# Mécanismes incitatifs pour les réseaux P2P de contenu Audio/Vidéo en temps réel

Rapport de stage de Master Réseaux

Etudiante : Jingxian.LU
Jingxian.lu@etu.upmc.fr
2761468

Encadrants :
Bénédict.LE GRAND
Olivier.FOURMAUX

Université Pierre et Marie Curie - Paris 6
Master Informatique- Réseaux

Août 2008

# Table des matières

# Remerciements

Ce stage a été effectué au laboratoire d'Informatique Paris 6(LIP6), Université Pierre et Marie Curie - Paris 6, dans le cadre du Master Recherche Réseaux.

Je remercie Mme. Bénédict Le Grand de m'avoir accueilli au LIP6 ainsi que M.Olivier Fourmaux de m'avoir accueilli et encadré au sein de son équipe dynamique et compétente.

Je remercie tout particulièrement Mme. Bénédict Le Grand. Mon travail n'aurait pas pu se faire sans son aide précieuse, sa grande disponibilité, sa patience, sa gentillesse et ses compétences.

Je présente enfin mes remerciements à Thomas Silviston. Ses conseils pratiques, ses connaissances en réseax et sa gentillesse m'ont aidé à m'adapter très vite au laboratoire.

# Abstract

With the progress of networks technologies, more and more peoples want to use P2P networking to communicate with each other and share files. Most current P2P file-sharing systems treat their users as anonymous, unrelated entities. They completely disregard any social relationships between them. Lack of cooperation (free-riding) is one of the key problems that confronts today's P2P systems. However, social phenomena, such as friendship and the existence of communities of users with similar tastes or interests, may be exploited in such systems in order to increase their usability and performance.

To increase the use and the performance of network, we implement social functions in the BitTorrent client-ABC by maintaining a social network and using "Buddycast" and "Cooperative Download". The name of this new software is "ABC social". Then, we present the architecture of "ABC social" and the functions of each part. After the test for "social ABC", we show the evidence that it brings a significant increase in file download speed by exploiting idle upload capacity of online friends.

**Keywords :** P2P, incentive mechanism, social community, P2P Streaming

# Résumé

Avec l'avancement des technologies des réseaux, de plus en plus de personnes ont la tendance à utiliser le réseau P2P pour des communications et des partages de fichier. Mais la plupart des systèmes P2P de partage de fichiers traitent leurs utilisateurs comme des anonymes, des entités sans relation. Ils négligent complètement des relations sociales entre eux. Le manque de coopération (*free-riding*) est un des problèmes principals qui affronte le système P2P aujourd'hui. Pourtant les phénomènes sociaux, comme l'amitié et l'existence de communautés d'utilisateurs avec les mêmes goûts ou les mêmes intérêts, peuvent être exploitées dans ces systèmes pour accroître leur utilisation et performance.

Afin d'accroître l'utilisation et la performance du réseau, nous avons implémenter les fonctions sociales au client BitTorrent-ABC en maintenant un réseau social et utilisant "*Buddycast*" et "Téléchargement coopératif". Ce nouveau logiciel est applé "ABC social". Ensuite, nous présentons l'architecture de "ABC social" et les fonctions de chaque partie. Et d'après le test pour "ABC social", nous montrons l'évidence qu'il a la capacité à réaliser une augmentation évidente pour la vitesse de téléchargement en employant la capacité oisive de upload des amis en ligne.

**Mots-clés :** P2P, mécanismes incitatifs, communauté social, P2P Streaming

# Chapitre 1

# Introduction

Dans la dernière décennie, le nombre d'utilisateur de l'Internet a augmenté extrêmement rapide. Mais il est ainsi devenu un phénomène social complexe, une communauté virtuelle où touts les utilisateurs interagissent ensembles et où toutes les types de comportements peuvent être identifiés facilement. De simples applications comme le web ou l'email, L'Internet permet maintenant de fournir des contenus multimédia riches et variées. Toutefois le multimédia utilise beaucoup de ressource, qui limite l'efficacité d'une solution basée seulement sur l'utilisation de serveur. Une alternative pour le modèle habituel de client-serveur à distribuer les données est l'utilisation du modèle Pair-à-Pair (P2P). Figure 1.2 montre que la différence entre un modèle client-serveur et

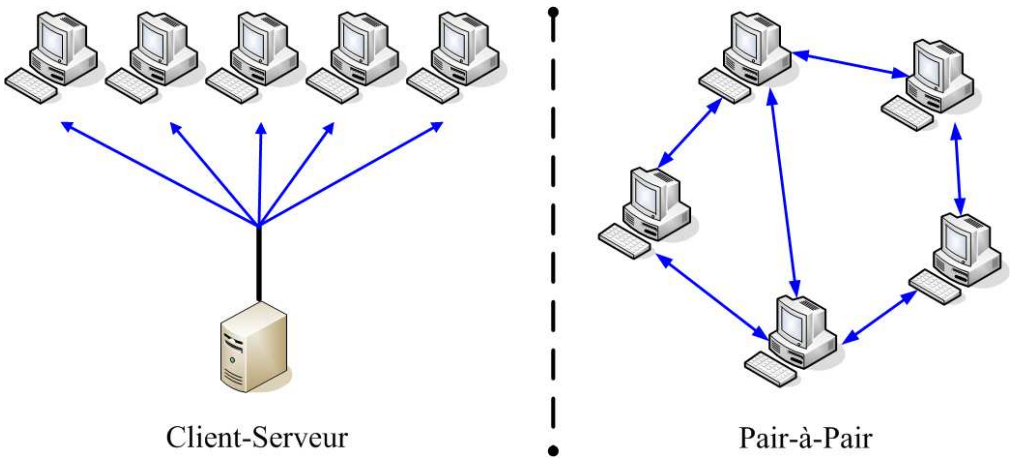


FIGURE 1.1 – Le modèle Client-Serveur et le modèle P2P

un modèle P2P. Dans l'architecture traditionnelle du réseau, les fichier sont stockés dans un serveur. Le serveur central peut servir les utilisateurs. Les utilisateurs peuvent connecter et demander un fichier d'un serveur. Mais cette

architecture est plus simple, il a des inconvénients. Le prenier est que le coût de scalabilité est plus élevé et le serveur central est cassé facilement. Mais dans l'architecture P2P, les fichiers sont stockés et distribués parmi les ordinateurs qui sont une partie du réseau. Un ordinateur est un client et il peut aussi fournir les services comme un serveur. Comme ils se révèlent efficace, les systèmes P2P sont de plus en plus populaires. Figure 1.2 [1] indique d'ailleur que leur utilisation ne peut qu'augmenter ces dernière années. Les réseaux P2P sont déjà largement utilisés autour de l'Internet, principalement pour le partage des fichiers.

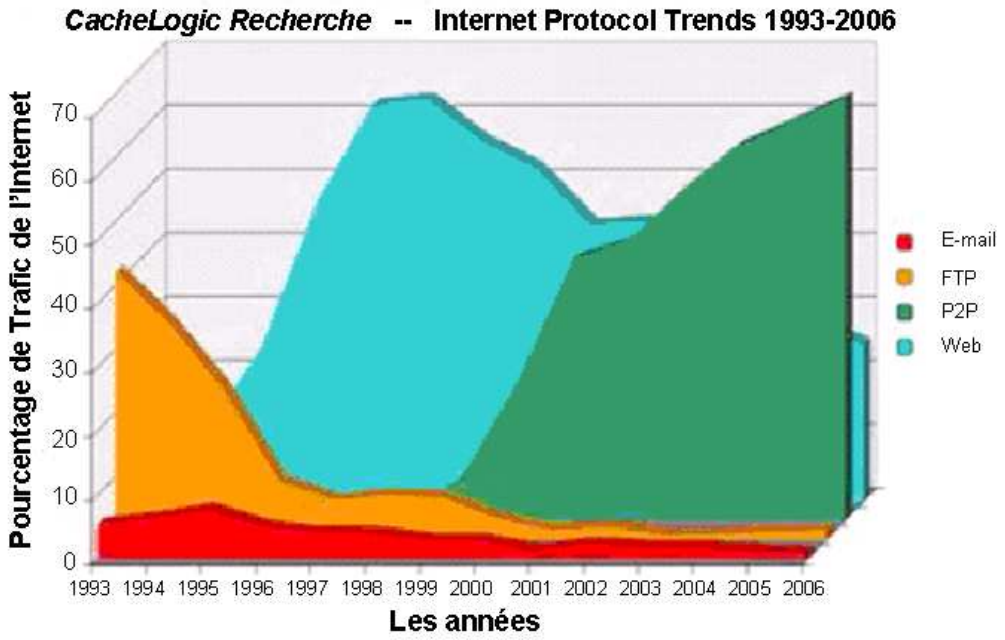


FIGURE 1.2 – Les volumes des données d'application utilisés de l'Internet

Les applications de réseaux P2P et les partages des fichiers constituent une importante tendance du trafic de l'Internet aujourd'hui et dans les habitudes de ses utilisateurs. En fait, l'utilisation de l'Internet d'aujourd'hui est beaucoup orientée par les contenus. L'évolution de P2P, qui est commencé par *Napster* [2] qui est l'émergence de partage de fichiers, a conduit à système de nouvelle application de live streaming, comme *PPLive*[3], *CoolStreaming*[4] ect. La nature de distribution de ces applications est un choix technologique qui vise à satisfaire un nombre d'utilisateur sans cesse croissant, ce qui ne peut être assuré par l'utilisation de plusieurs serveurs ou multiples serveurs. De plus, en l'absence d'un service IP multicast soutenu par l'infrastructure réseau (routeurs), une solution alternative devait être trouvée à la couche applica-

tive. Dans cette nouvelle approche, les nœuds sont organisés en une topologie d'overlay. Ils forment un réseau virtuel sur l'Internet. Ce réseau d'overlay est une couche de la topologie du réseau virtuel au-dessus du réseau physique. Il comprend tous les pairs participants comme les nœuds du réseau. Chaque deux nœuds du réseau d'overlay peuvent directement communiquer l'un et l'autre. Mais il n'y a pas de serveur central, c'est difficile de gérer tous les utilisateurs d'overlay. Donc sa sécurité et la protection de privé sont un grand problème.

Et plus, les interactions compliqués des utilisateurs dans les environnements virtuels ont réveillé la curiosité des psychologues et des sociologues, qui ont essayé d'expliquer comment l'appartenance au sein d'une communauté peut stimuler pour contribuer de façon désintéressée au sein de la communauté. Naturellement, après cela, les programmeurs et les ingénieurs croient qu'ils pourraient utiliser ces aspects intéressants du comportement humain et créer des mécanismes incitatifs pour la contribution, la coopération et l'altruisme dans les réseaux P2P. Derniéres recherches ont proposé la formation des communautés locales basées sur des critères géographiques qui pensent qu'il est plus facile pour les utilisateurs de faire confiance entre les uns et les autres s'ils ont de la chance d'être physiquement connu ou. Donc, l'unité des communautés P2P est une question de confiance.

## 1.1 Présentation de travaux du stage

La distribution Audio/Vidéo en temps réel (comme Live streaming) à large échelle est une application émergente qui pose de nombreux problèmes. L'évolution considérable du débit et du nombre de destinataire nécissite la mise en place d'infrastructures de distribution complexes. L'utilisation d'approches basées sur les modèles P2P suscite un vif intérêt[5, 6, 7]. Ces solutions tentent de substituer les coûteux services de distribution des flux continus par des approches distribuées ou la complexité serait reléguée au niveau des pairs. La qualité de l'échange réalisé au niveau des pairs dépend fortement de la structure et du comportement de la communauté d'utilisateurs impliquée. Pour approcher une modélisation réaliste, la combinaison d'aspects sociaux semble essentielle[8].

Dans le cadre de ce stage, nous voulons étudier l'impact des communautés au système P2P. Le réseau P2P a besoin des mécanismes incitatifs pour encourager les pairs à partager des fichiers et contribuer les ressources. Le réseau P2P live streaming demande le partage des ressources plus fortement que le réseau P2P traditionnel. Parce qu'il a une contrainte du temps, et il doit télé-

charger des ressources plus rapidement. Pour le réseau P2P live streaming, les mécanismes incitatifs sont vraiment très importants.

L'objectif principal du stage est de mettre en œuvre des mécanismes incitatifs basé sur la relation sociale pour les réseaux P2P de contenu Audio/Vidéo en temps réel (P2P Live streaming), malgré que notre expérience ne mentionne pas la partie d'Audio/Vidéo en temps réel dans ce rapport. Les codes du système P2P traditionnel sont plus simples et plus évidents pour mettre en œuvre les fonctions sociales. Donc, nous implémentons d'abord ces fonctions sociales dans le système P2P traditionnel-ABC pour voir si ce mécanisme social est bien utile. Selon les résultats, nous allons l'appliquer sur le P2P live streaming dans l'étape prochaine.

## 1.2 Organisation du rapport

Ce rapport est organisé comme de la façon suivante. Après cette introduction, le chapitre 2 introduit les mécanismes incitatifs existants pour les systèmes P2P et présente leur avantages et inconvénients de chaque type de mécanismes incitatifs. Ensuite, nous présentons les système P2P de distribution des contenus, et plus particulièrement, le fonctionnement principal de BitTorrent et l'extention-*BiTos* pour *Streaming*. Après, nous analysons les influences des phénomènes sociaux pour le motivation de humain et présente les réseaux des communautés sociales et un système P2P basé sur la société : *Tribler*. La chapitre 3 présente l'implémentation de *Social ABC* que je met en œuvre en ABC pour améliorer la performance et éviter le problème de *free-riding*. Ensuite, la chapitre 4, je vais présenter l'expérience pour tester le performance de ABC social en détails et les résuletats. Finalement, la chapitre 5 conclure ce rapport et propose les traveaux futures.

# Chapitre 2

# L'état de l'art

Dans cette chapitre, je vais introduire les mécanismes incitatifs existants pour les systèmes de P2P et faire un résume des avantages et des inconvénients de chaque type de mécanismes incitatifs dans la section 2.1. Et puis, dans la section 2.2, j'introduis les système P2P de distribution de contenu, particulièrementet, je présente le fonctionnement principal de BitTorrent et l'extention-*BiTos* pour *Streaming* basé sur BitTorrent. A la fin, dans la section 2.3, j'analyse les influences des phénomènes sociaux pour la motivation des humains et présente les réseaux des communautés sociaux et un système P2P basé sur les comportements sociaux : *Tribler*.

## 2.1 Les mécanismes incitatifs

Dans la plupart des systèmes P2P contemporaine, les utilisateurs sont naturellement découragés de coopérer, puisque les contributions affectent directement leurs ressources et leurs performances. En conséquence, si un utilisateur essaye de maximiser ses performances, son action aura un effet négatif immédiat sur la performance globale du système. Donc, il est nécessaire de compromettre le personnel pour le bien-être collectif et de fournir aux utilisateurs des motivations importantes pour la coopération, sinon ils vont naturellement devenir égoïste. Ce compromis est très difficile à faire en raison des défis actuels dans un système P2P :

- Les populations des pairs sont très grands et dynamiques, ce qui signifie que la durée de vie de pairs dans le système peut être très court.

- La transaction asymétrique : Utilisateur A s'intéresse au contenu possédé par B, et B s'intéresse au contenu possédé par C. En conséquence, un utilisateur peut demander un service qui ne peut pas donner en échange

immédiatement. Donc une politique tit-for-tat simpliste n'est pas appropriée pour un tel système.

- La majorité des systèmes P2P permettent aux utilisateurs de changer constamment d'identité(coût zéro pour l'identité). Cette caractéristique peut accepter les nouveaux venus et aider les réseaux à contribuer rapidement. Mais dans le même temps, elle ne pénalise pas les utilisateurs malveillants qui prennent l'avantage sur cette vulnérabilité pour exploiter le système. Ce comportement s'appelle "whitewashing" et est l'un des problèmes majeurs de ce type d'application.

- La plupart des systèmes P2P sont décentralisés et non-structurels. Cette absence d'une autorisé central impose les utilisation de solutions complexes pour gérer le système. [9]

Cependant, il y a certaines techniques qui peuvent nous aider à affronter ces difficultés :

- Combiner aléatoirement et informer le pair sélectionné. Cette façon choisira les nœuds déjà connues et encouragera le maintien de partenariats à un long terme entre eux. Et pour le sélection du pair aléatoire vise à explorer le réseau pour un meilleur service et faciliter l'intégration des nouveaux venus.

- Maintenir un historique des échanges entre nœuds.

- Adopter un politique adaptatif et singulier comme une défense pour le problème de "white-washing". Ce qui signifie qu'ils ont une politique souple qui s'adapte aux inconnus pour leur comportement récent. Le problème de cette solution est que il est aussi possible de pénalise les nouveaux venus innocents si la majorité des inconnus récents sont prouvées qu'ils sont malveillants ou égoïstes.

Essentiellement, il existe trois catégories de mécanismes incitatifs, bien que les applications P2P contemporains les combinent par nombreuses façons pour améliorer les performances. Donc, la distinction entre eux n'est pas toujours évidente.

### 2.1.1 Micro payment

Ces mécanismes visent à établir un équilibre entre les téléchargement total et les upload total de chaque utilisateur. Une solution très simple est de débiter les utilisateurs pour chaque octet qu'ils téléchargent et de les rémunérer pour chaque octet qu'ils envoient. L'avantage de cette méthode est qu'elle donne aux utilisateurs la opportunité de contrôler leur dettes par rapport au système et de gagner de l'argent s'ils veulent. [10]

• **Avantages(⊕) et Inconvénients(⊖)**

⊕ Le contrôle de la dette
⊖ Le problème des connections lentes et asymétriques
⊖ Il existe quelques dangers de l'exploitation quand les utilisateurs combinent ensemble.
⊖ C'est difficile pour évaluer le fichier rare ou le fichier demandé, afin de rémunérer d'avantage les utilisateurs qui les upload.

• **La discussion**

**Echange des fichiers**
Une variante de cette technique est la politique tit-for-tat qui est adopté par BitTorrent. Ce mécanismes approprie à ces systèmes en raison de sa simplicité et sa facilité de implémentation.

**Live streaming**
Cette approche ou le tit-for-tat politique n'est pas suffisant pour Live streaming. On doit utiliser une technique qui décourage les free-riders et encourage tous les nœuds à contribuer constamment et continuellement. Sinon la qualité de service va se détériorer. La vulnérabilité de ce type de mécanismes est que les utilisateurs peuvent accumuler des crédits et après ne contribuer pas une émission populaire lorsque la coopération est essentielle.

### 2.1.2 Services différenciés

Dans ce genre de mécanismes, la motivation principale des utilisateurs est la capacité de choisir les pairs fiables qui peut leur garantir une qualité prévisible de service. Donc, tous les utilisateurs sont libres de choisir leurs niveaux de contribution afin de maximiser la qualité de vidéo aperçue. Par exemple, tel mécanisme pourrait permettre les pairs de sélectionner seulement les nœuds, qui ont les scores inférieur ou équal aux scores d'eux, comme uploaders. Par

cette façon, un nœud avec un score zéro aura la meilleur qualité d'effort de service. S'il désire un streaming meilleur que le meilleur effort, il doit obtenir un score positif par contribuer au système.

Le score de chaque utilisateur est déterminé par une fonction de compter qui peut considérer et mesurer la contribution de chaque utilisateur ou la contribution diminuée la consommation de chaque utilisateur. La fonction de compter peut même introduire un facteur d'âge pour encourager les contributions régulières. Elle peut aussi rémunérer plus d'utilisateurs qui upload les fichiers rares ou demandés fortement. La possibilité est infinie.

• **Avantages et Inconvénients**

⊕ La qualité de service est directement relative au niveau de contribution.
⊕ Il n'y a pas de coût supplémentaire.
⊖ Injustice : Le même traitement pour les free-riders et les venus.
⊖ La nécessité de une entité central pour calculer les scores des utilisateurs. Solution : Chaque utilisateur calcule son score local et fait une approximation.
⊖ Le problème de l'intégrité : Les utilisateurs peuvent mentir pour exploiter le système.

• **La discussion**

**Echange des fichiers**
On a déjà trouvé qu'il n'y a pas de l'application de ce mécanisme dans le système d'échange des fichiers. C'est normal, parce qu'il introduit la complexité dans le niveau élevé. Cette complexité affecte sérieusement les performances du système. Cette complexité est redondant, puisque le système peut bien marcher avec un algorithme plus simple.

**Live streaming**
Il semble qu'une solution basé sur la différenciation de la qualité de service est proposée pour la relation de contribution de chaque utilisateur. L'objectif de systèmes P2P en temps réel doit aspirer à Live streaming. L'avantage de cette technique est qu'elle affecte directement la qualité de la vidéo perçue. Et de cette façon, les utilisateurs ne peuvent pas aider aux contributions. L'urgence de la vivacité de l'émission constitue une motivation supplémentaire. Bien sûr, cette technique est basée sur l'existence d'un système de pointage. On peut utiliser cette technique pour améliorer les fonctionnalités et la performance du système.

### 2.1.3 Réputation

L'utilisation plus simple des mécanismes de la réputation est d'aider les pairs avec une bonne réputation pour trouver les uns envers les autres et coopérer. L'idée principale est de construire un système de réputation qui attribue un score objectif pour chaque utilisateur en fonction de son comportement et ensuite détermine sa qualité de service par ce score. Ici c'est évident que une entité centrale est nécessaire de maintenir et de mettre à jour les scores, qui n'est pas souhaitable pour les applications P2P moderne. Une alternative pour la solution d'une entité centrale est que chaque nœud calcule son score propre. Ensuite, son score est mappé dans une pourcentage. Il peut connaître sa position dans la distribution globale en comparant son score avec cela d'autre. Et alors, il peut agir en conséquence.

• **Avantages et Inconvénients**

⊕ La flexibilité de décision.
⊕ C'est mieux d'être accepté par les utilisateurs.
⊕ La défense contre les utilisateurs malveillants.
⊖ Les problèmes de l'absence de l'autorité centrale.
⊖ Une entité centrale est très chère.
⊖ C'est nécessaire de maintenir une histoire et échanger les messages de sondage. En conséquence, le coût et l'overhead sont élevés.

• **La discussion**

**Echange des fichiers**
Une variation de cette technique est adoptée par "KaZaa" où les utilisateurs sont divisés en trois classes de contribution. La nécessité d'un élément central et ce problème devient vraiment compliqué Cette technique n'est pas si efficace et si approprié comme le tit-for-tat pour les systèmes.

**Live streaming**
Les mécanismes de réputation constituent une solution très intéressant vers la direction correcte. Toutefois, l'efficacité de cette solution proposée dépend des réponses dont l'on va donner à certaines questions : S'il y a une entité centrale ou pas ? S'il existe une histoire ? Quels sont les paramètres spécifiques qui sont maintenus par chaque utilisateur ? Comment la réputation va influencer les décisions déjà faites ? Et comment la réputation détermine la qualité de service ?

## 2.2 Système P2P de distribution de contenu

Les systèmes sont utilisés le décentralisé pour distribuer des contenus, comme la radio ou la distribution de télévision. Dans ce modèle, ce n'est plus un unique serveur qui utilise ses ressources (Bande passante etc.) pour distribuer des contenus, mais plutôt les utilisateurs qui partagent leur ressoures.

Chaque utilisateur du protocole est appelé un pair. Quand un utilisateur veut télécharger un fichier spécifique, il doit lui demander directement à d'autres pairs qui ont le fichier complet, ou une partie du fichier. Cet utilisateur est aussi appelé leecher, ils peuvent télécharger chaque partie du fichier chez d'autres pairs. Dès qu'il a obtenu sa première partie, l'utilisateur peut donner à chaque pair qui le veut. Quand il termine son téléchargement, il devient un *seeder*, et distribue à contribuer le fichier aux autres *leechers*.

Ce protocole est rapide, parce qu'il est possible de télécharger de nombreuses parties d'un fichier de nombreux pairs en même temps. La solution qui utilise un serveur central peut être plus vite, mais il est plus cher, parce que le serveur a besoin d'une grande capacité de stockage pour stocker des fichiers, et une grande bande passante. En outre, un système pair-à-pair est plus résistant à l'échec, comme tous les protocoles décentralisés. Il n'y a pas un serveur central qui peut paralyser le système entier en cas de collision.

### 2.2.1 BitTorrent

C'est nécessaire de présenter les fonctions principales de BitTorrent. Il est un pair-à-pair protocole qui définit que comment trouver des autres pairs qui ont les fichiers que les utilisateur veulent. Il est la plus réussi prédécesseur de l'application actuelle de Live streaming. Le but premier de BitTorrent est la réplication rapide d'un grand fichier par un groupe de nœuds.

#### Fonctionnement

Le système de BitTorrent utilise un serveur central, appelé "Tracker", qui coordonne le téléchargement seul (*swarms*). Il maintient la trace de tous les pairs actifs et stocke les informations relatives. Il peut aussi effectuer certaines tâches administratives et donner certaines limites pour les utilisateurs, par exemple, en autorisant uniquement les utilisateurs enregistrés. Les utilisateurs ont besoin d'un méta fichier, appelé un "Torrent", qui décrit le contenu en termes de taille et les valeurs de *hash*. Il est l'ensemble des pairs coopératifs pour télécharger le même contenu. *Torrent* contient aussi l'adresse de Tracker. Les clients demande le tracker pour communiquer les autre pairs qui ont le

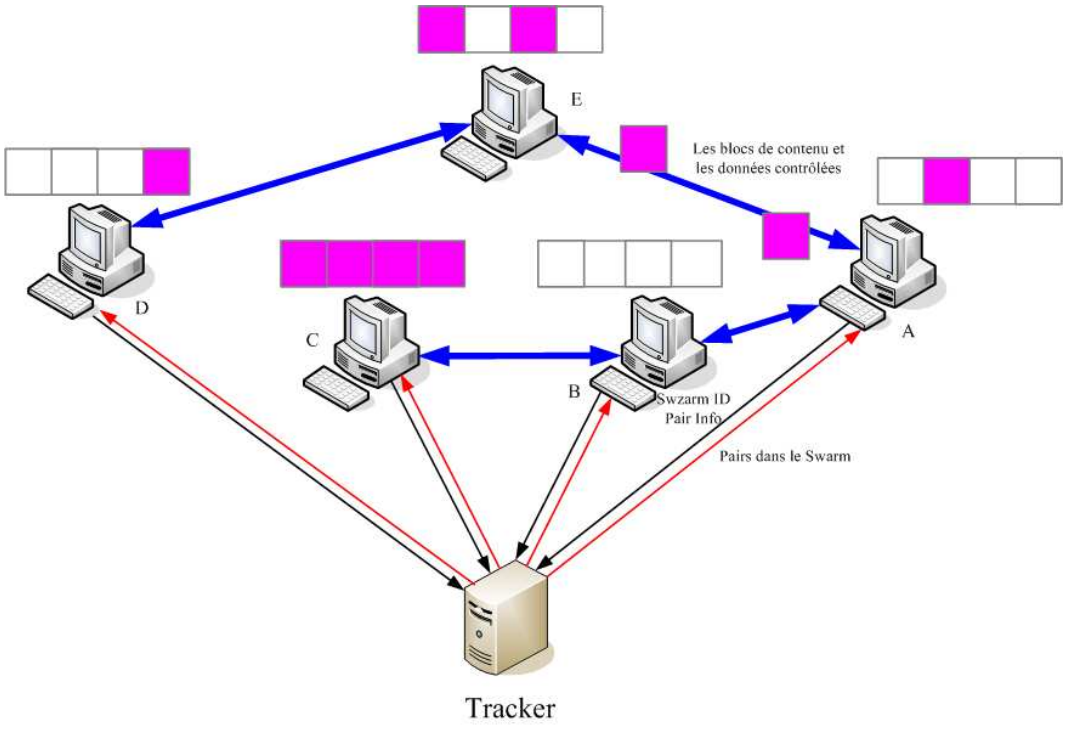


FIGURE 2.1 – L'échange de l'information dans le système de BitTorrent

contenu. Après obtenu l'adresse d'autres pairs, les connexions sont créées pour les pairs individuels. Les utilisateurs peuvent être mis en plusieurs *swarms* dans le même temps, qui sont indépendants pour l'un et l'autre. Le système divise le contenu en morceaux qui sont échangés entre les pairs. Parce que les morceaux peuvent être téléchargés de plusieurs pairs dans le même temps et le système est rapide et efficace. Les clients peuvent partager les morceaux qui ont déjà été acquis. Pour diminuer le nombre de *free-rider*, les utilisateurs stockent les données passés d'échanger pour un *swarm*. Les pairs qui contribuent beaucoup de ressources peuvent être servi plus vite. Le mécanisme de *tit-for-tat* est une incitation de partager, où chaque pair choisit de upload à ses pairs dès qu'il obtient quelques choses en retour. Figure 2.1 est un exemple de système de BitTorrent. Dans cet exemple, les blocs des données de le *swam* sont distribués parmi les pairs. Seulement le pairs C a le téléchargement complet, alors appelé un *seed*. Les autres pairs sont appelés *leechers*. Par exemple, le pairs A peut obtenir deux blocs de E et la dernière partie de B ou E, dès qu'ils l'acquièrent.

Dans l'article [11], les auteurs ont fait une analyse très intéressante montrant que les effets des mécanismes différents de BitTorrent sur les incitations de partage et *clustering*. Ils ont déduit que s'il y a un *seed* initial avec la capa-

cité de support un taux de upload élevé, les pairs de la même capacité d'upload ont tendance à former des *clusters*(groupes) et réciprocité avec des pairs dans le même *cluster*. Ce phénomène est évident dans la figure 2.2 [11] suivante : Les carrés plus foncés représentent des périodes de *unchoke* plus longes. Les pairs de 1 à 13 ont une limite de upload de 20kbps, les pairs de 14 à 27 ont 50kbps et ceux-là de 28 à 40 ont 200kbps, comme le *seed* initial. C'est évident que les trois catégories des pairs forment les *clusters* et interagissent à l'intérieur de *cluster*, sauf le cas d'un *optimistic unchoke*. On peut aussi voir que les nœuds lents *unchoke* souvent les nœuds moyens. Mais ce comportement n'est pas mutuel, les pairs moyens *choke* rarement les pairs lents car ils ne peuvent pas tirer profit des pairs lents et il n'y a d'intérêt à le faire.

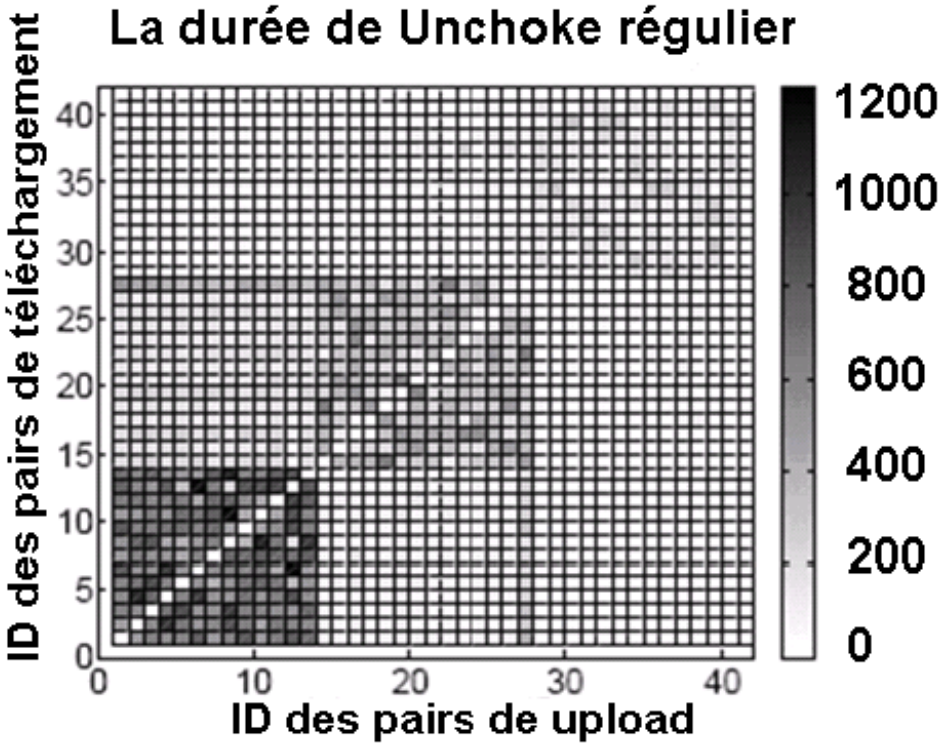


FIGURE 2.2 – "Clustering" et le partage incitatif dans le système de BitTorrent

### 2.2.2 BiTos-Extension pour streaming

Dans l'article [12], les auteurs proposent BitTorrent Streaming(BiTos), un protocole avec le capacité de supporter streaming basé sur BitTorrent. C'est une approche intéressante qui essaie de déterminer les modifications nécessaires minimums pour transformer BitTorrent en protocole qui a la capacité

de streaming. Les auteurs spécifient que la solution proposée concerne le Video on demand ant mais pas Live Streaming. Parce que les paquets sont dynamiquement créés dans le dernier cas, ils ne sont pas encore prêtes à commencer une session. Donc le problème est très compliqué de s'affronter à un seul modification du protocole existant. L'idée principale de leur proposition est de modifier l'algorithme de sélection de chunk pour donner une priorité élevée aux morceaux qui peuvent être reproduits rapidement par le media player : Essentiellement, ils introduisent deux catégories de morceaux, le *High Priority Set*, qui comprend les morceaux nécessaires pour la production d'une petite durée de la vidéo qui suit ; et le *Remaining Pieces Set*, qui comprend tous les morceaux qui n'ont pas encore été téléchargés et aussi n'appartiennent pas au *High Priority Set*. Figure 4.3[10] montre que comment Bitos fonctionne. De cette façon, chaque client décide de télécharger un morceau du *High Priority Set* avec une probabilité ***p*** et un morceau du *Remaining Pieces Set* avec une probabilité ***1-p***. De cette façon, on peut obtenir une meilleure performance. Encore, un long temps de buffer peut aussi augmenter la performance du protocole.

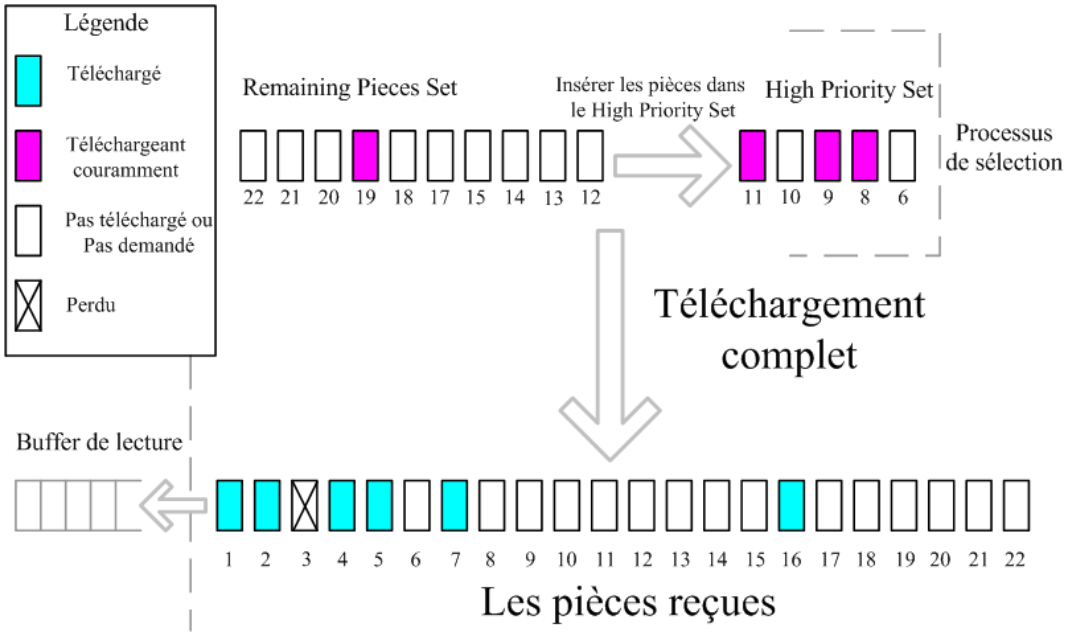


FIGURE 2.3 – BiTos-une approche pour supporter Streaming dans BitTorrent

Cette façon est faisable par le simulation. Et il peut facilement incorporer dans le protocole BT initial. Mais il manque de proposer une probabilité p qui a l'adaptation dynamique pour explorer et identifier les conditions. Si l'on fait

ça, il va être plus robuste dans l'environnement et peut améliorer mieux la performance de Streaming.

## 2.3 Les réseaux des communautés sociales

Une communauté dont les ordinateurs des membres sont reliés par une manière particulière que ils peuvent échanger des informations et communiquer les uns et les autres pour un but commun. Ces communautés ne sont pas nécessaires d'être défini par des frontières géographiques. Le réseau des communautés sociaux est destiné à créer un sentiment approprié sociale dans la communauté. Les réseaux P2P sont aussi une communauté sociale. Les pairs sont les membres de réseaux P2P. Donc, les enquêtes pour les comportements des participants dans la communauté sociale correspondent aussi aux les comportements des pairs dans le système P2P.

Les facteurs de motivation de humain ont déjà été examinés par Abraham Maslow et sa hiérarchie de la théorie du besoin de humain [13]. Dans un article sur l'utilisation de la psychologie sociale pour motiver les contributions aux communautés en ligne [14], une expérimentation a eu lieu où les problèmes de *under-contribution*(c'est-à-dire que un manque de contribution.) et *social loafing* ont été adressés. Dans cette article, comme la prévision de la théorie, les individus contribuent quand ils sont rappelés par leur unicité et quand ils ont obtenu les objectifs précis et défiés. Donc, c'est important de comprendre ce qui motive les personnes à contribuer et s'ils choisissent de contribuer aux communautés. C'est aussi important dans les systèmes P2P. Parce que les comportements ont lieu semblablement dans les systèmes P2P. La plupart des systèmes P2P sont basés sur les connexions anonymes parmi des pairs. Certains de ces réseaux sont entachés par des utilisations malicieux. Le problème majeur dans les systèmes existants est aussi l'absence d'incitation de participer au partage de contenu et de la bande passante. Beaucoup de personnes utilisent le système mais rien contribuent, appelé *free-rider*. La performance de la plupart des systèmes ont diminuée, après ils sont adoptés par les utilisateurs généraux qui ont tendance à être égoïste dans leur comportement. Seulement 10% de la communaut P2P partagent 99% de la bande passante. Les systèmes P2P existants sont soumises à des attaques ou au abus du réseau. Donc, afin de éviter que les utilisateurs ont tendance à maximiser leur profit et devenir un free-rider, le contrôle et la confiance sont distribués dans le réseau P2P. Et les phénomènes sociaux, comme l'amitié et l'existence de communauté d'utilisateurs qui ont les mêmes goûts ou les mêmes intérêts, peuvent être exploitées dans des tels systèmes pour augmenter les possibilités de leur utilisation et

performance. Des groupes fermés des utilisateurs ont tendance à mieux fonctionner en raison du contrôle social. L'exemple réussi est "Tribler" : un système P2P basé sur la société.

### 2.3.1 Les communautés virtuelles

Pendant la dernière décennie, beaucoup de personnes ont trouvé que l'Internet est une place facile pour contacter avec des personnes mondiaux. L'Internet est plus populaire et il est une partie de la vie sociale de personnes qui existent dans l'espace cybernétique et ne pas exister dans l'espace physique. L'existence de communautés virtuelles est un exemple. Une communauté virtuelle est un groupe de personnes sur l'Internet qui forme des agrégations sociaux basé sur des intérêts communs. Cette définition est tellement vaste, et l'on peut distinguer différents types de communautés virtuelles.

Exemples de communautés qui sont créés uniquement seulement pour les objectifs sociaux sont *Orkut* [15], *Friendster* [16], et *Hyves* [17]. Ces communautés organisent les relations sociales existantes dans une structure du réseau pour que les gens puissent naviguer sur le réseau et contacter avec des nouveaux amis. L'intérieur de la communauté, les membres peuvent constituer des sous-communautés et de composer leurs listes d'amis. Ces communautés en ligne ont aussi l'infrastructure de chat et d'échange de contenu.

Sauf les communautés purement sociales, il existe des autres communautés qui ont d'autres objectifs. Exemples des communautés de partage de l'information sont *Flickr* [18], où les gens peuvent publier et partager des photos, *YouTube* [19] pour le partage vidéo, et *Wikipedia* [20], qui est une encyclopédie écrité en collaboration par les contributeurs mondials.

Nous sommes particulièrement intéressés par les communautés de partage de fichiers, qui sont créés sur des réseaux de partage de fichiers comme le réseau BitTorrent.Ils combinent des contacts sociaux avec l'échange de fichiers. Ces communautés de partage de fichiers ont un moyen efficace pour protéger l'intégrité du contenu par affichant les observations et les discussions. Les contenus faux ou poorquality sont reconnus par les membres de la communauté et éliminées ou marqués comme faux. Donc, les communautés sociales peut former un filtre coopératif [21] pour les réseaux de partage de fichiers. De plus, les relations sociales qui sont créés dans les communautés forment des incitations supplémentaires à coopérer avec le réseau de partage de fichiers. Exemples de communautés populaires BitTorrent sont *Filelist*, *Piratebay*[22], et *Mininova*[23].

Les gens ont tendance à s'aider les uns les autres plus, quand le pair est considéré comme un vraie personne avec des intérêts similaires, en remplacement des ordinateurs. La fonctionnalité de téléchargement coopérative de Tribler (voir la section 2.3.2) est un exemple.

### 2.3.2 Tribler

Tribler [24] est un système P2P de partage des fichiers, il est construit sur le protocole de BitTorrent pour le transfert général de fichiers, et particulièrement dans ABC Client [25]. Figure 3.1 [24] illustre cette architecture, les rectangles représentent ses modules et fichiers, et les extrusions représentent les relations d'utilisation.

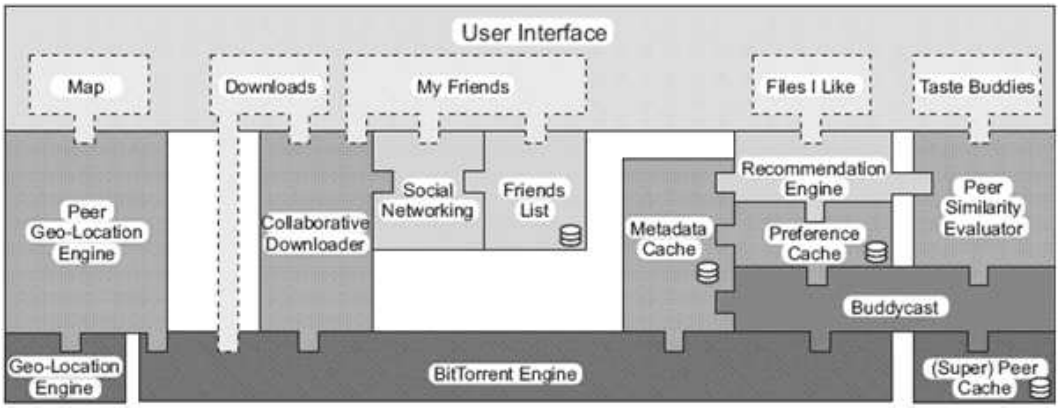


FIGURE 2.4 – L'architecture de Tribler

Tribler permet aux utilisateurs d'ajouter et enlever des amis. L'authentification améliorée basée sur l'identificateur permanent(PermID) est ajoutée à fournir les pairs identifiables. Le réseau social peut être utilisé de trouver les chemins parmi des pairs. Le réseau social est utilisé par la caractéristique extra de Tribler, comme un recommandation de contenu et l'algorithme de découverte, appelé *BuddyCast*. L'algorithme *BuddyCast* utilise un protocole épidémique pour échanger les listes de préférence avec *taste buddies*(c'est-à-dire que l'on connecte quelqu'un par le goût similaire). Tribler utilise aussi la découverte distribuée d'essaim(distributed swarm discovery) pour surmonter le problème de scalabilité avec un *tracker* central. Le téléchargement coopératif est une autre caractéristique, qui permet aux utilisateurs de demander aux amis pour l'assistance de télécharger. Des amis dénotent la bande passante disponible. Le *collaborative download protocol* appelé *2Fast* est utilisé

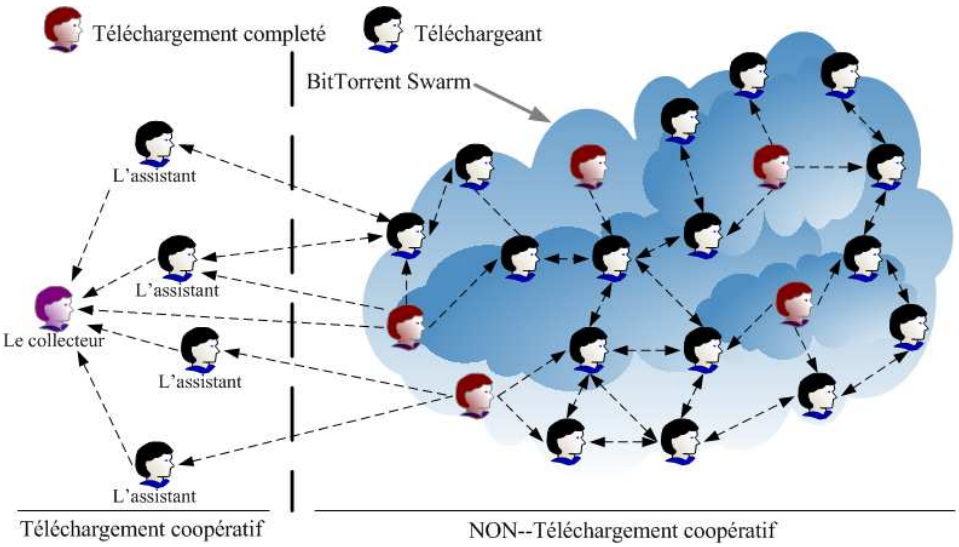


FIGURE 2.5 – La synthèse de téléchargement coopératif

dans Tribler. Le protocole *2Fast* utilise un groupe social, où les membres qui font confiance à les uns et les autres collaborent pour leur performance de télécharger. Figure 3.2 [25] montre la synthèse de téléchargement coopératif.

## 2.4 Conclusion de chapitre

Dans ce chapitre, d'abord j'ai présenté une introduction de l'état de l'art pour les mécanismes incitatifs, des avantages et des inconvénients de chaque type de mécanisme incitatif qui concerne les différents scénarios d'application P2P. Je trouve que le mécanisme de réputation constitue une solution très intéressante vers la direction correcte pour Live streaming. En raison de sa flexibilité et expansibilité, il a la valeur de recherche dans le développement de mécanismes incitatifs du réseau P2P à l'avenir.

Ensuit, c'est le présentation du fonctionnement principal de BitTorrent et l'extension-BiTos pour Streaming basé sur BitTorrent. Parce que BitTorrent est la plus réussi prédécesseur de l'application actuelle de Live streaming.

Enfin, avec les recherches, les influences des phénomènes sociaux pour le motivation de humain s'adaptent semblablement aux système P2P. Tribler ajoute quelques facteurs sociaux comme l'amitié et la confiance dans le système, les résultats montrent que ils améliorent la performance du système d'en-

courager les utilisateur à contribuer. Particulièrement, quand le téléchargement combiné est utilisé dans un BitTorrent swarm, il produit une augmentation évidente pour la vitesse.

# Chapitre 3

# Implémentation de fonctionnalité "sociale" dans ABC

L'objectif de mon stage est de mettre en œuvre des mécanismes incitatifs basé sur la relation sociale pour les réseaux P2P. Ce projet comprend trois parties :

- Nous allons ajouter des facteurs sociaux dans un client BitTorrent existant comme l'amitié et la confiance. Car la plupart des systèmes P2P sont basés sur les connexions anonymes parmi des pairs. Le problème majeur est l'absence d'incitation de participer au partage de contenu et de la bande passante. Beaucoup de personnes utilisent le système mais rien contribuent. C'est le phénomène de *free-riding*. Donc, nous pensons que les phénomènes sociaux, comme l'amitié, la confiance et un sens de la communauté, sont importants et ont une grande influence positive sur la facilité d'utilisation et les performances de P2P système de partage de fichiers . Et les utilisateurs comme les partenaires sociaux peuvent mieux atténuer le problème de *free-riding*.

- Nous allons ajouter le capacité de supporter streaming dans un client BitTorrent existant. La stratégie principale comprend deux parties. La première partie est de modifier l'algorithme de sélection de pièce pour donner une priorité plus élevée aux pièces qui sont plus proche du temps de lecture. La deuxième partie est d'ajouter un outil de Vidéo pour obtenir le capacité de lecture.

- Nous allons ajouter les deux parties présentées ci-dessus dans un client

BitTorrent en même temps.

Dans les travaux futurs, nous focaliserons comment ajouter un outil de Vidéo et comment mettre en œuvre le facteur social et de streaming en même temps.

Concrètement, afin de avoir une base du code testé pour notre implémentation et une grande base des utilisateurs dans une relativement courte période, j'essaie de mettre en œuvre des modifications et des extensions dans le open-source client BitTorrent-ABC.

Nous choisissons ABC, parce que'il est plus simple pour modifier. Et il supporte un système de files d'attente avec priorité, les préférences mondial global et local (par torrent) pour télécharger des torrents(y compris le taux limité de upload et télécharger), les options multiple de upload pour les fichiers complets, un display ajustable et un système nommé 'Upload Rate Manager(URM)' pour contraindre les torrents à déboîter la file d'attente, s'il n'y a pas un total prédéfini de l'activité de upload. Il y a aussi une Web interface extensive dans ABC, ce qui permet d'autres applications pour afficher et de changer ses torrents et ses préférences à distance. Ce sont le raison que pourquoi je choisi ABC pour mettre en œuvre l'expérience.

ABC est écrit en Python. Donc, j'utilise le Python pour modifier le code. Python est un langage de programmation interprété multi-paradigme. Il favorise la programmation impérative structurée, et orientée objet. Il est doté d'un typage dynamique fort, d'un système de gestion d'exception. Pendant la périod d'expérience, je trouve graduellement que Python est un langage de programmation très fort. Il offre un support puissant pour l'intégration avec d'autres langues et d'autres outils.

## 3.1 L'implémentation de ABC social

Selon les études antérieures pour le *Tribler*, nous nous intéressons les trois caractéristiques de *Tribler* : 1. Le réseau social ; 2.*Buddycast* ; 3. Téléchargement coopératif. Donc, afin de avoir une grande influence positive sur la facilité d'utilisation et les performances de P2P système de partage de fichiers, j'ai ajouté ces trois caractéristiques dans le client BitTorrent-ABC. Nous espérons un résuletat que nous peuvons réaliser une augmentation évidente pour la vitesse de téléchargement en employant la capacité oisive de upload des amis en ligne. Ce nouveau logiciel est applé *Social ABC*. Dans les parties suivantes,

je vais présenter les trois caractéristiques en détails et montre la réalisation du logiciel. Ensuite, ce sont des résultats de tester ce logiciel.

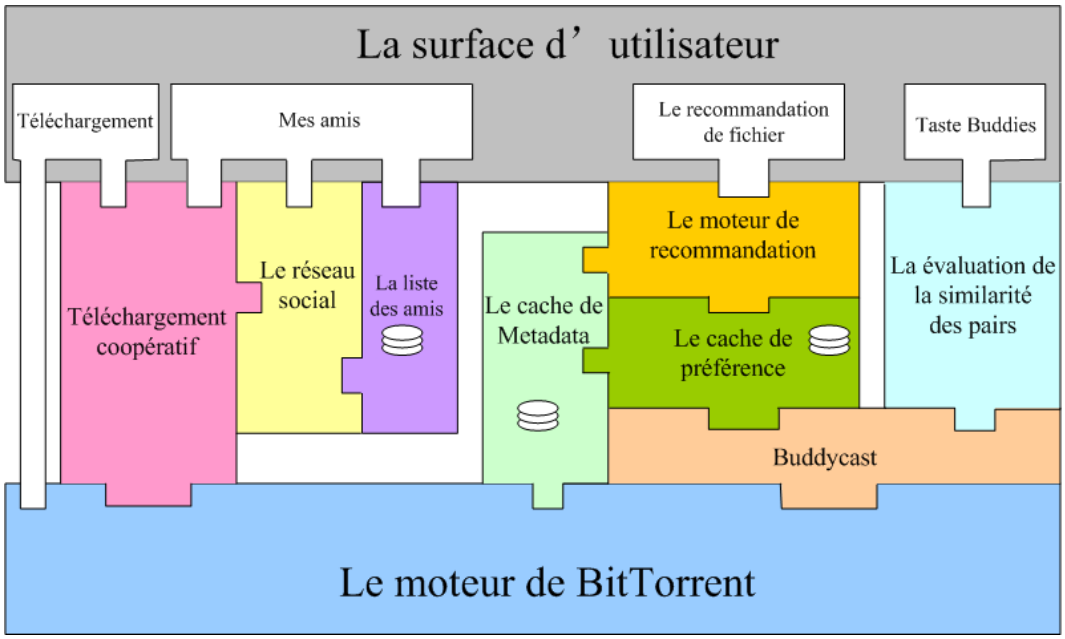


FIGURE 3.1 – L'architecture de ABC social

Figure 3.1 décrit l'architecture de ABC social. Les rectangles représentent les modules and fichiers. Et les extrusions représentent les relations d'utilisation. Maintenant, je décris les principes et fonctions de chaque module que j'ajoute.

### 3.1.1 Le réseau social

Le réseau social est responsable pour stocker et fournir les informations de groupe social. Les information comprennent les membres de group et leurs adresse de IP. Dans le réseau social que j'ajoute sur ABC, un pair doit pouvoir reconnaître ses amis, alors leur communiquer. Les utilisateurs utilisent un *PermID* pour reconnaître chaque pair et un *swarm* global qui contient tous les pairs pour permettre les communications entre eux.

Pour reconnaître les pairs, l'utilisation leurs adresses IP n'est pas suffisante, parce que ils changent par fois leurs adresses. C'est la raison pourquoi chaque pair a maintenant un PermID généré par hasard dans l'installation du nouveau logiciel. En fait le PermID est une clef publique d'une cryptographie

publique/privée. Elle est utilisée dans un protocole d'identification pour garantir l'identité des pairs. Les identificateurs de quelques pairs peuvent être alors enregistrés dans une liste des amis en fonction de leur similarité. Une quantité d'opérations privilégiées sont seulement disponibles pour les amis.

Pour permettre la communication entre les pairs, il faut introduire un *swarm* global. Dans le BitTorrent classique, les *swarms* sont gérés par le *tracker* et associés un torrent fichier unique, qui permet seulement les communications avec les pairs qui téléchargent le même fichier. Dans l'implémentation de *Social ABC*, il y a un *swarm* virtuel qui contenir tous les pairs qui utilisent ce système. Le *swarm* est appelé *Overlay swarm* qui travaille sans serveur. Il est géré par les pairs eux-mêmes. La liste entière des pairs n'est pas enregistrée dans un serveur local. Elle est distribuée à chaque pair de *swarm*. Ce *Overlay swarm* permet les pairs de communiquer avec les amis même si ils ne téléchargement pas le même fichier.

### 3.1.2 Le cache de Medadata

Dans le ABC social, les informations du contexte de toutes les pièces sont stockées localement dans le *Megacaches* pour chaque pair et ces information sont échargées à l'intérieur de groupe social en utilisant un protocole épidémie [26] (Le protocole de Buddycast, voir le section 3.1.4). Les icônes de la petite base de données dans le figure 4.1 identifient les trois *Megacaches* : "La liste des amis" avec les informations dans le réseau social, "La cache de Metadata avec les metadatas de fichier et "Le cache de préférence" avec la liste de préférence des autres pairs.

### 3.1.3 *Taste Buddies*

L'utilisation de module "Le recommandation de fichier" (voir le figure 4.1) est que chaque pair indique sa préférence pour certains fichiers. Par défaut, la liste de préférence d'un pair est remplie de son dernier téléchargement. Le algorithme de BuddyCast utilise un protocole épidémie à échanger des listes de préférence en utilisant le *Overlay Swarm* et il peut découvrir efficacement le *Taste Buddies* d'un utilisateur (voir le section 3.1.4). Le module "la évaluation de la similarité des pairs" dans le figure 4.1 peut comparer des listes de préférence et déterminer le montant de la différence pour le goût. Le module "le moteur de recommandation" peut compiler une liste des fichiers que un utilisateur veut fortement obtenir.

### 3.1.4 L'algorithme de *Buddycast*

Le algorithme de BuddyCast est une amélioration. Un des ses objectifs est de faire les *trackers* et les *search engines* externe inutile dans le protocle. Comparé avec un client normal, cette façon peut travailler même si les trackers sont cassés. De plus, il n'a pas besoins d'un *search engines* externe pour rechercher le nouveau contenu.

BuddyCast travaille en échangeant les messages entre les pairs et utilisant le *OverlaySwarm*. Les messages contiennent l'information sur les autres pairs et les torrents. L'information des autres pairs est utilisée pour trouver les nouveaux pairs avec l'intérêt similaire.

Un pair est connectable quand il peut accepter les noveaux connection. Chaque pair maintient deux listes : 1. La liste des pairs qui ont l'intérêt le plus similaire avec lui(liste A) ; 2. La liste des pairs aléatoire (liste B). Périodiquement, un pair peut connecter soit (a) un de liste A pour échanger la liste de préférence, soit (b) un nouveau pair aléatoire pour échanger les informations de leurs liste de préférence. Nous définissons un "pair connu" comme le pair dont nous connaissons la liste de préférence et un "pair aléatoire" comme le pair dont nous ne connaissons pas les informations. Une cible de Buddycast vient de deux sources : "pair connu" ou "pair aléatoire". Afin de équilibrer le choix entre les deux sources, un nombre aléatoire $r$ entre 0 et 1 est utilisé dans chaque circulation. Si $r<0.5$, un "pair connu" est sélectionné, sinon un "pair aléatoire" est sélectionné. Si la cible est un "pair connu", d'abord nous obtenons 100 pairs connus qui apparaissent fréquemment on ligne et sélectionnons un des pairs selon leurs similarités. Plus grande est la similarité, plus possible les pairs peuvent être sélectionné. Si la cible est un "pair aléatoire", d'abord nous obtenons tous les pairs aléatoires et sélectionnons un des pairs selon leurs âge. Le âge de pair est le nombre de secondes depuis sa dernière apparition on ligne. Plus petit est l'âge, plus possible les pairs peuvent être sélectionné. Le pseudo code de l'algorithme est comme le suivant (l'algorithme 1) :

**Algorithm 1** L'algorithme de Buddycast

```
loop
  if r < 0.5 then
    Sélectionner un pair connu
    Obenir 100 pairs connus qui apparaissent fréquemment on ligne
    if La similarité d'un pairs est le plus grande then
      Sélectionner ce pair
    end if
  else
    Sélectionner un pair aléatoire
    if L'âge d'un pairs est le plus petit then
      Sélectionner ce pair
    end if
  end if
end loop
```

### 3.1.5 Le téléchargement coopératif

Le module "Téléchargement coopératif" a le capacité à réaliser une augmentation évidente pour la vitesse de téléchargement en employant la capacité oisive de upload des amis en ligne. Ici, j'ai utilisé le protocole de téléchargement coopératif qui est applé *2Fast*. Ici, un utilisateur peut demande à ses assistants pour élever le téléchargement.

Le protocole de *2Fast* [27] cible les environnement où les fichiers larges sont téléchargés par beaucoup d'utilisateurs dans le même temps. Nous focalisons sur l'optimisation de la vitesse du transfert de data. Dans le section 2.2.1, j'ai déjà présenté les fonctions principales de BitTorrent. Afin de faciliter le processus d'échange, les fichiers sont divisés en parties plus petites appelées *chunks*. La taille d'un *chunk* dans un BitTorrent est à l'échelle des centaines de kilobits. En sélectionnant les *chunks* à télécharger de l'un et l'autre, les pairs suivent le politique de *rarest-first* [28] qui signifie que le *chunk* avec le plus petite quantité de réplique est premièrement sélectionné pour télécharger. Un pair a un fichier complet seulement après obtenir tous les chunks qui composent ce fichier. Un pair qui possède une copie entière d'un fichier est appelé un *seeder*, alors que un pair dont le téléchargement est encore en cours est appelé un *leecher*. Le groupe des pairs (*leecher* et *seeder*) téléchargeant un fichier particulier est appelé un *swarm*.

Dans le protocole de *2Fast*, un pair prendra un de deux rôles : un collecteur

ou un assistant (voir le figure 2.5). Le collecteur est un pair qui s'intéresse au obtenir le fichier particulier complet. Afin de améliorer la performance du téléchargement, un collecteur forme une collaboration composée par les pairs qui veulent devenir ses assistants. Les assistants téléchargent certaines *chunks* de fichier des pairs hors de la collaboration, et envoient les *chunks* au collecteur sans rien demander. Le collecteur optimise sa performance de téléchargement en sélectionnant dynamiquement la source de data le plus disponible dans les assistants et les autre pairs dans le network. Les assistants donnent la priorité à la demande de collecteur, et donc sont préférés comme les source de date. Les assistants ne sont pas nécessairement aux contenus que ils sont en train de télécharger. Le pseudo code de l'algorithme est comme le suivant (l'algorithme 2) :

**Algorithm 2** L'algorithme du téléchargement coopératif

```
Chaque fois le collecteur avoir une demande d'aide
Ouvrir une connection d'Overlay-Swarm aux les assistants
Le collecteur envoyer un message Download_Help pour demander l'aide
if L'assistant a reçu le message Download_Help then
  if L'assistant veut aider then
    L'assistant obtient un fichier de Torrent du collecteur par le message
    Get_Metadata
    L'assistant envoie un message Reserve_Pieces pour demander des
    pièces
    Le collecteur répond un message Pieces_Reserved
  else
    L'assistant ne veut pas aider
  end if
end if
Commencer un téléchargement coopératif
```

La justice du partage de la bande passante est renforcée par l'approche hybride. La bande passante est rendu compte différemment entre les pairs non collaborant, et entre un collecteur et ses assistants. Les pairs qui ne sont pas les membres de la même collaboration peuvent échanger les *chunks* avec le *tit-for-tat* algorithme du standard de BitTorrent. La stratégie d'échange de BitTorrent *tit-for-tat* assure que la quantité de revenue de data pour un pair est approximativement égal à la quantité de dépense.

Le *tit-for-tat* mécanisme garantit seulement la justice pour une session simple de téléchargement (le téléchargement d'un fichier). La nature asymétrique de la relation entre le collecteur et l'assistant implique que la bande

passante contribué par l'assistant afin d'obtenir les *chunks* pour le collecteur dans le téléchargement actuel peut seulement être réclamé pendant les téléchargement plus tard. Nous établissons la justice entre les pairs de collecteurs sur une notion d'une promesse. Un collecteur recrute les assistants en leur donnant une promesse que le largeur de la bande passante investi à la performance du téléchargement de collecteur sera retourné. Les assistants qui contribuent leurs bandes passantes actuel savent que les rôles pourront être renversés–un assistant deviendra un collecteur et le collecteur actuel remboursera pour la bande passante consumé en participant le téléchargement comme un assistant.

Le rôle d'une promesse est de fournir un mécanisme incitatif pour la collaboration. Cette rôle peut seulement être accompli quand le système peut garantir que la promesse sera délivrée. Le *2Fast* système sous son forme basique utilise le mécanisme incitatif sociale pour renforcer la livraison de promesse. Le phénomène social, tel que l'amitié et l'existence des communautés des utilisateurs qui se font confiance, est utilisé dans le P2P protocole, comme j'ai déjà présenté dans le *Social Networking*. L'observation que les membres de communautés sociales tendent à avoir une grande quantité des amies dans la combinaison avec les assistants peu nombreux. Ils sont demandé à avoir bonne performance de téléchargement donne une raison pour croire que un collecteur ne doit pas avoir un problème pour trouver un assistant. Dans le système décrit dans [10], les utilisateurs sélectionnent les pairs que ils veulent aider en spécifiant explicitement combien de confiance ils ont. Le valeur de la confiance peut être modifié tous le temps pour refléter le changement de la confiance des utilisateurs en crédibilité de l'un et l'autre.

Le protocole de *2Fast* permet un nouveau, peu restrictif modèle de partage de ressource dans le P2P network de distribution de donnée. Les systèmes actuels de P2P distribution basent leurs mécanismes incitatifs sur l'échange de contenu, mais *2Fast* introduit le mécanisme incitatif basé sur l'échange de la bande passante.

Tous les mécanismes incitatifs basés sur l'échange de contenu sont pour obtenir le droit de télécharger un fichier du network, en retour, pour upload un fichier stocké localement dans le network. Dans ce modèle, les fichiers ou les contenus générales sont échangés entre les pairs pour renforcer la justice de partage. Pour permettre l'échange de contenu entre deux pairs, il demande que les deux pairs concernés ont des contenus qui sont intéressants pour l'autre. En conséquence, l'échange de contenu peut avoir lieu seulement entre les pairs qui ont l'intérêt mutuel de contenu. Le protocole de BitTorrent est un exemple d'une variante le plus restrictive de l'échange de contenu, parce que les pairs

de BitTorrent échangent les *chunks* de la même fichier. Le système tel que *Scrivener* [29] adopte un mécanisme d'échange plus avancé appelée *transitive trade* , qui établit un *credit path* [29, 30] de nœud demandant à nœud qui a le fichier désiré. Les crédits sont alors transférés par ce chemin et le téléchargement peut commencer. La découverte de credit path est, pourtant, un problème complexe, et il y a toujours une probabilité qu'il n'y a pas de path entre deux pairs particuliers.

Dans le modèle incitatif de l'échange de la bande passante, tel que *2Fast*, c'est la bande passante plutôt que le contenu qui est échangé. Tous les pairs peuvent agir comme un assistant à l'autre pair en investissant la bande passante sous-utilisé et pourront obtenir cette bande passante retourné. Donc il n'y a pas d'intérêt mutuel demandé entre les assistants et les collecteurs.

## 3.2 ABC pour streaming

Selon les études antérieures pour l'extension de BitTorrent pour *sreaming-BiTos*, je m'intéresse le modification pour l'algorithme de sélection de *chunks*. Donc, j'ajoute cet algorithme dans un client BitTorrent-ABC. Ce nouveau logiciel est applé *Streaming ABC*. Essentiellement, tous les pièces d'un fichier sont classifiés en trois groupes : 1.*Received Pieces* ; 2*High Priority Set* ; 3.*Remaining Pieces Set*.

- **Received Pieces** : Contenir tous les pièces téléchargés d'un flux de Vidéo, que les pairs ont déjà téléchargé. L'état d'une pièce peut être "Téléchargé", "Non-téléchargé" ou "Perdu". Si une pièce ne peut pas finir le téléchargement avant le temps de lecture, son état est "Predu".

- **High Priority Set** : Contenir les pièces d'un flux de Vidéo qui n'ont pas encore été téléchargés, mais ses état n'est pas "Predu" et ils sont plus proche du temps de lecture. Ainsi, ces pièces ont une priorité plus élevée à demander que le reste des pièces. Ce groupe a une taille fixe de pièces, et cette taille est un paramètre du système. Une pièce de ce groupe peut être dans les deux états suivants : "Non-demandé" ou "Téléchargeant-actuellement".

- **Remaining Pieces Set** : Contenir les pièces qui n'ont pas été téléchargés, ses état n'est pas "Predu" et ils ne sont pas dans le groupe de *High Priority Set*. Une pièce de ce groupe peut être aussi dans les deux états suivants : "Non-demandé" ou "Téléchargeant-actuellement".

Dans ce processus de sélection, le pair choisit une pièce du *High Priority Set* à la télécharger avec une probabilité $\boldsymbol{p}$ ($\boldsymbol{p}$=0.8 par défaut) et une pièce du *Remaining Pieces Set* avec une probabilité $\boldsymbol{1\text{-}p}$. La probabilité $\boldsymbol{p}$ représente l'équilibre entre le besoin immédiat d'une pièce et l'acquisition d'une pièce dans le futur.

A cause de la limite du temps, je n'ajoute pas encore un outil de Vidéo pour obtenir le capacité de lecture. Dans les travaux futurs, nous allons réaliser cette fonction et le tester dans un environnement réel pour obtenir et connaître les résultats que nous voulons.

## 3.3 Conclusion de chapitre

Dans ce chapitre, j'ai déjà introduit l'objectif et la condition simple de mon stage. Ensuitem, je présente les fonctions sociales que j'ajoute dans le ABC original. Ce nouveau logiciel-*Social ABC* peut réaliser les fonctions d'ajouter les amis et du téléchargement coopératif. Ensuite, je montre l'implémentation du nouveau logiciel-ABC social et décris les algorithmes concernés. Enfin, je vais introduire l'algorithme de sélection de pièce pour donner une priorité plus élevée aux pièces qui sont plus proche du temps de lecture.

# Chapitre 4

# Expérience et les résultats

Dans ce chapitre, je vais présenter l'expérience pour tester le performance de ABC social en détails et les résuletats pour comparer les performances de ABC original avec *ABC social* dans la section 4.1.

## 4.1 Le test pour *Social ABC*

Dans cette parie, je vais montrer les résultats de tester *Social ABC*. Nous utilisons quatre ordinateurs personnels (PCs) avec 3.2GHz CPU et 1GB de RAM pour tester. Ici, les PCs seront applés nœuds. Le système de manipulation (OS) est *Windows XP*, parce que tous les applications ont déjà été implémentés pour ce OS. Les quatre nœuds sont situés dans notre réseau de campus. Et ils connectent directement avec l'Internet et avec le 100 Mbps accès de l'Ethernet. Le figure 4.1 montre l'architecture d'expérience.

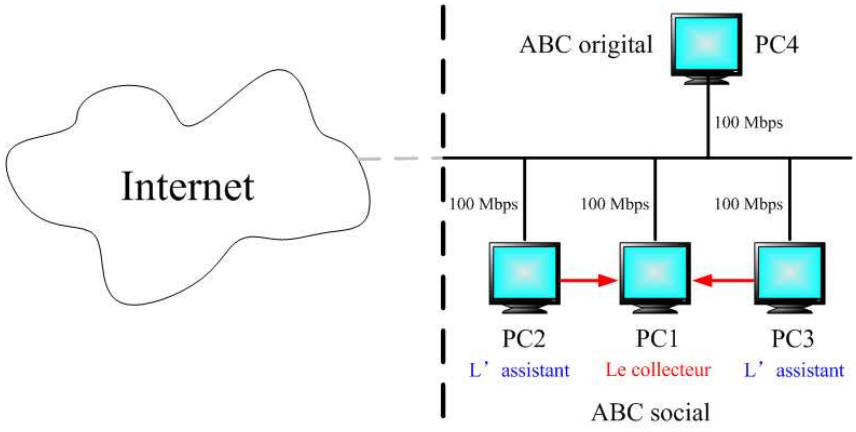


FIGURE 4.1 – La comparaison pour le débit de téléchargement

Entre les quatre PCs, il y a trois PCs (PC1, PC2 et PC3)qui utilisent notre implémentation du logiciel *Social ABC*, un PC (PC4) restant utilise le logiciel ABC original. PC1 et PC4 veulent télécharger un même fichier, et PC2 et PC3 vont être l'assistant de PC1 pour aider PC1 à télécharger le fichier. Mais pendant le processus d'expérience, PC3 échoue de connecter à l'Internet. En fait, il y a seulement trois PCs peuvent bien marcher. PC2 n'est que l'assistant de PC1 et il ne télécharge que des certains pièces d'un fichier que PC1 veut obtenir. Donc, nous comparons seulement la performance de PC1 et PC4. Nous choisissons un fichier ".torrent" de vidéo plus populaire du site "Pirate". Chaque fichier ".torrent" a l'âge. L'âge de fichier ".torrent" est le nombre de jours depuis sa première apparition on ligne. Normalement, l'âge est plus grande, plus petit est le nombre des pairs qui le téléchargement. C'est-à-dire que l'activité de fichier ".torrent" va diminuer. Alors nous ne peuvons pas trouver beaucoup de ressources sur l'Internet. Celà va influencer la performance de notre expérience. Donc, nous devons choisir le fichier ".torrent" plus populaire pour bien effectuer notre expérience. Nous voulons voir les performances du téléchargement des deux logiciel et comparer les deux pour connaître l'impact de fonctionnalité "social". Nous espèrons que la vitesse de téléchargement a une augmentation évidente en utilisant le téléchargement coopératif. Nous utilisons les configurations suivantes dans notre expérience :

1.La taille de Vidéo = 702.16 MB ;
2.La taille de *Chunk* = 1024 KB

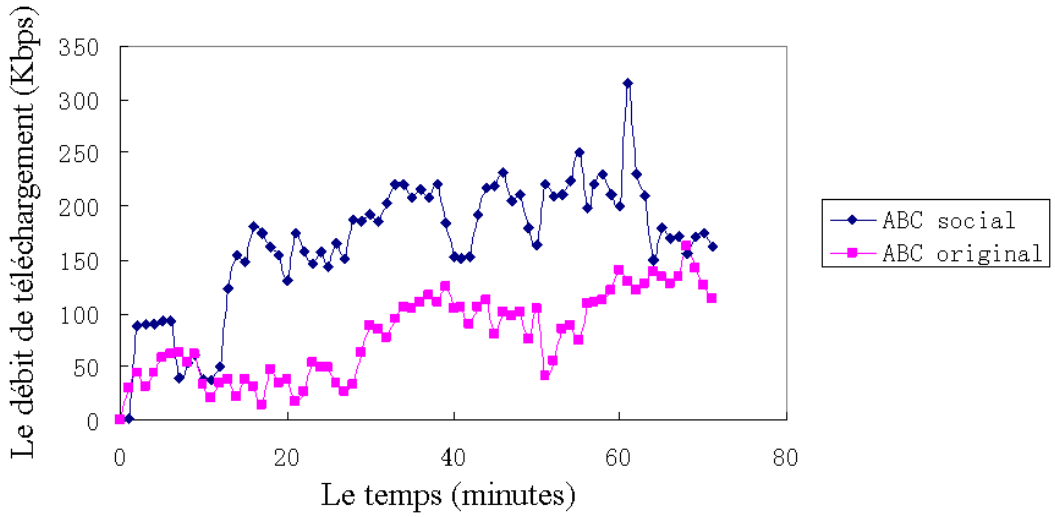


FIGURE 4.2 – La comparaison pour le débit de téléchargement

Le figure 4.2 montre que le débit de téléchargement de ABC social (PC1) a un augmentation évidente comparé avec ABC original (PC4). Nous pouvons voir que au début PC1 et PC4 utilisent le même protocole BitTorrent pour téléchargement. La différence du débit de téléchargement entre les deux n'est pas très évidente. Parce que PC1 n'ai pas demandé aux assistants. Après 16 minutes, le débit de téléchargement de PC1 a un augmentation évidente. La raison est que PC2 commence à aider PC1 à télécharger les pièces. La différence du débit de téléchargement devient le plus grande sur le point de minute 60. Le débit de ABC social est 229.9Kbps et le débit de ABC original est 112Kbps. Il illustre que ABC social peut utiliser plus efficacement la bande passante que ABC original en durant le même temps. La raison principale est que PC1 élève le capacité de téléchargement sous l'aide de PC2. Donc cette expérience confirme que nous pouvons améliorer le performance de téléchargement en ajoutant les éléments sociaux, et nous pouvons utiliser efficacement la bande passante dans certain temps. Et cette façon peut promouvoir les aides entre les nœuds. Ils téléchargent les fichiers en fournissant les upload à le réseau, celà peut accroître la vitalité du réseau entier et aussi peut abaisser efficacement le problème de *free-riding*.

| | ABC social | ABC original |
|---|---|---|
| Le temps total du téléchargement | 4260 s | 7020 s |
| Le débit moyen du téléchargement | 168.76 Kbps | 102.42 Kbps |
| La taille totale de upload | 1.49 GB | 1.02 GB |
| Le nombre de seeder connecté | 108 | 81 |
| Le nombre de leecher connecté | 31 | 21 |

FIGURE 4.3 – La comparaison pour les autres paramètres

D'après le figure 4.3, tous les paramètres montre que le performance de ABC social est meilleur que le ABC original. Le débit moyen du téléchargement de ABC social est 66.34Kbps plus vite que ABC original. Et ABC social peut trouver plus de seeder et leecher pour télécharhement. La raison peut être que PC1 connecte les autres pairs sous l'aide de PC2. De plus, le téléchargement coopératif peut fournir plus de ressources de upload à le système P2P.

Même si je n'ai pas appliqué les éléments sociaux dans P2P streaming pendant mon stage, nous croyons que ils apporteront des influences positives dans P2P streaming d'après leurs influences que nous avons vu pour le système P2P classique. Parce que P2P streaming demande une contrainte du temps plus élevé que le système P2P classique, il a fortement besion de les mécanismes incitatifs.

## 4.2 Conclusion de chapitre

Dans ce chapitre, je présente l'implémentation d'expérience pour tester le *ABC Social*, nous pouvez voir la performance améliorée pour téléchargement et upload. Ces résultats expliquent que les facteurs sociaux peuvent organiser efficacement les pairs à faire le téléchargement coopératif dans le P2P système. Je n'ajoute pas encore les facteurs sociaux sur le P2P streaming système, mais je pense que les impacts des facteurs sont évidents dans le système de BitTorrent classique, et ils sont sont aussi applicables pour le système de BitTorrent streaming. Nous pourrions les confirmer dans les travaux futurs.

# Chapitre 5

# Conclusion et les travaux futurs

Dans ce chapitre, je vais montrer une conclusion de cette rapport. Ensuite, je vais proposer les recherches futures.

## 5.1 Le sommaire et les conclusions

D'après les recherches pour les mécanismes incitatifs existants pour les systèmes de P2P et le P2P système de distribuer des contenus, particulièrement, pour l'extention-*BiTos* pour *Streaming* basé sur BitTorrent et Tribler. Et nous pensons que les influences des phénomènes sociaux pour le motivation de humain peut être appliqué dans le P2P système pour encourager les nœuds pour contribuer aux réseaux de P2P. Car la plupart des P2P systèmes sont basé sur les connexions anonymes parmi des pairs. Le problème majeur est l'absence d'incitation de participer au partage de contenu et de la bande passante. Beaucoup de personnes utilisent le système mais rien contribuent. Donc, nous pensons que les phénomènes sociaux, comme l'amitié, la confiance et un sens de la communauté, sont importants et ont une grande influence positive sur la facilité d'utilisation et les performances de P2P système de partage de fichiers . Et les utilisateurs comme les partenaires sociaux peuvent mieux atténuer le problème de *free-riding*.

Selon les recherches sur les phénomènes sociaux, j'ai ajouté les modules sociaux,comme "Le réseau social", "*Buddycast*" et "Téléchargement coopératif" dans le client BitTorrent-ABC. Nous espérons que les performances du P2P système ont l'amélioration évidente en ajoutant ces modules sociaux. Dans le nouveau logiciel ABC social, les utilisateurs peuvent ajouter et enlever des amis et ils font confiance à l'un et les autres et commencer le téléchargement coopératif. D'après le test pour ABC social, les résultats montre que il a le capacité à réaliser une augmentation évidente pour la vitesse de téléchargement

en employant la capacité oisive de upload des amis en ligne. Il illustre que ABC social peut utiliser plus efficacement la bande passante que ABC original en durant le même temps. Et son performance est meilleur que le ABC original selon tous les paramètres de test. Il confirme notre hypothèse pour améliorer le performance de système en ajoutant les éléments sociaux.

## 5.2 Les travaux futures

Nous espèrons appliquer les influences sociales dans P2P streaming. Parce que P2P streaming demande une contrainte du temps plus élevé que le système P2P classique, il a fortement besion de les mécanismes incitatifs pour élever la performance de téléchargement. Maintenant, de plus en plus de personnes utilisent le système P2P pour voir le film sur ligne. C'est une tendance d'application pour le système P2P futur. Dans les travaux futurs, nous voulons choisir un client BitTorrent plus simple et le modifier en P2P streaming. Et nous testons ce P2P streaming basé sur BitTorrent pour voir que si sa performance du lecture est plus meilleur que le P2P streaming basé sur l'architecture traditionnelle-client-serveur. Ensuite nous ajouterons les fonctionnalité "social" dans le P2P streaming pour voir les impacts.

# Bibliographie


[1] "Source : Cachelogic," 2006.

[2] "Napster," *http ://www.Napster.com.*

[3] "Pplive," *http ://www.pplive.com.*

[4] "Coolstreaming," *http ://www.coolstreaming.us.*

[5] M.Bawa, H.Deshpande, and H.Garcia-Molina, "Transience of peers and streaming media," *in Proc. of HotNets 1*, 2002.

[6] X.Zhang, J.Liu, and T.P.Yum, "Coolstreaming/donet : A datadriven overlay network for peer-to-peer live media streaming," *in Proc. of IEEE INFOCOM*, 2005.

[7] T. Silverston and O. Fourmaux, "Measuring p2p iptv systems," *ACM NOSSDAV*, 2007.

[8] S.Wasserman and K.Faust, "Social network analysis," *Cambridge university Press*, 1994.

[9] M. Feldman, K. Lai, I. Stoica, and J. Chuang, "Robust incentive techniques for peer-to-peer networks," *Association for Computing Machinery*, vol. Session 4, pp. 102 – 111, 2004.

[10] P. Golle and K. Leyton-Brown, "Incentives for sharing in peer-to-peer networks," *Lecture Notes in Computer Science*, vol. 2232, 2001.

[11] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, "Clustering and sharing incentives in bittorrent systems," *SIGMETRICS '07 Conference Proceedings*, vol. Session : Networking 2, no. 2, pp. 301 – 312, 2007.

[12] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "Bitos : Enhancing bittorrent for supporting streaming applications," *IEEE Global Internet*, 2006.

[13] M. Abraham and H. Harper, "Motivation and personality," *Collins Publishers*, 1987.

[14] K. Ling, B. Gerard, and L. Pamela, "Using social psychology to motivate contribution to online communities," *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, vol. Communities, pp. 212 – 221, 2004.

[15] "Orkut, online social community," *http ://www.orkut.com.*

[16] "Friendster, social community," *http ://www.friendster.com.*

[17] "Hyves, social community," *http ://www.hyves.nl.*

[18] "Flickr, photo sharing," *http ://www.flick.com.*

[19] "Youtube, online video streaming and sharing service," *http ://www.st.ewi.tudelft.nl/koala.*

[20] "Wikipedia, online collaborative encyclopedia," *http ://www.wikipedia.org.*

[21] J.Wang, J. Pouwelse, R. Lagendijk, and M. Reinders, "Distributed collaborative filtering for peer-to-peer file sharing systems," *In Proc. of the 21st Annual ACM Symposium on Applied Computing*, 2006.

[22] "The piratebay, bittorrent community and tracker," *http ://www.thepiratebay.org.*

[23] "Mininova, bittorrent community and search engine," *http ://www.mininova.org.*

[24] J.A.Pouwelse, P.Garbaki, J.Wang, A.Bakker, J.Yang, A.Iosup, D.H.J.Epema, M.Reinders, M. Steen, and H.J.Sips, "Tribler : a social-based peer-to-peer system," *Concurrency and Computation :PRACTICE AND EXPERIENCE*, vol. Concurrency Computat : Pract, no. 20, pp. 127–138, 2008.

[25] "Abc [yet another bittorrent client]," *http ://www.pingpong-abc.sourceforge.net.*

[26] V. S. M. Jelasity M, "Large-scale newcast computing on the internet," *Technical Report*, vol. IR-503, 2002.

[27] P. Garbacki, A. Iosup, D. Epema, and M. van Steen, "2fast : Collaboration downloads in p2p network," *Peer-to-Peer Computing, Sixth IEEE International Conference*, pp. 23–30, 2006.

[28] B. Cohen, "Incentives build robustness in bittorrent," tech. rep., P2PEcon'03, Berkeley, CA, May 2003.

[29] A.Nandi, T.-W.Ngan, A.Singh, P.Druschel, , and D.S.Wallach, "Scrivener : Providing incentives in cooperative content distribution systems," *Middleware 2005*, November 2005.

[30] K.G.Anagnostakis and M.B.Greenwald, "Exchange-based incentive mechanisms for peer-to-peer file sharing," *ICDCS'04*, March 2004.