MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE ABDELHAMID IBN BADIS DE MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique
Département de Mathématiques et d'Informatique
Filière Informatique

MEMOIRE DE FIN D'ETUDES
Pour l'Obtention du Diplôme de Master en Informatique
Option : Ingénierie des Systèmes d'Information

ETUDE ET ANALYSE DES GRANDS GRAPHES D'INTERACTION ET EXPLOITATION DES SYSTEMES P2P POUR LA RECHERCHE DE DONNEES

Etudiants :
LARIBI Zineb.
REBAI Amina.

Encadrant(e) :
KENNICHE Ahlem.

Année Universitaire 2014/2015

Sommaire

Résumé

Le paradigme pair-a-pair (p2p), qui considère que tous les utilisateurs d'un système sont égaux, est principalement utilisé pour partager des données et les échanges. Certains systèmes p2p sont complètement distribues et jouent le rôle de client et de serveurs en même temps.

Dans ce mémoire, nous avons présenté la théorie des grands graphes d'interaction, les principes du phénomène p2p, plus les outils de mesure et d'analyse. Notre travail consiste principalement à l'analyse d'un réseau conçu selon le modèle des graphes d'interactions scall-free et l'étude de certaines propriétés de celui-ci.

Mots Clés : *P2P, graphe d'interaction, systèmes décentralisés, réseau scall-free*

Introduction générale

Ces dernières années, les systèmes pair-à-pair (désormais P2P) sont devenus très populaires. Cette popularité vient des bonnes caractéristiques offertes par ces systèmes comme : la grande échelle, l'autonomie des nœuds et le contrôle décentralisé. Ce type de systèmes offre une bonne opportunité pour répondre aux limites des systèmes basés sur le paradigme Client/Serveur. En évitant d'éventuels goulets d'étranglement et en offrant une bonne tolérance aux pannes, les systèmes P2P sont bien adaptés aux environnements distribués à grande échelle dans lesquels les nœuds (appelés indifféremment nœuds ou pairs) peuvent partager leurs ressources (e.g. puissance de calcul, capacité de stockage, bande passante) d'une façon autonome et décentralisée.

Chaque nœud peut jouer le rôle d'un serveur en offrant ses ressources aux autres nœuds, d'un client en consommant les ressources des autres nœuds existants dans le système, d'un routeur en propageant les requêtes et les messages venant des autres nœuds et d'un hôte1 de source de données en partageant ses propres données avec d'autres nœuds.

Problématique et objectif :

Plusieurs problèmes apparaissent dans les systèmes peer to peer nous nous intéressons dans notre cas aux systèmes de recherches d'informations et de partage de fichiers peer to peer. En effet ses systèmes sont très pauvres en fonctionnalités d'indexation et de recherche d'information. L'objectif et d'arrivé à une recherche aussi efficace que dans les systèmes client/ serveur.

Beaucoup de travaux se font dans ce sens mais le problème principale et qu'il n'existe pas jusqu'à maintenant une plate forme de simulation qui peut permettre d'évaluer ses systèmes vue la difficulté de concevoir des systèmes réel et dynamique tél que le paradigme P2P.

L'objectif de notre travail est d'implémenté un réseau peer to peer en utilisant les propriétés des graphes d'interactions. En effet, ses graphes partage des propriétés intéressante avec les réseaux peer to peer, c'est pour cela que nous allons modéliser notre réseau en utilisant les graphes d'interactions, et étudié les propriétés de ce

graphe. Donc notre travail porte sur l'aspect physique du réseau (construction de réseau) et ne traite pas l'aspect de recherche d'information.

Ce mémoire est structuré en trois chapitres :

- Dans le premier chapitre, nous précisons les notions générales des systèmes pair-à-pair : définitions, différentes applications de la technologie, architectures p2p.
- Dans le deuxième chapitre, nous présentons les graphes d'interaction, nous présenterons leurs différentes caractéristiques et propriétés, les modèles existants et nous terminerons par une étude comparative entre les différents modèles.

- Nous présentons dans le chapitre trois, le modèle de graphe d'interaction que nous proposons pour l'implémentation du réseau p2p ainsi que quelques vue de notre application.

Nous terminerons ce mémoire par une conclusion générale.

Liste des figures

Liste des tables

I.1. Introduction:

Dans ce chapitre, nous allons décrire les différents concepts des systèmes P2P. Nous y fournissons leur définition, les motivations d'utilisation des ces systèmes et leurs principales caractéristiques. Ensuite, nous détaillerons les différentes classes des systèmes P2P, et la recherche d'information sur les systèmes P2P, ainsi que les avantages et les inconvénients des systèmes P2P.

I.2. Présentation des systèmes pair-à-pair :

I.2.1. Historique et Définitions [3] :

Historiquement, le P2P est devenu très populaire en 1999 aux Etats Unis, avec le logiciel de partage de musique en ligne Napster [Napster]. Rapidement, ce dernier a connu un grand problème de copyright pour les compagnies et les éditeurs de disque, ce qui a mené Napster à des poursuites judiciaires en 2000.

Ce problème a été la cause principale de l'apparition des systèmes P2P totalement décentralisés, pour rendre le contrôle judiciaire plus difficile. Cependant, il y a des nombreuses utilisations légales qui sont aujourd'hui impliquées dans le monde du P2P, mais il est dur d'interdire l'échange de fichiers illégaux (musique, films...).

Définition [10]: Le pair-à-pair (en anglais, peer-to-peer, abrégé P2P) est un modèle de réseau informatique proche du modèle client-serveur. À la différence du modèle client-serveur où un seul gros ordinateur (serveur) dessert de l'information à de nombreux terminaux (clients), dans le modèle P2P, chaque client est aussi un serveur. Tous les ordinateurs récupèrent de l'information et resservent l'information obtenue.

I.2.2. Caractéristiques des systèmes P2P [3] :

Comme nous allons le présenter plus tard dans ce chapitre, il existe plusieurs types des réseaux P2P avec plusieurs architectures, protocoles et mode de fonctionnement. Cependant, quelques caractéristiques sont communes à tous ces systèmes, les plus importantes sont :

- Un système P2P doit être constitué d'au moins de deux pairs.

- Les nœuds d'un réseau P2P sont généralement de nature volatiles (les pairs peuvent joindre ou quitter le réseau en toute liberté).
- Un point ou plusieurs nœuds centralisés peuvent jouer le rôle des serveurs dédiés dans un système P2P, selon la nature de l'application. Ces nœuds sont connus sous le nom de « Super-Pairs ». Les systèmes P2P qui ne contiennent pas de serveurs dédiés sont dits systèmes P2P purs.
- Les pairs peuvent appartenir à différents propriétaires. Dans le cas de « clusters », un pair peut être membre de plusieurs groupes en même temps.

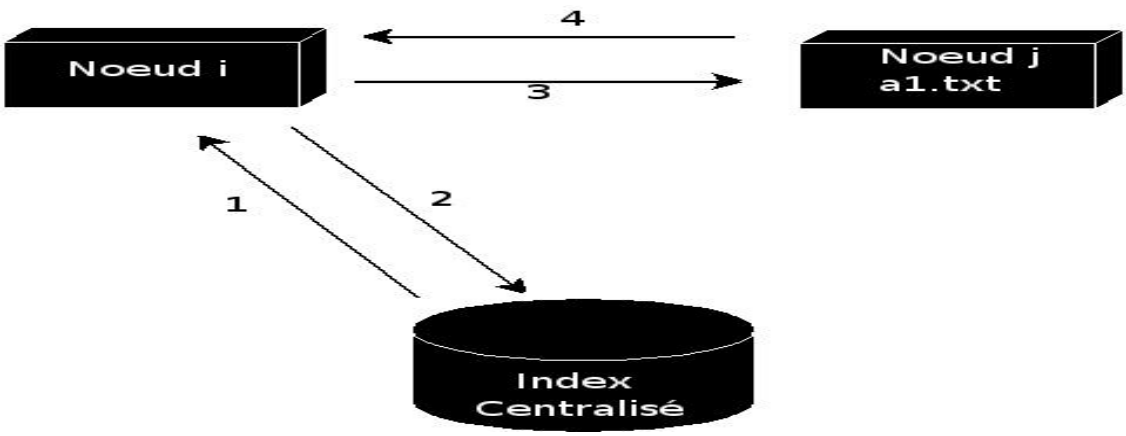
I.3. Les grandes classes de systèmes pair-à-pair :

I.3. 1. Classe non structuré [2] :

Le principe de résolution de requêtes y est le suivant : un client soumet sa requête à un serveur quelconque du réseau (le plus proche de lui par exemple), celui-ci résout la requête localement et la propage récursivement à un certain nombre de ses voisins (aléatoirement choisis). La recherche s'arrête quand une certaine profondeur de recherche a été atteinte. Cette approche est bien sûr très inefficace et conduit à générer un grand nombre de messages. Elle fonctionne assez bien lorsque les ressources recherchées sont fréquemment les mêmes. Nous présentons deux exemples des systèmes les plus représentatifs de la classe non structuré :

1. Napster [1]:

Napster Etait une plate forme d'échange de fichiers musicaux de type mp3 crée par Shawn Fanning et Sean Parker en 1999 dans laquelle chaque pair pouvait mettre à disposition le contenu de sa bibliothèque musicale. Dans cette architecture, chaque pair envoie la liste de ses fichiers à un serveur qui stocke l'association *adresse du pair* ↔ *données + métadonnées*. Lorsqu'un pair recherche un nom de fichier, il envoie une requête sous forme de mots clés au serveur d'index auquel il est connecté. Ce dernier lui retourne une liste de n-uplets *(ip, fichier, métadonnées)*. Une fois que l'utilisateur a choisi la donnée *fichier i* qui l'intéresse, l'échange se fait directement entre le pair demandeur et l'hébergeur *ip i*. Quand le fichier est téléchargé, le pair demandeur devient à son tour un hébergeur qui est indexé dans le serveur central pour cette donnée (**Fig.I.1**).



1. Enregistrement du noeud i. Ajout des fichiers de i dans l'index.
   Demande du fichier "a.txt".
2. Réponse contenant les noeuds disposant de "a.txt".
3. Demande directe de i à j pour télécharger "a.txt".
4. Téléchargement de "a.txt", et ajout dans la base de ressources partagées.

Fig.I.1. Exemple de P2P Napster [8].

2. **Gnutella [6] :** est un protocole décentralisé développé en mars 2000 par Tom Pepper et Justin Fränkel et initialement conçu pour le partage de fichiers. Il a pris la suite en résolvant le problème de centralisation de Napster. Lors d'une recherche, le nœud envoie la requête à ses voisins, qui font de même et ainsi de suite, jusqu'à atteindre les nœuds à distance 7 du demandeur. Les types de requête sont présentés dans le Tab.I.1.

Types des requêtes [5] :

Tab.I.1.Les requêtes Gnutella [5].

Type	Description	Information
Ping	Annonce la disponibilité, et lance une recherche de pair	Vide
Pong	Réponse à ping.	IP et de port .Nombre et taille des fichiers partagés.
Query	Requête	Bande passante minimum demandée. Critères de recherche.
QueryHit	Réponse à query si on possède la ressource.	IP+N° de port +Bande Passante. Nombre de réponses + descripteurs réponses.
Puch	Demande de téléchargement pour les pairs dernière un firewall.	IP du pair, index du fichier demandé. Adresse IP + N° de port où envoyer le fichier.

***Mécanisme de routage* [5]**:

Un nœud se connecte sur le réseau Gnutella, il commence par rechercher tous les nœuds Gnutella présents, pour cela il transmit une trame d'identification (***ping***) à tous ses voisins qui eux-mêmes le transmettons à leurs voisins. Ces envois sont encapsulés dans une trame TCP. Le mécanisme de recherche joue sur le TTL (***Time To Live***) de la trame pour borner devient le nombre de sauts des messages (**Fig.I.2**).A chaque nœud du réseau, le TTL est décrémenté et lorsqu'il égal à zéro, la retransmission est stoppée. Un mécanisme permet d'éviter les boucles causées par les transmissions successives, lorsqu'une trame est reçue, elle est stockée pendant un court laps du temps. Si le nœud reçoit pendant ce laps du temps une trame identique, il la rejette, car elle est déjà traitée. Lorsqu'un nœud est identifié, il envoie à l'émetteur une trame de réponse (***pong***).

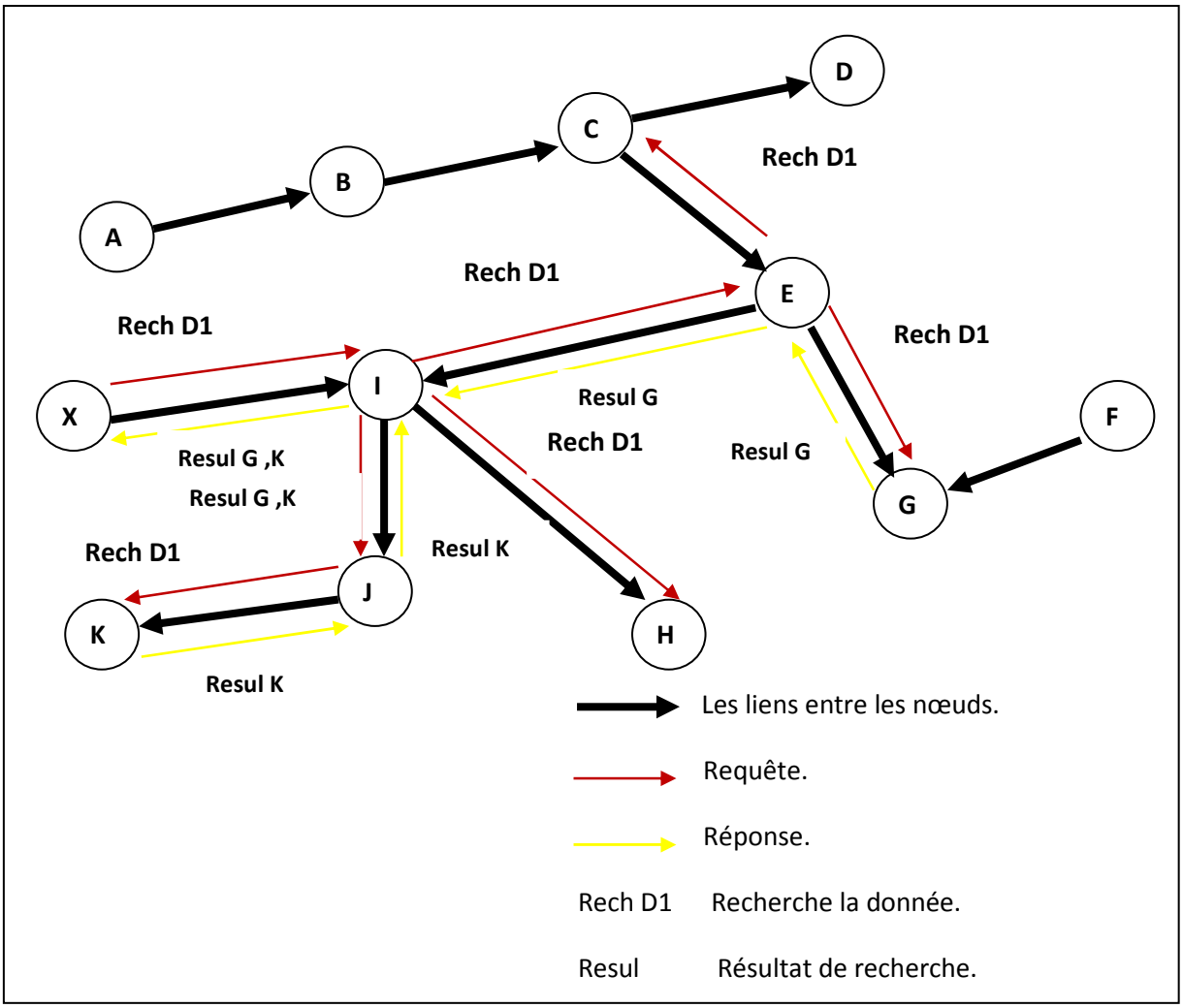


Fig.I.2. Mécanisme de routage dans Gnutella [5].

Recherche de données* [5]*:

Le nœud qui cherche la donnée transmit sa requête vers ses voisins, 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***) (**Fig.I.3**).

Dans l'exemple décrit sur la **Fig.I.3**, le nœud *X* recherche la donnée *D1*, il envoie une requête *Query* à tous ses voisins, après l'expiration de TTL (*TTL = 0*), la donnée est trouvée dans les nœud *K* et *G*, ces derniers répondent par des *Query Hit*, le nœud *X* envoie un *Push* aux deux nœuds *K* et *G* en utilisant leurs adresses IP retournées dans *Query Hit*.

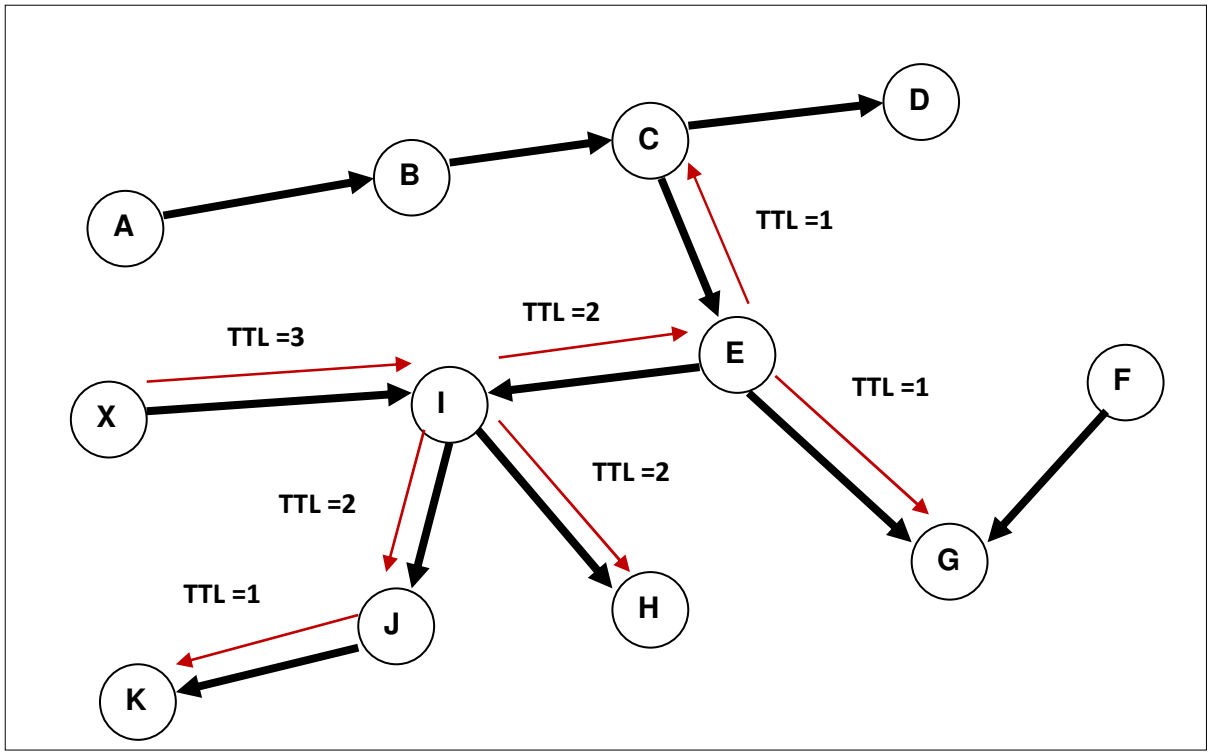


Fig.I.3. Le mécanisme de recherche dans Gnutella [5].

I.3.2. Classe hiérarchique [2] :

Le réseau est hiérarchisé en distinguant deux catégories de nœuds, les nœuds « puissants » (en terme de cpu, de bande passante) et les nœuds « faibles ». Les nœuds faibles sont regroupés et sont associés à un nœud puissant. Seuls ces derniers fonctionnent en mode P2P, alors qu'à l'intérieur d'un groupe il s'agit d'un mode client-serveur classique, voir la Fig.I.4.

Exemple :

Kazaa[5]: est une application P2P basée sur le réseau FastTrack. Niklas Zennström fondateur de Kazaa, est également développeur de Skype. Kazaa fut le premier logiciel à offrir une décentralisation de serveur. Les fichiers partagés sont stockés dans un dossier spécialisé, ce dossier étant sur l'ordinateur de chaque utilisateur.

I.3. 3. Classe structuré [2] :

Améliore la fonction de recherche en organisant l'espace de stockage selon une structure d'ordre et en utilisant cette structure pour placer les ressources sur les nœuds (on parle également de Distributed Hash Table ou **DHT**). Plusieurs algorithmes comme Chord (Stoica et al. 2001), **CAN** (Ratsanamy et al. 2001), P-Grid (Aberer 2001), Pastry (Rowstron et al. 2001) ont été proposés qui diffèrent entre eux selon la structure d'ordre choisi

(anneau, espace multi-dimensionnel, arbre binaire, …). Ces solutions amènent l'efficacité mais au prix de la perte d'autonomie des pairs puisque le placement des ressources est décidé par le système et non les utilisateurs.

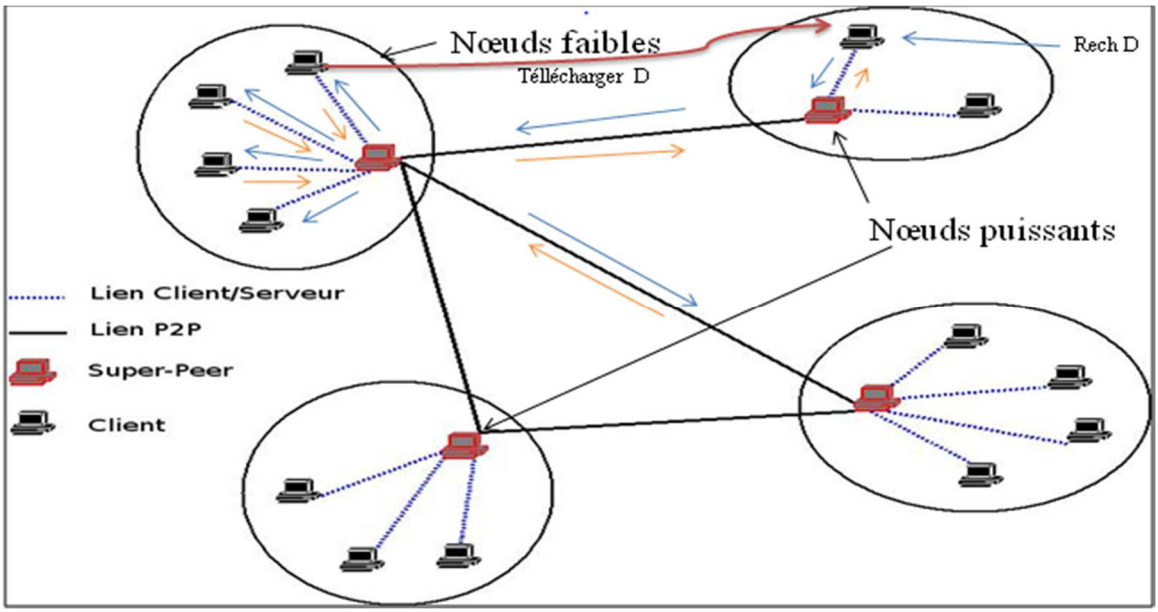


Fig.I.4.Exemple de P2P hiérarchique [7].

Définitions :

DHT[5] : Les tables de hachage distribuées sont une catégorie de réseaux décentralisés structurés qui fournissent deux primitives put et get. Soit une donnée *v* possédant une clé *k* définie sur le même espace d'adressage que les pairs, la primitive put(k,v) stocke la donnée *v* sur le pair en charge du sous-espace contenant *k*. Symétriquement, la primitive get(k) récupère la donnée associée à la clé *k* en interrogeant le pair responsable du sous-espace contenant *k*.

CAN [5] : *(Content Adressable Network)* Est une infrastructure décentralisée et distribuée. Il est organisé autour d'un espace virtuel de coordonnées cartésiennes de dimension *d* (*d-tore*). L'espace des coordonnées est partitionné dynamiquement entre les nœuds. La figure montre un exemple d'espace de coordonnées cartésiennes de deux dimensions [0, 1] [0, 1] partitionné entre six nœuds.

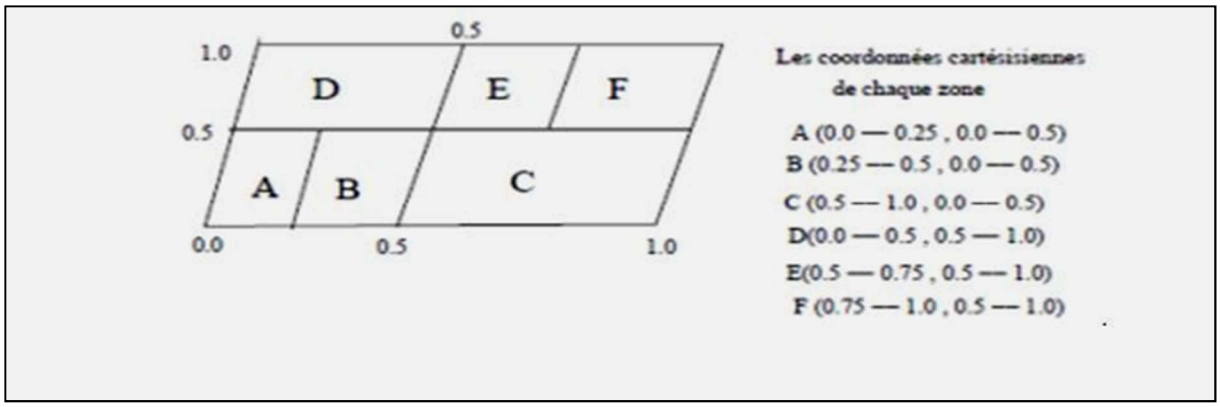


Fig.I.5. Exemple de CAN[5] .

I.4.La recherche d'information sur les systèmes P2P :

Dans un système P2P, les résultats d'un système de recherche d'information ne dépendent plus seulement de la manière dont sont indexés les documents et de la mesure utilisée pour comparer les documents avec les requêtes. En effet la répartition des données, la topologie du réseau et les algorithmes de routage utilisés pour diffuser les requêtes vont avoir une grande influence dans les performances (pertinence des résultats, temps d'exécution et charge du réseau).

I.4.1.Caractéristiques de la recherche d'information dans les systèmes P2P :

- ***La répartition des données :***

Dans un système P2P, les pairs contiennent des données qui leur sont propres. Selon la répartition des données dans le réseau, les résultats du système de recherche d'information seront plus ou moins bons. Par exemples, si les documents pertinents sont contenus dans les voisins directs du pair initiateur de la requête, alors les résultats seront bons, même avec un faible TTL. Par contre, si les documents pertinents sont très éloignés du pair initiateur, ils ne sont pas nécessairement accessibles les résultats seront donc moins bons.

- ***La topologie du réseau :***

La topologie du réseau a aussi un impact important sur les résultats. En effet, plus la connectivité des pairs est importante et plus les pairs sont accessibles rapidement. Cela permet donc d'accéder aux données pertinentes de manière efficace. Par contre, le nombre de messages échangés est plus important.

- ***Les algorithmes de routage***

L'algorithme de routage choisi pour rechercher l'information dans un système P2P a une influence sur les résultats retournés par le système de recherche d'information. Par exemple, il est évident que la valeur du *TTL* a une influence sur la qualité des résultats retournés par le système de recherche d'information. Plus le *TTL* est grand et plus le nombre de pairs recevant la requête est important : cela augmente les chances d'accéder à des sources de documents pertinents. Par contre, il faut noter que plus le *TTL* est grand plus le nombre de messages échangés est important et donc plus le temps de réponse est long.

I.5. Avantages et inconvénients des systèmes P2P [3].

Nous pouvons résumer les différents avantages et inconvénients des systèmes P2P comme suit :

I.5.1. Avantages :

A partir de propriétés citées ci-dessus, nous pouvons affirmer que les systèmes P2P sont plus adaptés aux environnements qui possèdent de grandes quantités de ressources à partager.

Par apport à une approche client/serveur, les systèmes P2P permettent:

- ✓ Eviter la création de goulots d'étranglement.
- ✓ Passer à l'échelle supérieure.
- ✓ Exploiter des ressources importantes distribuées sur un grand nombre de nœuds du réseau.
- ✓ Améliorer l'utilisation de la bande passante, des ressources de traitements et des espaces de stockage.
- ✓ Accélérer l'accomplissement des tâches en réduisant le temps de traitement à travers les liens directs entre les pairs.
- ✓ Eviter le seul point d'échec grâce à la distribution redondante des ressources et la décentralisation des systèmes P2P. Cette caractéristique offre à ce type de systèmes plus de fiabilité et de robustesse.
- ✓ Augmenter les performances du système en partageant la charge du travail et
- ✓ l'accroissement de l'autonomie et l'agrégation des ressources, ce qui augmente la performance des réseaux P2P.
- ✓ Permet aux utilisateurs de maintenir le contrôle de leurs ressources. En plus, ils peuvent joindre ou quitter le système à tout moment.

I.5.2. Inconvénients :

Les systèmes P2P, présentent aussi des inconvénients majeurs à savoir :

- ***Problème de sécurité:*** la sécurité n'est pas toujours assurée lors des communications entre les nœuds du réseau à cause des pairs malveillants.
- Pairs égoïstes: des pairs qui exploitent les ressources des autres et n'offrent rien.
- ***Difficulté d'administration:*** la décentralisation rend le système difficile à administrer, et une connaissance globale de l'état des ressources et du réseau est presque impossible.
- ***Hétérogénéité et faible interopérabilité:*** Vue la diversité des architectures, des protocoles utilisés, en plus de l'absence de standardisation d'un système P2P à un autre, l'intégration et la communication entre les logiciels P2P n'est pas une tâche toujours faisable.

- Problème de disponibilité des ressources: lorsqu'un nœud quitte le réseau, toutes les ressources qu'il fournit disparaissent aussi.
- Problème de publication et de découverte des ressources: dans certains types de systèmes P2P, les mécanismes efficaces de publication et de découverte des ressources manquent particulièrement dans le cas où de nouveaux pairs joignent le réseau.

I.6. Conclusion

Le P2P a connu et connaît toujours un franc succès auprès du grand public grâce aux logiciels de partage et de communication. Néanmoins, le peer to peer se développe également auprès du monde professionnel de par son utilisation notamment dans le calcul distribué et le travail collaboratif.

Dans de nombreux cas, les technologies P2P peuvent servir d'alternative au mode client serveur, ce qui permet de décharger les serveurs.

II.1.Introduction :

Beaucoup de systèmes réels peuvent être modélisés sous forme de graphes d'interactions. Ces graphes sont utilisés pour modéliser des interactions complexes tels que : les réseaux sociaux, les réseaux biologiques, les réseaux de collaboration, les réseaux de transport, l'Internet et les réseaux d'information... Dans ce chapitre, nous allons définir le graphe d'interaction, ainsi nous représentons la modalisation des réseaux d'interactions.

II.2. Définition d'un graphe d'interaction [7] :

Les graphes d'interaction sont une modélisation utilisée dans de nombreuses disciplines. A un instant de l'évolution du réseau, les acteurs sont les nœuds d'un graphe où une arête modélise une interaction entre les deux nœuds qu'elle relie. Ce réseau est bien sûr dynamique, des nœuds peuvent être ajoutés ou supprimés durant son évolution. Les arêtes peuvent aussi être dynamiques selon la définition de l'interaction considérée.

Une interaction, dans un tel réseau, est définie selon ce que l'on cherche à modéliser, par exemple :

-Les réseaux issus de l'Internet : Un des réseaux les plus connu est le réseau Internet (ou encore les réseaux d'Internet). Plusieurs applications utilisent ce réseau pour transmettre de l'information, tel que « World-Wide-Web » ou encore « pair-à-pair ». Les instances de ces applications sont représentées par des nœuds, et les relations qu'elles entretiennent entre elles, peuvent être vues comme des arêtes.

-Les réseaux sociaux : Les graphes d'interactions sont utilisés couramment en sciences sociales pour modéliser des interactions entre individus. Les sommets représentent alors les entités ou individus, et les arêtes ou arcs une relation ou interaction entre eux.

-Les réseaux de cooccurrence lexicale : Le graphe de cooccurrence lexicale d'un document donné est un graphe dont les sommets sont des mots d'un document. Deux mots sont reliés s'ils apparaissent proches dans le document.

II.3. Propriétés communes des réseaux d'interactions [7]:

L'étude des réseaux d'interactions en tant que graphes a révélé que ces derniers, bien que issus de différents domaines, partagent quelques propriétés intéressantes. Dans cette section, nous présentons trois propriétés communes aux réseaux d'interactions :

- distribution des degrés en loi de puissance,
- distance moyenne faible (ou petit diamètre),
- fort coefficient de regroupement (*coefficient de clustérisation*).

II.3. 1. Distribution des degrés en loi de puissance :

La distribution des degrés d'un réseau d'interactions suit une loi de puissance de type :

$$P(k) = C.k^{-\lambda}$$ Où :

k : est le degré.

C : est un paramètre qui dépend de la taille du réseau.

Et λ : est l'exposant de la loi de puissance. Dans la pratique la valeur de λ est comprise entre 2 et 3 et ne dépend pas de la taille du graphe.

Une distribution des degrés en loi de puissance signifie qu'il existe beaucoup de sommets de faible degré et très peu de sommets de fort degré. L'exposant représente la vitesse de décroissance de la courbe des degrés. Plus il est grand, plus la probabilité d'obtenir des sommets de fort degré est petite. Le degré moyen dans un graphe en loi de puissance n'est pas significatif car l'écart-type est très important. Les graphes en loi de puissances sont appelés graphes sans échelle (scale-free graphs).

II.3. 2. Petit diamètre et distance moyenne faible :

- ***Le diamètre d'un graphe :*** est le plus long des chemins parmi l'ensemble des plus courts chemins pour tout couple de nœuds dans le graphe.
- ***La distance moyenne :*** est la moyenne des longueurs des plus courts chemins entre tous les couples de nœuds d'un graphe.

Les réseaux d'interactions ont un petit diamètre et une distance moyenne faible de l'ordre de log(n) où n est la taille du graphe. Le calcul de la distance moyenne et du diamètre étant coûteux en temps, une estimation de la distance moyenne est faite en l'évaluant seulement pour un certain nombre de paires de sommets.

II.3. 3. Coefficient de regroupement fort :

Le coefficient de regroupement (d'agrégation, de clustérisation ou de clustering) est une mesure de la connectivité d'un graphe non orienté, introduite en 1998 par Watts et Strogatz. Le coefficient d'agrégation d'un graphe est égal à la moyenne des coefficients d'agrégation de l'ensemble de ses nœuds.

Dans différents réseaux d'interactions ont montré que le coefficient d'agrégation de ces réseaux est élevé. Dans un réseau social, par exemple, cela signifie que les amis d'un même individu ont une grande probabilité d'être amis entre eux.

II.4.Modélisation des réseaux d'interactions [7] :

L'étude des grands graphes d'interaction a permis de découvrir les principales propriétés de ces graphes. De nombreux scientifiques ont dés lors cherché à trouver un modèle qui se rapprocherait le plus des grands réseaux réels. Historiquement, ce genre de

graphe ont été d'abord assimilés à des graphes aléatoires mais de nombreuses différences subsistent.

Dans les points qui suivent nous allons tenter de présenter les principaux modèles proposés ainsi que les points de similarités et de divergences avec les réseaux réels.

II.4.1. Les graphes aléatoires uniformes :

Rien, à priori, ne relie les différents graphes réalistes. Ce qui amène à penser tout simplement que ces graphes ne sont que le résultat d'interconnexions établies au hasard entre les nœuds.

La théorie des graphes aléatoires a été introduite par Paul Erdös et Alfred Rényi après la découverte d'Edrös qui démontre que des méthodes probabilistes peuvent être très utiles pour venir à bout de certains problèmes dans la théorie des graphes. La modélisation des grands réseaux d'interaction par des graphes aléatoires était donc une première tentative. Elle était considérée, jusqu'à récemment, comme la seule méthode, par défaut, pour les réseaux réels.

- **Le modèle aléatoire d'Erdös et Rényi :**

Lors d'une série de séminaires entre 1950 et 1960, Paul Erdös et Alfréd Rényi ont proposé et étudié les premiers modèles de réseaux d'interactions appelés les graphes aléatoires. Erdös et Rényi ont apporté plusieurs versions de ce modèle, le plus étudié est construit comme suit :

1. Ajouter n sommets.
2. Pour tout couple de sommets, une arête est ajoutée avec une probabilité p.

- **Propriété 1** : un graphe aléatoire G = (V, E) de probabilité p est caractérisé par n sommets et un ensemble d'arêtes E contenant chaque couple de sommets (vi, vj) avec une probabilité p. Si n est l'ordre du graphe, on défini : $\boldsymbol{\lambda = pn}$.
- **Propriété 2 :** La distribution des degrés peut être approchée par une loi de Poisson de Paramètre l. Soit k un entier positif ou nul, la probabilité que le degré d'un nœud égale k vaut (**voire Fig.II.1**):

$$\mathbf{P(k) = e^{-\lambda}\ \frac{\lambda^k}{\lambda!}} \qquad \textbf{Avec } \lambda = \langle k \rangle$$

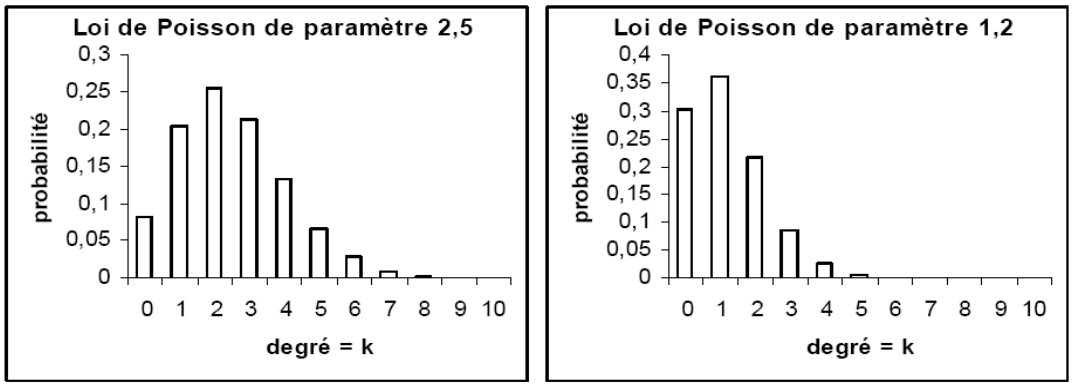


Fig.II.1. lois de Poisson de paramètre l = 2,5 et l = 1,2

Bilan de ce modèle :

Les récentes observations expérimentales (qui étaient impossibles ou très partielles auparavant) ont mis en lumière d'importantes différences. Le modèle de graphe aléatoire est dit « démocratique » dans le sens où la probabilité de connexion d'un nœud est la même pour tous. Ainsi, la distribution des degrés dans les graphes aléatoire d'Erdös et Rényi suit une loi de Poisson (exponentielle) alors qu'il s'agit d'une loi de puissance pour la grande majorité des réseaux réels. Ce graphe a un faible ***coefficient de clustering*** puisque les voisins d'un nœud n'ont aucune raison d'être d'avantage reliés entre eux que deux nœuds pris au hasard. Néanmoins, les graphes d'Erdös et Rényi partagent la caractéristique du petit diamètre avec les réseaux réels.

II.4.2. Les graphes petit-monde :

Les graphes petit monde (Small World) sont nés après l'apparition du concept des « six degrés ». Le concept des six degrés de séparation a été imaginé par le Hongrois Frigyes Karinthy en 1929. Cette idée théorique évoque la possibilité que toute personne sur le globe peut être reliée à n'importe quelle autre à travers une chaîne de relations individuelles comprenant au plus cinq autres maillons.

Cette théorie fut reprise en 1967 par le psycho-sociologue Stanley Milgram qui a réalisé une série d'expériences dont le résultat est connu sous le nom de «six degrés de séparation». La conclusion de ses études est, que deux personnes quelconques dans le monde ont une chaîne de connaissances de longueur six en moyenne. En d'autres termes, il y a cinq personnes intermédiaires qui séparent les deux personnes étrangères l'une de l'autre. Par contre, après plus de trente ans, le statut de cette idée comme description de réseaux sociaux hétérogènes reste une question ouverte. Des études sont encore menées actuellement sur le « petit monde ».

- **Le modèle de Kleinberg :**

En 2000, Kleinberg propose le premier modèle de petit monde présentant la propriété de navigabilité, c'est-à-dire le premier modèle de graphe dont le diamètre est poly

logarithmique en nombre de nœuds et dont des chemins poly logarithmiques peuvent être découverts par un algorithme décentralisé entre tout couple de sommets.

Voici les étapes de construction du graphe selon ce modèle (voir figure 2.4):

1. Construire une grille à deux dimensions où chaque croisement dans la grille est un nœud, et chaque nœud est lié à ses quatre voisins immédiats : haut, bas, droit et gauche. Ces liens sont appelés « liens courts ».
2. Ajouter à chaque nœud u un nombre k > 0 constant d'arc. La destination du $j^{ème}$ arc de u est le nœud v avec une probabilité proportionnelle à $1/|u\text{-}v|^s$, où |u-v| est la distance entre u et v dans la grille, s est une constante positive. Ces liens sont appelés « liens longs ».

Bilan de ce modèle :

L'idée de Kleinberg est de relier un sommet à tous ses proches voisins et à quelques sommets lointains. La première démarche permet d'augmenter la valeur du coefficient de regroupement et la deuxième de diminuer la distance moyenne. Le principal défaut de ce modèle est qu'il ne produit pas la distribution des degrés observée sur les réseaux réels, puisque chaque sommet a un degré constant.

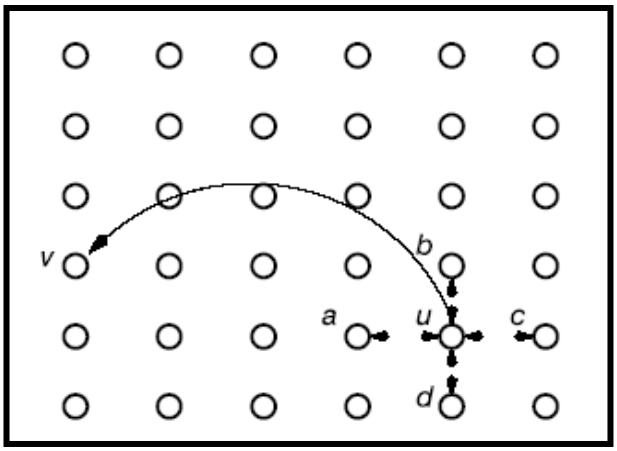


Fig.II.2. Construction de réseau petit-monde : modèle de Kleinberg[6] .

II.4.3.Les réseaux scale-free:

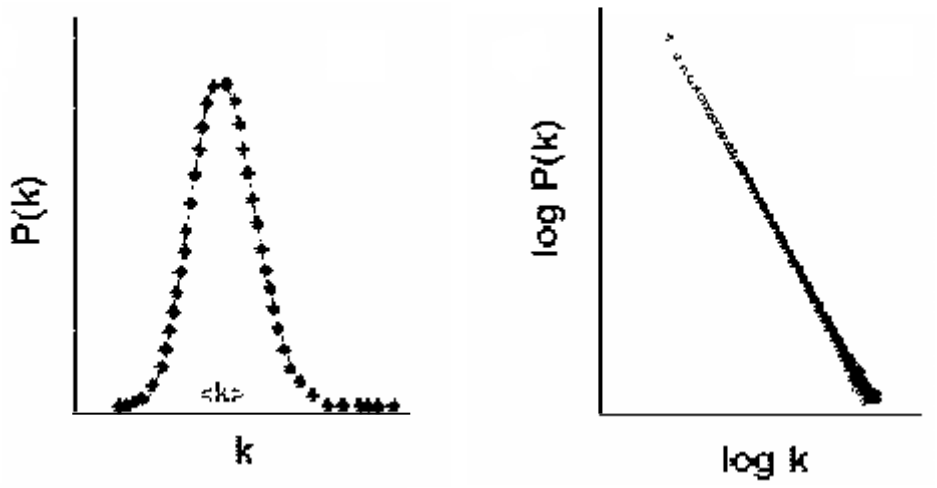
Dans les systèmes aléatoires, on a la probabilité *P(k)* qu'un nœud ait *k* liens. En moyenne, chaque nœud a donc *k* liens. Le système suit alors une $loi\ de\ Poisson$ avec un pic *k* où *P(k)* est à peu près nul pour des *k* élevés (voir **Fig.II.3 -a-**).

Il a été remarqué par la suite que la plupart des systèmes ne suivaient pas la loi de Poisson mais une loi de puissance où $P(k) = k^{-\gamma}$, tel le WWW, ils sont alors appelés scale-

free. Ce modèle prend en compte la *dynamicité* des nœuds. Ici le nombre de nœuds ayant un petit *k* est très important, il décroit au fur et à mesure que *k* augmente (**voir Fig.II.3 -b**-). L'existence de nœuds ayants un fort degré garantit la connexité du graphe.

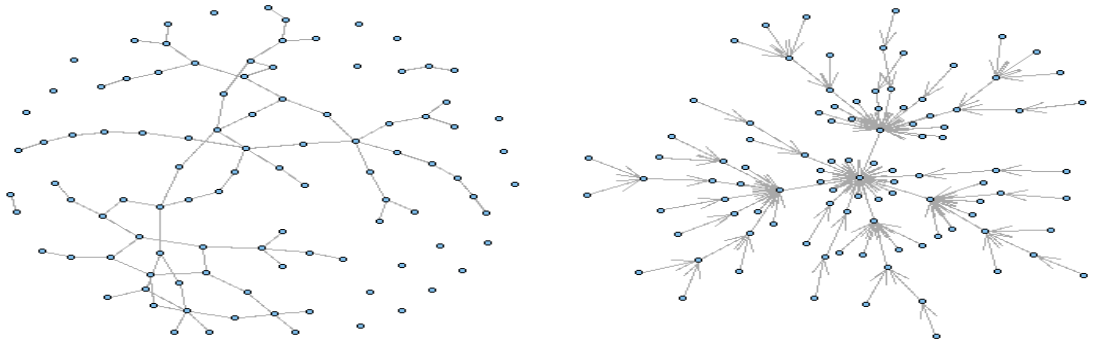
On peut prendre comme exemple le trafic aérien : en effet, la majorité des aéroports n'accueillent qu'un petit nombre de vols alors qu'un petit nombre d'aéroports accueillent un nombre importants de vols.

Dans **scale-free**, les nouveaux nœuds se connecteront avec une forte probabilité à des nœuds ayant un fort degré. C'est ce qu'on appelle : l'attachement préférentiel. La Fig.II.3 montre l'exemple de deux systèmes : -a- constituant un graphe aléatoire et -b- un graphe scale free, construit selon le principe de *l'attachement préférentiel*.



- a - cas de graphes aléatoires **- b -** cas de graphe scale-free

Fig.II.3 : Distribution des degrés des nœuds.



-a- **-b-**

Fig.II .4 : Graphe aléatoire. Graphe scale-free.

- **Le modèle à base d'attachement préférentiel de Barabàsi-Albert :**

En 1999, Albert et Barabàsi ont popularisé un modèle dynamique permettant d'obtenir une distribution des degrés suivant une loi de puissance. Ce modèle consiste à construire un réseau nœud par nœud, en reliant chaque nouveau nœud préférentiellement aux sommets existants de plus hauts degrés. Il est connu sous le nom de l'attachement préférentiel.

Un graphe du modèle de Barabàsi-Albert (BA) est construit de la manière suivante :

1. n_0 sommets sont créés.
2. à l'étape t, un sommet est ajouté ainsi que m ($0<m\leq n_0$) arêtes pour le relier au reste du graphe. La probabilité qu'un sommet v soit choisi comme destination de l'arête est égale au degré de v sur le nombre d'arête à l'étape t-1.

Avec l'attachement préférentiel, un sommet choisi à l'étape t a une probabilité plus élevée d'être choisi à l'étape t+1.

Bilan de ce modèle :

L'intérêt de ce modèle est sa construction dynamique, puisque dans de nombreux réseaux réels, des nœuds et des liens sont fréquemment ajoutés et enlevés au cours du temps (on peut penser au réseau des pages web par exemple). La construction d'un graphe suivant le principe de l'attachement préférentiel engendre la propriété de la distribution des degrés en loi de puissance. L'étude de ce genre de graphes a démontré que la distance moyenne entre les sommets est faible ce qui prouve bien que cette propriété est implicitement induite par les graphes présentant une distribution de degrés en loi de puissance.

Malgré les nombreux points en communs qui existent entre les graphes réels et les graphes obtenus avec ce modèle, ces derniers ne sont pourtant pas fidèles aux trois grandes propriétés connues des graphes réels puisque le coefficient de clustering reste relativement faible.

II.5. Comparaison entre les différents modèles [5] :

Tab.II.1 : Comparaison entre les différents modèles de graphes d'interaction étudiés [7].

Modèle	Dynamique	Loi de puissance	Petit diamètre	Clustering
Erdös et Rényi	Non	Non	Oui	Non
Watts et Strogatz	Non	Non	Oui	Oui
Kleinberg	Non	Non	Oui	Oui
Barabàsi-Albert	Oui	Oui	Oui	Non

Le Tab.II.1 : montre qu'en règle générale, les modèles ne couvrent pas toutes les propriétés communes aux réseaux d'interactions. En effet, les premiers modèles aléatoires possèdent uniquement la propriété du petit diamètre, ils sont les moins appropriés pour modéliser les grands graphes réels. Les modèles des graphes small Word et des graphes sans échelles, possèdent tous les deux, deux propriétés, les premiers ont un petit diamètre et un fort coefficient de clustering, et les seconds ont une distribution des degrés suivant une loi de puissance et un petit diamètre. Le fait que les graphes sans échelles soient dynamiques les rend plus représentatifs des grands graphes réels.

II.6.Conclusion :

Les systèmes pair-à-pair décentralisés non-structurés peuvent être représenté part des graphes d'interaction uni-bipartie. Plusieurs modèles existent pour modéliser ses grands réseaux, mais il n'existe pas encore de modèle conforme aux graphes réels. De nombreuses recherches sont menées pour trouver un modèle qui réunit à la fois les trois propriétés de ce genre de graphes à savoir : une distribution des degrés en loi de puissance, une distance moyenne faible, et un fort coefficient de clustering.

Dans ce chapitre, nous avons représenté le graphe d'interaction et ainsi leur caractéristique, les différents modèles de modélisation des grands réseaux d'interaction.

III.1. Introduction :

Après avoir donné une vue théorique d'ensemble sur les différents aspects qu'il nous on été nécessaires d'étudier. Dans ce chapitre nous passons à la partie conception et implémentation de notre application, aussi nous présentons les différents résultats obtenus.

III.2. Choix du langage de programmation :

Java est à la fois un environnement d'exécution portable et un langage de programmation informatique à objets développé par la société Sun Microsystème.

Nous avons choisit d'utiliser ce langage de programmation afin de développer notre application.

III.2.1. Pourquoi l'utilisation de la plate forme eclipse ?[4] :

Eclipse est une plate-forme java open source est un environnement de développement intégré (Integrated Development Environment) développé par I.B.M, dont le but est de fournir une plate-forme modulaire pour permettre de réaliser des développements informatiques. Eclipse utilise énormément le concept de modules nommés "plug-ins" dans son architecture. D'ailleurs, hormis le noyau de la plate-forme nommé "Runtime", tout le reste de la plate-forme est développé sous la forme de plug-ins. Ce concept permet de fournir un mécanisme pour l'extension de la plate-forme et ainsi fournir la possibilité à des tiers de développer des fonctionnalités qui ne sont pas fournies en standard par Eclipse.

III.2.2. La bibliothèque JFreeChart [9]:

JFreeChart est une bibliothèque java libre. Elle est assez facile à utiliser. L'avantage du langage de programmation Java est que l'on peut trouver sur Internet de nombreuses API gratuites, qui peuvent être utilisées dans les applications. JFreeChart est une fonctionnalité qui permet de générer des graphiques.

Le JFreeChart propose de nombreux types d'affichages de graphiques possibles : histogramme, camemberts, graphiques d'évolution, courbes...

III.3.Les ressources utilisées :

III.3.1. les ressources matérielles :

- Processus : Intel® Atom™ CPU N450 @ 1.66Hhz 1.67.
- Mémoire installée (RAM) : capacité : 1,00 Go.

III.3.2. les ressources logicielles :

- Système d'exploitation : Windows 7 Edition Familiale Premium.
- Type du système : Système d' exploitation 32 Bits.
- Type de langage de programmation : JAVA.

III.4. Présentation de l'application :

III.4 .1.Construire un réseau :

Le réseau que nous construisons est un réseau scale-free construit selon le principe de l'attachement préférentiel décrit par l'algorithme d'Albert-Barabasi.

Le principe est le suivant :

- Construire des liens entre m_0 nœuds initiaux de façon aléatoire. La connexité de ce réseau doit être vérifiée afin qu'il n'y ait pas de nœuds isolés. Chaque nœud de ce sous-ensemble sera connecté à tout autre nœud de ce même sous ensemble avec une certaine probabilité qui sera définie au départ.
- A chaque arrivé d'un nouveau nœud, un nombre aléatoire de liens m, compris entre 0 et $m_0/3$ lui seront attribués de la manière suivante :

Exemple d'un reseau decentralisé gnutella :

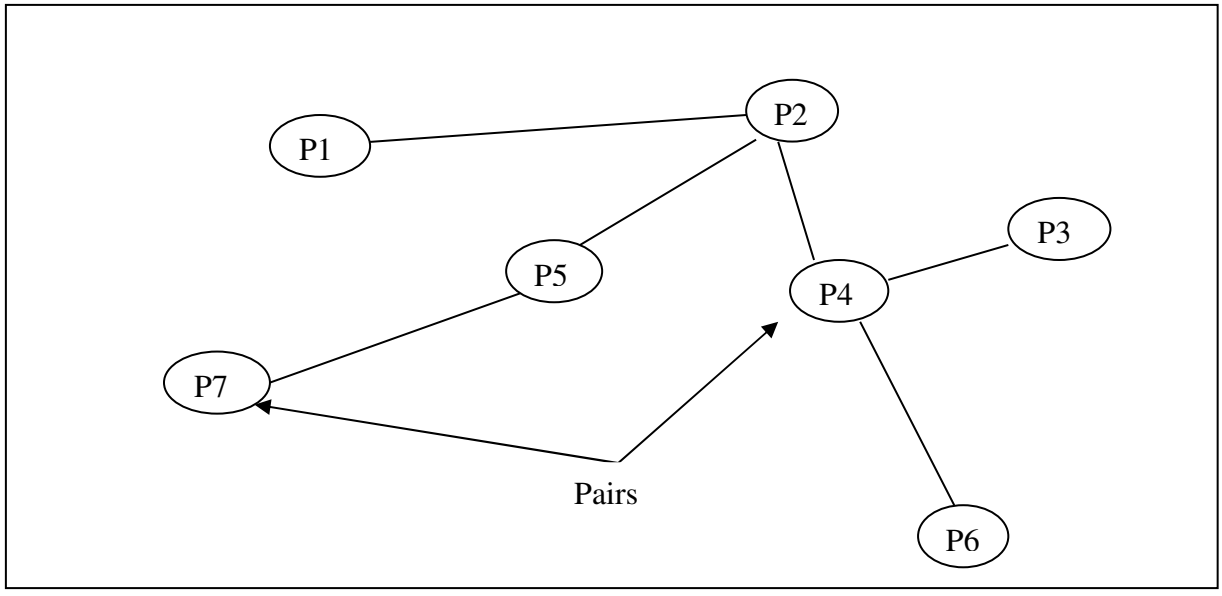


Fig.III.1.Exemple d'un réseau.

Algorithme de Barabsi :

```
public BarabasiAlbertGenerator(Factory<Graph<V,E>> graphFactory,
Factory<V> vertexFactory, Factory<E> edgeFactory,
int init_vertices, int numEdgesToAttach, Set<V> seedVertices) {
this(graphFactory, vertexFactory, edgeFactory, init_vertices, numEdgesToAttach, (int)
System.currentTimeMillis(), seedVertices);
}
// Initialisation des nœuds.
private void initialize(Set<V> seedVertices) {
mGraph = graphFactory.create();
vertex_index = new ArrayList<V>(2*init_vertices);
index_vertex = new HashMap<V, Integer>(2*init_vertices);
for (int i = 0; i < init_vertices; i++) {
V v = vertexFactory.create();
mGraph.addVertex(v);
vertex_index.add(v);
index_vertex.put(v, i);
seedVertices.add(v);
}
mElapsedTimeSteps = 0;
}
// La fonction aléatoire pour la connexivité entre les nœuds
private void createRandomEdge(Collection<V> preexistingNodes,
V newVertex, Set<Pair<V>> added_pairs) {
V attach_point;
boolean created_edge = false;
Pair<V> endpoints;
do {
attach_point = vertex_index.get(mRandom.nextInt(vertex_index.size()));
endpoints = new Pair<V>(newVertex, attach_point);
if (!(mGraph instanceof MultiGraph))
{
if (added_pairs.contains(endpoints))
continue;
if (mGraph.getDefaultEdgeType() == EdgeType.UNDIRECTED &&
added_pairs.contains(new Pair<V>(attach_point, newVertex)))
continue;
}
double degree = mGraph.inDegree(attach_point);

double attach_prob = (degree + 1) / (mGraph.getEdgeCount() + mGraph.getVertexCount() - 1);
if (attach_prob >= mRandom.nextDouble())
created_edge = true;
}
```

```
    while (!created_edge);

     added_pairs.add(endpoints);
if (mGraph.getDefaultEdgeType() == EdgeType.UNDIRECTED) {
        added_pairs.add(new Pair<V>(attach_point, newVertex));
    }
  }
        //Instruit l'algorithme pour évoluer le graphique pas N.
  public void evolveGraph(int numTimeSteps) {

    for (int i = 0; i < numTimeSteps; i++) {
      evolveGraph();
      mElapsedTimeSteps++;
    }
  }
  private void evolveGraph() {
    Collection<V> preexistingNodes = mGraph.getVertices();
    V newVertex = vertexFactory.create();
    mGraph.addVertex(newVertex);

    Set<Pair<V>> added_pairs = new HashSet<Pair<V>>(mNumEdgesToAttachPerStep*3);

    for (int i = 0; i < mNumEdgesToAttachPerStep; i++)
      createRandomEdge(preexistingNodes, newVertex, added_pairs);

    for (Pair<V> pair : added_pairs)
    {
      V v1 = pair.getFirst();
      V v2 = pair.getSecond();
      if (mGraph.getDefaultEdgeType() != EdgeType.UNDIRECTED ||
                !mGraph.isNeighbor(v1, v2))
          mGraph.addEdge(edgeFactory.create(), pair);
    }
 vertex_index.add(newVertex);
    index_vertex.put(newVertex, new Integer(vertex_index.size() - 1));
  }
     // Rapporte le nombre total de pas écoulé.
  public int numIterations() {
    return mElapsedTimeSteps;
  }
    // Créez dans interface.
  public Graph<V, E> create() {
    return mGraph;
  }
```

Fig.III.2.Algorithme barabasi generator.

III.5.Analyse des besoins :

Notre objectif, dans ce travail, est d'implémenter un réseau peer to peer et d'étudier les propriétés vue dans le chapitre deux. Pour cela, nous sommes amenés à développer une application de simulation pour les réseaux p2p. Nous considérons les systèmes p2p non structurés de topologie décentralisée où tous participants jouent le même rôle. Ensuite durant la période de simulation, nous ne gérons pas l'aspect dynamique physique du réseau (déconnexion et connexion des pairs).

Lorsque l'on simule, on va modéliser la partie du système que nous voulons observer et simplifier au maximum le reste.

Afin de répondre à nos besoins, nous avons modélisé un réseau p2p c'est pour sa qu'il est nécessaire d'avoir une architecture à une deux couches:

- **Niveau réseau physique :** on va trouver ici les notions de pair ainsi que les mécanismes de gestion du réseau de pairs. (Adressage, table de voisinage des pairs, requête p2p, propagation de requête p2p).
- **Niveau gestion de documents et du contenu :** on va trouver ici les notions de pair gérant un ensemble de documents (avec toutes les fonctions de gestion, d'indexation et d'organisation associées). Cette étape est représentée comme étant une perspective pour notre travail.

Donc nous définissons notre architecture comme un ensemble de structures de données structurées en deux couches (la couche logique est pertinente dans notre cas).

Le programme développé prend en charge les fonctionnalités suivantes :

- Générer un réseau p2p decentralisé en permettant d'introduire le nombre total de pairs.
- Verifier le voisinage de chaque pair.
- Observer les propriétés du reseau construit en utilisant les propriétés des graphes d'interaction qui sont : le petit diametre, coefficient de regroupement, distribution des degrés en loi de puissance.

Acteurs :

Le principal acteur dans cette application est l'utilisateur, c'est lui qui crée le réseau en fournissant le nombre total de pairs. Les principaux cas d'utilisation peuvent être résumés par Use Case (Fig.III.3).

Use Case :

- Introduire le nombre total de pairs du réseau.
- Creation du reseau .
- Creation du graphe.

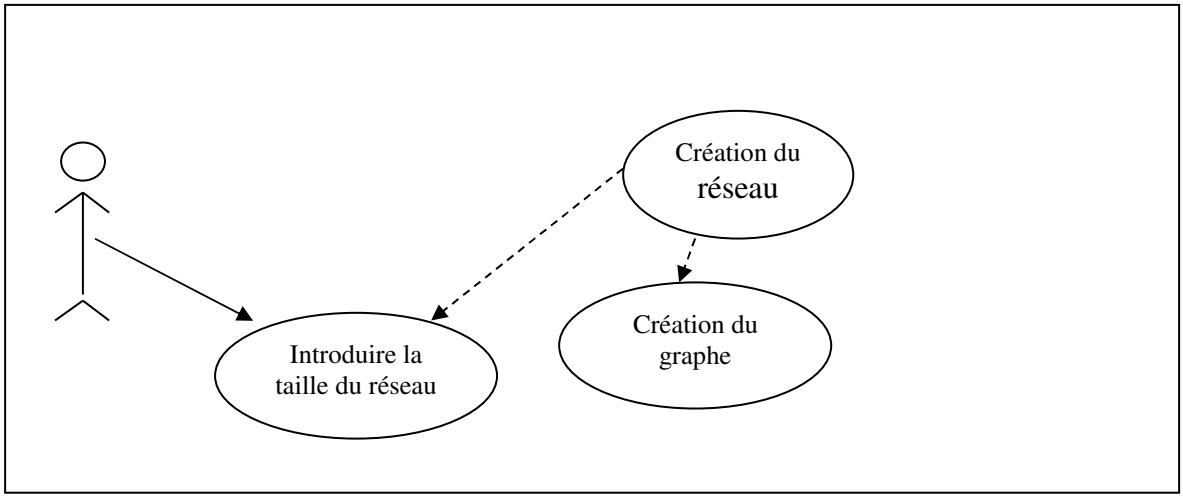


Fig.III.3. Diagramme UML des cas d'utilisation (Use-Case).

III.6.Etapes de construction de système :

Nous présentons un diagramme de cas d'utilisation (Fig.III.3) pour récapituler les étapes de construction de notre système.

- **Configuration :** L'utilisateur du logiciel va saisir, à partir de l'interface graphique les paramètres de construction du système (taille du réseau).
- **Observation :** Ce cas d'utilisation permet à l'utilisateur de suivre l'évolution du réseau simulé et différents changements.

III.6.1.La topologie du réseau :

Nous avons modélisé notre réseau en se basant sur la topologie décentralisé de Gnutella0.4, donc le réseau construit aura une structure purement décentralisé.

III.6.2.Description des données en entrée :

Pour modéliser un système, on doit lui présenter ses paramètres d'entrées. Notre projet à besoin des données qui décrivent :

III.6.2.1.Les nœuds du système :

Il existe un seuls type de nœuds dans le système :

III.6.2.1.1.Les nœuds pairs :

Chaque nœud est représenté par un pair ayant les capacités de communiquer, rechercher, partager, apprendre de nouveaux documents, de nouveaux profiles et de nouveaux voisins.

III.6.2.1.2.Les voisins :

Un pair *v* est dit voisin du pair *u*, s'il connait son identifiant (on travaille sur un graphe non-orienté, donc l'un connait l'identifiant de l'autre).

Chaque pair maintient localement une liste de voisins de son cluster avec lesquels il peut communiquer directement et dont les identifiants figurent dans la liste des voisins. De nouveaux voisins sont ajoutés à cette liste grâce aux échanges des différents messages.

III.6.2. Etude des propriétés de réseau :

III.6.2.1.La distribution des degrés des nœuds de loi de puissance:

Nous avons utilisé un vecteur degré et nous avons calculé le degré de chaque nœud et à l'aide de la bibliothèque JfreeChart ,nous avons pu donner la courbe de la distribution des degrés (voir **Fig.III.4**).

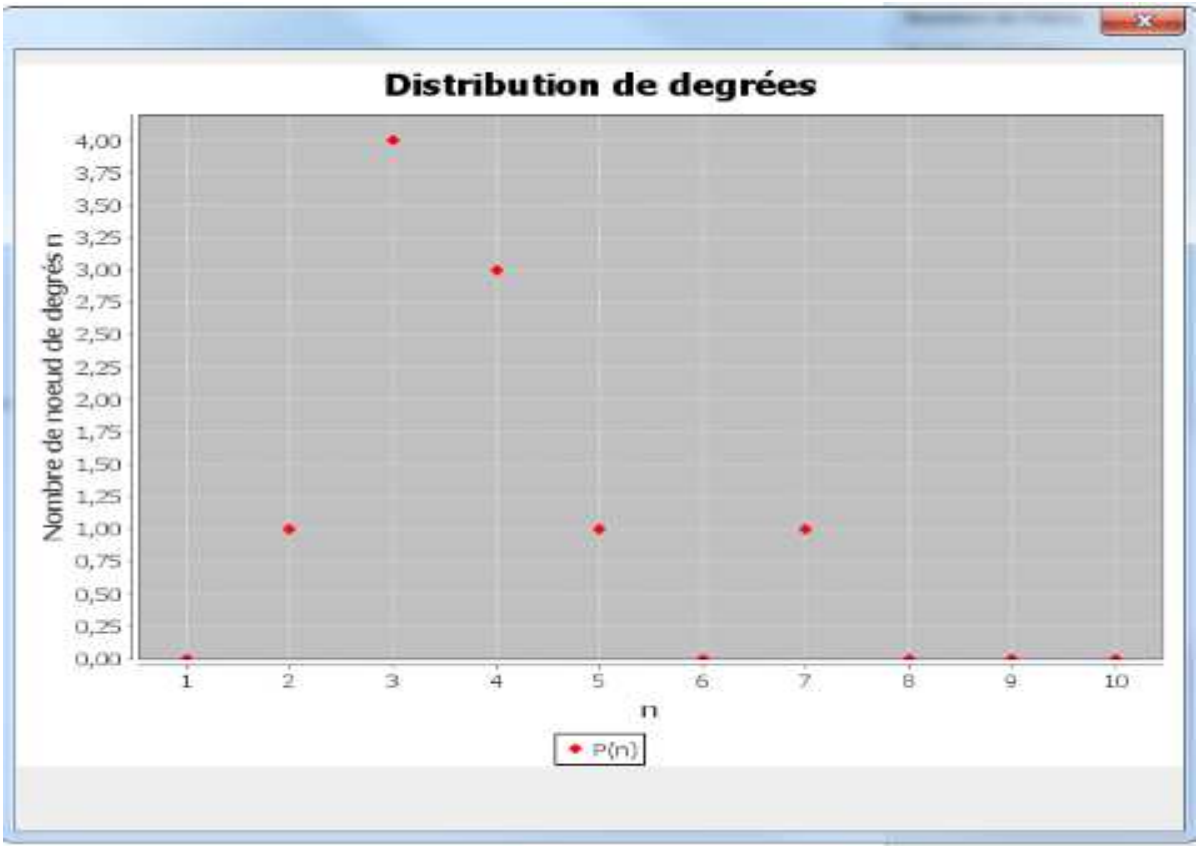


Fig.III.4.La distribution de degrés des nœuds dans un réseau construit.

La Fig.III.4 est représente la distribution des degrés des nœuds de ce même système. Le graphe obtenu est un graphe à queue lourde, ce qui caractérise les réseaux scale-free. La distribution suit bel et bien une loi de puissance, puisque nous avons obtenu un graphe à queue lourde.

III.6.2.2.Le coefficient de regroupement fort :

Nous allons aussi calculer le coefficient de clustering du graphe C_g qui la moyenne des coefficients de clustering C_i de chaque nœud i, ce dernier est calculé par la formule suivante :

$$C_i = \frac{3 \times (Ntg)}{Ntp}$$

Où *Ntg* est le nombre de triangles (les voisins d'un nœud sont voisins entre eux) dont i fait partie et *Ntp* est le nombre de triplet (chaine de trois nœuds liés) dont i fait partie aussi.

- Si : C_g < 0.2 alors on dit qu'on a un faible clustering.
- Si : 0.2 < C_g< 0.4 alors on dit qu'on a un clustering moyen.
- Si : C_g > 0.4 alors on dit qu'on a un fort clustering.

L'algorithme de la Fig.III.5 présente le calcul du coefficient de regroupement dans notre application.

```
public static Map clusteringCoefficients(ArchetypeGraph graph)
{
Map coefficients = new HashMap();

for (Iterator v_iter = graph.getVertices().iterator(); v_iter.hasNext(); )
{
ArchetypeVertex v = (ArchetypeVertex)v_iter.next();
int n = v.numNeighbors();
if (n == 0)
coefficients.put(v, new Double(0));
else if (n == 1)
coefficients.put(v, new Double(1));
else
{
// combien de voisin sont connécté entre eux ?
ArrayList neighbors = new ArrayList(v.getNeighbors());
double edge_count = 0;
for (int i = 0; i < neighbors.size(); i++)
{
ArchetypeVertex w = (ArchetypeVertex)neighbors.get(i);
for (int j = i+1; j < neighbors.size(); j++ )
{
ArchetypeVertex x = (ArchetypeVertex)neighbors.get(j);
edge_count += w.isNeighborOf(x) ? 1 : 0;
}
}
double possible_edges = (n * (n - 1))/2.0;
edge_count += w.isNeighborOf(x) ? 1 : 0;
}
}
double possible_edges = (n * (n - 1))/2.0;

}
```

```
coefficients.put(v, new Double(edge_count / possible_edges));
}
}
return coefficients;
}
```

Fig.III.5. L'algorithme de coefficient de regroupement.

Le graphe de la Fig.III.6 présente le coefficient d'un réseau de 5 nœuds construit selon le principe de l'attachement préférentiel (algorithme de barabasi).

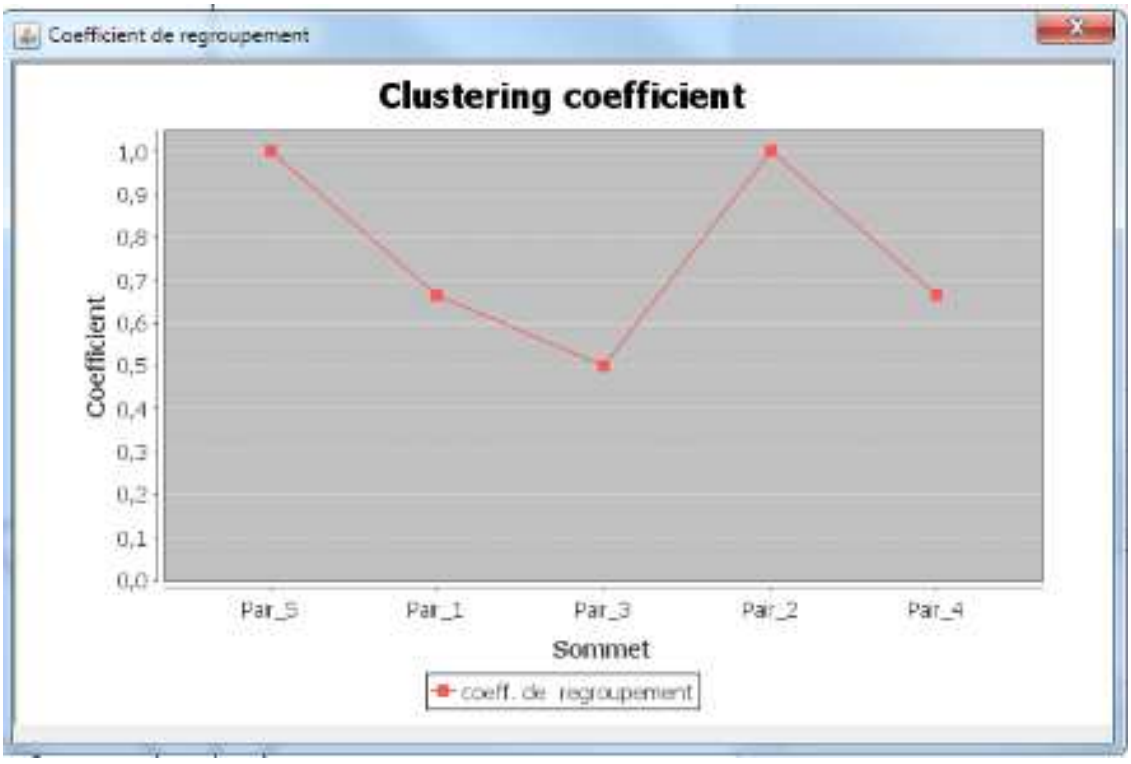


Fig.III.6. Le coefficient de regroupent d'un réseau construit.

Pour notre cas nous avons eu le 0,5 <Cg<1,0 donc on est devant un fort coefficient de clustering.

III.6.2.3.La distance moyenne et le petit diamètre :

Il nous reste à calculer le diamètre de notre graphe. Le diamètre d'un graphe est le plus long des chemins parmi l'ensemble des plus courts chemins pour tout couple de nœuds dans le graphe.

L'algorithme de la distance moyenne de notre application est le suivant :

```
public static Map averageDistances(ArchetypeGraph graph, Distance d)
   {
      Map avg_dist = new HashMap();
      Set vertices = graph.getVertices();
      int n = graph.numVertices();
      for (Iterator outer = vertices.iterator(); outer.hasNext(); )
      {
         ArchetypeVertex v = (ArchetypeVertex)outer.next();
         double avgPathLength = 0;
         for (Iterator inner = vertices.iterator(); inner.hasNext(); )
         {
            ArchetypeVertex w = (ArchetypeVertex)inner.next();
            if (v != w) // don't include self-distances
            {
 Number dist = d.getDistance(v, w);
               if (dist == null)
               {
                  avgPathLength = Double.POSITIVE_INFINITY;
                  break;
               }
               avgPathLength += dist.doubleValue();
            }
         }
         avgPathLength /= (n - 1);
         avg_dist.put(v, new Double(avgPathLength));
      }
      return avg_dist;
   }
```

Fig.III.7. L'algorithme de la distance moyenne.

Le graphe obtenu de la Fig.III.8 représente la distance moyenne entre les nœuds dans un réseau construit.

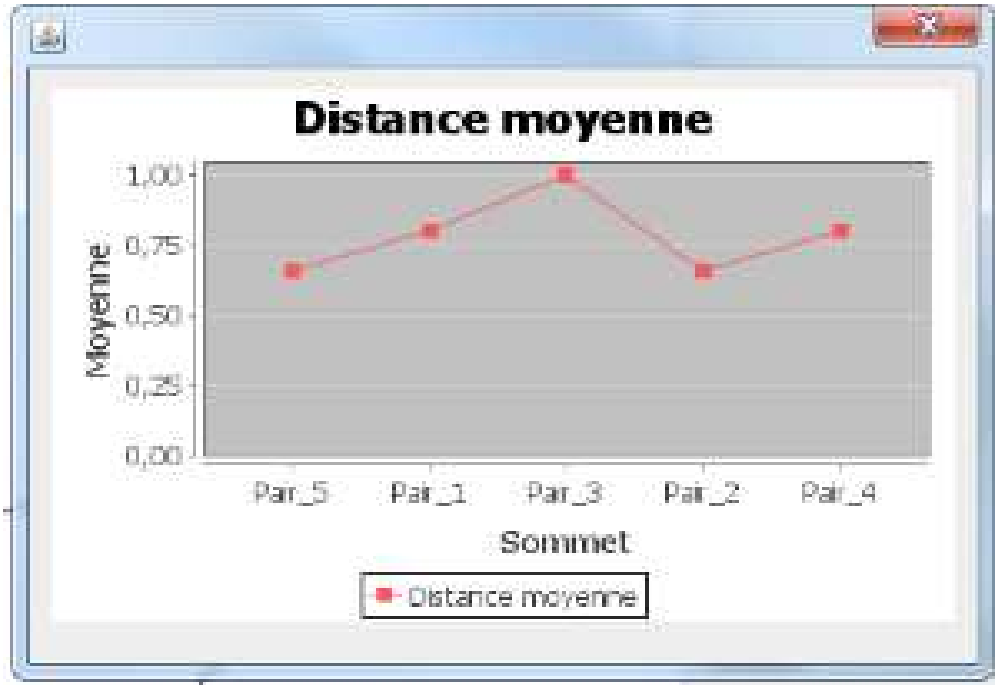


Fig.III.8.La distance moyenne entre les nœuds dans un réseau construit.

Dans ce graphe la distance moyenne =1,0 et voilà une distance moyenne faible.

III.7. Quelques vues d'application :

III.7.1. Le fenêtre principale :

La fenêtre principale de la Fig.III.9 est notre porte d'entrée à l'application. A partir de cette fenêtre, nous pouvons paramétrer une nouvelle configuration pour lancer une étude, il suffit de cliquer sur « graphe» et la fenêtre permettant une nouvelle configuration apparaitra.

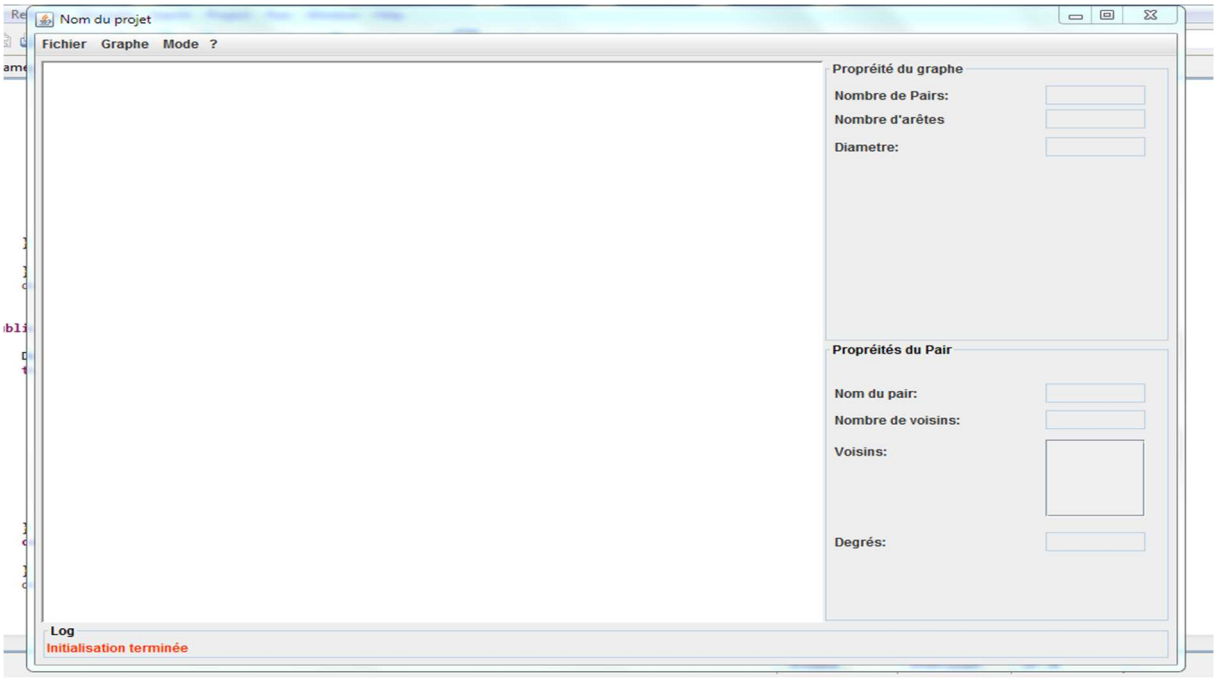


Fig.III.9.La fenêtre principale.

III.7.2. Fenêtre de création de réseau :

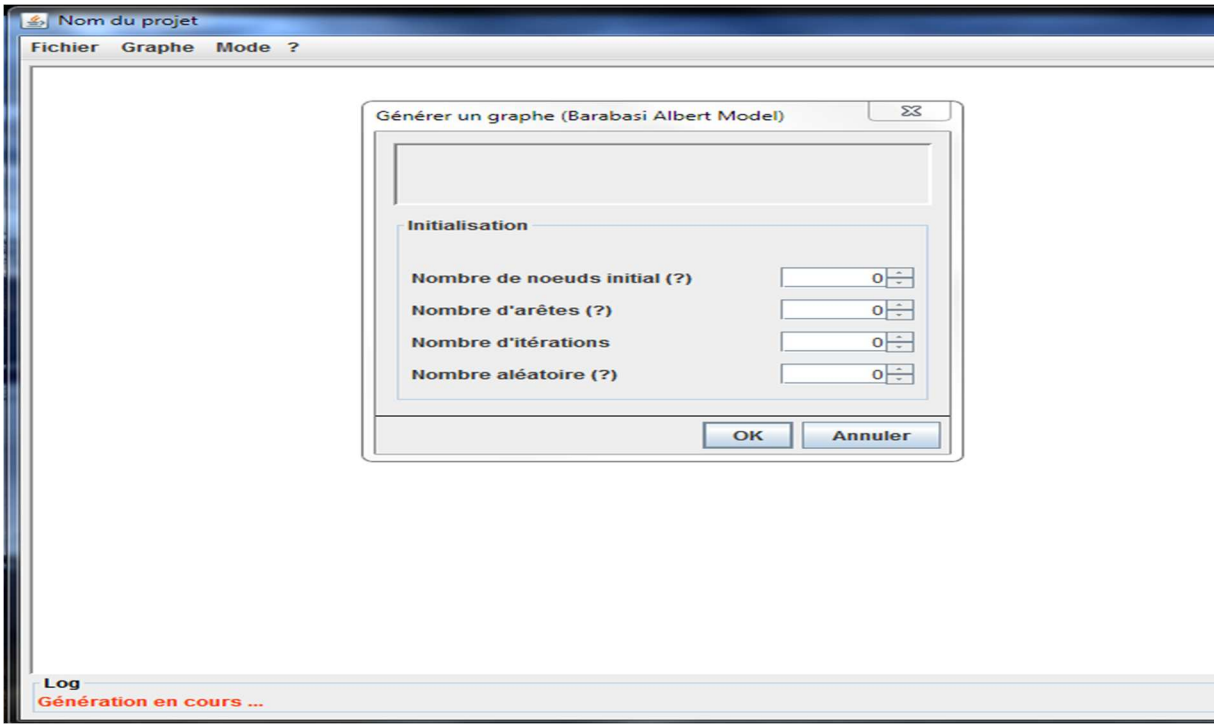


Fig.III.10. Générer un graphe.

La fenêtre de la Fig.III.10 illustre la construction du graphe en utilisant l'algorithme d'Albert Barabasi.

III.7.3. Propriétés de réseau :

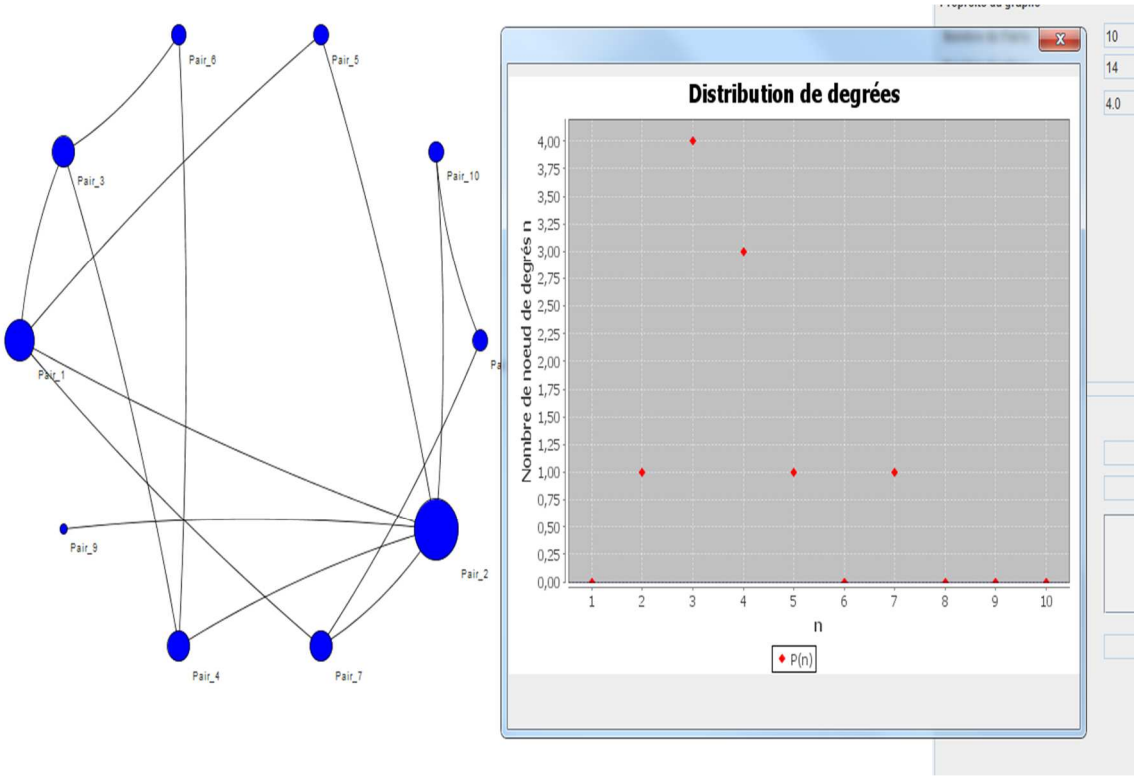


Fig.III.11.Distribution de degrés des nœuds.

Après la construction du graphe nous pouvons directement étudier les propriétés de celui-ci, dans la Fig.III.11 la distribution de degrés des nœuds est présentée.

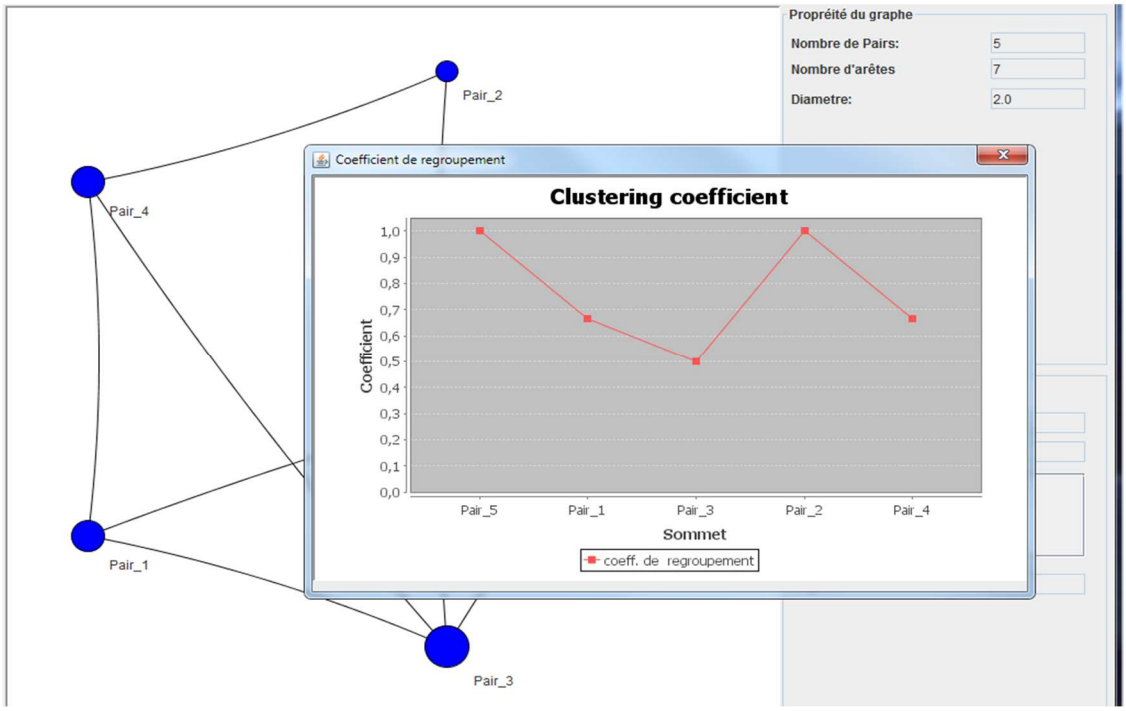


Fig.III.12.Coefficient de regroupement.

Le Fig.III.12 montre le coefficient de regroupement d'un réseau.

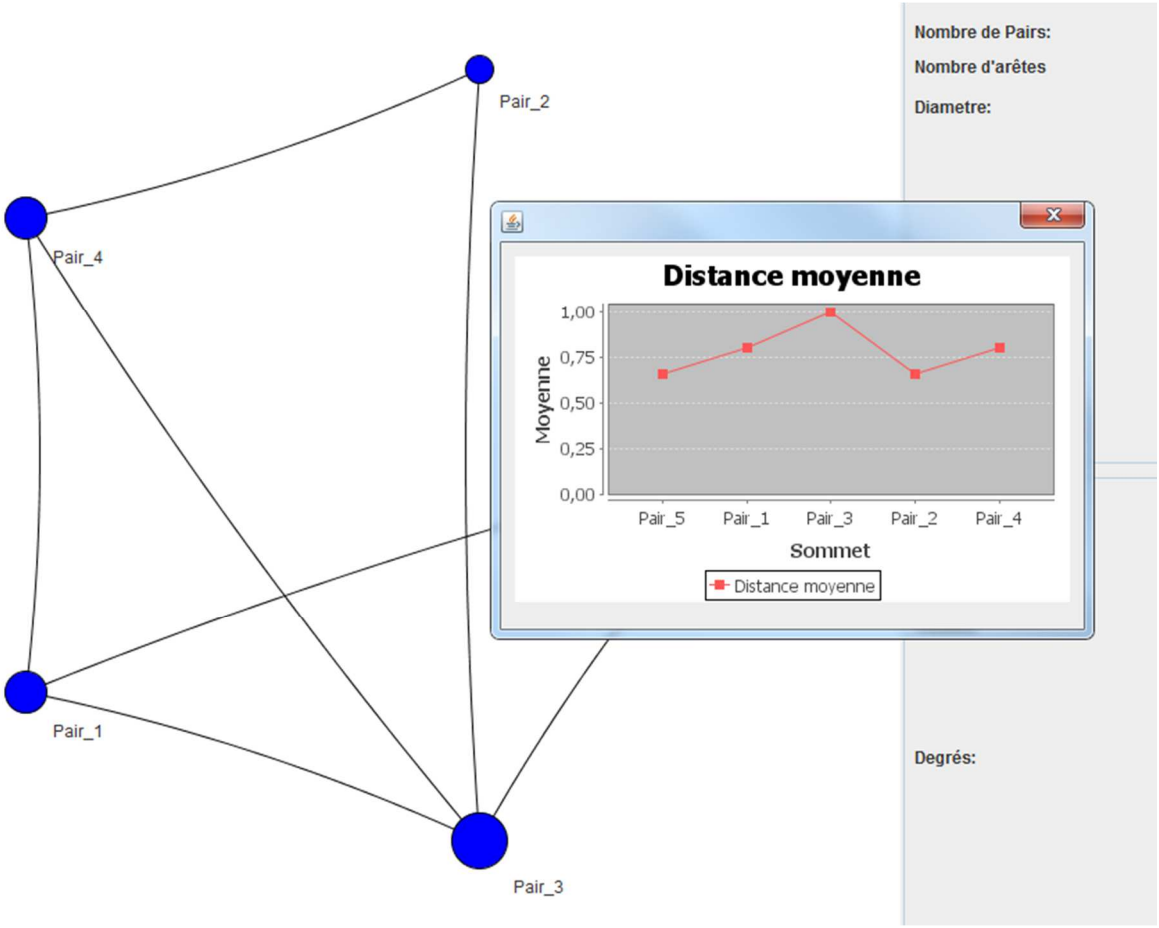


Fig.III.13.Distance moyenne.

La Fig.13 montre la distance moyenne faible construite pour ce même réseau.

III.8. Conclusion :

Dans ce chapitre, nous avons présenté la description de l'application. En présentant les étapes nécessaires pour la création de notre projet avec les différents modes d'exécution.

Notre application permet la création du réseau et étudié les propriétés de ce même réseau.

Le graphe qu'on a analysé répond bien aux trois propriétés des grands réseaux d'interaction : distribution des degrés selon les lois de puissance, un fort clustering, et une faible distance moyenne faible. Cependant nous ne pouvons pas supposer cela pour de grand graphe car le test qu'on a effectué était sur un graphe avec un nombre de nœud restreint. Donc une perspective à cour terme serai d'étudier les répercutions du passage à l'échelle sur ses propriétés.

Conclusion Générale

Les systèmes pair-à-pair sont devenus très populaires car ils offrent la possibilité aux utilisateurs de partager et d'accéder à des ressources diverses, distribuées à large échelle. Chaque pair se comporte à la fois comme client et serveur, et fournit une partie de l'ensemble des informations de l'environnement distribué sans s'appuyer sur une administration centrale.

Le réseau scall-free répond aux trois propriétés fondamentales qui caractérisent les grands réseaux d'interaction : faible distance moyenne, fort clustering, distribution des degrés selon les lois de puissance. Et son analyse nous a donné une vue plus formelle de la réalité des grands réseaux et leur complexité, ce qui nous a poussé à modéliser un réseau p2p décentralisé en utilisant les reseau Scall-Free.

Dans ce mémoire nous avons présenté une généralité sur systèmes P2P ensuite nous avons entamé les grands graphes d'interactions et leurs propriétés.

Notre objectif était d'appliquer et d'utiliser les propriétés d'étude de graphe d'interaction pour modélise un graphe pair-à-pair.

Plusieurs perspectives se présente pour ce travail, car concevoir une plate forme de simulation nécessite plusieurs étapes, notre travail représentait que la partie construction du réseau, donc parmi ses perspective on peut citer :

- Prendre l'aspect dynamique du réseau en compte (connexion et déconnexion des pairs).
- Echanges de requête entre les pairs.
- Ajout des Index de recherche pour chaque pair.
- Implémenter une couche de réseau logique.

Bibliographie

[1] Benoît Romito, Stockage décentralisé adaptatif : autonomie et mobilité des données dans les réseaux pair-à-pair, Thèse de doctorat, Université de Caen Basse-Normandie, le Décembre 2012.

[2] Bruno Defude**,** Article **:** Organisation et routage sémantiques dans les systèmes pair-à-pair**.**2007.

[3]Gharzouli Mohamed, Composition des Web Services Sémantiques dans les systèmes Peer-to-Peer, le 25 Septembre 2011.

[4] Kenniche Ahlem, Indexation de données à large échelle : mémoire de fin d'études pour l'obtention du diplôme de magister, spatialité informatique, option : informatique et automatique, Université d'Oran - Es Sénia.

[5] Margurite Fayçal, Routage efficace pour les réseaux pair-à-pair utilisant des tables de hachage distribuées, Thèse de doctorat, Ecole Doctorale d'Informatique, Télécommunication et Electronique de Paris, le Mai 2010.

[6] Stéphane Raux, Dynamiques des réseaux sociaux en ligne recommandations et interactions, Mémoire de Thèse en Informatique de l'Université Paris-Diderot Mémoire de Thèse en Informatique de l'Université Paris-Diderot, le 12 décembre 2014.

[7] Taibi Djamel et Touati Ilies, Etude et analyse spectrale des grands graphes et exploitations des ultra-pair pour la recherche de données : Mémoire de fin d'études pour l'obtention du diplôme d'Ingénieur d'Etat en Informatique**,** Université d'Oran - Es Sénia, Juin 2008.

Webographie :

[8]http://www.google.fr/Etude et utilisation des technologies des P2P.html/. Dernière date de consultation 18 Novembre 2014**.**

[9]http://www.google.fr\JFreechart : créer des graphes et diagrammes en Java .html**.** Dernière date de consultation à 20 Avril 2015.

[10]http://www.google.fr//P2P\p2p - Documentation Ubuntu Francophone.html**.**Denière date de consultation 18 Novembre 2014**.**