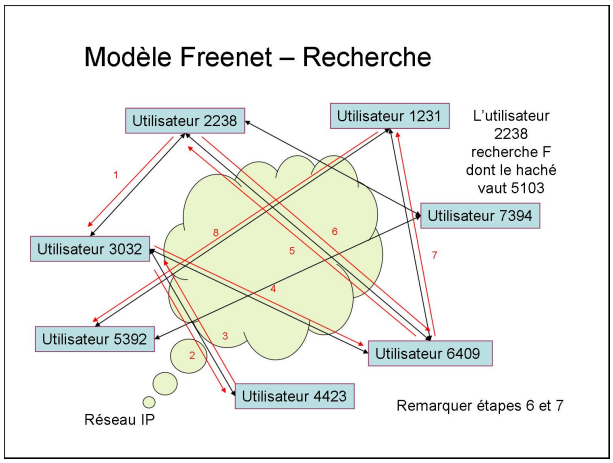

FIG. 7 – Recherche avec Freenet

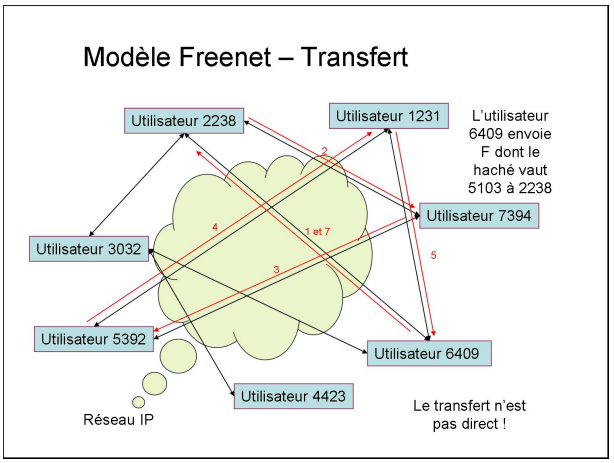

FIG. 8 – Transmission avec Freenet

Ensuite le transfert se fait en utilisant le réseau virtuel pour éviter que les parties puissent savoir avec qui elles s'échangent leur fichier en passant toujours par le tiers, mais avec des tables de routages maintenant mieux définies. Les utilisateurs sur le chemin sont utilisés pour stocker le fichier, et sont amenés a se déclarer comme origine du fichier en envoyant de plus une clé publique d'encryption qui permet a la source de réaliser un chiffrage qui sera décodé au cours du cheminement du fichier, mais dont seul l'utilisateur final verra le contenu original. On peut voir sur la figure 8 à quel point ce type de réseaux est inefficace du point de vue du transfert de données : un utilisateur peut être amené à télécharger un fichier une fois en tant qu'intermédiaire et une autre fois pour lui-même. Par contre dans ce dernier cas on devine bien le niveau de sécurité du protocole puisqu'un utilisateur ne sait même pas quand il participe à télécharger un fichier qu'il a lui-même demandé.

1.5 D'autres pistes

1.5.1 Bit Torrent et ses clones

Pour commencer on peut citer un programme qui marche très bien, Bit Torrent [3]. Il n'implémente pas de fonction de recherche P2P, mais permet à des serveurs (que l'on nomme des Seeds dans les hordes) de distribuer la charge en utilisant les capacités d'upload des personnes qui téléchargent un fichier en utilisant une méthode de de horde. La recherche comme dans le cas des anciens téléchargements par site FTP interposé sont organisés grâce à des forums de discussion. Certains clients Bit Torrent implémentent des fonctions de recherche P2P en utilisant le protocole de Gnutella. Il est à noter que, sous certaines conditions, des hordes peuvent exister sans la présence du serveur initial. Cependant le serveur est le point d'accès à la horde, il est donc nécessaire si on n'apporte aucune amélioration à l'algorithme de recherche dans ce sens.

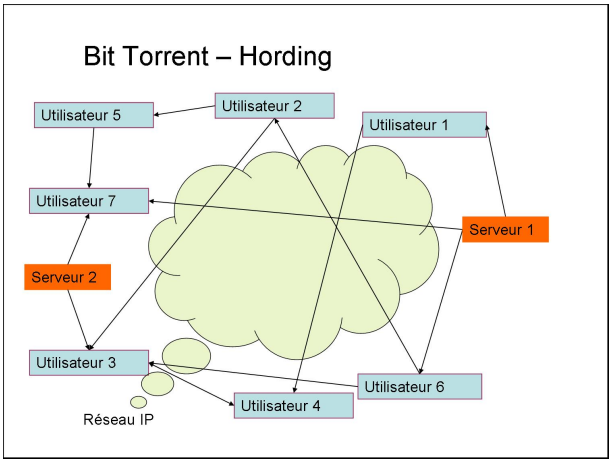

FIG. 9 – Téléchargement en horde

Il existe aussi des versions commerciales qui sont des implémentations de Kazaa lite (donc des clones de Gnutella) comme Brillant Soft, Ren Swoosh ou Kontiki [4]. Il est à noter que contrairement à Bit Torrent, ces derniers installent toutes sortes de logiciels indésirables par la même occasion et qu'ils ne sont pas libres. De plus si Bit Torrent est pensé pour améliorer l'idée de horde, les versions commerciales reposent sur un logiciel qui développe à peine l'idée de la horde et qui est assez loin de l'état de l'art (du moins au moment où je parle).

1.5.2 Nevrlate

Ensuite bien que peu diffusée pour l'instant c'est une amélioration qui peut permettre de fabriquer un réseau commercial de type P2P. Il permet de réaliser des recherches quasi exhaustives dans le réseau tout en réglant le problème du passage à grande échelle de réseaux P2P. L'idée est de diviser les N utilisateurs du réseau P2P en $\sqrt{N}$ groupes de $\sqrt{N}$ utilisateurs qui forment une clique (ceci est rendu nécessaire par les opérations de mitose d'un groupe). Chaque utilisateur est connecté en plus des individus de sa clique à un individu dans chacune des autres cliques. La diffusion d'information se fait en envoyant dans chaque clique un exemplaire du lien symbolique vers l'utilisateur. Pour la recherche on diffuse la demande dans la clique puis si personne n'est en mesure d'apporter une réponse positive, la demande est relayée dans d'autres groupes. Le nombre d'utilisateurs dans un groupe peut fluctuer au gré des arrivées et des départs, Nevrlate permet de maintenir par un mécanisme de fusion de groupes et de mitose des groupes de tailles contrôlées.

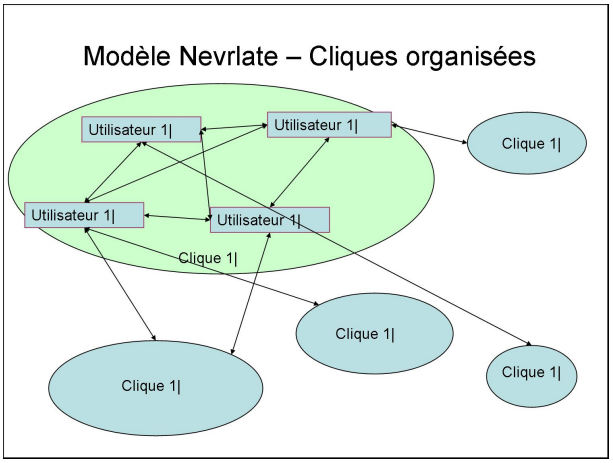

FIG. 10 – Les cliques de Nevrlate

1.5.3 Les clones de Freenet

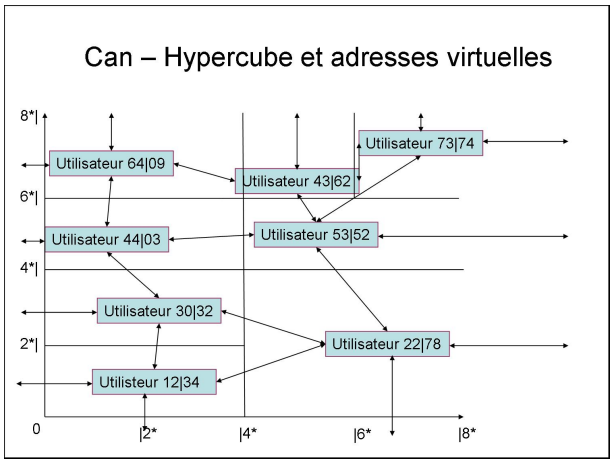

FIG. 11 – Architecture de CAN

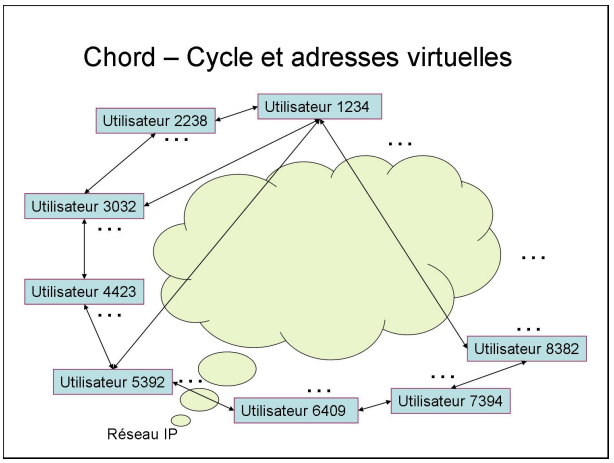

FIG. 12 – Architecture de Chord

Enfin il existe plusieurs architectures certaines assez fantaisistes (Chord) qui reprennent l'idée de la table de hachage distribuée pour mettre en place des algorithmes de routage. On peut citer CAN [8] qui repose sur un quadrillage multi-dimensionnel dans lequel

les utilisateurs viennent se placer, il peut être vu comme une amélioration de la partie réseau de Freenet. Il existe aussi Chord [9] qui a fait beaucoup de petits (Pasty et Taperty [10] par exemple) et qui repose pourtant sur une architecture qui n'a aucune chance de fonctionner avec des utilisateurs qui arrivent ou qui partent puisque les utilisateurs sont sensés maintenir des liens avec leur prédécesseurs et leurs successeurs pour l'ordre des identités virtuelles, ainsi que des liens vers des voisinages de leur identité plus un 2^k pour les $k < \log_2(ID_{max})$. Les défauts de Chord surtout par rapport à son homologue CAN sont criants : même si la complexité de routage est en log pour Chord et en racine de l'ordre de la dimension pour CAN sans liaisons longues (dim=2 dans la figure 11 ; on rappellera que $\sqrt[4]{10^6} = 30; \log_2(10^6) = 20$), CAN peut maintenir sa structure car les liens obligatoires sont tous locaux par rapport à la structure. Dans le cas de Chord, par contre on voit mal comment la structure résiste aux arrivées et aux départs des utilisateurs. J'ai même vu des démonstrations (que je ne citerai pas par bonté, mais qu'on trouvera dans les références) de maintient de la structure de Chord ou on suppose que chaque utilisateur connaît à tout moment son prédécesseur et son successeur. Je me bornerai à dire que c'est une grave erreur de raisonnement, mais il est probable que ce soit de la malhonnêteté intellectuelle caractérisée. Curieusement Chord et ses avatars font l'objet d'un gros effort de recherche au point que les idées intéressantes d'améliorations des réseaux P2P peuvent passer inaperçues.

2 Organisation d'un P2P

L'appellation P2P regroupe en fait deux briques de base :
- la recherche d'information de localisation du fichier,
- le rapatriement du fichier.

qui reposent sur la structure que l'on a bien voulu donner au réseau. Un même réseau peut être organisé de manière identique pour la recherche et le téléchargement comme dans le cas de Freenet ou du modèle client serveur. Les structures peuvent aussi être totalement différentes, ce qui est devenu le cas le plus courant.

2.1 la structure du réseau : un pas préalable à la construction d'un P2P

C'est elle qui permet par la suite d'implémenter des protocoles de recherche de fichiers et de téléchargement. L'un des extrêmes est le modèle client-serveur ou tout le monde est relié à un point donné. De l'autre coté on trouve Gnutella où chacun se connecte comme il le désire. Entre les deux on trouve deux axes de développement :
- les réseaux à identités virtuelles comme Freenet d'une part
- et les réseaux par groupes ou les utilisateurs se regroupent dans différents amas comme Nevrlate d'autre part.

Notons d'une part que Nevrlate a un système d'identités virtuelles, d'autre part qu'il fournit le moyen d'adresser un groupe en entier comme une entité unique qui est elle résistante aux pannes individuelles. Mais ce dernier point n'est pas encore développé.

Le choix de la structure du réseau est déterminant pour ce qui peut être fait par la suite, cependant il faut se méfier d'un écueil : une structure trop complexe dont les performances sont "optimales" et qui n'est pas faite pour résister aux pannes n'est pas adaptée au contexte. Chord est l'exemple typique qui ne doit pas être imité.

2.2 la recherche de l'information

Elle passe la plupart du temps par la publication des données des nouveaux arrivants (avec peut-être des mises à jour pour se prémunir contre les pannes). La recherche elle-même se fait par flooding ou par routage, essentiellement on ne peut pas espérer faire de routage vers l'information si on ne sait pas à l'avance où la trouver. Ainsi pour trouver une information sur la localisation d'un fichier par une méthode de routage il est nécessaire que les utilisateurs utilisent une méthode commune les amenant à se connecter au même tiers. La méthode de la table de hachage distribuée est un moyen très simple de réaliser cela.

2.2.1 Le flooding

Cette méthode est particulièrement adaptée au contexte P2P, la demande est diffusée à tous dans le voisinage du demandeur avec un nombre de sauts déterminé. Les fichiers très demandés finissent par arriver au demandeur essentiellement par un mécanisme de diffusion, qui doit pouvoir faire l'objet d'une étude mathématique.

2.2.2 Les tables de hachage distribuées

La méthode est triviale : on choisit une fonction de l'espace des fichiers partagés vers l'espace des adresses virtuelles, et la personne qui recherche un fichier et celle qui en possède un occurrence vont se connecter via l'intermédiaire qu'ils vont déterminer de la même façon.

2.3 le rapatriement du fichier

2.3.1 transfert par l'intermédiaire du réseau virtuel

Selon le niveau de sécurité/anonymat recherché le transfert se fait soit directement par FTP/HTTP soit par l'intermédiaire du réseau virtuel pour que les parties ne puissent pas se connaître comme dans le cas de Freenet. Dans ce dernier cas il est nécessaire d'avoir une structure du réseau qui supporte le routage. Si ce niveau de sécurité n'est pas requis, il ne faut pas utiliser le transfert par le réseau virtuel car il produit une surcharge importante du réseau (contrairement à ce qui est affirmé parfois de manière assez douteuse, il faut imaginer un facteur multiplicateur de l'ordre de 20 plus qu'un petit overhead de 10%).

2.3.2 Algorithmes de hordes

Les utilisateurs qui téléchargent un même fichier s'organisent entre eux pour améliorer les performances. C'est un domaine très peu étudié, mais qui est utilisé par la plupart des Gnutella-like et dans Bit Torrent (dont c'est le principal apport).

3 Précisions à propos des erreurs communes autour des P2P

3.1 Problème du routage

Le routage requiert un format précis sur les demandes. Par exemple si on recherche une musique de Bach, un système de routage est voué à l'échec puisqu'une musique de Bach n'aura à coup sur pas le nom "Bach" ou "Bach.mp3", il faut soit que le demandeur soit capable de demander "Bach-Piano bien tempéré-artiste libre de droit.mp3" ce n'est pas pensable, soit que la publication du fichier se fasse sur toute les demandes raisonnables que l'on fera sur le fichier, "Bach", "Piano" (disons que les majuscules et les minuscules sont identiques), "tempéré" (avec ou sans accent). Mais ceci pose un gros problème parce que "Piano" est trop générique, de même que "Bach", il faut imaginer le travail qu'aura le tiers qui doit se charger du mot "the" par exemple. Pour résumer le routage P2P est du domaine de l'utopie, il est tellement plus facile de traiter une demande et de voir si on possède un fichier qui peut y répondre, on peut même supporter les fautes d'orthographe ! En résumé, à moins d'avoir résolu le problème du format des noms de fichiers disponibles (c'est à dire dans un contexte où les noms sont attribués de manière centralisée en gros), ou bien de faire un hachage sémantique, ce qui est un problème assez délicat à régler, le routage ne devrait pas s'appliquer en dehors de Freenet où les utilisateurs s'échangent des données ultra-confidentielles. La question à se poser est en fait de savoir si on souhaite vraiment participer à fabriquer un P2P dont le but est le transfert d'images prohibées !

Cependant, il existe principalement trois cas où le routage peut être utile : le DNS distribué, le service de messagerie instantanée ou de téléphone sans serveur central. Dans ces cas, l'orthographe des adresses est de toute façon très stricte.

Il est à noter que l'idée de routage n'a de sens que si on réussit à assurer la connexité du réseau.

3.2 Efficacité et anonymat sont antagonistes

Un des points de base pour s'assurer l'anonymat est d'être connu par un minimum des personnes et que ces personnes ne puissent pas savoir si les données qu'elles reçoivent de moi ou que je leur envoie m'intéressent ou si je ne suis qu'un simple relais (et ni l'origine ni la destination). Ceci interdit par exemple de router des paquets avec les vraies adresses, on doit nécessairement faire appel à des adresses virtuelles, de plus

ces adresses ne doivent pas pouvoir fournir d'informations sur l'utilisateur, elles sont donc tirées au hasard. La conséquence est qu'on ne doit pas espérer d'améliorations du routage par des considérations géographiques.

De plus l'anonymat suppose que la transmission effective des données (le gros fichier) se fera par le réseau virtuel en repassant par le chemin inverse de la demande de renseignement c'est à dire par le tiers.

Enfin à moins d'imaginer que le tiers ait une information sur les personnes téléchargeant un certain fichier, il doit juste servir d'intermédiaire et ne doit pas avoir d'information en clair sur ce pour quoi il est intermédiaire. Tout ceci multiplie la charge globale du réseau physique par un facteur très important, bien au delà de ce qui peut être simplement négligé (il n'y à qu'à voir le cheminement d'un fichier téléchargé dans la figure 8).

3.3 La recherche dans un réseau P2P ne doit pas être exhaustive

Si on est dans le cas où des personnes décident librement de partager des données et d'arrêter de les partager, il est impossible d'assurer une recherche exhaustive d'un fichier à un temps donné sous des hypothèses qui restent raisonnables et sans centralisation des données. En effet il paraît naturel, même avantageux, qu'une donnée rare soit plus difficile à trouver que d'autre plus courantes. On ne voit pas bien pourquoi il faudrait avantager la recherche équitables des données que presque personne ne veut. Un réseau P2P est fait pour partager des données populaires, et pour des données très rares on fera avantageusement appel à une publication vers un serveur.
Pour illustrer mon propos, imaginons qu'un jour la Bibliothèque de France décide de se connecter à un réseau P2P, soit on accepte qu'elle publie des informations redondantes sur ses fichiers ce qui va inonder le réseau, soit on accepte que les recherches aillent jusqu'à la BNF, mais alors le réseau va se trouver inondé par les demandes.

4 Conclusion : Points à retenir pour fabriquer un bon P2P

4.1 Les algorithmes de hordes, une amélioration de TCP

En ce qui concerne la partie du transfert de fichiers, à part du point de vue des problèmes de protection de l'information, on a tout intérêt à adopter les algorithmes de hordes. Même si de nombreuses questions restent posées quant à la sélection de paramètres optimaux vis-à-vis de certaines contraintes, il résulte de leur utilisation une amélioration de la capacité des serveurs. Surtout l'idée de la horde permet de passer d'un domaine où plus une information est recherchée plus elle est dure à obtenir à cause de la saturation du serveur à une situation où le nombre joue un rôle positif du point de vue du serveur, moins saturé, et des utilisateurs qui téléchargent plus vite leur information. C'est pourquoi à mon sens un bon programme de partage de fichier P2P doit reposer sur un bon programme de hording. Pour bien faire il doit pouvoir utiliser

le programme de horde comme un module externe (Plug in) qui peut être changé au fil des améliorations.

4.2 L'architecture du réseau : Peut-on l'imposer ou doit-on la subir ?

La question reste pour l'instant largement ouverte. En effet les implémentations viables et efficaces (ce qui écarte Freenet) de partage de fichiers P2P reposent pour l'instant toutes sur une architecture aléatoire comme dans le cas de Gnutella. Les utilisateurs ne font pas d'effort particulier pour avoir une organisation cohérente. Des solutions proposées semblent pouvoir venir d'architectures comme CAN ou Nevrlate, mais sans que l'on retienne forcément les protocoles de recherche proposés pour chacun d'entre eux.

4.3 Structure conseillée

Il semble bien que malgré les efforts consentis pour faire disparaître l'idée de serveur, la seule réponse raisonnable à l'heure actuelle pour l'organisation des utilisateurs d'un réseau virtuel soit celle adoptée par Gnutella. Ceci avec un réseau de connaissances établis de manière aléatoire. Pour donner une illustration dans la vie courante, on pourra prendre comme exemple le domaine commercial. Le même type de problèmes se pose pour l'échange de ressources. Les réponses trouvées passent soit par les connaissances locales (réseaux de collecte/distribution), soit par les marchés de plus ou moins grande taille. Rechercher une marchandise quelconque qui quelque-part dans le monde sans passer par des réseaux de proximité ou des marchés centralisés est à peu près impossible.

C'est pourquoi on aura sans doute intérêt à garder deux moyens de transmission des fichiers et trois type d'objets dans la structure :

- une structure locale très dense du type de Gnutella
- des lieux de publication des fichiers du type collecte
- des lieux de publication des demandes du type centrale d'achat.

Il est à noter que Gnutella forme des réseaux qui suivent des lois de puissance pour l'incidence des sommets du graphe virtuel ce qui n'est pas nécessairement souhaitable, en tout cas ce n'est pas du tout égalitaire. De plus il faut décider à un moment si certains utilisateurs ne doivent pas devenir des super-utilisateurs pour contrôler et organiser le réseau.

Références

[1] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for massive graphs. pages 171–180, 2000.

[2] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet : A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009 :46+, 2001.

[3] B. Cohen. Incencitive build robustness in bit torrent. Technical report, May 2003.

[4] Kontiki corp. Kontiki. Technical report, 2003.

[5] Andrew C. Fuqua, Tsuen-Wan Johnny Ngan, and Dan S. Wallach. Economic behavior of peer-to-peer storage networks, 2003.

[6] N. Harvey, M. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet : A scalable overlay network with practical locality properties, 2003.

[7] P. Keleher, B. Bhattacharjee, and B. Silaghi. Are virtualized overlay networks too much of a good thing, 2002.

[8] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content addressable network. In *Proceedings of ACM SIGCOMM 2001*, 2001.

[9] Ion Stoica, Robert Morris, David Karger, M. Francs Kaashoek, and Hari Balakrishnan. Chord : A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.

[10] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry : An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley, April 2001.