République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane MIRA – Bejaïa
Faculté des sciences Exactes
Département d'Informatique

Mémoire de fin d'étude

En vue de l'obtention du diplôme de Master Professionnel en Informatique

Option : Administration et Sécurité des Réseaux

Thème

Etude comparative des systèmes de media streaming dans les réseaux sociaux pair à pair (P2P)

Encadré par :

M^me^ MAMMERI Karima

Réalisépar :

M^r^. AYAD Farid

M^r^. CHERIET Tahar

Devant le jury composé de :

Présidente : M^me^ LARBI Wahiba	MCB	Université A.Mira de Bejaia
Examinatrice : Mme GHERBI Meriem	MAA	Université A.Mira de Bejaia

Année 2019-2020

Remerciements

Nous tenons tout d'abord à remercier ALLAH le tout puissant, qui nous a donné l'aide, la force, la volonté dans les moments difficiles et la patience d'accomplir ce modeste travail.

Nous souhaitons ensuite exprimer notre gratitude et notre reconnaissance à notre promotrice madame MAMMERI Karima pour sa patience, sa modestie, le temps qu'elle nous a consacré, la qualité de son encadrement, ses encouragements et pour ses précieux conseils et remarques.

Nous exprimons, aussi, nos vifs remerciements aux membres de jury de notre soutenance d'avoir pris le temps de lire et juger ce mémoire et le travail qu'il représente.

Une pensée particulière est adressée à tous nos collègues et amies de l'université de A. Mira-Bejaia, les moments passés ensemble ont été très agréables.

Enfin, que tous ceux qui ont contribué de près ou de loin, par leurs encouragements et conseils à l'accomplissement de ce travail, trouvent ici l'expression de notre profonde reconnaissance.

Dédicace

Je dédie ce modeste travail Aux êtres les plus chers au monde, mon père et ma mère qui ont sacrifié leurs vies pour ma réussite.

A mes chers frères et sœurs.

A toute ma famille.

A tous mes amis (es).

Farid

Dédicace

Je dédie ce modeste travail à ma princesse mère, à mon cher père, Aucun hommage ne pourrait être à la hauteur de l'amour Dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie. À mes adorables frères et sœurs. À toute ma famille. à toutes mes amies de l'université de Bejaia. A tous qui m'ont connu de près ou de loin.

Tahar

Résumé

Le but de ce travail est d'établir une étude comparative des systèmes de media streaming dans les réseaux sociaux en P2P. Nous nous sommes intéressés au système NetTube qui est un système de partage de vidéo dans les réseaux sociaux utilisant l'architecture P2P , nous avons réalisé la simulation de ce système afin de démontrer l'existence d'un problème qui peut faire face au bon fonctionnement de celui-ci qui est représenté par « la surcharge des nœuds fournisseurs potentiels». Puis, nous avons proposé une approche qui consiste à réduire la surcharge des nœuds fournisseur potentiels ainsi l'amélioration de l'architecture NetTube.

Mots clés : Réseaux P2P, Réseaux Sociaux, Media streaming, NetTube, Netbeans 8.

Abstract

The aim of this work is to establish a comparative study of media streaming systems in P2P social networks. We were interested in the NetTube system which is a video sharing system in social networks using the P2P architecture, we performed the simulation of this system in order to demonstrate the existence of a problem that can face the proper functioning of this one, which is represented by "the overload of potential supplier nodes". Then, we proposed an approach which consists in reducing the overhead of potential provider nodes thus improving the NetTube architecture.

Keywords :P2P Networks, Social Networks, Streaming Media, NetTube, Netbeans 8.

Table des matières

Chapitre 2 : Media streaming dans les réseaux pair-à-pair

Chapitre 3 : Systèmes de media streaming dans les réseaux sociaux en P2P

Chapitre 4 : Amélioration de l'architecture de NetTube

Liste des figures

Liste des tableaux

Liste des abréviations

ALM: Application Level Multicast

CPU: Central Processing Unit

CAN: Centent Adressable Network

DHT: Distributed Hash Table

F2F: Friend to Friend

ICQ: Internet Chat Query

IDE : Integrated Development Environment

IP: Internet Protocol

IPTV: Internet Protocol Television

PC: Personal computer

PDA: Personal Degital Assistant

P4L: P2P Four Layers

P2P: Peer To Peer

P2PTV: Peer to Peer Television

QoS: Quality of Service

RPC: Remote Procedure Call

SHA: Secure Hash Algorithm

TTL: Time to Live

TCP: Transmission Control Protocol

VoD: Video on Demand

VCR: Vidéo Cassette Recorder

Introduction générale

Le pair à pair (*peer to peer*) ou P2P est une nouvelle technologie réseau qui est utilisé avec succès dans plusieurs domaines d'application comme : le partage de fichiers, l'échange de messages instantanés et le partage de capacité de calcul. Il est devenu très populaire ces dernières années vu ses bonnes caractéristiques offertes, comme : le passage à l'échelle, l'autonomie des nœuds et le contrôle décentralisé et grâce à ces caractéristiques avantageuses, de nouveaux domaines se dirigent vers l'utilisation de ces réseaux dans de nouvelles applications.

Le media streaming est l'une des applications de plus en plus populaire dans les réseaux sociaux en ligne comme (Facebook, Youtube…). Cependant, l'architecture client/serveur déployée par ces réseaux sociaux présente des problèmes, telles que l'espace de stockage limité, des frais pour l'utilisation du réseau social ou pour profiter de quelques services des réseaux sociaux, etc. Pour surmonter ces problèmes, L'adoption des architectures P2P est une meilleure alternative.

L'objectif de notre travail consiste à étudier les systèmes de media streaming dans les réseaux sociaux utilisant les architectures P2P afin d'analyser leurs performances. Nous allons particulièrement nous intéresser au système NetTube afin de comprendre en détail son fonctionnement, puis proposer une amélioration à propos du problème de la surcharge des nœuds fournisseurs potentiels que nous avons détecté.

Notre mémoire est structuré de la manière suivante :

Le premier chapitre sera consacré à une étude générale sur les réseaux pair-a-pair, leurs architectures, leurs caractéristiques, ainsi que leurs domaines d'application et les réseaux P2P les plus populaires.

Dans **le deuxième chapitre**, nous étudierons le media streaming P2P, les différents champs d'application de media streaming et les topologies de media streaming P2P avec des exemples pour chaque topologie. Ainsi que les techniques de transfert des blocs et les stratégies de sélection des blocs et des nœuds.

Le troisième chapitre sera consacré à l'étude de quelques systèmes de media streaming dans les réseaux sociaux utilisant les architectures P2P, avec une description détaillée de leurs architectures et leurs algorithmes de construction.

Dans **le quatrième chapitre**, nous étudierons l'un des problèmes auxquels le système NetTube pourrait faire face et qui est représenté par la charge sur les nœuds fournisseurs potentiels. Alors, nous allons réaliser des simulations et nous allons montrer l'existence de ce problème dans l'architecture du système NetTube. Puis, nous allons décrire notre proposition d'amélioration de NetTube.

Une conclusion générale viendra clore de ce travail résumant les grands points qui ont été abordés dans ce mémoire.

CHAPITRE 1

Généralités sur les réseaux pair à pair (P2P)

1.1 Introduction

Ces dernières années, les réseaux pair à pair ou P2P sont devenus de plus en plus populaires, cette popularité est obtenue grâce aux caractéristiques avantageuses offertes pour leurs usagers tel que l'échange de l'information et le partage des ressources.

Dans ce chapitre, nous allons tout d'abord définir les réseaux pair à pair, décrire leurs caractéristiques, ainsi que leurs domaines d'application. Ensuite, nous présenterons les différentes architectures qu'on peut rencontrer et les reseaux les plus populaires.

1.2 Définition

Plusieurs définitions du pair-à-pair ont été proposées dans la littérature (voir figure 1.1), nous citons ici deux parmi celles définies par la communauté P2P qui couvrent les concepts de partage de ressources, d'auto-organisation, de décentralisation, de topologies et de réseaux logiques.

Selon S. Androutsellis-Theotokis and D. Spinellis. [1]: « Les systèmes pair-à-pair sont des systèmes distribués composés de nœuds interconnectés capables de s'autoorganiser en topologies réseau afin de partager des ressources telles que des contenus, des cycles CPU, de l'espace de stockage et de la bande passante. Ils sont capables de s'adapter aux fautes et de tolérer les groupes de nœuds ayant un comportement instable tout en conservant un niveau de connectivité et de performances acceptable sans recourir à un serveur global centralisé ni d'une quelconque autorité. » ;

Selon Buford*et al* [10] : « Un réseau logique *(overlay network)* est une couche réseau applicative virtuelle dans laquelle les nœuds sont adressables, qui assure la connectivité et l'acheminement des messages entre les nœuds. Les réseaux overlay sont fréquemment utilisés comme support au déploiement de nouveaux services réseaux ou pour fournir une structure de routage qui n'est pas accessible à partir du réseau physique sous-jacent. Beaucoup de systèmes pair-à-pair sont des réseaux overlay qui fonctionnent au-dessus d'internet. ».

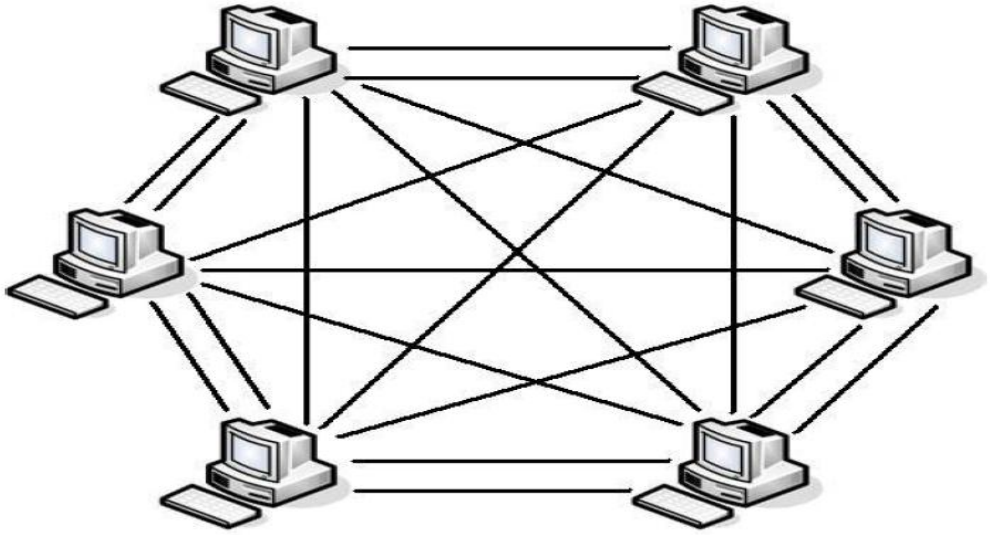

Figure 1.1 : Architecture d'un réseau P2P

1.3 Caractéristiques des réseaux Peer to Peer

Les réseaux pair-à-pair présentent généralement les caractéristiques suivantes [2]:

1.3.1 Passage à l'échelle

C'est le fait d'augmenter le nombre d'utilisateurs pour partager leurs ressources afin d'avoir une bonne performance du système. Cela permet d'obtenir un environnement dans lequel un grand volume de données est partagé entre un grand nombre de nœuds via un réseau largement distribué.

1.3.2 Décentralisation

Le fait que chaque nœud gère ses propres ressources permet d'éviter la centralisation de contrôle. Un système P2P peut fonctionner sans avoir aucun besoin d'une administration centralisée ce qui permet d'éviter les goulets d'étranglements et d'augmenter la résistance du système face aux pannes et aux défaillances.

1.3.3 Auto-organisation

La participation d'un nouveau nœud à un système P2P ne nécessite pas une infrastructure couteuse. Pour se connecter au système, il suffit d'avoir un point d'accès à l'Internet et de connaitre un autre nœud déjà connecté.

1.3.4 Hétérogénéité

Dans les systèmes P2P, les architectures matérielles et logicielles sont hétérogènes à cause de l'autonomie des nœuds. Ces systèmes doivent posséder des techniques pour résoudre les problèmes liés à l'hétérogénéité de ressources.

1.3.5 Dynamisme

Dans les systèmes Peer to Peer, le pair peut quitter le système en n'importe quel moment et de nouveaux pairs peuvent être ajoutés. Cela crée des problèmes de l'instabilité du système. Le système devrait résister à ce phénomène appelé "churn" et faire les mises à jour appropriées afin de s'auto-organiser et garantir la disponibilité des ressources dans un système distribué pouvant être face à des pannes et des départs de nœuds à tout moment.

1.3.6 Autonomie des nœuds

Les nœuds décident quelle partie de leurs données à partager. Ils peuvent aussi se connecter et se déconnecter à n'importe quel moment. Ils possèdent également l'autonomie de gérer leurs puissance de calcul et leur capacité de stockage. Donc, chaque nœud gère ses ressources d'une façon autonome.

1.4 Domaines d'application des réseaux P2P

Les réseaux pairs à pairs sont utilisés dans de nombreux domaines applications, parmi lesquels nous citons [3] [4]:

1.4.1 Le partage et la distribution de contenu

1.4.1.1 Partage de fichiers : Ces applications permettent le partage des fichiers de types différents (*texte, image, ...*) entre plusieurs ordinateurs connectés. Les protocoles de partage de fichiers sont nombreux, Parmi les plus connues, on trouve Napster, Gnutella, Kazaa, E-Mule, Bit-Torrent.

1.4.1.2 Streaming P2P : Sont des systèmes qui permettent la distribution des flux (stream) de données multimédias (audio et vidéo) en temps réel. Cette technologie consiste en un ensemble de nœuds qui coopèrent pour s'échanger du contenu vidéo entre eux. Un grand nombre d'applications de streaming P2P sont actuellement utilisées sur Internet, comme les services de télévision IP(*P2PTV*), tels que PPStream et TvAnts.

1.4.2 Calcul distribué

Le calcul distribué est probablement le domaine qui peut tirer les grands bénéfices d'une architecture P2P. Il Consiste à utiliser les machines connectées à l'internet pour faire des petites portions d'un grand calcul, en exploitant les ressources (*CPU, mémoire*….) inutilisées des PC en réseau en vue d'accroitre le potentiel réseau.

1.4.3 Programmes de messagerie

Les services de messageries électroniques basées sur le principe P2P permettent aux utilisateurs d'échanger des emails, messages… d'une façon sécurisée sans passer par le serveur central pour stocker les messages d'une façon temporelle, ce qui assure la confidentialité des correspondants, ils utilisent des systèmes d'authentification et de cryptage pour assurer la protection des contenus.

Parmi les programmes de messagerie, on cite la messagerie instantanée qui utilise le serveur juste pour la résolution d'adresses comme ICQ, AIM.

1.4.4 Communication et collaboration

La communication dans les systèmes p2p peut être audio (Skype) ou par simple message texte (chat et plateforme Jabber) ou permettre aux utilisateurs de jouer ensemble (jeux multi joueurs) ou même de travailler ensemble (plateforme de travail collaboratif GROOVE).

1.5 Comparaison entre le modèle P2P et le modèle client/serveur

L'architecture client/serveur est basé sur une seule entité centrale très puissante (serveur) et plusieurs entités généralement de puissances inférieures (les clients), le serveur est le seul fournisseur de service au clients , les clients consomment les services executés par le serveur, sans partager aucune de ses ressources.

Dans un modèle P2P, les pairs peuvent jouer en même temps le rôle de serveur et le rôle du client et sont à égalité de devoirs et de droits.

Les performances du modèle client/serveur se dégradent au fur et à mesure que le nombre d'utilisateurs augment, par contre dans le réseau P2P, la qualité et la quantité des données disponibles augmentent à mesure que le nombre d'utilisateur augmente, les performances du réseau augmentent donc avec sa popularité.

Dans le modèle client/serveur, des solutions pour les problèmes relatifs au passage à large échelle sont bien connus, c'est l'utilisation des machines plus puissantes (serveurs) mais ceci

a un coût considérable [8]. Par contre, les systèmes Peer to Peer décentralisés nécessitent toujours des solutions algorithmiques pour résoudre les problèmes de passage à l'échelle puisqu'il n'y a rien de centralisé pour ajouter plus de ressources informatiques. Le tableau 1.1 montre les différences entre les deux modèles [5].

Critère	**Mode client/serveur**	**Mode P2P**
Gestion	Supervisée	Auto-organisée
Accès aux ressources	Recherche	Découverte
Organisation	Hiérarchique	Distribuée
Mobilité	Statique	Dynamique
Disponibilité	Dépendante du serveur	Indépendante des pairs
Modèle de programmation	RPC	Asynchrone

Tableau 1.1 : Comparaison entre le modèle client/serveur et le modèle P2P.

1.6 Classification des réseaux P2P

Les réseaux P2P sont subdivisés en trois catégories de système (voir figure 1.2) : centralisé, décentralisé et hybride. La catégorie décentralisée peut encore subdiviser en deux types : structuré et non structuré. La différence principale entre ces systèmes est le mécanisme utilisé pour rechercher des ressources dans le réseau Peer to Peer.

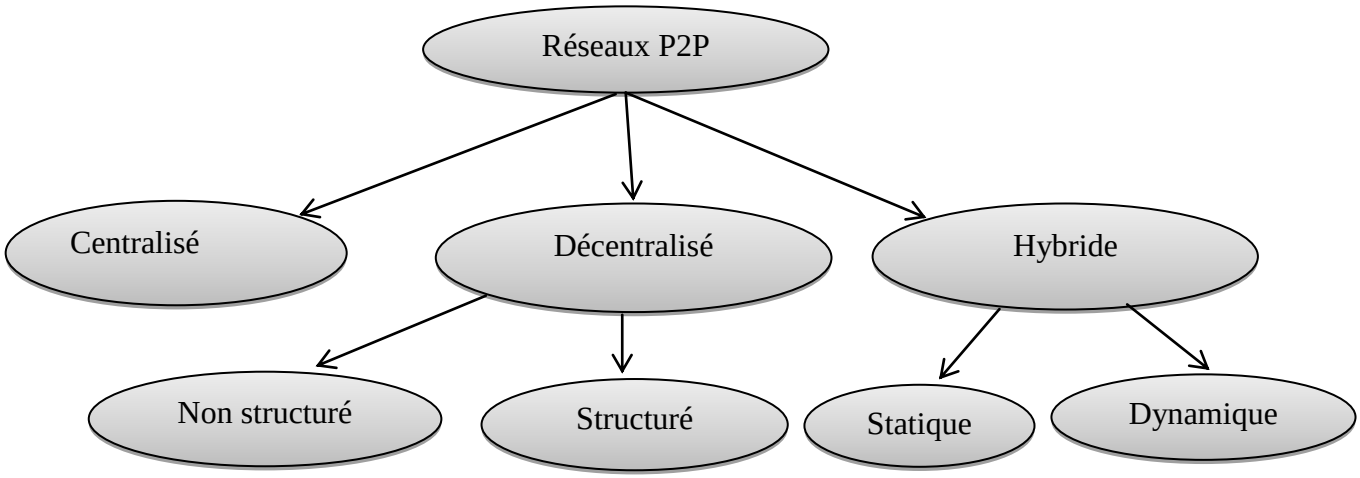

Figure 1.2 : Classification des réseaux P2P.

1.6.1 Architecture centralisée

L'architecture centralisée repose sur un serveur central, qui joue le rôle d'un index et qui stocke des informations concernant la description des ressources partagées (nom, taille,…) et d'autres sur les utilisateurs qui les hébergent (nom utilisé, IP, nombre de fichiers partagés,…). Les ressources étant toujours hébergées sur les pairs. Une fois les opérations de découverte et de localisation sont effectuées sur le serveur central, les échanges de services sont réalisés directement entre pairs (voir figure 1.3).

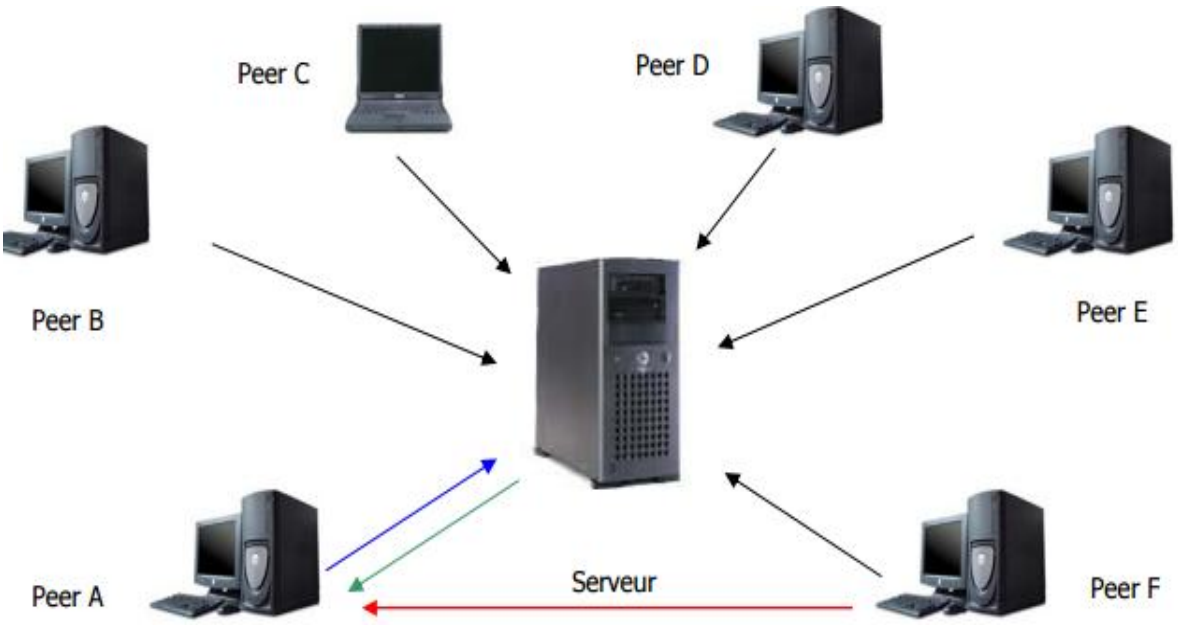

Figure 1.3 : Architecture centralisée.

L'exemple type d'une telle architecture est Napster, c'est le premier programme P2P conçu pour le partage de fichiers, spécialement pour l'échange de fichiers audio/vidéo.
L'inconvénient de cette architecture est le risque de surcharge de l'index qui peut devenir un point de défaillance du système.

1.6.2 Architecture décentralisée

Les nœuds jouent le rôle de serveur et de client en même temps, ils remplissent les mêmes tâches, ce qui donne une nature plate à l'architecture (voir figure 1.4).
Cette architecture présente l'avantage d'être robuste, elle ne dispose pas d'un serveur central, mais la recherche de fichiers prendra plus de temps si des milliers d'utilisateurs sont présents, chaque requête sera adressée à chaque utilisateur connecté.
On peut distinguer deux types d'architectures décentralisées : non structurée et structurée.

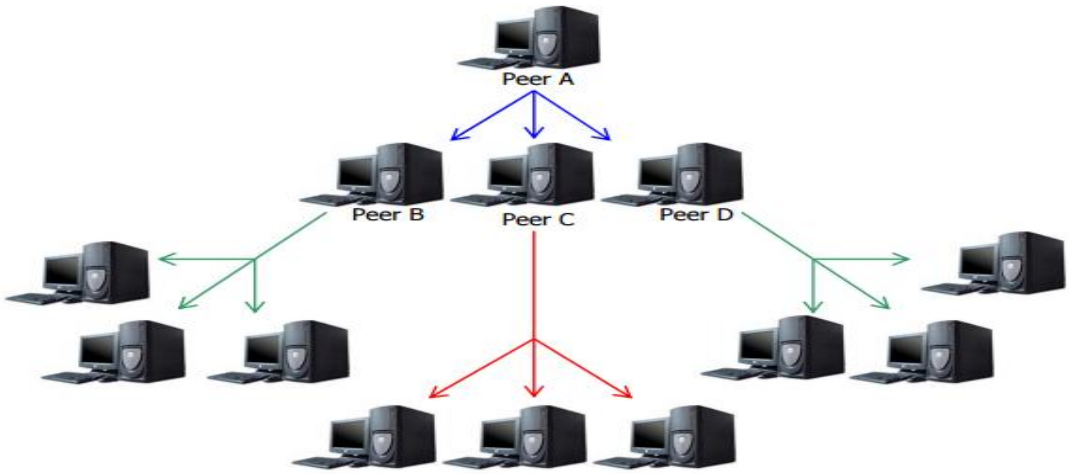

Figure 1.4 : Architecture décentralisée

1.6.2.1 Architecture décentralisée non structurée

Les nœuds sont autonomes et ne sont pas respensables que de leurs propres ressources , il n'y a aucune relation entre le nom d'un objet et le nœud qui en est responsable.

Les protocoles Gnutella et BitTorrent sont des exemples de réseaux non structurés. Ils sont basés sur un index global, où ils utilisent des algorithmes de diffusion pour la localisation et la découverte d'autres pairs voisins. La maintenance de cette architecture de réseau est globalement simplifiée, tandis que la sociabilité représente un grand obstacle.

1.6.2.2. Architecture décentralisée structurée

Les réseaux structurés sont basés sur les tables de hachage distribuées (**D**istributed **H**ash **T**ables (DHTs)) pour gérer les opérations de recherche, Une entrée dans la table contient la paire (clé, valeur), où la clé est l'identifiant associé à un objet partagé et la valeur correspond à l'information de localisation d'objet recherché. Cycloid, Chord, CAN… sont des exemples des réseaux structurés, Ils sont basés sur le concept de table de hachage distribuée (DHT). Avec l'approche DHT, chaque nom d'entité dans le système peut être met en cohérence sur le même espace de recherche, en utilisant une fonction de hachage telle que SHA-1 ou SHA-2. Cependant, toutes les entités dans le système ont une vue consistante de cette correspondance. Étant donné cette vue consistante, plusieurs structures d'espace de recherche sont définies pour localiser ces entités. A titre d'exemple, dans Chord, l'espace de recherche est basé sur une topologie en anneau. Dans P4L, il est structuré sous forme d'anneaux hiérarchiques.

1.6.3 Architecture hybride

Le modèle hybride est un couplage entre le modèle P2P et l'architecture client/serveur dont les nœuds disposant d'une bonne bande passante sont organisés en super-peers (les serveurs) et les nœuds disposant une faible bande passante sont reliés en mode client/serveur à un super-peer. Nous distinguons deux types de réseaux hybrides (voir figure 1.5).

1.6.3.1 Hybrides statiques

Les super-peers sont manuellement désignés dés le début, ces derniers peuvent jouer en même temps le rôle de client, comme on le retrouve dans l'exemple du réseau eDonkey.

1.6.3.2 Hybrides dynamiques

Dans certaines conditions, le logiciel client décide de transformer un nœud client en super-peer, comme dans l'exemple de Kazaa .

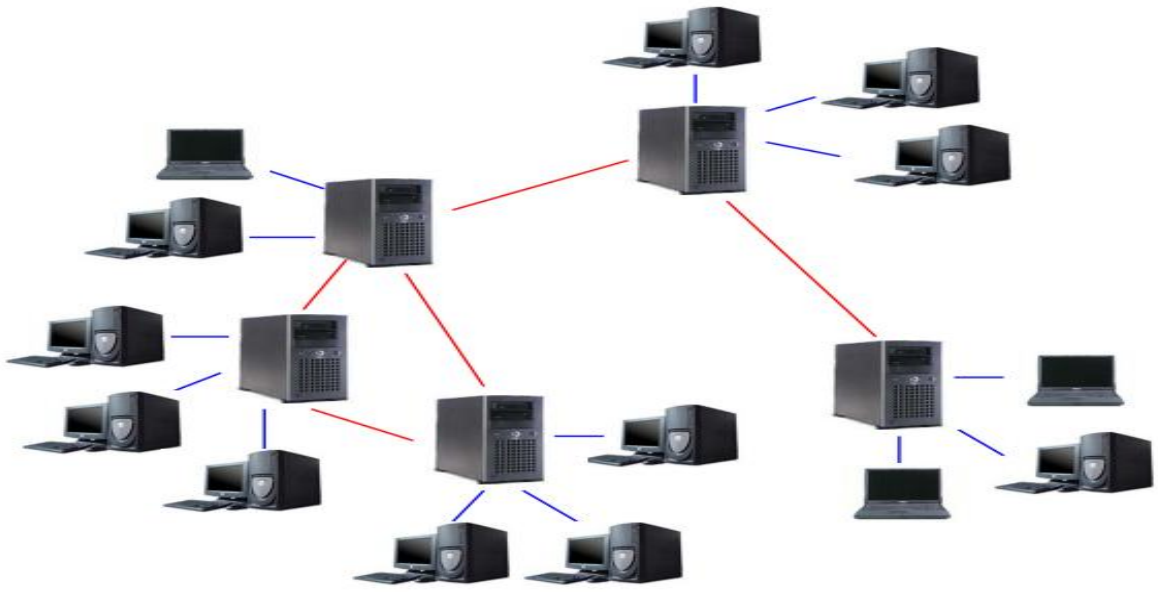

Figure 1.5 : Architecture hybride

1.7 Avantages et inconvénients des architectures P2P

Les tableaux suivant représentent les avantages et les inconvénients des architectures P2P [9].

Avantages	Inconvénients
- Charge minimale sur les peers - Accés aux ressources - Recherches évoluées	- Disponibilité du serveur - Coût de serveur - Confiance dans l'administration du serveur

Tableau1.2 : Réseau pair-à-pair centralisé

Avantages	Inconvénients
- Stockage des données dans les DHT - Passage à l'echelle/charge homogéne - Décentralisation - Distribution complète	- Recherches exactes uniquement - Structure à maintenir

Tableau 1.3 : Réseau pair-à-pair décentralisé structuré

Avantages	Inconvénients
- Décentralisation - Recherches non-exactes - Distribution complète	- Passage à l'échelle limité(inondation) - Charge non-homogène des pairs (super pairs)

Tableau 1.4 : Réseau pair-à-pair décentralisé non structuré

Avantages	Inconvénients
- Présence du serveur centrale : facile à administrer et à controler. - Evite les recherches couteuses. - Service novateur qui généralise le P2P.	- Pas d'anonymat partout - Probléme de disponibilité et de passage a l'echelle

Tableau 1.5 : Réseau pair-à-pair hybride

1.8 Exemples de réseaux P2P de partage de fichier

1.8.1 Napster

Est le premier système pair-à-pair créé en 1999 par Shawn Fanning [7]. Il propose un service de partage de fichiers musicaux, adopte une architecture centralisée dont laquelle un serveur est contacté pour obtenir des méta- informations (identité du pair sur lequel sont stockés les informations), mais les données restent distribuées sur les pairs et les échanges de données se font directement d'un pair à un autre.

Dans cette architecture, chaque utilisateur désirant partager des fichiers doit établir une connexion avec le serveur central Napster et lui déclare les fichiers partagés, le serveur Napster maintient un index avec toutes les adresses IP des pairs participants, ainsi que l'index de leurs données partagées. Les pairs sont donc les fournisseurs de fichiers, et les échanges vont se faire directement entre eux. La recherche et le téléchargement vont se passer de la manière représentée sur la Figure 1.6, un pair recherchant un fichier envoie sa requête au serveur central (1), qui va lui répondre par une liste triée de pairs qui hébergent le fichier recherché (2). Le pair va alors choisir parmi les réponses celle qui lui convient et contacte directement le pair choisi. Le transfert s'effectue alors directement entre les deux pairs (3).

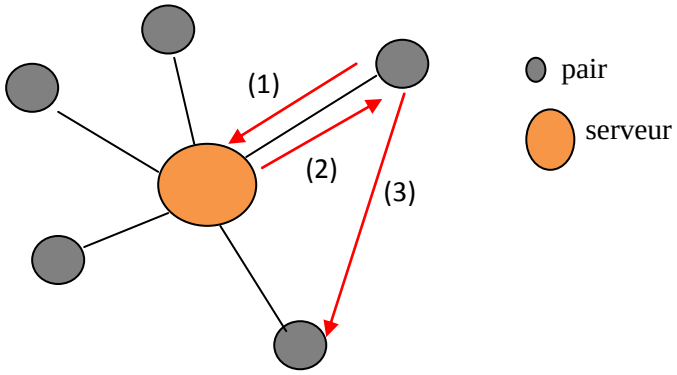

Figure 1.6 Architecture de Nepster

Nepster est basé sur un serveur central qui permet une recherche simple, il est facile à administrer et à contrôler, cependant il présente un inconvénient majeur, il n'est pas robuste car la surcharge ou la panne du serveur central rend tout le réseau indisponible.

1.8.2 Gnutella

Gnutella est un protocole de partage de fichiers totalement décentralisé, développé en mars 2000 par Justin Frankel et Tom Pepper [8].Chaque pair joue à la fois le rôle de cli**ent** et de **serv**eur « *servent* ». Le protocole Gnutella définit cinq types de messages :

Ping : Il est utilisé pour découvrir les autres servents sur le réseau. Un servent qui reçoit un *Ping* doit répondre avec un ou plusieurs *Pong*.

Pong : C'est la réponse à un *Ping*, elle contient l'adresse IP et le numéro du port du servent, ainsi que le nombre et la quantité de données partagées.

Query : Est utilisé pour la recherche d'un fichier

QueryHit : Pour la réponse à une requête *Query*.

Push : Permet de télécharger des données depuis un servent situé derrière un firewall.

Ces messages sont transmis par une technique d'inondation, chaque message est muni d'un champ TTL (*Time To Live*) qui limite le nombre de sauts, la valeur attribuée au TTL est généralement 7, lorsqu'il arrive à zéro, la requête n'est plus renvoyée.

Un pair qui souhaite rejoindre le réseau Gnutella, il commence par identifier les nœuds présents sur le réseau, pour cela, il envoie un message d'identification *Ping* à ses voisins, qui l'envoient à leur tour à leurs voisins et ainsi de suite, la retransmission est stoppée lorsque le

TTL devient 0. Chaque pair recevant un *Ping* répond avec une réponse *Pong* contenant l'adresse IP, le numéro de port, le nombre et la taille des fichiers partagés.
Pour chercher une ressource dans le réseau, un pair lance une requête *Query* en spécifiant les critères de recherche, si un voisin ne possède pas la donnée recherchée, il continue de transmettre la requête vers ses voisins et ainsi de suite, jusqu'à l'expiration de la requête (*TTL=0*),chaque nœud qui a reçu la requête et qui possède la donnée, renvoie une requête de réponse *QueryHit* au voisin qui lui a retransmis la requête, spécifiant son adresse IP et son numéro de port TCP où l'objet peut être téléchargé. La réponse remonte de proche en proche jusqu'au pair initiateur. Ce dernier sélectionne ensuite les fichiers à télécharger, en envoyant directement une requête de téléchargement au pair possédant le fichier.

L'inconvénient majeur du protocole Gnutella est que la recherche n'est pas exhaustive, c'est-à-dire qu'une recherche sans réponse ne signifie pas que la donnée cherchée n'existe pas dans le réseau, mais simplement qu'elle n'est pas trouvée.

1.8.3 BitTorrent

BitTorrent [11] est un protocole développé au début des années 2000et couramment utilisé sur Internet. Celui-ci offre des fonctionnalités de partage de fichiers et repose sur une stratégie dite gagnant-gagnant (tit for tat) et sur une diffusion multi-sources afin de permettre un téléchargement efficace. Le protocole BitTorrent fut tout d'abord conçu comme un modèle hybride : les données sont stockées et échangées directement entre les pairs, l'accès au réseau, tout comme l'indexation et la recherche de méta-données, reposent sur l'utilisation de super-pairs appelés ici trackers. Afin d'améliorer la robustesse du protocole et son aspect décentralisé, diverses extensions ont été proposées : L'utilisation d'une table de hachage distribuée basée sur le protocole Kademelia, indexant à la fois les nœuds et les méta-données, permet au pairs de s'échanger directement des contenus en utilisant un modèle structuré. Une seconde modification repose sur un système d'échanges de pairs et un système permettant aux pairs de s'échanger les méta-données directement entre eux. Enfin, une troisième extension Tribler, complète la seconde et permet d'utiliser un modèle par inondation où les pairs transmettent les requêtes les uns des autres à l'intérieur du réseau. Ces différentes extensions ont permis d'augmenter la robustesse du protocole et d'en améliorer les performances. Le protocole BitTorrent repose donc à la fois sur les modèles centralisés structurés et non structurés.

1. 9 Les problèmes des réseaux P2P

Dans cette section nous présentons quelques problèmes courants des réseaux P2P [6].

1.9.1 Equilibrage de charges

Les pairs dans les réseaux P2P jouent le même rôle, quel que soit leurs participations (soit serveur, soit client). Ils assurent aussi avec une responsabilité partagée toutes les fonctionnalités de réseau (auto-organisation, routage,…). Les nœuds qui ont des capacités faibles tels que les PDAs ne peuvent pas assurer ces fonctionnalités.

1.9.2 Media streaming en temps réel

Le transfert des données multimédia (vidéos, audio,…) consomment beaucoup de bandes passantes pour les acheminer vers leurs destinations, les algorithmes classiques tel que DJIKSTRA, BELMAN FORD,… ne ressoudent pas ce problème. Donc, il faut des algorithmes spécifiques pour chaque architecture.

1.9.3 Le freeloading

Pour que le réseau P2P fonctionne d'une façon performante, il faut la participation active de tous ses membres. Les réseaux P2P souffrent d'un problème de freeloading, ou les freeloaders prennent sans donner, ils bénéficient des ressources partagées sans partager les leurs, ce qui cause un problème dans l'équilibre du réseau.

1.9.4 Calcul parallèle

Un très grand nombre de problèmes nécessitent un calcul énorme et puissant dans lequel les machines les plus puissantes dans le monde ne peuvent pas l'accomplir toutes seules. Des architectures spécialisées (parfois matérielle) sont très bien adaptées à ce type d'applications, mais malheureusement, ces architectures ne sont pas à la portée de tout le monde, car elles sont coûteuses et réservées aux grands centres de recherche et d'entreprises commerciales. Pour ces raisons, quelques solutions essayent de faire participer les machines oisives de réseaux Internet afin de réaliser des calculs complexes.

1.9.5 Sécurité

La sécurité est parmi les grands problèmes des réseaux P2P, les mécanismes de sécurité matérielle ne sont pas compatible avec l'architecture, et aussi, à cause de la nature ouverte de

ce type de réseau (pas de contrôle d'accès), les mécanismes logiques sont très difficile a implémenté.

1.10 Conclusion

Les systèmes P2P représentent un domaine type de réseaux très actif grâce à leurs utilisations et leurs applications, ils sont classés selon leurs architectures. Chacun de ces réseaux possède des caractéristiques bien différentes, ils offrent beaucoup d'avantages tels que: la répartition de la charge et la résistance aux pannes, mais aussi quelques inconvénients liés à la distribution du contenu tel que le média streaming.

Le chapitre suivant sera consacré aux systèmes de media streaming dans les réseaux P2P.

CHAPITRE 2

Média streaming dans les réseaux pair à pair

2 .1 Introduction

La technique du streaming vidéo consiste à transférer des données multimédia sous forme de flux (stream) régulier et continu. Le streaming ne nécessite pas de téléchargement préalable de la séquence. Il permet à l'utilisateur de lire un contenu audio et vidéo au fur et à mesure de son téléchargement. Le flux peut être en direct (live) ou préenregistré sur un serveur.

Dans ce chapitre, nous allons étudier le media streaming dans les réseaux P2P. Nous décrivons les champs d'application du streaming, les topologies de streaming P2P, les stratégies de sélection des blocs et des nœuds ainsi que les techniques de transfert des blocs.

2.2 Les champs d'application du streaming

Dans les réseaux P2P, les applications de streaming peuvent être classifiées en deux principaux types : vidéo à la demande (VoD) et streaming en direct [4].

2.2.1 Vidéo à la demande

La vidéo à la demande, également appelée VOD, est proposée par de nombreux opérateurs Internet et téléphone, Il s'agit d'un service qui permet de commander et de regarder une vidéo en ligne, lorsqu'on le souhaite. Par rapport à la diffusion en direct, la VOD offre plus de flexibilité et de commodité aux utilisateurs. La VOD a été identifiée comme la caractéristique clé pour attirer les consommateurs vers le service IPTV.

Après un court délai de latence, le client peut commencer à visionner sa séquence vidéo choisie. Il a la possibilité d'arrêter la diffusion en cours, de la reprendre, de retourner en arrière et d'avancer en mode accéléré, exactement comme avec un magnétoscope (VCR).

2.2.2 Streaming en direct

Un contenu vidéo en direct est diffusé à tous les utilisateurs en temps réel. Il s'agit d'une forme particulière de streaming, dans laquelle la diffusion de la vidéo est simultanée à sa capture, sans possibilité de montage ni d'édition. Le live streaming est particulièrement adapté à la diffusion d'événements, permettant à ceux qui visionnent la vidéo de partager l'événement avec les spectateurs y assistant en live avec un décalage d'au plus quelques secondes. Ce mode de transmission de type streaming est donc souvent utilisé par exemple pour retransmettre une manifestation au moment où elle a lieu.

2.3 Topologies de streaming P2P

Les systèmes de distribution du media streaming P2Ppeuvent être classifiés en deux types de topologies, la topologie basée sur les arbres et la topologie basée sur le maillage [12] [13].

2.3.1 Topologie basée sur les arbres

Le streaming basé sur un arbre provient de l'architecture d'un arbre de diffusion IP multicast (formé par les routeurs au niveau réseau). Dans le streaming basé sur les arbres, les utilisateurs qui participent à la diffusion d'une vidéo forment un arbre à la couche application, cette arborescente appelée Application-Level Multicast (ALM), dans laquelle chaque utilisateur s'inclut à un certain niveau de l'arborescence. Il reçoit le flux vidéo à partir de son nœud père se trouvant à un niveau supérieur et distribue le flux vidéo pour ses nœuds fils positionné dans le niveau inférieur.

Pour la construction d'un arbre avec un nombre donné de pairs, il existe plusieurs possibilités pour composer le graphe de l'arbre. Chaque algorithme à sa propre stratégie pour le faire, bien que la majorité des algorithmes se base sur deux principaux facteurs :

- La profondeur de l'arbre (le nombre de niveaux dans l'arbre).
- Le débit de diffusion des nœuds internes (ou le nombre de fils que peut porter un nœud interne, qui est en relation avec la bande passante).

Nice[15], SpreadIt [16], PeerCast[17], ESM [18] et Overcast [19] sont des systèmes qui se basent sur un arbre, où la racine est la source du stream, les autres nœuds coopèrent dans la retransmission des pièces de la vidéo déjà reçues.

Ces systèmes présentent quelques inconvénients comme la surcharge du nœud parent, la synchronisation des nœuds, l'hétérogénéité des nœuds, etc.

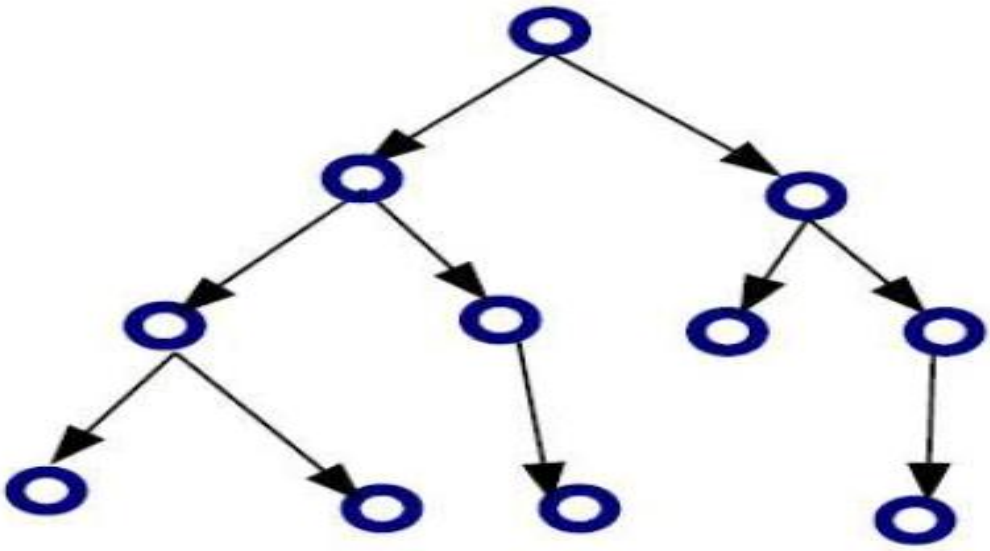

Figure 2.1 : Topologie basée sur les arbres.

- **Topologie basée multi-arbres (foret)**

Ce mode d'organisation répond aux problèmes soulevés dans la diffusion basée sur un arbre. Dans ce type de topologie en multi-arbres, le flux multimédia est subdivisé en plusieurs sous-flux (si par exemple en prend un flux subdivisé en deux sous-flux, le premier aura le flux des paquets pairs du flux original, et le deuxième comprendra les paquets impairs), la subdivision du flux en plusieurs sous flux conséquent la création de plusieurs sous arbres au lieu d'un seul arbre dont chaque sous arbre pour un sous flux. Un nœud fait partie de chaque sous arbre crée. Le nœud participant à la diffusion du vidéo peut être positionné comme nœud interne ou nœud terminal, néanmoins pour des propos d'optimisations, la majorité des systèmes basées foret font en sorte que le nombre de fois où le nœud doit paraitre comme nœud interne, est en relation avec sa bande passante sortante, elle doit être utilisée au maximum. Dans ce mécanisme, chaque nœud recueille les différents sous-flux des différents sous-arbres où il appartient, pour assembler les sous-flux et synthétiser le flux original.
Les travaux tels que SplitStream [20], Bullet [21] et CoopNet [22] se basent sur ce type de topologie.

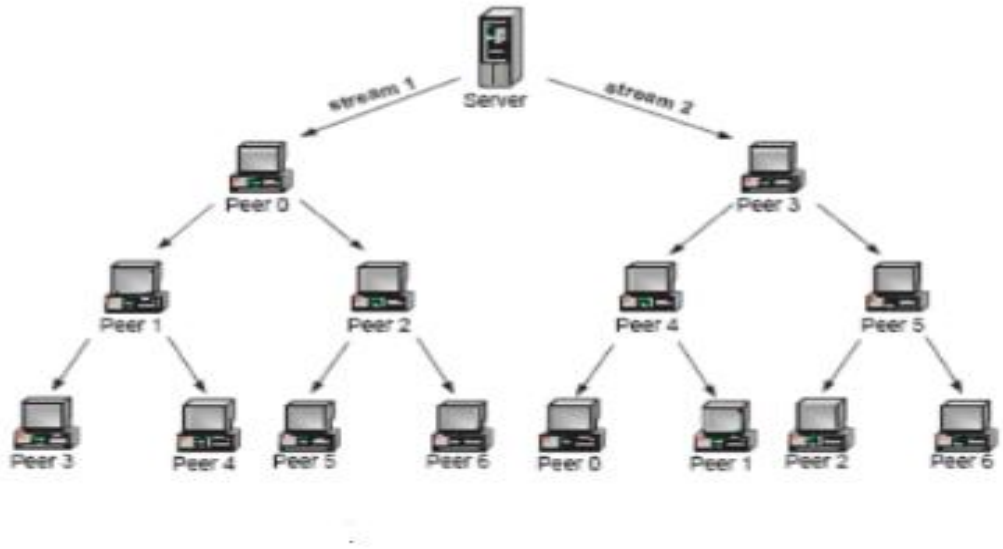

Figure 2.2 : Topologie basée sur les multi-arbres.

2.3.2 Topologie basée sur le maillage

Les systèmes basés sur des arbres permettent uniquement aux pairs d'avoir un parent dans une seule arborescence de streaming à partir duquel il télécharge le morceau, ce qui provoque un seul point de défaillance. Si le parent du pair quitte, le pair et tous ses fils cessent de récupérer des informations jusqu'à ce que le pair se connecte à nouveau à un parent différent. Pour faire face à ce problème, de nombreux systèmes de streaming P2P adoptent une approche basée sur le maillage, dans laquelle un nœud ne collabore pas avec un seul nœud mais avec plusieurs. La principale caractéristique du système de streaming basé sur le maillage est qu'il n'y a pas de topologie de streaming statique. Un pair détient un nombre donné de voisins avec lesquels il peut télécharger ou transmettre du contenu vidéo. Si un voisin viendrait à quitter sa liste de partenaires, il peut tout aussi télécharger le contenu média à partir d'autres pairs.

La différence entre les protocoles basés sur le maillage se fait généralement sur deux entités de grande importance. La première est le sélecteur de voisins, c'est le compartiment responsable de la sélection, la connexion, la déconnexion des voisins. Sa stratégie pour effectuer ces opérations se base principalement sur la disponibilité des ressources comme le nombre de connexions dans un pair, la bande passante montante et descendante, la CPU, la mémoire, et la qualité de la connexion avec le voisin comme le délai… etc. D'après ces critères, un sélecteur de voisin peut choisir le voisin qui convient le plus au pair. La deuxième

entité est le scheduler, son algorithme est le principal facteur de performance pour un système basé mailles. Son rôle est de choisir le voisin dont le pair doit lui transmettre le bloc vidéo, et le voisin dont le pair doit télécharger le bloc vidéo. En général, le scheduler se base sur les critères suivants : La disponibilité du bloc parmi ses voisins, le deadline du bloc vidéo (le temps où le bloc doit être joué) et la bande passante du voisin.

Les plus connus des systèmes de streaming qui implémentent ce mode de distribution de contenus sont : PRIME [23] et DONet/CoolStreaming [24].

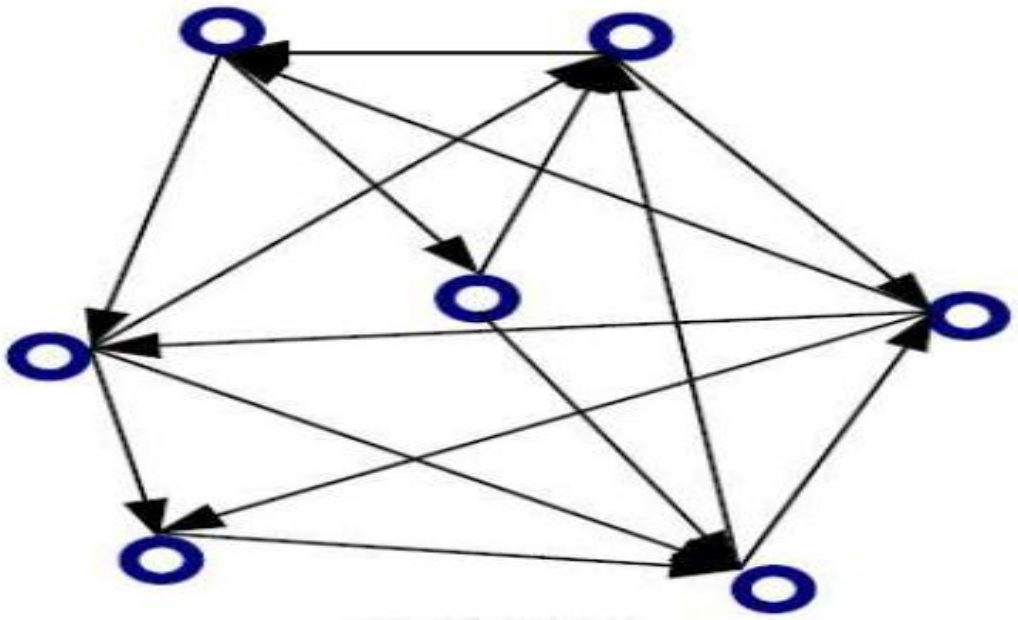

Figure 2.3 : Topologie en mailles.

2. 3.3 Comparaison entre les topologies en arbre et en mailles

Dans le tableau suivant, nous présentons une comparaison entre la topologie basée sur les arbres et la topologie basée sur le maillage [14].

Aspects	Topologies	
	Arbre	Mailles
Source	Un seul nœud offre la vidéo	Plusieurs nœuds contribuent
Auto-organisation	A l'arrivée, il s'attache en bas de l'arbre, en profondeur	A l'arrivée, le nœud obtient une liste de nœuds avec qui il va coopérer
Maintenance	Difficile à maintenir, après le départ d'un parent, tous ses descendants deviennent orphelins.	Messages hello (Ping) suffisent à chaque nœud pour maintenir ses voisins.
Dynamique	La stabilité du système est détériorée par la dynamique des nœuds.	Pas d'effets sur la topologie grâce aux connexions aléatoires renouvelées à chaque départ/arrivée
Utilisation optimale de la bande passante	Pas de contribution de débit montant des nœuds feuilles	Bonne coopération du débit montant et descendant. Surtout que chaque nœud envoie/reçoit a/de plusieurs voisins à la fois.
Mobilité	La topologie des arbres ne résiste pas au départ des nœuds même s'il est actif autre part.	Pendant son déplacement le nœud prévient ses voisins et obtient une nouvelle liste dès qu'il arrive à sa destination.

Tableau 2.1 : Comparaison entre la topologie en arbre et en maillage.

Dans ce tableau comparatif, il apparait clairement que la topologie en maille est beaucoup plus robuste que celle basée sur les arbres. Dans la structure en arbre, un nœud ne reçoit que de son parent direct. Donc, lorsque celui-ci tombe en panne, il se déplace ou quitte tout simplement les feuilles, tous le sous-arbre (composés par l'ensemble de descendants du nœud en question) ne parviennent pas à obtenir le flux jusqu'à ce que l'arbre soit reconstruit. Dans les réseaux de streaming basés sur le maillage, la vidéo est obtenue à partir de plusieurs voisins, et dans ce cas, si l'un d'entre eux échoue, d'autres continueront à fournir le stream, encore faut-il qu'au moins une copie de chaque bloc existe.

Les structures en mailles rencontrent un problème, en général, est celui de ne pas compter sur une structure réseau prédéfinie et sont donc, plus difficiles à étudier que les topologies en arbre [14].

2.4 Les techniques de transfert des blocs

2. 4.1 Le mode push

Dans le mode push [14], l'expéditeur est à l'initiative, et envoie le bloc déjà reçu aux voisins choisis auparavant. Nous notons que le mode push dans un arbre est assez diffèrent de celui du maille. Le premier n'a pas besoin de consulter les récepteurs pour la validité du bloc, ce qui permet de réaliser un plus faible retard d'acheminement.

La stratégie push est utilisé généralement dans les architectures basées arbre en raison de la nature mono-direction (du père vers le fils) de la structure arbre. Pour les architectures basées mailles, la relation père-fils est inexistante, et une stratégie push peut poser de problème, pour la principale raison qu’un pair peut aveuglement transmettre un bloc vidéo que le voisin dispose déjà. De ce fait, la stratégie push a le sérieux inconvénient de gaspiller la bande passante montante des pairs avec la redondance des blocs vidéo, par conséquent, cette technique est peu utilisée dans les architectures basée mailles.

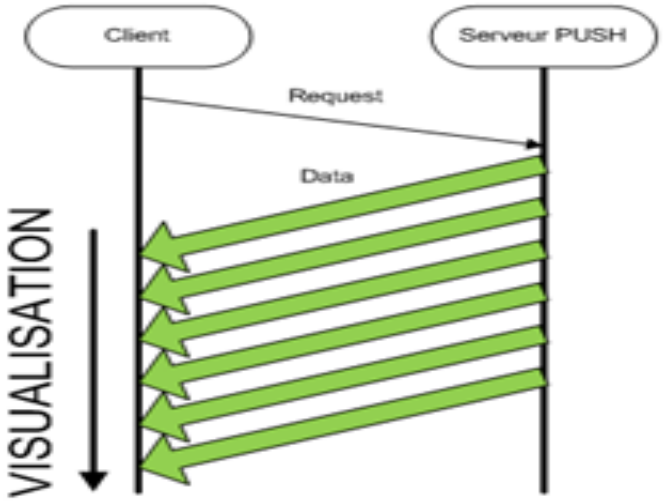

Figure 2.4 : Le mode push [4].

2.4.2 Le mode pull

Dans le mode pull [14], le récepteur a l'initiative de prendre le bloc qu'il veut de son voisin. Ce mode est plus avantageux par rapport au maintien de la qualité de services (QoS) pour le récepteur. Parce que celui-ci peut choisir les blocs à recevoir, il a donc un meilleur control du processus de livraison, et peut facilement assurer la livraison à temps d'un bloc, ceci est essentielle dans le media streaming P2P.

Dans ce mode de diffusion, c'est le client qui prend en charge la gestion des données vidéo. Ainsi, à chaque fois que le client a besoin de blocs de données (flux vidéo), il envoie au serveur un ensemble de sous-requêtes d'où l'appellation « client PULL » [4].

Le système basé maille utilise le mode pull pour éviter la situation décrite précédemment (paragraphe 2.4.1**)(**gaspillage de la bande passante montante des pairs avec la redondance des blocs vidéo). Puisque c'est le pair en question qui exprime explicitement sa demande du bloc vidéo dont il a besoin. Cette technique permet aux pairs d'échanger périodiquement la disponibilité des blocs en fonction des cartes de tampons. Une carte de tampon contient le numéro de séquence de blocs actuellement disponibles dans le tampon d'un pair, de cette façon, un pair peut décider à partir de quels pairs télécharger quels blocs, en évitant les transmissions redondantes de blocs qui étaient présentes en utilisant le mode push.

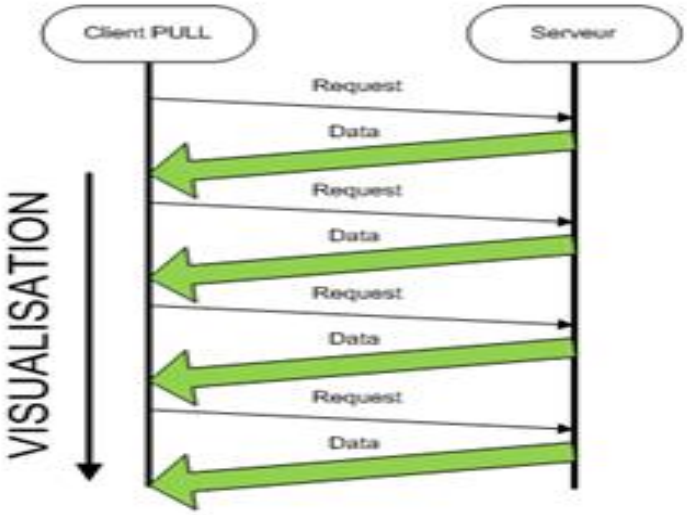

Figure 2.5 : le mode pull [4].

2.4.3 Le mode hybride

Dans un mode hybride [4], le récepteur et l'émetteur peuvent prendre l'initiative de négocier le transfert de blocs. Les blocs y sont classés en deux catégories: des blocs pull et des blocs push. Un nœud lorsqu'il rejoint le réseau utilise le mode pull pour obtenir les premiers blocs, puis il utilise le mode push pour envoyer les blocs à ses voisins.

2.5 Les stratégies de sélection de blocs et de nœuds

Quel que soit le protocole P2P, le nœud doit décider d'une stratégie de sélection de bloc, c'est-à- dire, quel bloc demander/offrir, et de/a qui le bloc est reçu/donne. Chaque stratégie possède ses avantages et inconvénients. Nous appelons ces stratégies : la sélection de pairs et la sélection de blocs.

2.5.1 La sélection de pairs

Dans cette sélection, un pair choisit le nœud à qui il va donner des blocs. Pour le mode pull, le récepteur choisit généralement les pairs émetteurs en fonction de leurs capacités d'envoi, par exemple, choisir des pairs qui offrent un moindre délai entre le moment de la demande et celui de la réponse. Pour le mode push, l'expéditeur peut sélectionner ses destinataires sur la base de leurs performances.

2.5.2 La sélection de bloc

Cette sélection choisit un bloc sur une liste de blocs disponibles chez un voisin. Il existe plusieurs schémas de sélection de bloc, nous citons [25]:

- Séquentiel: Sélectionne le premier bloc disponible.
- Rarest: Choisi le bloc le plus rare dans le voisinage local du pair. Il n'existe aucune méthode pour départager les blocs avec la même rareté.
- Random: Sélectionne un bloc au hasard.
- Randomrarest: Choisi le bloc le plus rare dans le voisinage local du pair. Si plusieurs blocs sont de même rareté, un bloc est sélectionné au hasard.

2.6 Conclusion

Le streaming en direct et la vidéo à la demande sont les applications multimédias les plus populaires sur Internet ces dernières années. Les solutions traditionnelles de streaming vidéo client-serveur entraînent des coûts de fourniture de bande passante coûteux sur le serveur. Alors, La mise en réseau P2P est un nouveau paradigme pour construire des applications réseau distribuées. Récemment, plusieurs systèmes de streaming P2P ont été déployés pour fournir des services de streaming vidéo en direct et à la demande sur Internet à faible coût de serveur. Ces systèmes réduisent la charge de serveur et fournissent une distribution de contenu évolutive.

Le chapitre suivant sera consacré aux systèmes de media streaming dans les réseaux sociaux en P2P.

CHAPITRE 3

Systèmes de media streaming dans les réseaux sociaux en P2P

3.1 Introduction

Les réseaux sociaux sont en plein essor et jouent un rôle important dans notre vie culturelle, sociale, économique et professionnelle. Le media streaming est l'une des applications de plus en plus populaire dans les réseaux sociaux en ligne comme (Facebook, Youtube…). Cependant, l'architecture client/serveur déployée par ces réseaux sociaux présente des problèmes, telles que l'espace de stockage limité, des frais pour l'utilisation du réseau social ou pour profiter de quelques services des réseaux sociaux, etc. Pour outrepasser ces problèmes, L'adoption des architectures P2P est une meilleure alternative.

Dans ce chapitre, nous allons étudier quelques systèmes du média streaming dans les réseaux sociaux en P2P.

3.2 Media streaming dans les réseaux sociaux P2P

Dans cette partie, nous allons présenter quelques travaux de recherche liés aux architectures du média streaming dans les réseaux sociaux P2P.

3.2.1 NetTube

Les auteurs ont proposé NetTube [25], une nouvelle architecture P2P de partage de vidéos de petite taille dans les réseaux sociaux.

Dans cette architecture, le serveur stocke toutes les vidéos et les fournit aux clients. Les clients partagent également les vidéos par des communications P2P, chaque client désigné ainsi sous le nom d'un pair. Un pair met en cache toutes ses vidéos précédemment regardées, et les rend disponibles pour la redistribution. Pour un client intéressé par une vidéo particulière, tous les pairs qui ont précédemment téléchargé cette vidéo deviennent des fournisseurs, formant un overlay pour cette vidéo comme un Swarm, motivé par la structure semblable dans BitTorrent, Ainsi, un pair de NetTube peut apparaître dans plusieurs Swarms et dans chaque Swarm, on trouve au moins un fournisseur.

Comme le montre la figure 3.1, un exemple, où il y a cinq Swarms de vidéos v1 à v5, tel que dans le Swarm v1, le pair P1 est un fournisseur, P2 et P3 sont des clients qui ont téléchargé la vidéo.

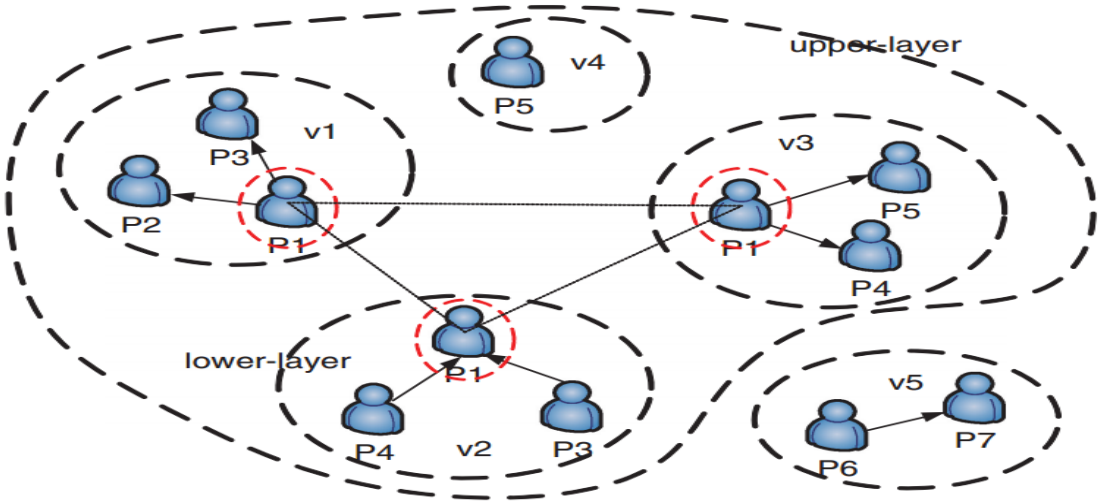

Figure 3.1 : Illustration de l'overlay en deux niveaux.

En général, comme les clients regardent une série de vidéos, il est nécessaire de localiser rapidement les fournisseurs potentiels pour la prochaine vidéo et de permettre une transition plus souple. À cet effet, NetTube présente un overlay upper-layer au-dessus des Swarms de différentes vidéos, dans cette overlay, pour un pair donné, des relations de voisinage sont établies entre tous les Swarms qui contiennent ce pair. C'est une relation conceptuelle qui ne sera pas utilisé pour la distribution de données, mais, elle permet la recherche rapide des fournisseurs de vidéos dans le réseau social. Comme le montre la figure 3.1, le pairP1existe dans les Swarms de v1, v2 et v3, ainsi ces trois Swarms ont des relations de voisinage mutuelles, reliées par le pairP1. Le Swarm v4 est également dans cet overlay conceptuel, en raison du pair commun P5, cependant, le swarm v5n'a aucune relation avec les autres Swarms.

Quand un nouveau Pair rejoint le système NetTube, d'abord, il envoie une demande de sa première vidéo au serveur, Le serveur redirige ce dernier au swarm correspondant en lui fournissant la liste des pairs fournisseurs potentiels, puis, l'opération d'ajout de ce pair sera réalisée par l'algorithme de construction des swarms. Si le pair est le premier à regarder la vidéo, dans ce cas, le serveur fournira directement le service de streaming.

Quand un client termine de regarder une vidéo et commence à télécharger la prochaine, il joindra un autre Swarm, mais restera dans les Swarms précédents en tant que fournisseur potentiel, jusqu'à ce qu'il quitte le système NetTube. Pour joindre le prochain Swarm,

d'abord, il vérifie la disponibilité de la prochaine vidéo dans tous les nœuds voisins des swarms où il appartient, Si non, les nœuds voisins vérifieront plus loin avec leurs voisins dans leurs Swarms voisins, Intuitivement, ceci exploite la relation amis à amis (freind to freind (F2F) entre les pairs, parce que ces pairs dans différents Swarms ont probablement des intérêts similaires.
Par exemple, supposons P2 a fini de regarder v1 et il clique sur v2, il a seulement un voisin P1et P1ne cache pas v2, mais, quand P1cherche la vidéo dans ses voisins, il trouve que P3 et P4 ont caché v2.Ainsi P2peutcontacter P3et P4 à partir de P1 et alors télécharger v2.

Dans ce système, pour une recherche rapide et efficace des vidéos, un modèle d'indexation et de recherche de vidéos est proposé, dont le quel :

Chaque pair maintient un vecteur de ses vidéos. Vu le nombre importants des vidéos, la structure du vecteur doit être efficace et scalable pour la recherche. Pour cette fin, chaque vecteur est représenté par *Bloom filer* [26], qui est une structure de données Space/Time d'indexation.

Pour chaque nouveau pair, pour localiser les fournisseurs de sa première vidéo, le serveur utilise une table d'indexation, dont laquelle, il maintient seulement les identificateurs de quelques pairs fournisseurs de la vidéo. Si le nouveau pair (ou un pair existant) ne peut pas localiser assez de fournisseurs, le serveur peut servir de fournisseur additionnel.

Le système adopte une organisation overlay basée sur le maillage, où les pairs extraient les données attendues d'un ensemble de partenaires (d'autres pairs ou le serveur) via un algorithme d'ordonnancement.

NetTube a été évalué par des simulations, qui montrent qu'il réduit considérablement la charge du serveur, scalable et améliore la qualité de lecture.

3.2.2 SocialTube-2014

Les auteurs dans [27] ont proposé SocialTube, une architecture P2P pour le partage de vidéos basée sur les relations sociales et l'intérêt similaire entre les pairs.
SocialTube est composé de deux algorithmes :

- Un algorithme de construction de l'overlay P2P basé sur un réseau social.
- Un algorithme de Prefetching (prélecture) qui réduit le retard de démarrage des vidéos.

Dans ce système, un serveur est utilisé pour représenter toutes les vidéos des serveurs sources, et les serveurs externes des vidéos. Les nœuds stockent les vidéos qu'ils ont regardées précédemment afin de les redistribuer aux autres utilisateurs. Une vidéo est divisée en petits blocs de taille fixe, ainsi, les nœuds qui veulent regarder la vidéo, il suffit qu'ils téléchargent les blocs correspondants aux blocs de la vidéo.

3.2.2.1 Algorithme de construction du SocialTube

Pour identifier les abonnés et les non-abonnés d'un nœud source pour la construction de la structure, SocialTube prédéfinit deux seuils, Th et Tl. Si la valeur de pourcentage d'un viewer d'une vidéo $\geq Th$ alors le viewer est un abonné. Si pourcentage $Tl<x\leq Th$. Alors, le viewer est un non-abonné.

SocialTube :

1. établit un overlay P2P pour chaque nœud source. Il se compose de pairs de moins de 2 sauts de la source qui regardent au moins un certain pourcentage (> T1) des vidéos de la source. D'autres pairs peuvent toujours récupérer des vidéos sur le serveur. Comme le montre la figure 3.2, ces pairs d'un nœud source S dans le réseau social constituent un overlay P2P pour le nœud source.
2. construit une structure hiérarchique qui connecte un nœud source à ses abonnés socialement proches, et connecte les abonnés avec d'autres non-abonnés. Ainsi, les abonnés peuvent recevoir rapidement des morceaux du nœud source, et également distribuer les morceaux à d'autres amis. La source envoie en mode push le premier morceau de sa nouvelle vidéo à ses abonnés. Le morceau est mis en cache dans chaque abonné et a une forte probabilité d'être utilisé car les abonnés regardent presque toutes les vidéos de la source. En outre, les non-abonnés partageant le même intérêt sont regroupés dans un cluster d'intérêt (interest-cluster-peers) pour le partage de vidéos. Un nœud qui a plusieurs intérêts se trouve dans plusieurs clusters d'intérêts du nœud source.

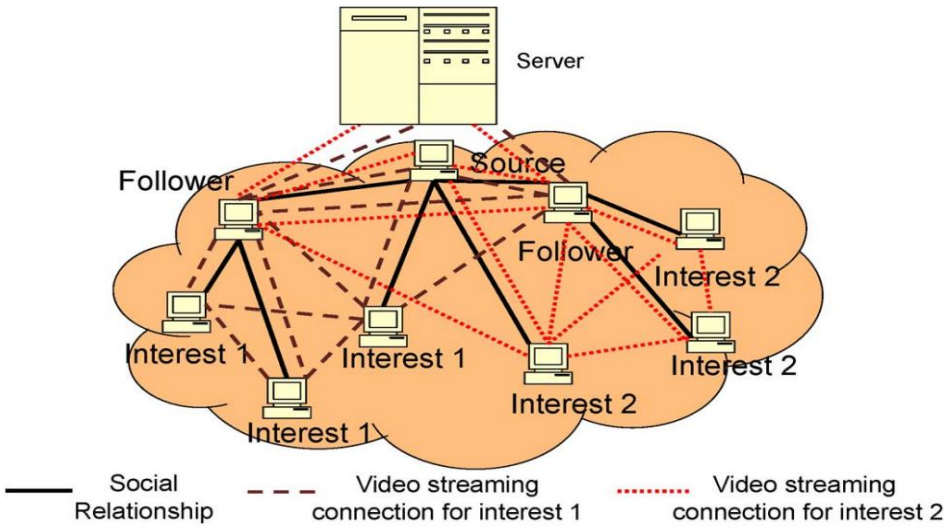

Figure 3.2 : Architecture du SocialTube.

3.2.2.2 Algorithme de prefetching (prélecture)

Pour réduire la latence de démarrage vidéo, un mécanisme de Prefetching basé sur la stratégie Push est proposé. Lorsqu'un nœud source upload une nouvelle vidéo au serveur, il envoie également le préfixe (le premier segment) de la vidéo à ses abonnés et aux pairs de groupe d'intérêt dans les groupes d'intérêt correspondant au contenu de la vidéo. Les récepteurs de préfixe stockent le préfixe dans leurs caches. Lorsque le nœud source envoie le préfixe, ces pairs de groupe d'intérêt et les abonnées qui ne sont pas en ligne le recevront automatiquement du nœud source ou du serveur une fois qu'ils seront en ligne. Après le départ du nœud source, le serveur prend la responsabilité d'envoyer le préfixe. Etant donné que ces abonnées et les pairs de groupe d'intérêt sont très susceptibles de regarder la vidéo, les préfixes mis en cache ont une forte probabilité d'être utilisé.

Une fois que les nœuds demandent les vidéos, le préfixe stocké localement peut être lu immédiatement sans délai, pendant ce temps, le nœud essaie de récupérer les morceaux vidéo restants de ses pairs swarm.

SocialTube permet à un demandeur de demander quatre nœuds en ligne en même temps pour fournir le contenu vidéo afin de garantir la disponibilité du fournisseur et d'atteindre un faible délai en récupérant des morceaux en parallèle. Il contacte d'abord les pairs d'intérêt (cluster-peers), puis les abonnées, puis le nœud source. Si le demandeur ne peut pas trouver quatre fournisseurs après que le nœud source est contacté, il recourt au serveur en tant que seul

fournisseur. Cet ordre de requêtes permet de répartir la charge de distribution de blocs entre les pairs de swarms, tout en fournissant la disponibilité élevée de blocs.

SocialTube a été évalué par des simulations, les résultats prouvent que SocialTube fournit un faible retard de démarrage de vidéo et une faible demande de trafic serveur, il offre la disponibilité des vidéos sur le réseau P2P, même si le nombre d'utilisateurs en ligne est faible.

3.2.3 SocialTube-2018

Les travaux précédents comme NetTube construisent un overlay P2P pour chaque vidéo, ce qui fournit une disponibilité limitée des fournisseurs de vidéo et produit une surcharge de maintenance de l'overlay élevée. Pour gérer ces problèmes, les auteurs dans [29] ont proposé Social-Tube, qui construit un overlay P2P pour les abonnés d'un canal (chaine) et regroupe également les nœuds d'intérêt commun à un niveau supérieur.

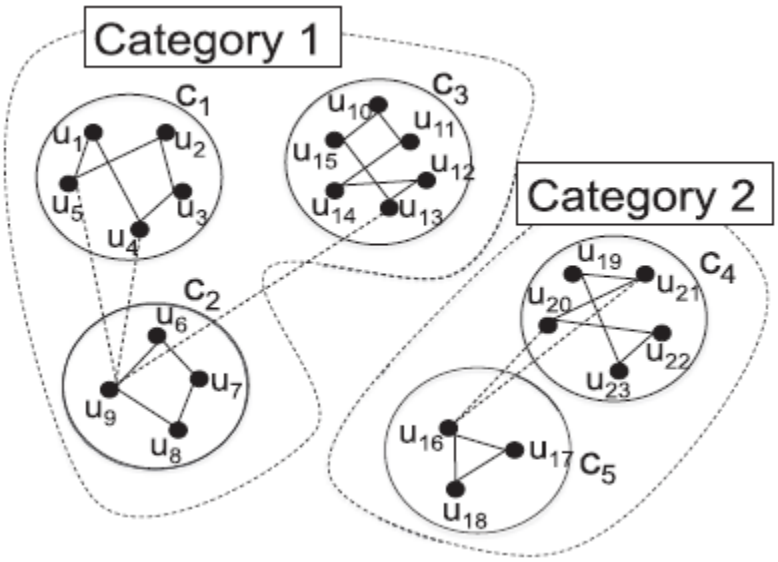

Figure 3.3 : Structure de SocialTube.

La figure 3.3 montre un exemple de la structure du réseau SocialTube. Au niveau inférieur, les nœuds de chaque canal (c'est-à-dire c1 – c5) sont formés en un overlay représenté par un petit cercle. Les canaux de la même catégorie d'intérêt (c'est-à-dire de la catégorie 1, 2) sont formés en cluster de niveau supérieur, où les nœuds de chaque canal sont connectés à travers les canaux. Étant donné qu'un abonné regarde toujours les vidéos dans sa chaîne souscrite, le regroupement des abonnés dans le même canal permet d'améliorer la disponibilité des fournisseurs de vidéo et de réduire la charge du serveur. Comme les utilisateurs ont également

tendance à regarder des vidéos dans leur intérêt, ils peuvent regarder des vidéos sur les chaînes auxquelles ils ne sont pas abonnés. La superposition de niveau supérieur regroupant tous les utilisateurs de chaînes au sein d'une même catégorie d'intérêt aide les utilisateurs à trouver des fournisseurs de vidéo.

SocialTube limite le nombre de liens d'un nœud à un seuil Nl dans sa superposition de canal de niveau inférieur, et limite le nombre de liens d'un nœud à un seuil Nh dans son cluster de canaux de niveau supérieur. Nous appelons les liens d'un nœud dans ses liens internes de superposition de canal de niveau inférieur et les liens d'un nœud dans ses interconnexions de cluster de canaux de niveau supérieur; en conséquence, les voisins connectés par les liens sont appelés voisins internes et inter-voisins,

Lorsqu'un nœud rejoint le système et recherche des vidéos dans SocialTube, comme le montre la figure 3.4, par exemple, le nœud u9 demande initialement une vidéo dans un canal, il contacte le serveur. Si le nœud s'est abonné au canal et qu'il existe un ou plusieurs nœuds dans la superposition de canal, le serveur choisit au hasard un nœud dans la superposition de canal, disons u6, auquel u9 se connecte. Le serveur choisit également au hasard un nœud dans chaque canal dans la superposition de niveau supérieur de ce canal, par exemple u4 et u13, auquel u9 se connecte. S'il n'y a pas de nœud existant dans la superposition de canal ou si u9 ne s'est pas abonné à la chaîne, le serveur choisit au hasard un nœud dans chaque superposition de canal (y compris un nœud avec la vidéo) dans la superposition de niveau supérieur de l'intérêt de la vidéo et recommande les à u9. Dans le premier cas, u9 devient également le premier nœud de la superposition de canal. Ainsi, pour les abonnés qui ne sont pas abonnés à une chaîne, SocialTube les aide toujours à localiser les fournisseurs de vidéo en utilisant la superposition de haut niveau basée sur les intérêts. Ils peuvent facilement trouver des vidéos de leurs pairs d'intérêt commun.

Après avoir rejoint le système, u9 commence à rechercher la vidéo souhaitée. Il recherche d'abord dans sa superposition de canal, puis recherche dans sa superposition d'intérêt de niveau supérieur dans les sauts TTL (Time To Live). Plus précisément, il envoie la demande d'abord à son voisin interne, u6, avec un TTL.

Si u6 n'a pas la vidéo, alors il transmet la demande à ses voisins internes (u7) dans la superposition de canal, et chaque voisin décrémente TTL et transmet la demande à ses voisins s'il n'a pas la vidéo et TTL≠0. Si un récepteur de demande à la vidéo (disons u8), alors il fournit la vidéo directement à u9. Ensuite, u9 se connecte au fournisseur vidéo et ignore les

autres réponses. De cette manière, le nœud u9 nouvellement joint construit ses liens vers d'autres nœuds dans la superposition de canal de niveau inférieur jusqu'à ce que le nombre de ses liens internes atteigne Nl. Ainsi, u9 se connecte à des nœuds qui ont tendance à regarder les mêmes vidéos que u9 plus tard.

Si u9 ne peut pas trouver sa vidéo désirée dans TTL le long des liaisons internes dans sa superposition de canal, alors il envoie sa requête à ses inter-voisins (u4 et u13). Dans chaque superposition de canal, la demande est transmise le long de sauts TTL. Si un fournisseur de vidéo est trouvé, disons u5, alors il fournit la vidéo à u9, et u9 se connecte à u5 si le nombre de ses interconnexions est inférieur à Nh. Si une vidéo ne peut pas être trouvée dans l'interrogation inter-canal, alors u9 recourt au serveur pour la vidéo. Le TTL est utilisé pour limiter les sauts de transmission des messages et donc la surcharge de recherche vidéo.

Sur la figure 3.3, en tant que nœud existant, l'utilisateur u9 regarde actuellement des vidéos dans le canal c2, il maintient donc des liens vers les utilisateurs u6 et u8 dans c2. Il maintient également des liens vers les utilisateurs u4 et u5 dans le canal c1 et u13 dans le canal c3. u9 ne maintient aucun lien vers des utilisateurs en dehors de sa catégorie. Lorsque u9 veut regarder la vidéo v1, il envoie une requête à u6 et u8. Si aucun des pairs n'a la vidéo, ils transmettent la demande à leurs voisins dans c2. Si la vidéo ne peut pas être localisée dans la superposition de canal après TTL, alors u9 envoie une requête à u4, u5 et u13 dans c1 et c3, qui transmettent la requête à leurs pairs de chaîne jusqu'à ce que TTL = 0. Si la vidéo n'a toujours pas été localisée, u9 envoie la demande au serveur.

Un nœud quitte toutes ses superpositions dans SocialTube lorsqu'il se déconnecte du système. La prochaine fois, lorsque le nœud se connecte, il essaie d'abord de se connecter à ses voisins précédents. Si aucun des voisins internes n'existe dans la superposition de canal de niveau inférieur ou aucun des inter-voisins n'existe dans la superposition de niveau supérieur, puis le nœud contacte le serveur pour créer des liens comme s'il se joignait pour la première fois au système SocialTube.

SocialTube réduit la charge du serveur et les frais généraux de maintenance de l’overlay et améliore la qualité de service pour les utilisateurs.

3.2.4 Stir

Le taux de désabonnement des pairs dans les systèmes P2P en directe est très élève, car les départs et les arrivées des utilisateurs dépendent de leurs intérêts dans la session, par exemple

les utilisateurs partent rapidement parce qu'ils s'intéressent pas au contenu, donc, la connaissance des intérêts des utilisateurs est essentielle pour gérer le désabonnement des pairs.
Les amis dans un réseau social n'ont pas toujours des intérêts similaires, ils ne peuvent pas nécessairement rejoindre ou rester longtemps dans la même session P2P.

Pour résoudre ces problèmes, Les chercheurs dans [28] ont proposé Stir, un système social de streaming P2P qui fournit des services de streaming satisfaisants et personnalisés à un grand nombre d'utilisateurs.

Stir est un système de streaming P2P multi-canal, qui fournit du contenu streaming en direct, par exemple, IPTV basé sur P2P. Chaque utilisateur de Stir dispose d'un profil contenant un identifiant unique, un mot de passe et une liste d'amis.

Un utilisateur U doit se connecter au système avant de regarder n'importe quel canal. Une fois l'utilisateur U est authentifié, le système renvoie le profil à U simultanément, le serveur de streaming envoie la liste des canaux à U. Lorsque U sélectionne un canal, le serveur envoie une liste d'adresses IP de certains pairs disponibles dans le canal à U. Maintenant, l'utilisateur U peut se connecter à d'autres utilisateurs pour le téléchargement de données.

Dans Stir, les relations d'amitiés entre les utilisateurs doivent être établit rapidement, car un groupe d'utilisateurs regardent le même canal. En regardant le canal, l'utilisateur U peut envoyer des commentaires en ligne à l'un des forums de Stir. Ce forum n'est visible que par ceux qui regardent la même chaîne avec U. U peut avoir une conversation privée avec un autre utilisateur, par exemple V, et le chat peut être entièrement effectué dans un navigateur web ou le canal a joué en direct. L'utilisateur U peut ajouter V à sa liste d'amis. La liste d'amis constitue un état qui passe d'une session à l'autre, c'est-à-dire, être dans la liste d'amis de U signifie que U connaît le statut (si V est dans le système ou non, quel canal regarde V) de V chaque fois que U se connecte au système.

Stir est un système de streaming basé sur la méthode Pull. Chaque pair extrait des données vidéo à partir des autres pairs en fonction des échanges basés sur un bufferMap. Chaque pair Stir a deux listes pour les partenaires potentiels :

- La liste des voisins.
- La liste des amis contient des amis qui regardent également la même chaîne.

Chaque élément de ces listes contient l'adresse IP et certaines données statistiques sont recueillies à partir des activités sociales des utilisateurs qui sont stockées dans le journal social. Ces données statistiques seront utilisées par le gestionnaire des partenaires et l'ordonnanceur. Le gestionnaire des partenaires sélectionne les partenaires potentiels à partir des listes pour demander des données en fonction de l'état du tampon de lecture, des facteurs sociaux et des paramètres du réseau. L'ordonnanceur ordonnance les demandes de données et les envoie aux partenaires sélectionnés. Les paquets reçus sont stockés dans le tampon de lecture et seront envoyés au lecteur lorsque la date limite de lecture sera atteinte. La figure 3.4 montre les composants et leurs interactions dans le système Stir.

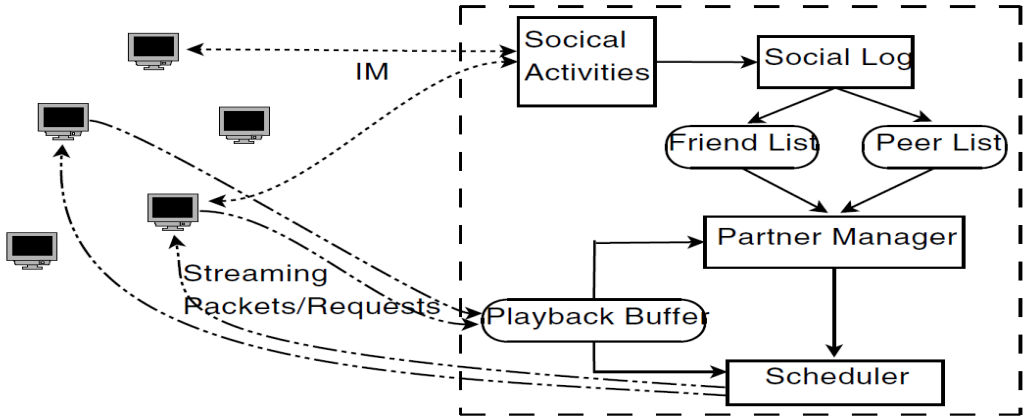

Figure 3.4 : Architecture de Stir.

L'architecture de Stir encourage les utilisateurs de faire des amis et de partager les intérêts par sa conception qui donne aux amis une plus grande priorité que les autres, indépendamment de leurs capacités sociales ou de leur bande passante.

Les résultats de simulations de Stir montrent qu'il exploite efficacement les relations sociales et offre la stabilité des pairs.

3.3 Comparaison des systèmes étudiés

Le Tableau 3.1 résume l'ensemble des systèmes présentés précédemment et donne une vue globale sur leurs différences selon un ensemble de critères choisis.

Système	**Type de streaming**	**Architecture**	**Paramètres sociaux**	**Echange de données**	**Algorithme de pre-lecture**	**Overlay P2P**
NetTube	VOD	Hybride Serveur/P2P	Intérêt similaire	Pull	Oui	Par vidéo
SocialTube-2014	VOD	Hybride Serveur/P2P	Intérêt similaire.	Push	Oui	Pour chaque nœud source
Stir	Streaming en direct.	Hybride Serveur/P2P	Relation d'amitié.	pull	Non	Non structuré
SocialTube-2018	VOD	Hybride Serveur/P2P	Intérêt commun	Pull	Oui	Par catégorie de vidéo

Tableau 3.1 : Tableau comparatif des systèmes étudiés.

3.4 Conclusion

Dans ce chapitre, nous avons étudié quelques systèmes de vidéo streaming dans les réseaux sociaux utilisant des architectures P2P, qui réduisent considérablement la charge du serveur.

Nous avons étudié les systèmes NetTube, SocialTube et Stir en décrivant les caractéristiques et le principe de fonctionnement de chaque système. Ainsi, nous avons établi une comparaison entre ces systèmes.

Le chapitre suivant sera consacré à la description de notre simulation de NetTube et notre proposition d'amélioration de l'architecture de NetTube.

Chapitre 4

Amélioration de l’architecture de NetTube

4.1 Introduction

Dans notre travail, nous nous intéressons aux architectures de partage de vidéo dans les réseaux sociaux en P2P, plus particulièrement, nous allons réaliser la simulation de NetTube afin d'évaluer ses performances par rapport à la surcharge des nœuds fournisseurs potentiel.

Dans ce chapitre, nous allons présenter quelques notions de simulation. Ensuite, nous allons implémenter le système NetTube. Puis, et pour l'étude du problème de la charge sur les fournisseurs potentiels, nous avons réalisé une série de simulation de ce système en utilisant NETBEANS 8. Enfin, nous avons proposé une idée pour traiter ce problème.

4.2 Définition de la simulation

La simulation est une technique de modélisation du modèle réel. Elle permet de représenter le fonctionnement d'un système composé de différents centres d'activités, de mettre en évidence les caractéristiques de ceux-ci et les interactions entre eux, de décrire la circulation des différents objets traités par ces processus et d'observer le comportement du système dans son ensemble et son évolution dans le temps.

4.3 Etapes d'une simulation

Les étapes à suivre pour entreprendre une simulation d'un système sont les suivantes:

- **Formulation du problème** : cette étape consiste principalement à identifier et analyser le problème, en déterminant ses composantes, et les frontières entre le système et son environnement.
- **Elaboration du modèle :** cette étape consiste à extraire un modèle aussi fidèle que possible du système réel, dans le but d'expliquer et de prédire certains aspects de son comportement.
- **Identification et collecte des données :** la phase de l'identification du type de données à entrer dans le modèle est une phase très délicate et essentielle. La collecte des données est indispensable pour l'estimation des paramètres du modèle. Ceci requiert une connaissance des méthodes statistiques et des tests d'hypothèses.
- **Exécution de la simulation :** le concepteur doit pouvoir mettre à l'épreuve le modèle en agissant sur les paramètres qui le configurent. Il s'agit d'effectuer plusieurs exécutions et de recueillir les résultats obtenus.

- **Validation du modèle :** cette étape consiste à évaluer les performances du modèle en les comparants à celles du système réel.
- **Analyse et interprétation des résultats :** une fois les résultats obtenus, le concepteur passe à l'analyse et à l'interprétation de ces résultats pour donner des recommandations et des propositions.
- **Conclusion et exploitation du modèle :** cette étape consiste à évaluer les perspectives d'exploitation du modèle pour d'autres préoccupations.

4.4 Outils de simulation

Afin d'évaluer les performances de l'architecture de NetTube, nous avons utilisé l'environnement de développement Netbeans 8.2 qui est basé sur le langage de programmation JAVA.

NetBeans 8.2 : est un environnement de développement intégré (IDE) pour Java qui permet de développer des applications à partir d'un ensemble de composants logiciels modulaires appelés modules. NetBeans fonctionne sous Windows, macOS, Linux et Solaris. En plus du développement Java, il a des extensions pour d'autres langages comme PHP, C, C ++, HTML5 et JavaScript. Les applications basées sur NetBeans, y compris l'EDI NetBeans, peuvent être étendues par des développeurs tiers.

Langage JAVA : c'est un langage de programmation orienté objet, développé par Sun Microsystems, il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage peut être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur (jsp).

4.5 Simulation de NetTube

Pour la simulation du système NetTube, nous avons utilisé un ensemble de paramètres qui est représenté par un ensemble de pairs, un serveur et un ensemble de vidéos distinctes. Le débit de streaming minimum est de 300 kb/s pour prendre en charge une diffusion fluide.

Ces paramètres sont représentés dans le tableau 4.1.

paramètre	**Valeur**
Débit	300 kb/s
Le nombre de nœuds	15 nœuds
Le nombre de vidéos	10 vidéos

Tableau 4.1: Paramètres de simulation

Les nœuds possèdent un identificateur (P1, P2, P3…. P15) et une adresse IP. Ainsi, les vidéos aussi possèdent un identificateur (v1, v2, v3, ….v10).

La figure 4.1 représente l'interface de la simulation qui est composé par un espace où on peut visualiser le déroulement de la simulation et un ensemble de boutons :

Nombre de nœuds (cercle rouge) : pour la précision de nombre de pairs utilisés dans la simulation.

Nombre de données (cercle bleu) : pour la précision de nombre de données utilisés dans la simulation.

Vitesse de simulation : pour controler la vitesse de déroulement de la simulation.

Le bouton démarrer/redémarrer : pour construire l'architecture du systéme.

Le bouton démarrer simulation : pour démarrer la simulation.

Le bouton graphe des liens : pour afficher la destribution des liens dans l'architecture.

Nombre de liens (cercle vert) : pour calculer le nombre des liens utilisés dans la simulation.

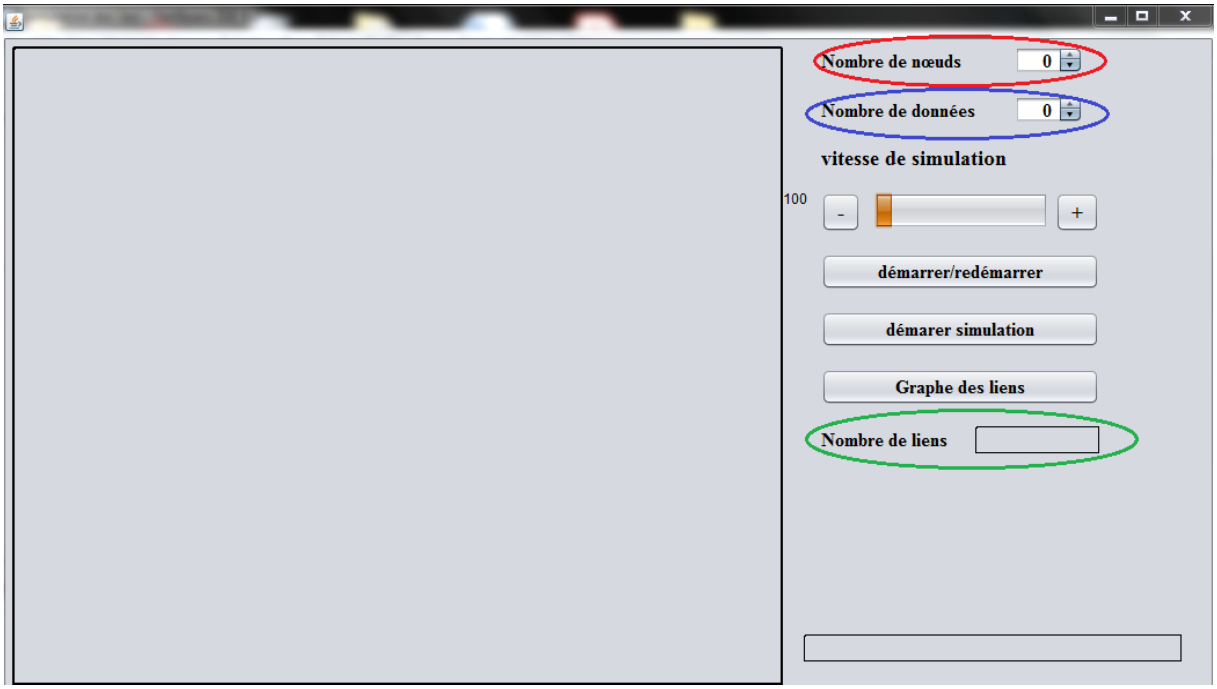

Figure 4.1 : L'interface de la simulation

- **Déroulement de la simulation**

Selon le principe de fonctionnement du système NetTube, les nœuds P1, P2, P3 et P4 sont des nouveaux nœuds qui veulent respectivement regarder les vidéos v4, v5, v2 et v3. Alors, ils envoient des demandes au serveur, ce dernier fournit les vidéos voulues et enregistre leurs adresses IP.

Maintenant, les nœuds P5, P12 et P14 veulent regarder la vidéo v4, ils envoient des demandes au serveur. Ce dernier, les oriente vers le nœud P1, puisque ce dernier a déjà téléchargé la vidéo à partir du serveur et devient donc un fournisseur. Alors, le nœud P1 leurs fournit la vidéo v4.

Aussi, les nœuds P6, P7, P9, P11, P13 et P15veulent regarder la vidéo v4, ils envoient des demandes au serveur. Ce dernier les oriente vers le nœud P2, puisque ce nœud a déjà téléchargé la vidéo à partir du serveur et devient un fournisseur. Alors, le nœud P2 leurs fournit la vidéo 5.

Le nœud P8 veut regarder la vidéo v2, il envoie une demande au serveur. Ce dernier l'oriente vers le nœud P3 puisque ce nœud a déjà téléchargé la vidéo à partir du serveur et devient un fournisseur. Alors, le nœud P3 lui fournit la vidéo v2.

Le nœud P10 veut regarder la vidéo v3, il envoie une demande au serveur. Ce dernier le orienté vers le nœud P4 puisque ce nœud a déjà téléchargé la vidéo à partir du serveur et devient un fournisseur. Alors, le nœud P4 lui fournit la vidéo v3.

Ce déroulement est montré dans la figure 4.2

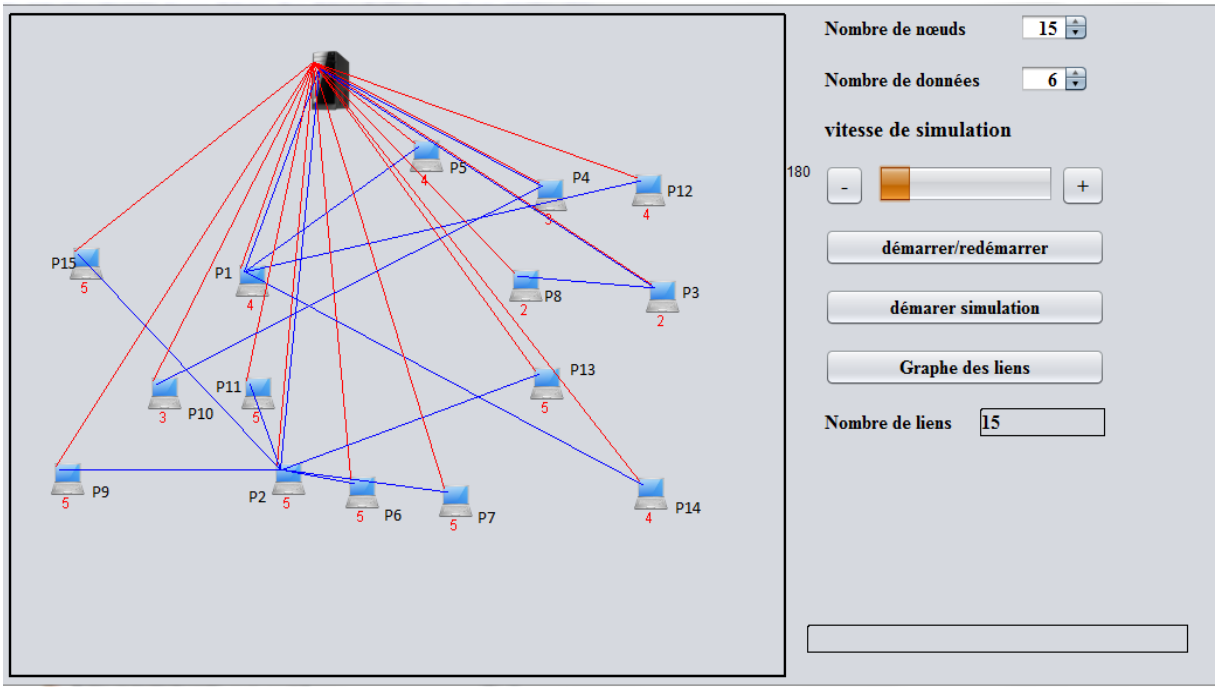

Figure 4.2 : Le déroulement de la simulation (NetTube).

Selon les résultats de la simulation, le serveur a reçu 15 demandes de téléchargement des vidéos et il a fourni 4 vidéos. Donc, on a constaté que le principe de fonctionnement de NetTube sert à réduire la charge du serveur comme un principal objectif.

Mais, on voit que les fournisseurs potentiels sont les seuls donneurs des vidéos après le serveur, et leurs voisins sont que des consommateurs qui prend les vidéos sans les partagé aux autres pairs.

La figure 4.3 montre le graphe des demandes des vidéos par rapport au nombre de vidéos fournit au niveau du fournisseur potentiel P2.

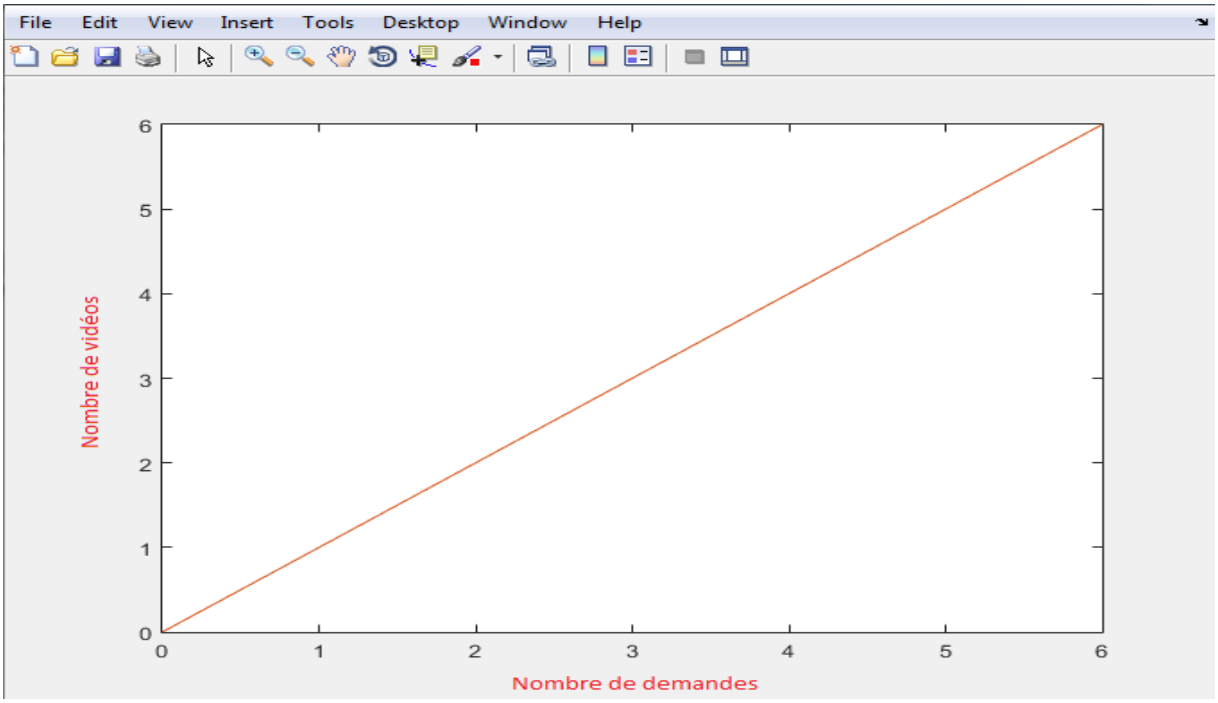

Figure 4.3 : Graphe des nombre de vidéos fournit par rapport au nombre de demandes reçues au niveau de p2 (NetTube).

D'après ces résultats, on remarque que le fournisseur potentiel P2 a reçu 6 demandes de téléchargement des vidéos et il a fourni 6 vidéos. Donc, on distingue que les fournisseurs potentiels peuvent souffrir de la charge avec l'augmentation de nombre des demandes.

4.6 Proposition

Pour la résolution du problème de la surcharge des nœuds fournisseurs potentiel, nous proposons une approche qui consiste en :

1- La limitation de nombre de vidéos fournit par les fournisseurs potentiels.
2- Si le nombre des vidéos fournit par le fournisseur potentiel égal au nombre maximal des vidéos, le fournisseur potentiel oriente le demandeur vers un de ces voisins d'une façon aléatoire.

Dans cette proposition, nous limitons le nombre de vidéos qui seront fourni par les fournisseurs potentiels a un nombre Nl vidéos, donc, si le nœud fournisseur potentiel reçoit

un nombre $nb > Nl$ de demandes, ce dernier, oriente le demandeur correspondant vers un de ces voisins aléatoirement.

La figure 4.4 montre le déroulement de la simulation.

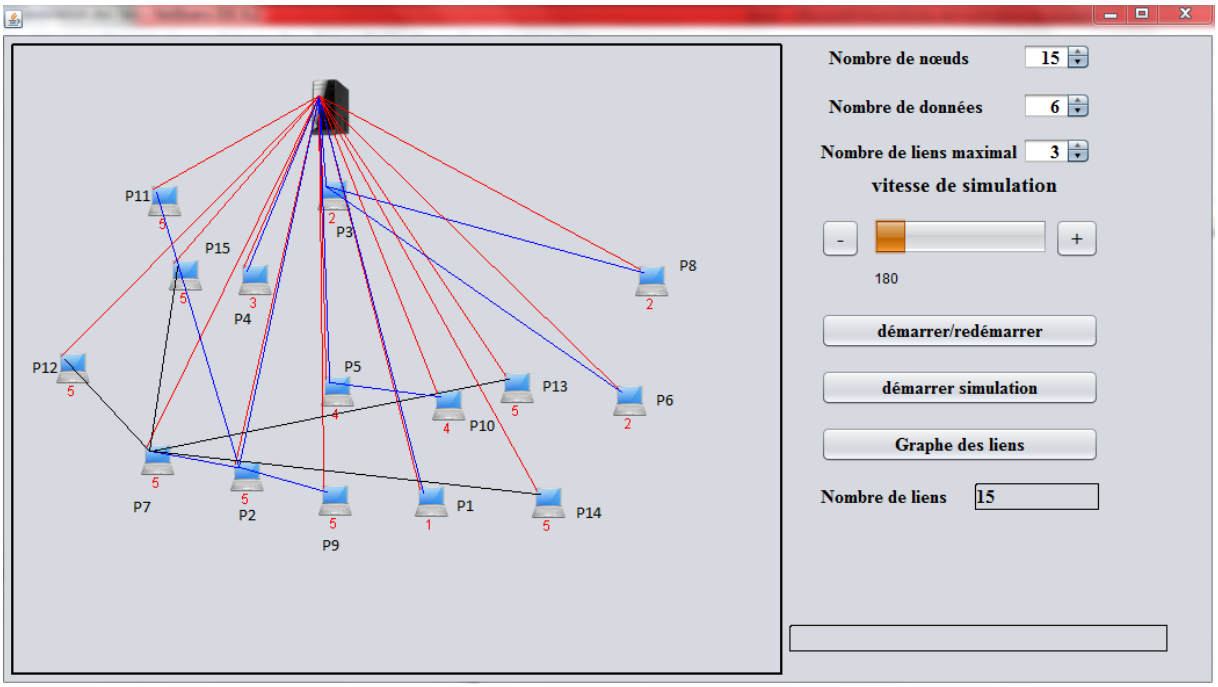

Figure 4.4 : Déroulement de la simulation (notre proposition).

- **Déroulement de la simulation**

Les nœuds P1, P2, P3, P4 et P5 sont des nouveaux nœuds qui veulent respectivement regarder les vidéos v1, v2, v3, v4 et v5. Alors, ils envoient des demandes au serveur et ce dernier, leurs fournit les vidéos voulues et enregistre leurs adresses IP.

Les nœuds P6 et P8 veulent regarder la vidéo v5, ils envoient des demandes au serveur. Ce dernier, les oriente vers le nœud P2, puisque ce dernier a déjà téléchargé la vidéo à partir du serveur. Alors, le nœud P2 leurs fournit la vidéo v5.

Les nœuds P7, P9 et P11 veulent regarder la vidéo v2, ils envoient des demandes au serveur. Ce dernier les oriente vers le nœud P3 puisque ce dernier a déjà téléchargé la vidéo à partir du serveur. Alors, le nœud P3 leurs fournit la vidéo.

Le nœud P10 veut regarder la vidéo v4, il envoie une demande au serveur. Ce dernier, l'oriente vers le nœud P5 puisque ce nœud a déjà téléchargé la vidéo à partir du serveur. Alors, le nœud P5 lui fournit la vidéo v4.

Les nœuds P12, P13, P14 et P15 sont des nouveaux nœuds qui veulent respectivement regarder la vidéo v5. Alors, ils envoient des demandes au serveur et ce dernier les oriente vers le fournisseur P2, ce dernier de son tour les oriente vers son voisin P7 car il a déjà arrivé au nombre maximal des vidéos fournit. Donc, le voisin P7 fournit la vidéo v5 à ces demandeurs.

La figure 4.5 montre le graphe des demandes des vidéos par rapport au nombre de vidéos fourni au niveau du fournisseur potentiel P2.

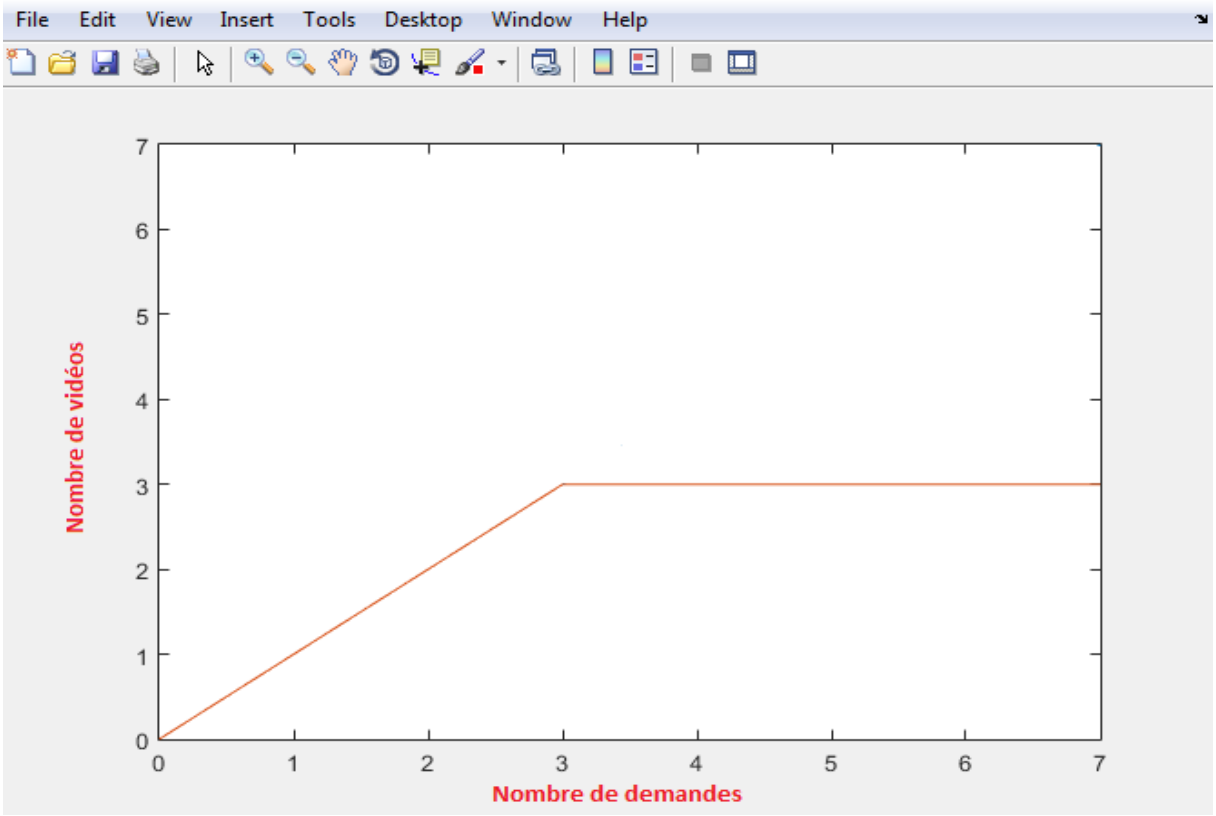

Figure 4.5 : Graphe de nombre de vidéos fournit par rapport au nombre de demandes reçues au niveau de p2 (notre proposition).

D'après ces résultats, on remarque que le fournisseur potentiel P2 a reçu sept demandes de téléchargement des vidéos et il a fourni que trois vidéos. Donc, ils prouvent que la charge au niveau des fournisseurs potentiels est réduite même si le nombre des demandes des vidéos sur lui augmente.

4.7 Conclusion

Dans ce chapitre, nous avons réalisé la simulation du système NetTube afin d'évaluer les performances de celui-ci par rapport à la surcharge des nœuds fournisseurs potentiel. Puis nous avons proposé une approche qui permet de réduire la surcharge des nœuds fournisseurs potentiels en limitant le nombre de vidéos à fournir.

Conclusion générale

Le pair à pair (P2P) est une nouvelle technologie réseau qui est apparue pour but de remédier les problèmes lié à l'utilisation de l'architecture client/serveur. Ces dernières années, les réseaux P2P sont devenus très populaires. Cette popularité vient des bonnes caractéristiques offertes par ces systèmes, et grâce à leurs caractéristiques avantageuses, de nouveaux domaines se dirigent vers l'utilisation de ces réseaux dans de nouvelles applications.

Dans ce mémoire, nous nous sommes intéressés aux systèmes de media streaming dans les réseaux sociaux en P2P. Nous avons réalisé une partie théorique où nous avons présenté un état de l'art sur les réseaux P2P, ensuite nous avons réalisé un état de l'art sur le media streaming dans les réseaux P2P, enfin, nous avons présenté quelques systèmes de media streaming dans les réseaux sociaux P2P que nous avons comparés selon un ensemble de critères choisis.

Nous nous somme par la suite intéressé à réaliser la simulation du système NetTube afin de démontrer l'un des problèmes qui peut faire face au bon fonctionnement de celui-ci, est qui est représenté par « la surcharge au niveau des nœuds fournisseurs potentiels ». Puis, nous avons proposé une approche permettant la résolution de ce problème ainsi l'amélioration de l'architecture NetTube.

Nous envisageons comme perspective d'établir une étude du système NetTube en ce qui concerne le mécanisme de prefetching (prélecture).

Bibliographie

[1]: S. Androutsellis-Theotokis et D. Spinellis. A survey of peer-to-peer content distribution technologies. ACM Computing Surveys, 2004.

[2]: H.Ragib, A.Zahid, W.Yurcik, L.Brumbaugh, and Roy H. Campbell, A survey of peer-to-peer storage techniques for distributed file systems., ITCC (2)'05, 2005.

[3]: M.Amad. Performance et optimisation des architectures P2P pour les applications à grande échelle. Thèse de doctorat. Université A Mira de Béjaia, 2012.

[4] : A.Chaouche. Gestion de la qualité de service pour le streaming de contenu audiovisuel sur les réseaux Pair-à-Pair. Mémoire de magister. École Nationale Supérieure d'Informatique. Algérie, 2010.

[5] : H.Hafi. Protocole pour la sécurité des réseaux sans fil Peer to Peer. Université KasdiMerbah-Ouargla, 2010.

[6] : Y.MOUALKIA. Performances et optimisation des architectures P2P pour les applications des réseaux sociaux. Mémoire de magister. Université A Mira de Bejaia, 2016.

[7]**:** C.Shirky, Peer-to-peer : Harnessing the Power of Disruptive Technologies, chapterListening to Napster, pp. 21–37, O'Reilly& Associates, 2001 .

[8]: G.Kan, Gnutella, and GoneSilent.com, Peer-to-peer: Harnessing the Power of Disruptive Technologies. ChapterGnutella, pp. 94–122, O'Reilly& Associates, 2001.

[9] : T.Cholez. Supervision des réseaux pair à pair structurés appliquée à la sécurité des contenus. Thèse doctorat de l'université Henri Poincaré. Nancy 1. 2011.

[10]: J. Buford, H. Yu, and E. K. Lua. P2P Networking and Applications. Morgan Kaufman 2008, Publishers Inc., San Francisco, CA, USA, 2008.

[11]: B. Cohen, Bittorrent, “http://www.bittorrent.org/”, 2010 .

[12]: K.Hareesh et D.H. Manjaiah. Peer to peer live streaming and video on demand design issues and its challenges. International Journal of Peer to Peer Networks (IJP2P) Vol.2, No.4, October 2011.

[13]: A.Kara. Le développement d'un système embarqué implémentant le Peer-to-Peer pour la TVoIP. Mémoire de magister. Université Ferhat Abbas-Sétif, Algérie.

[14] : M.NAFA. Optimisation des applications de streaming peer to peer pour des réseaux ad hoc mobiles. Thèse de doctorat. Université d'Évry-Val-d’Essonne, 2009.

[15] : S.Banerjee, B.Bhattacharjee, and C.Kommareddy, “Scalable application layer multicast”, in ACM Sigcomm, 2002.

[16] :H.Deshpande, M.Bawa, and H.Garcia-Molina, “Streaming Live Media over a Peer-to-Peer Network”, Technical Report Stanford 2001.

[17] :J.Zhang, L.Liu, and L.Ramaswamy, PeerCast : “Churn-Resilient End System Multicast on Heterogeneous Overlay Networks”, Journal of Network and Computer Applications (JNCA), Elsevier, 2007.

[18] :Y.H.Chu, S.G.Rao, and H.Zhang, “A case for end system multicast” in Proceedings of ACM Sigmetrics 2002.

[19] : J. Jannotti, D.Gifford, K. Johnson, M. Kaashoek, and J.O’Toole, Overcast : “Reliable multicasting with an overlay network”, in Porceedings the Fourth Symposium on Operating Systems Design and Implementation, pp. 197–212, 2000.

[20] : M.Castro, P.Druschel, A.M.Kermarrec, A.Nandi, A.Rowstron and A.Singh, SplitStream : “ High-bandwidth multicast in a cooperative environment”, SOSP'03, Lake Bolton, New York, October, 2003.

[21] : N. Vratonjic, P. Gupta, N. Knezevic, D. Kostic, and A. Rowstron, “Enabling DVD-like features in P2P Video-on-demand Systems”, in ACM SIGCOMM 2007 Data Communications Festival, Peer-to-Peer Streaming and IP-TV Workshop (P2P-TV’07), Kyoto, Japan, 31 August 2007.

[22] : V.N. Padmanabhan, H.J. Wang, P.A. Chou, and K. Sripanidkulchai, “Distributing streaming media content using cooperative networking”, in Proceedings ACM NOSSDAV, pp. 177–186, Miami Beach, May 2002.

[23] : N. Magharei and R. Rejaie, PRIME : “Peer-to-Peer Receiver-driven MEsh-based Streaming”, in proceedings of IEEE Infocom pp. 1415–1423, Anchorage, Alaska, May 2007.

[24] : X. Zhang, J. Liu, B. Li and T-S. P. Yum, CoolStreaming/DONet : A Data-Driven Overlay Network for Efficient Live Media Streaming, in Proceedings of IEEE Infocom’05, Miami, FL, USA, March 2005.

[25]: X. Cheng and J. Liu. Exploring interest correlation for peer-to-peer socialized video sharing. ACM Trans. Multimedia Comput.Commun. Appl. 8, 1, Article 5 (January 2012).

[26]: B. Bloom, Space/Time Trade-offs in Hash Coding with Allowable Errors. Communications of the ACM, 1970.

[27] :H. Shen, Z. Li, Y. Lin and J. Li. SocialTube: P2P-Assisted Video Sharing in OnlineSocial Networks. IEEE transactions on parallel and distributed systems, September 2014.

[28] :A. T. Nguyen, B. Li, M. Welzl and F. Eliassen. Stir: Spontaneous Social Peer-to-Peer Streaming. In Proceedings of the 2011.IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS).

[29]: H. SHEN, H. CHANDLER and H. WANG. Toward Efficient Short-Video Sharing in the YouTube Social Network. ACM Transactions on Internet Technology, Vol. 18, No. 3, Article 33. Publication date: March 2018.