Agrégation des résultats dans les systèmes de recherche d'information pair-à-pair non structurés

Rim Mghirbi

Université d'Évry Val d'Essonne

Université de Tunis - El Manar

Thèse de doctorat en co-tutelle entre Télécom SudParis & La Faculté des Sciences de Tunis

Spécialité Informatique

Par :

Rim MGHIRBI

Agrégation des résultats dans les systèmes de Recherche d'Information Pair-à-Pair non structurés

Soutenue le 18 Janvier 2013 devant le jury composé de :

MM.	Sadok BEN YAHIA	M.C. FST, Tunis	Président
	Rim FAIZ	Pr. IHEC - Université de Carthage	Rapporteur
	Bénédicte LE GRAND	Pr. Université Paris 1	Rapporteur
	Philippe MULHEM	Pr. UBP, Clermont Ferrand	Examinateur
	Bruno DEFUDE	Pr. TÉLÉCOM SudParis	Directeur de thèse
	Yahya SLIMANI	Pr. ISAMM, Manouba	Directeur de thèse

Thèse n°2013TELE0003

Remerciements

Par ces quelques lignes, je tiens à exprimer mes plus vifs remerciements à toutes les personnes qui ont participé de près ou de loin au bon déroulement de cette thèse et sans lesquelles ce travail n'aurait pas pu voir le jour, en espérant n'avoir oublié personne...

Je voudrais tout d'abord exprimer ma reconnaissance envers tous les membres du jury pour la grande attention qu'ils ont bien voulu porter à mon travail.

Je tiens à remercier spécialement mes deux directeurs de thèse, Bruno Defude et Yahya Slimani, pour le temps et la patience que vous m'avez accordés tout au long de ces années en me fournissant d'excellentes conditions logistiques. Je garderai dans mon cœur votre générosité, votre compréhension et votre efficacité.

Si Yahya, vous êtes le titre de la rigueur scientifique, vos remarques m'ont bien guidée et m'ont aidée à bien faire les choses. Je n'oublierai jamais que c'était grâce à votre aide que j'ai eu l'acceptation de mon premier papier. Merci de participer à m'enseigner plusieurs règles scientifiques et je sais bien que votre rigidité a toujours été pour mon bien.

Bruno, je me rappellerais toujours de votre fameuse phrase " pense bien à manger "!!!! J'y penserais!!!!! (après la thèse). Merci d'avoir rendu notre communication assez facile et assez cool sans me stresser avant de vous demander quoi que ce soit. Merci pour toutes vos idées qui viennent ouvrir mes horizons quand j'arrive à des impasses scientifiques. Merci spécialement pour cette dernière année et toute l'aide que vous m'avez prodiguée en rédigeant ce manuscrit.

Mes plus chaleureux remerciements vont également à Khédija Arour, mon encadrante qui a dirigé mes recherches tout au long de ces années. Je vous exprime ma gratitude pour l'aide compétente que vous m'avez apportée, vos encouragements et la confiance que vous m'avez toujours témoignée. Je vous remercie aussi d'être une très bonne amie, ayant subi le plus mon stress et mes états dépressifs pendant plusieurs moments. Merci d'être toujours à mes côtés.

Je remercie très sincèrement mes rapporteurs Rim Faiez et Bénédicte Le Grand pour avoir bien accepté d'être mes rapporteurs et pour avoir bien voulu lire et évaluer mon travail de thèse. Je les remercie pour leurs lectures approfondies de mon mémoire de thèse, pour tout le temps qu'ils m'ont accordé et pour les remarques très constructives qu'ils m'ont données.

Je remercie également Sadok Ben Yahya et Philippe Mulhem qui m'ont fait honneur en acceptant de faire partie de mon jury de thèse. Leur participation dans l'évaluation de ce travail est d'une très grande valeur.

Je remercie tous les membres des équipes MOSiC et SIMBAD et des départements informatiques de la faculté des sciences de Tunis et de Télécom SudParis. Un remerciement spécial à Brigitte Houassine qui nous simplifie énormément la vie au sein du département informatique de Télecom Sud paris et à Zarrouk à la FST.

Je remercie aussi toutes les personnes que j'ai connues au cours de cette thèse et qui l'ont rendue plus agréable :Imen, Dorsaf, Mouna, Tawfik, Manel, Saloua, Amira, Hanen, Samir, Alda, Amel, Jérôme, Marie, Mohamed, Rami, Zahra sans oublier mes étudiants insatiens (de l'INSAT) Yassine, Amina, Hamza et Mohamed et mon beau frère Ahmed. Pour tout ce que vous m'avez donné, je vous remercie très sincèrement.

Je remercie mes chers amis Dora et Zied Sehil pour tout ce qu'ils ont fait pour moi pendant mes séjours en France.

La famille!!!!!

Rien à dire vous étiez le charbon qui m'a aidée à avancer. Que vous trouviez ici l'expression de ma gratitude et ma reconnaissance.

Je remercie spécialement mes parents qui se sont impliqués dans les détails quotidiens de ma vie pour me la rendre plus facile. Merci maman pour tous les pots que tu m'as préparés. Merci papa pour toutes les courses que tu m'as faites. Merci pour vos prières.

Mes beaux parents, Merci pour votre patience et votre compréhension. Je souhaite être par ce travail à la hauteur de vos attentes. Ami Rafik merci pour tes encouragements. Merci pour vos prières aussi.

J'ai dit que je te laisserai en fin de liste, je rigole... Merci d'être toi, merci d'être à côté de moi, Merci d'assumer tout mon stress et dépression. Merci d'être l'autre face de moi. Je t'ai fatigué... Je sais!!!! Rien à dire!!!! Merci, chandelle de ma vie. Je souhaite que j'assure le jour j et que je t'offre toutes mes années qui viennent. Baby

La liste est longue...!

Je remercie également ma sœur Ramla, mon frère Aymen. Vous avez attendu cet événement impatiemment. Désolée d'être trop absente. Je vous aime. Un coucou spécial à Drissou qui a allumé mes nuits en France en partageant avec moi la même chambre. Je t'aime chéri. Sarra et YouYou mes bouts de chou, j'aime "Tia" grâce à vous.

Marwaaaaaaaa, Panditta, merci pour l'encouragement et les prières je t'aime.

Hichem, Rihab, Meriem, Aymen, Ahmed, Ameni, Hbib, Ingrid vous avez subi une part de mon stress. Je vous aime tous merci.

Ma Grami Khaddouja je t'aime, voila le jour et inchallah tu es là.

Un remerciement spécial va aux familles Mghirbi, Fehri, Debbiche (mes chers oncles et tantes) et Mabrouk. Je remercie aussi mes amis qui ne cessent de m'encourager par telephone, mail et sur facebook : Yosra, Thou, Hayfa, Houda, Ramzi, Takoua, Said, sans oublier l'après-midi scientifique passé avec Leila...

J'ai le tournis.

Et je dis toujours Dieu Merci.

À l'âme de mes grands parents Ahmed, Mohamed, Jamila, Milad....

Avec mon amour.

Rim.

Table des matières

Table des figures

Liste des tableaux

Glossaire

AFC

Analyse Formelle de Concepts

BC

Borda Count

CombSum

Combinaison Somme

CORI net

COllection Retrieval Inference Network

DHT

Distributed Hash Table

GS

Global Score

IDF

Inverse Document Frequency

IPD

Importance Passée du Document

IPP

Importance Passée du Pair

KB

Knowledge Base (base des connaissances)

LM

Modèle de langage

LP

Light Profile

MAJ

Mise À Jour

MAP

Mean Average Precision

P2P

Peer-to-Peer (Pair-à-Pair)

PBA

Profile-Based Aggrégation

PV

Positional Value (Valeur positionnelle)

RARE

Routage optimisé par Apprentissage de Requêtes

RI

Recherche d'Information

RID

Recherche d'Information Distribuée

RIDLE

Recherche d'Information Distribuée à Large Échelle

RIM

Regular Increasing Monotone

RIP2P

Recherche d'Information Pair-à-Pair

RP

Précision Relative

RR

Round Robin

SDLE

Systèmes Distribués à Large Échelle

SSL

Semi-Supervised Learning

TF

Term frequency (fréquence d'un terme)

TREC

Text REtrieval Conference

Introduction Générale

Depuis son avènement, l'Internet a accéléré le développement d'applications dans tous les domaines. Les réseaux informatiques, qui constituent une infrastructure d'échange de plus en plus performante, ont permis une forte dissémination de données, de services et d'applications sur des serveurs distants, à des volumes impressionnants. Pour gérer cette évolution exponentielle, deux approches sont aujourd'hui utilisées : (i) une approche centralisée, qui trouve son incarnation la plus aboutie dans le cloud computing [Hennion 2011] ; et (ii) une approche décentralisée incarnée dans les systèmes Pair-à-Pair (P2P) [Leuf 2002]. La première approche a montré son efficacité avec des services à l'échelle planétaire, comme le moteur de recherche Google [Dean 2009], le commerce en ligne, avec Amazon ou les outils collaboratifs de type Googledocs. Les inconvénients de cette approche sont essentiellement dûs à la situation quasi-monopolistique qu'elle implique et les risques de non respect de la vie privée. De plus, elle est par définition, très coûteuse à mettre en place, et donc restreinte à de grands groupes [Tanenbaum 2006].
L'approche Pair-à-Pair prend le contre pieds de cette vision centralisatrice et propose un modèle basé sur l'utilisation d'un grand nombre de pairs, pour construire un service à grande échelle. L'application phare est le partage de fichiers (notamment musique et vidéo) [Leuf 2002, Yee 2005] sur Internet dont les principaux avantages sont, son faible coût et sa capacité à passer à l'échelle. Par contre, elle n'est pas forcément économe en ressources utilisées et le fait qu'il n'y ait pas de contrôle centralisé peut impliquer des problèmes juridiques, notamment l'absence de vérification des droits sur les ressources partagées [Nap 2006].

Contexte de travail

Dans le cadre de cette thèse, nous nous sommes intéressés essentiellement à la Recherche d'Information dans les Systèmes Distribués à Large Échelle (ou RI-SDLE) et plus précisément au problème d'agrégation des résultats dans de tels systèmes. Les défis d'une Recherche Distribuée à Large Échelle (RIDLE), dans une approche P2P (ou RIP2P) s'avèrent à première vue, analogues à ceux rencontrés dans la Recherche d'Information Distribuée classique (RID), (souvent appelée méta-recherche), dans laquelle il s'agit de :

- Sélectionner les "bons" pairs permettant de satisfaire une requête utilisateur (problème de routage requête).
- Fusionner les résultats provenant de ces pairs afin de restituer seulement les résultats intéressants du point de vue utilisateur (problème d'agrégation de résultats).

Dans le cadre de la RID, l'agrégation de résultats, est faite par un serveur courtier (le méta-moteur), détenant une vision globale du système (même si cette vision

n'est pas forcément parfaite). Ce serveur collecte et maintient des informations concernant les Systèmes de Recherche d'Information (SRI) distants, et les utilise pour rechercher les SRIs pertinents à une requête et d'en agréger les résultats. Ces informations peuvent être soit :

- **Un index global** : il s'agit de construire un index global sur une entité centralisée (généralement le serveur courtier). Cet index permet de définir une fonction qui mesure la distance (ou similarité) entre l'ensemble des documents et requêtes selon un référentiel commun. C'est le cas idéal, mais il est très coûteux à construire et maintenir, en terme d'espace et de calcul. De plus, il suppose que les SRIs acceptent de céder toutes leurs données.
- **Des statistiques globales** : pour réduire le coût de l'index global et pour augmenter l'autonomie des SRIs, des statistiques globales (ou résumés) peuvent être construites sur le courtier. Ces statistiques sont généralement construites à partir d'informations sur les index locaux (la taille des collections, les fréquences de certains termes dans les documents, les fréquences documentaires pour un ensemble de termes, etc). Ces informations, bien qu'incomplètes, permettent d'approximer un index global à un coût moindre, et en respectant mieux l'autonomie des SRIs.
- **Des scores locaux** : dans cette approche, on n'utilise pas de statistiques globales mais seulement les scores (mesures de pertinence) fournis par les différents SRIs. Ces scores sont alors utilisés pour classer les résultats en une liste unique. Lorsqu'un même document est retourné par plusieurs SRIs, un consensus doit être fait entre les différents scores retournés (par exemple prendre la moyenne, la somme, etc.), comme proposé dans [Fox 1994]). Cependant, cette approche n'est intéressante, que lorsque les collections des différents SRIs impliqués sont suffisamment homogènes (même taille des collections, ou presque, même modèles et fonctions de recherche, même type de contenu).
- **Des rangs locaux** : Cette approche peut être considérée la plus défavorable puisque l'information remontée des SRIs est restreinte aux rangs des résultats. La difficulté du classement, en se basant sur cette approche, émane de la difficulté de comparer des simples rangs, un venant de chaque SRI. Cependant, cette approche présente l'avantage de maximiser l'autonomie des SRIs. En plus, si le système est très homogène (en termes de collections, de modèles de RI, de contenus), une agrégation basée sur les rangs locaux peut donner de bons résultats qui peuvent concurrencer l'utilisation de statistiques globales [Aslam 2001].

Dans ces différentes approches (hormis l'utilisation d'un index global), un recours à l'apprentissage peut être utilisé conjointement à ces approches afin d'aider soit à normaliser des informations locales, soit à récupérer un échantillon des index locaux des collections des SRIs.

Problématique et principales contributions

Le problème d'agrégation en RIP2P ne peut être abordé de la même manière que dans le cadre distribué classique. En effet, en RIP2P, le facteur d'échelle augmente fortement puisqu'on passe de quelques dizaines de SRIs, dans la RID, à plusieurs centaines, voire des milliers, en RIP2P. Cela rend, notamment, le recours à un serveur courtier inadéquat.

De même, le recours à l'index global est une solution coûteuse qui risque de surcharger le réseau (pour maintenir l'index à jour) et d'avoir des conséquences sur le passage à l'échelle. L'utilisation de statistiques globales est envisageable, à condition qu'un système contribuant à la réponse accepte de diffuser des informations sur lui-même. De plus, à ce facteur d'échelle, il est sûr que les SRIs vont offrir un niveau d'hétérogénéité fort, avec lequel, il sera très difficile d'interpréter correctement les informations locales (que ce soit des statistiques, des scores, voire même, des rangs). C'est ainsi que notre approche doit être en mesure de respecter toutes les caractéristiques des systèmes distribués à large échelle, en général, et des systèmes de RIP2P, en particulier, à savoir essentiellement, l'hétérogénéité des collections, la décentralisation totale et le passage à l'échelle. De même, étant une solution de RI, notre approche doit être efficace en terme de mesures d'évaluation mais surtout du point de vue utilisateur.
Nous proposons donc de suivre l'approche à base de rangs, en l'étendant par l'utilisation d'informations visant, notamment, à prendre en compte l'hétérogénéité des SRIs. À cet effet, nous proposons d'utiliser les préférences des utilisateurs recensées localement à partir de l'historique de leurs interactions avec les réponses fournies à leurs requêtes passées (profils). L'intuition derrière ce choix a été motivée par le fait que, d'une part, le feedback de l'utilisateur est un véritable juge de la qualité des résultats qui lui sont fournis en réponse à sa requête et que, souvent, cet utilisateur ré-exprime des besoins similaires à ses requêtes passées (au moins sur une période donnée), d'autre part.
En se basant sur l'hybridation des poids issus des interactions passées de l'utilisateur avec le système (les profils), avec les rangs, nous calculons des scores globaux pour l'ensemble des documents retournés. Par ailleurs, nos contributions portent sur deux principaux aspects :
(i) la définition et la formalisation d'un profil centré utilisateur, utilisable dans un contexte de large échelle ; et (ii) la définition d'une méthode hybride de calcul de score global qui intègre la connaissance issue de l'utilisateur.

La réalisation des contributions citées a passé à travers l'étude de plusieurs problématiques secondaires, à savoir :

- Quelle connaissance faut-il utiliser ?
- Comment la faire évoluer, en fonction des évolutions du système et des besoins utilisateurs ?
- Comment calculer les scores ?
- Comment évaluer la validité de nos propositions ?

Quelle connaissance faut-il utiliser ?

Nous proposons ici d'utiliser l'association entre les requêtes passées, les pairs qui y ont contribuées et les retours (feedbacks) des utilisateurs. Pour cela, nous avons étudié la possibilité d'extraire automatiquement la relation pairs-termes pour rechercher l'ensemble des pairs souvent sollicités dans la réponse à un ensemble de termes de requêtes données et l'ensemble de documents sollicités pour un même ensemble de termes et réunir cet ensemble "sémantique" dans un profil enregistré dans une base des Connaissances (appelée dans la suite $\mathcal{KB}$). Afin d'extraire un ensemble de corrélations, l'utilisation des règles associatives a été possible. Cependant, afin de réduire l'ensemble de règles produites par cette technique, nous avons opté plutôt pour l'Analyse Formelle de Concepts qui a été utilisée de manière adaptée (nous ne parlons plus d'un domaine et d'un co-domaine mais plutôt d'un triplet,(T,P,D).

Comment faire évoluer la base des connaissances en fonction des évolutions du système et des besoins utilisateurs ?

L'évolution de la base des connaissances est nécessaire pour maintenir l'efficacité de l'agrégation. Une évolution simplement basée sur le temps est inadéquate, car elle peut nous amener dans une des situations suivantes : (i) la base est mise à jour alors que le système n'a pas encore évolué, ou (ii) la base n'est pas mise à jour alors que le système a lui évolué. Nous étudions dans cette thèse, une approche d'évolution basée sur l'estimation de l'écart entre l'état du système et celui des bases de connaissances.

Comment calculer les scores ?

L'utilisation des profils doit nous permettre de calculer deux types de scores, (i) un score de pair contributeur basé sur la confiance donnée à ce pair parce qu'il a su bien répondre à des requêtes passées "similaires" à la nouvelle requête et (ii) le score du document qui se base sur le même principe. A cet effet, une formule de score hybridant les rangs et les scores basés sur les profils et calculée de manière purement distribuée sera proposée. En fait, il s'agit de "contextualiser" le rang qui sera attribué à chaque réponse retournée pour un besoin spécifique. L'intégration de ce contexte dans la phase d'agrégation permet de rendre l'information retournée à l'utilisateur plus intelligible.

Comment évaluer ?

La quatrième problématique de cette thèse est de concevoir un cadre pour l'évaluation des Systèmes de Recherche d'Information distribuée à large échelle. Le problème dans ce contexte, consiste à définir :

– Les métriques à utiliser pour évaluer.

– Les collections de test à utiliser pour tester l'efficacité de l'approche d'agrégation proposée.
– La plate-forme d'évaluation à choisir pour évaluer les résultats.

Les métriques classiques telles que la précision et le rappel semblent une solution naturelle au premier problème. Cependant, elles ne prennent en compte que l'efficacité du système relativement à la Recherche d'Information. Dans un système distribué à large échelle, l'aspect performance du système doit être prise en compte. Cela comprend, notamment, les coûts de communication et de stockage de données. Le problème est que efficacité et performance sont, en général, antagonistes. Ainsi, c'est le compromis efficacité/performance que nous allons évaluer.

Pour répondre au deuxième problème, il manque aujourd'hui des collections de test décentralisées gratuites et bien établies dans la communauté comme TREC [Voorhees 2005]. Par collection de test, nous entendons un ensemble de requêtes, un ensemble de documents et des jugements de pertinence d'un ensemble de documents par rapport aux requêtes. Ainsi, l'un des problèmes importants de notre thèse est de construire des collections décentralisées. De plus, les caractéristiques des systèmes P2P doivent être respectées dans la construction de ces collections, à savoir, l'hétérogénéité en tailles et contenus des collections et des modèles de Recherche d'Information. Pour les jugements de pertinence, nous ne cherchons pas ici à avoir une évaluation dans l'absolu, mais à nous comparer aux systèmes centralisés. C'est la raison pour laquelle, nous utiliserons, comme jugements de pertinence, les résultats retournés par une version centralisée de notre système.
Quant au choix de la méthode d'évaluation, nous sommes amenés à choisir entre une des solutions suivantes :

– Mettre en place un environnement distribué à large échelle, mais c'est bien évidemment coûteux en nombre de ressources demandées et difficile à réaliser.
– Utiliser une plate-forme existante d'évaluation distribuée à large échelle comme Grid5000 et reproduire les conditions réelles du système.
– Simuler le système distribué avec ses différents nœuds en considérant une seule machine.

Organisation du mémoire

L'ensemble de nos propositions sont résumées dans ce mémoire en six chapitres, en plus d'une introduction générale et d'une conclusion générale. Les deux premiers chapitres présentent le contexte dans lequel se situent nos travaux.
Le chapitre 1 décrit les éléments de base de la Recherche d'Information, ses modèles et son évolution depuis les années 50, jusqu'à l'arrivée de la Recherche d'Information Distribuée à Large Échelle (RIDLE). Nous décrivons particulièrement, la Recherche d'Information en environnement Pair-à-Pair (RIP2P), en partant des systèmes P2P et en introduisant la notion de P2P sémantique.
Le Chapitre 2 est consacré à une revue critique de la littérature concernant les

approches d'agrégation classiques adoptées en RID et en RIP2P. Nous présentons également quelques approches basées sur les comportements des utilisateurs ou profils.
À la lumière des limites de ces approches, quant à leur adéquation aux propriétés intrinsèques des systèmes P2P, le chapitre 3 vient proposer une nouvelle méthode d'agrégation de résultats basée sur l'hybridation des rangs et des scores utilisant les profils des utilisateurs. Un modèle de profils basé sur l'Analyse Formelle des Concepts est proposé. Il définit un ensemble d'opérateurs indispensables à la manipulation des profils.
Afin de valider notre méthode d'agrégation, nous définissons dans le chapitre 4 une démarche d'évaluation basée sur la réutilisation des métriques d'efficacité quantitatives et la proposition de deux métriques qualitatives permettant de comparer l'efficacité d'un classement par rapport à un classement de référence (celui de la collection centralisée). Une étude de performance de notre algorithme d'agrégation est également discutée. La validation de l'efficacité et la performance constituent l'objet du chapitre 5 qui propose un cadre d'expérimentation basé sur la simulation. Différents scénarii, visant à appréhender des situations réalistes, sont décrits. Les résultats des expérimentations menées sur trois collections de test construites sont présentés à la fin de ce chapitre.
La force de notre méthode d'agrégation à base de profils vient de sa capacité à profiter, le mieux possible, de ce que le système a appris au cours de la résolution des requêtes passées. Cela suppose donc que notre base des connaissances ou de profils ($\mathcal{KB}$) soit toujours cohérente par rapport à l'état courant du système, ce qui nécessite un mécanisme d'évolution. Ainsi, Nous proposons dans le chapitre 6 un mécanisme d'évolution contrôlée de notre base des connaissances ($\mathcal{KB}$).
Enfin, la conclusion générale synthétise les principaux apports de la thèse et propose quelques perspectives de prolongement de ce travail.

Chapitre 1

De la Recherche d'Information centralisée à la Recherche d'Information à Large Échelle

1.1 Introduction

La Recherche d'Information (RI) est un problème qui, malgré son ancienneté, continue à s'imposer comme un besoin essentiel à la plupart des utilisateurs. En effet, le besoin d'avoir de l'information pertinente est crucial d'autant plus que les systèmes de partage de contenus évoluent et prennent plusieurs formes qui, dans la plupart des cas cohabitent ensemble. L'information est disponible sous forme de grands volumes sur des serveurs centralisés, décentralisés, dans les blogs, chez des utilisateurs, chez des communautés, etc [add online 2009, Informatica 2010].
Les concepts de base de la recherche d'information sont toujours adoptés mais selon le système de partage d'information et le volume de données, de nouvelles contraintes affectant toutes les phases d'un processus de recherche d'information apparaissent et les techniques de recherche sont appelées à évoluer.
Dans ce chapitre, nous présentons les concepts de base d'un système de Recherche d'Information. Nous décrivons les principes de base de la Recherche d'Information, puis nous détaillons les phases de la Recherche d'Information Distribuée pour arriver à la fin aux problèmes spécifiques aux systèmes de Recherche d'Information largement distribués tels que les systèmes Pair-à-pair [Defude 2007]. Nous évoquons de même, le défi de l'intégration de la sémantique, en vue d'améliorer les processus de recherche, dans les systèmes Pair-à-Pair.

1.2 Les Systèmes de Recherche d'Information Centralisés

La Recherche d'Information (RI) documentaire est un concept qui remonte aux années 50 lorsque Mooers [Mooers 1950] a inventé ce terme pour la première fois. Mooers définit un Système de Recherche d'Information (SRI) comme un système d'information permettant, à une personne exprimant un besoin, d'extraire, à partir d'une collection documentaire, les documents qui correspondent à son besoin.
Cette définition met en évidence deux éléments essentiels dans un processus de

Recherche d'Information : une **requête** et un ensemble d'informations appelé collection, à partir duquel le système délivre des **ressources pertinentes** à la requête utilisateur. La pertinence d'une réponse par rapport à une requête est généralement déterminée via des métriques d'évaluation que l'on détaillera plus tard dans le chapitre 4.

De manière formelle, et en référence au modèle de [Baeza-Yates 1999], un SRI peut être défini comme un quadruplet $M = (D, Q, F, S(q_i, d_j))$, tel que :

- D : représente un ensemble de vues logiques pour des documents dans une collection.
- Q : est un ensemble de vues logiques pour des requêtes utilisateurs.
- F : ensemble de modèles de représentation de documents ou de requêtes.
- $S(d_j, q_i)$: fonction de classement ou plus précisément de similarité du document d_j par rapport à la requête q_i.

Tous ces éléments sont intégrés dans le processus de recherche d'information dont nous décrirons l'architecture dans la section suivante.

Processus de recherche d'information centralisé

Les SRIs sont largement utilisés dans différents domaines d'activités [Andrieu 2011]. Les systèmes "historiques" les plus connus, sont Smart dans les années 80 [Salton 1989] et Medline, qui se présente parmi les systèmes commerciaux, actuellement accessible depuis [PubMedline 2012]. Le concept de SRI a été popularisé avec l'avènement du web sous la forme des moteurs de recherche dont les plus connus sont notamment, le leader Google [Page 1998] ou "son equivalent" Bing de Microsoft [center 2012] .
La Figure 1.1 représente les briques de base d'un SRI centralisé. Un SRI doit être en mesure d'indexer, d'une part les documents présents dans son corpus documentaire et d'autre part, les besoins en information exprimés par les utilisateurs, puis de rechercher les documents les plus pertinents par rapport à une requête utilisateur et de les ordonner.
Compte tenu des éléments du quadruplet M, un processus de recherche d'information centralisé comporte au moins trois modules de base : le module d'indexation, le module d'interrogation et le module d'appariement.

Module d'indexation

Le module d'indexation consiste à représenter l'ensemble de documents présents dans un corpus documentaire, sous forme d'un ensemble de vues logiques [Baeza-Yates 1999], visant à repérer les contenus de ces documents, afin d'en faciliter la recherche et l'accès.
L'indexation concerne également les requêtes afin de les écrire dans un langage Q qui est l'ensemble de vues logiques représentant une requête (ce qui correspond au deuxième élément du quadruplet M) comparable à celui des documents (D). Souvent, la même méthode d'indexation est utilisée pour indexer aussi bien les do-

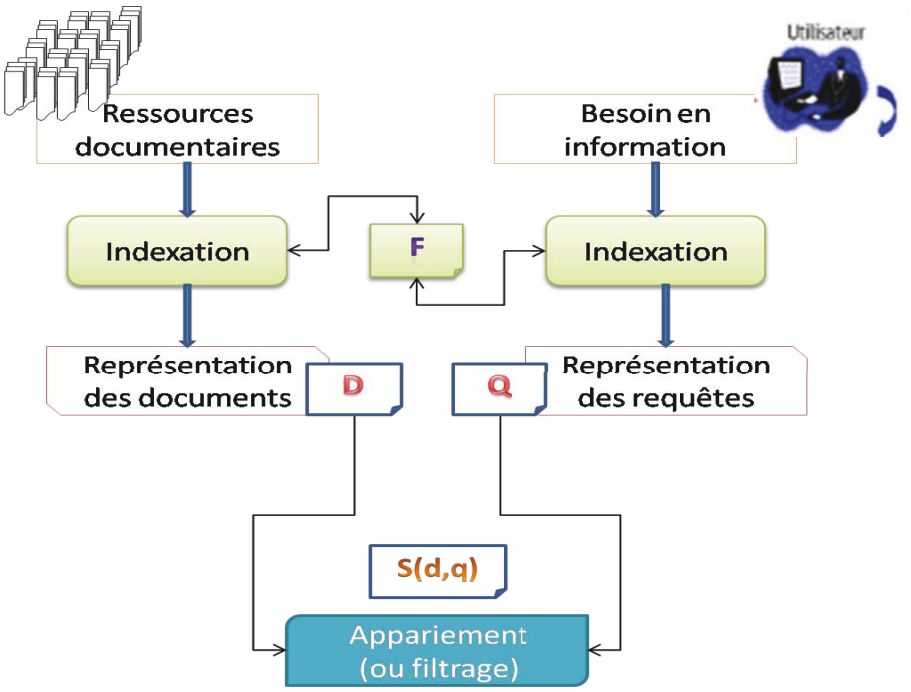


FIGURE 1.1 – Processus de recherche d'information centralisé

cuments que les requêtes. Ainsi, pour rechercher une information à partir d'une collection de données, un SRI doit accéder à ces informations d'une façon unique en comparant une requête à un ensemble de clés formant l'index de la collection. La façon par laquelle on représente les documents ou les requêtes dépend d'un modèle de représentation F. Plusieurs modèles de représentation de contenus existent [Salton 1975, Ponte 1998, Baeza-Yates 1999, Savoy 2001].

Dans la section 1.2.1, nous décrivons succinctement les modèles piliers de la RI : le modèle booléen, le modèle vectoriel et le modèle probabiliste.
Le modèle vectoriel de Recherche d'Information, par exemple représente les documents par une liste de termes couplés avec la fréquence de ces termes dans le document (tf), ainsi que l'inverse des fréquences des documents contenant ces termes dans la collection (idf).
La requête doit être représentée de la même manière afin d'être appariée aux bons contenus dans cette collection. Des méthodes plus récentes de représentation de données ont continué à émerger, telles que les représentations sémantiques à base d'ontologies [Köhler 2006].

Module d'interrogation

Après avoir indexé une requête, le module d'interrogation lance la recherche sur les documents afin d'en sélectionner ceux qui correspondent à la requête. Cepen-

dant, cette recherche peut ramener un nombre très important de documents, ou, au contraire, un nombre très réduit parce que la requête est soit, trop généraliste (ou ambiguë), soit trop spécialisée.
Plusieurs propositions ont été faites pour pallier ces problèmes. La première catégorie utilise l'expansion des requêtes, c'est-à-dire la réécriture d'une requête initialement formulée par l'utilisateur en lui ajoutant des termes, ou en modifiant les termes utilisés [Xu 1996]. La deuxième catégorie utilise un feedback de l'utilisateur [Rocchio 1971] (relevance feedback ou bouclage de pertinence) sur le résultat retourné par la requête pour la reformuler selon le sens souhaité par l'utilisateur. Ce feedback peut être explicite ou implicite et peut d'ailleurs être soit positif (documents jugés comme particulièrement pertinents), soit négatif (documents jugés comme particulièrement non pertinents).

Le module d'appariement ou de filtrage

Ce module constitue le cœur de tout le processus de Recherche d'Information. En effet, c'est au cours de la phase d'appariement qu'un Système de Recherche d'Information produit les réponses jugées adéquates satisfaisant le mieux possible le besoin de l'utilisateur. Ceci passe par l'utilisation d'une fonction d'estimation de la similarité (ou fonction de scoring) $S(d_i, q_j)$ entre le document d_i et la requête q_j. Faisant référence au score calculé par la fonction de similarité, un SRI procède généralement au tri des réponses qu'il retourne à l'utilisateur (voir Figure 1.1).
La pertinence des réponses retournées par un SRI et leur ordre dépendent d'une procédure d'évaluation qui vise à réduire la distance entre les réponses du système (ou pertinence système) et les intensions de l'utilisateur (ou pertinence utilisateur). La fonction de similarité utilisée est très différente d'un SRI à un autre et dépend du modèle de RI adopté par le système. Les Systèmes de Recherche d'Information intègrent différents algorithmes pour mesurer la similarité d'une représentation d'un document relativement à celle d'une requête et pour évaluer par conséquent la pertinence du document relativement à la requête. Par exemple, Les fonctions cosinus, dice, jaccard et overlap [Egghe 2010] constituent une famille de fonctions de similarités largement utilisées dans le modèle vectoriel [Salton 1989] (voir plus loin dans ce chapitre). Le modèle probabiliste utilise un autre ensemble de fonctions de correspondance, telle que la fonction okapi BM25 [Robertson 1995]. Des fonctions de similarité sémantiques existent également, comme par exemple, la similarité de Salton [Salton 1989].

Les méthodes d'appariement utilisent des logiques différentes pour classer les documents par rapport à une requête. De ce fait, elles peuvent diverger nettement dans l'ordre qu'elles affectent à leurs réponses même s'il s'agit de classer les mêmes réponses retournées. Cette divergence de classement peut être remarquée pour une même famille de méthodes d'appariement. Ainsi, une fonction *cosinus* peut donner un ordre différent de celui donné par *dice* pour la réponse à une requête q. Ceci pose le problème de l'estimation de l'efficacité des modèles de RI et rend essentiel

le recours à leurs évaluations.
Des métriques d'évaluation doivent être utilisées. La précision et le rappel sont les plus communément utilisées et nous les détaillerons dans le chapitre 4.

Nous présentons dans ce qui suit quelques modèles de base pour la Recherche d'Information. Ces modèles proposent des approches de représentation (F) et des fonctions d'appariement $S(d_i, q_j)$ différentes.

1.2.1 Revue de quelques modèles de Recherche d'Information

Plusieurs variantes de modèles de Recherche d'Information existent dans la littérature [Greengrass 2000]. Nous avons choisi de présenter les premiers modèles qui sont considérés comme piliers dans le domaine de la Recherche d'Information. La table 1.1 résume les notations utilisées pour illustrer les différents modèles.

Symbole	Signification
q	requête
d	document
i	un terme particulier d'une requête q
j	un terme d'un document d
$tf(i,j)$	la fréquence du terme i dans le document d sachant les termes j
$df(i)$	fréquence des documents contenant le terme i
$idf(i)$	inverse de fréquence documentaire $df(i)$

TABLE 1.1 – Table des notations - Similarité requête-réponse

1.2.1.1 Modèle booléen

Le modèle booléen constitue la première famille des modèles de Recherche d'Information. Issu de la théorie des ensembles [Johnson 1972] et de l'algèbre de Boole [Boole 1848], il offre une logique simple pour la recherche. Une requête est représentée sous forme d'un ensemble de termes reliés par des opérateurs logiques (AND, OR, NOT). Un document est également représenté sous forme d'un ensemble de termes.
La fonction d'appariement retourne tous les documents dont la représentation vérifie la requête, c'est-à-dire qui satisfont la condition logique portant sur les termes de la requête. Le modèle booléen est considéré comme exact puisqu'il ignore la notion de correspondance partielle (i.e, un document satisfait la requête ou ne la satisfait pas, mais il ne peut la satisfaire partiellement). Pour cette raison, aucun classement de documents relativement à une pertinence n'est fourni au sein de ce modèle.

1.2.1.2 Modèle vectoriel classique

Le modèle vectoriel est l'un des modèles phares pour la Recherche d'Information Web ou la recherche textuelle classique. Il fournit un espace pour représenter les documents et les requêtes sous forme de vecteurs de poids de termes. Chaque poids est

calculé sur la base des informations explicitées dans une matrice termes-documents. Ces informations concernent essentiellement les valeurs de :

- $tf(i,j)$
- $df(i)$

Souvent, c'est l'inverse de la fréquence documentaire qui est considéré afin de présenter le pouvoir discriminatif d'un terme donné : l'*idf* [Salton 1975]. Les schémas de pondération pour la représentation des documents et des requêtes sont multiples [Robertson 1976, Lan 2005]. L'un des modèles les plus utilisés consiste à utiliser la formule suivante :

$$W_{ij} = tf(i,j) \times idf(i) \tag{1.1}$$

Sur la base de cette représentation de termes, plusieurs fonctions de similarité de documents par rapport à une requête $S(d_i, q_j)$ existent [Greengrass 2000]. Elles sont dérivées de la relation géométrique entre les vecteurs document et requête dans un espace à t dimensions (t étant le nombre de termes d'indexation). Les fonctions les plus connues sont [Egghe 2010] :

$$Cosine(d,q) = \frac{\sum_{k=1}^{n} w_{ki} \times w_{kj}}{|d_i| \times |q_j|} \tag{1.2}$$

$$Dice(d,q) = \frac{\sum_{k=1}^{n} w_{ki} \times w_{kj}}{\sum_{k=1}^{n} w_{ki} + \sum_{k=1}^{n} w_{kj}} \tag{1.3}$$

$$Jaccard(d,q) = \frac{\sum_{k=1}^{n} w_{ki} \times w_{kj}}{\sum_{k=1}^{n} w_{ki} + \sum_{k=1}^{n} w_{kj} - \sum_{k=1}^{n} w_{ki} \times w_{kj}} \tag{1.4}$$

$$Overlap(d,q) = \frac{\sum_{k=1}^{n} w_{ki} \times w_{kj}}{\min(\sum_{k=1}^{n} w_{ki}, \sum_{k=1}^{n} w_{kj})} \tag{1.5}$$

La liste finale de documents retournés à l'utilisateur sera produite par le classement des documents selon les scores calculés. Le principe du modèle vectoriel est de fournir une approximation effective des propriétés statistiques du système de recherche étudié. Cependant, le modèle représente une difficulté d'interprétation et délaisse toute dépendance sémantique entre les termes indexés, notamment les relations de synonymie (plusieurs termes pour désigner le même concept) et polysémie (même terme désignant des concepts différents). Ceci augmente fortement l'espace des termes dans la matrice documents-requêtes finale et constitue une vraie limite de l'approche.

1.2.1.3 Modèle probabiliste

Le modèle probabiliste modélise le problème de RI en utilisant un cadre probabiliste [Fuhr 1992]. Dans une logique probabiliste, toute requête possède un ensemble de réponses idéales sur la base de la description de cet ensemble. Le modèle probabiliste repose sur une estimation de la probabilité qu'un document sera pertinent vis-à-vis d'une requête utilisateur. Il suppose que cette probabilité dépend seulement de la requête et des représentations de documents. L'ensemble de réponses idéales selon la logique de ce modèle doit maximiser cette probabilité de pertinence.

La représentation des documents et des requêtes dans le modèle probabiliste est basée, dans la plupart des cas, sur le schéma $tf \times idf$ [Robertson 1976].
Quant aux fonctions d'appariement, plusieurs ont été adoptées, parmi lesquelles, nous pouvons citer la fonction BM25 utilisée par le moteur de recherche OKAPI [Robertson 1995]. Elle constitue l'une des méthodes probabilistes de référence dans le domaine de la Recherche d'Information. Le calcul des scores de documents est régi par le principe suivant : étant donné une requête q, constituée des termes i, le score BM25 d'un document d se calcule selon la formule suivante :

$$score(d,q) = \sum_{i=1}^{n} idf(i) \times \frac{tf(i) \times (k_1 + 1)}{tf(i) + k_1 \times (1 - b + b \times (\frac{|d|}{avgdl}))} \tag{1.6}$$

Où :
- $tf(i)$ est la fréquence du terme i de la requête q dans le document d,
- $|d|$ est le nombre de termes contenus dans le document,
- $avgdl$ constitue la longueur moyenne d'un document.
- k_1 et b sont des paramètres d'ajustement.

Il est à noter que la division par la longueur moyenne du document est la méthode de normalisation proposée par OKAPI et que les valeurs des paramètres qui sont considérées comme efficaces pour l'approche sont respectivement 2.0 pour k_1 et 0.75 pour b [Robertson 1995].
idf se calcule selon l'approche d'Okapi comme suit :

$$idf(i) = \log \frac{N - n(i) + 0.5}{(n(i) + 0.5)} \tag{1.7}$$

où N est le nombre total de documents retournés par le système de recherche et $n(i)$ est le nombre de documents contenant le terme i.
Comparée à la fonction de similarité vectorielle, $BM25$ donne toujours de meilleurs résultats comme présenté dans [Savoy 2001, Savoy 2002, Lu 2005]. Cependant, la méthode souffre des mêmes limites recensées dans les méthodes de pondération du modèle vectoriel car elle est basée sur un espace de termes et suppose leur indépendance.

Après avoir présenté les concepts de base de la RI, nous allons présenter une expérience pratique qui fait référence dans le monde de la recherche d'information en général et la recherche web en particulier, à savoir Google.

1.2.2 Google : une solution centralisée pour de très gros volumes

Pour la grande communauté des utilisateurs d'Internet, la Recherche d'Information est une opération facile et quotidienne qui se compose de 3 phases : (i) taper

un ensemble de mots clé dans le moteur de recherche Google [Page 1998] ; (ii) récupérer les réponses sur quelques dizaines de pages ; et, (iii) consulter les meilleures réponses généralement sur les 10 premières réponses retournées. Google étant le moteur de recherche le plus présent sur le web, ceci explique pourquoi le concept de Recherche d'Information s'est fortement ancré chez le grand public avec le moteur de recherche Google [Dean 2009].
Pour l'indexation, Google utilise des robots logiciels (les spiders) qui, à partir de quelques points d'entrée, vont explorer en profondeur et ce récursivement, toutes les pages qui leur sont liées. Chaque page explorée est envoyée sur le site central de Google où elle va être indexée et contribuer à la construction d'un gigantesque index (termes - documents).

La recherche consiste, quant à elle, à rechercher dans cet index toutes les pages incluant les termes de la requête utilisateur, ce qui permet de retrouver en général un très grand nombre de résultats.
L'avancée principale de Google est sa capacité à indexer des milliards de pages et à traiter en simultané des milliers de requêtes utilisateurs. Ceci est possible par le recours à des dizaines de gros clusters de serveurs gérant cette charge en parallèle (cela représente aujourd'hui plusieurs dizaines de milliers de serveurs).
En ce qui concerne le classement des pages, Google intègre plusieurs critères de décision dont le plus important est le "PageRank" [Brin 1998] ou popularité d'une page. L'algorithme pageRank consiste à mesurer l'importance d'une page en fonction du nombre de liens externes qui mènent vers cette page. D'autres critères considèrent les relations de synonymies, l'importance des termes de la requête dans les pages sélectionnées et leurs fréquences [Andrieu 2011].
Google a réussi à avoir un bon compromis entre le nombre de documents indexés, la fraîcheur des index et le traitement des requêtes par seconde [Dean 2009]. En effet, la réponse à une requête ne dépasse pas aujourd'hui $0.5s$ avec des taux de mise à jour très importants (les pages les plus dynamiques selon le log du moteur d'index sont mises à jour quotidiennement et même plusieurs fois par jour alors qu'auparavant les mises à jour se faisaient mensuellement) [Dean 2009].

Google propose également des produits de recherche spécialisés tels que *Google scholar* pour la recherche scientifique, *Google Maps* pour la recherche géographique, *Google Books* pour l'exploration des livres, etc.
Malgré l'investissement énorme rendu nécessaire pour mettre en place et maintenir l'infrastructure de dizaines de milliers de serveurs, Google offre cependant, des performances limitées. En effet, le principe d'exploration des robots ne garantit pas l'exhaustivité de l'indexation et des pans entiers de l'Internet ne sont pas indexés par Google. On peut citer notamment tout ce qui est qualifié de web caché (the hidden web) et de web profond [Bergman 2001] (the deep web), c'est-à-dire les informations présentes derrière des formulaires web ou bien qui nécessitent une identification préalable. L'autre limite porte sur la qualité de la RI proposée par Google. Google utilise une approche purement statistique applicable sur un corpus

énorme et très hétérogène (taille, structure, langues, etc).

Enfin, pour des raisons économiques (les grands coûts dépensés par Google), il n'a pas de place pour un autre Google, ce qui rend la mise en place de nouveaux systèmes centralisés à l'échelle du web impossibles (même si on vise des sous-ensembles restreints à une langue ou une spécialité, par exemple). Ceci laisse la place à d'autres approches basées sur une vision décentralisée moins coûteuse à mettre en place.

Avant de s'attaquer à la vision distribuée de recherche, nous proposons une revue de l'histoire de l'evolution de la RI.

Dans ce qui suit, nous tenons à retracer les principaux événements qui ont impacté le monde de la RI et ses principales évolutions. Les événements concernent le côté matériel, le volume de données et les principales innovations.

1.3 Evolution de la Recherche d'Information

Nous présentons, à travers la Figure 1.2, les différentes dimensions qui expliquent l'évolution de la Recherche d'Information. Ces dimensions sont notamment, les *larges volumes de données*, *l'échelle de la distribution* ainsi que l'avènement de nouveaux *environnements et plateformes logiciels et matériels*.

Dans la figure 1.2, les dimensions horizontales représentent la progression des principaux événements depuis l'année 1962, peu après l'avènement du concept de RI, jusqu'à aujourd'hui où coexistent le web 2.0, le P2P, les grilles, le cloud, les équipements ubiquitaires, etc.

Nous présentons sur l'axe vertical à gauche, l'impact des événements Internet sur l'augmentation du volume des données échangées. Ces données ont évolué de l'échelle de quelques Ko en 1984 [add online 2009] jusqu'à 1.2 Zo en 2010 (1.2 million de péta octets) [Informatica 2010]. L'échelle traduite par le nombre d'équipements connectés sur Internet est représentée sur l'axe vertical à droite comme une autre conséquence de l'évolution technologique accentuée par l'évolution des événements Internet. L'échelle est passée de 1000 composants connectés en 1984 à 1000,000,000,000 prévus en 2013 [Informatica 2010].

Nous présentons dans ce qui suit, par référence à la Figure 1.2, les étapes importantes par lesquelles est passé le concept de RI. Nous parlerons du passage d'une recherche centralisée classique vers une Recherche d'Information Distribuée à Large Échelle (RIDLE).

A l'ère pré-Internet, avant les années 90, les travaux de RI se sont focalisés sur les thèmes d'indexation, d'interrogation et de filtrage. Des modèles de recherche ont été mis en œuvre, notamment, le modèle vectoriel classique [Salton 1975], le modèle booléen [Lancaster 1973], etc.

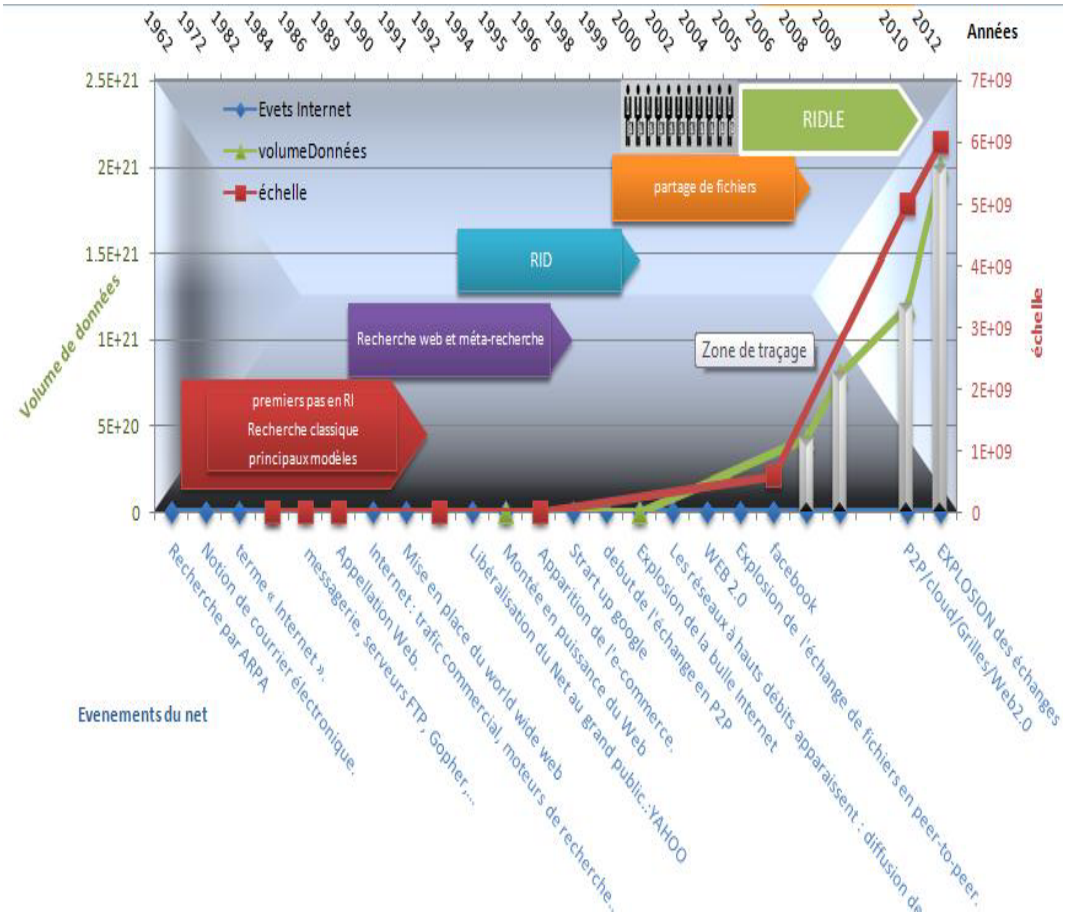


FIGURE 1.2 – Principales étapes d'évolution de la RI

À l'époque, il s'agissait d'une recherche centralisée classique dont l'objectif essentiel était l'amélioration de la qualité des résultats.

La notion du réseau s'est bien imposée peu après les années 50 pour relier effectivement un nombre très reduit d'ordinateurs. La recherche est restée toujours centralisée en utilisant un serveur qui gère l'indexation des données, la réception des requêtes clients et la restitution des résultats.

Durant les années 90 après l'avènement d'Internet et du Web (voir Figure 1.2), le volume de données a augmenté de façon remarquable pour atteindre 0.9 To en 1995 comparé à 2Ko en 1984 [add online 2009, Informatica 2010]. L'année 1995 fut comme l'indique la Figure 1.2, l'année de la montée en puissance du Web qui s'est vu rapidement renforcé par l'apparition du moteur de recherche Google en 1998. Le nombre d'ordinateurs connectés a atteint à ce moment, plus de 10000000 ordinateurs [add online 2009, Informatica 2010]. Ainsi, on a associé la recherche

d'information en web et on a commencé à paler de recherche Web. L'émergence du Word Wide Web a créé une révolution informationnelle par l'offre de grandes possibilités aux internautes pour créer et rechercher de l'information à travers un support logique appelé page web. Ces pages peuvent avoir sans aucun contrôle un contenu modifié ou répliqué. Le changement rapide de contenu des pages web pose un problème de mise à jour presque instantané des index des moteurs de recherche afin de suivre les changements de contenu.

Aux débuts du Web, la compétition porte sur le nombre de pages indexées et sur la qualité du processus de classement des pages. L'incomplétude des moteurs de recherche (le fait de n'indexer qu'une fraction du web) a conduit à l'émergence de méta-moteurs, couche logicielle permettant à un utilisateur d'interroger un ensemble de moteurs via une interface unique. Cette couche logicielle gère d'une part la diversité des langages de requêtes des différents moteurs et leur choix éventuel relativement à une requête donnée. Un nouveau problème qui apparaît est celui de la construction de la liste finale des résultats, à partir des listes retournées par les différents moteurs : comment interclasser des résultats retournés par des moteurs hétérogènes ? Comment classer un document retourné par plusieurs moteurs ?, etc.

Les méta-moteurs sont le premier exemple d'une nouvelle génération de RI, appelée, RI Distribuée (RID). Les principes de cette nouvelle génération seront présentés plus en détails dans la section 1.4.

Cette génération a ensuite évolué par la prise en compte de volumétries toujours plus importante. Ceci a donné lieu à ce que nous appelons la Recherche d'Information Distribuée à Large Échelle (RIDLE) dont l'une des plus importantes variantes est la Recherche d'Information en Pair-à-Pair (RIP2P).

En effet, selon la Figure 1.2, nous remarquons que l'année 1999 a été marquée par la mise en œuvre du système Napster [Nap 2006] pour le partage de fichiers en Pair-à-Pair (P2P). Le développement des systèmes P2P a été renforcé par l'apparition massive des Réseaux Haut Débits en 2002. Le partage d'information en P2P suit la tendance du centrage des applications autour de l'utilisateur. Ce dernier est en effet, à la fois, utilisateurs de ressources (notamment celles des autres pairs) mais aussi contributeur en partageant ses propres ressources (et/ou) en offrant des ressources de calcul ou de stockage.

L'échange d'information entre pairs, a contribué à l'augmentation exponentielle du volume de données échangé pour atteindre les 3.72 To avec un nombre d'ordinateurs connectés dépassant les 25000000 (voir Figure 1.2). L'échange en P2P a soulevé un engouement spécial pour la communauté des utilisateurs vu sa facilité et son accès à des ressources quasi-illimitées.

Dans ce qui suit, nous allons présenter les principes de la Recherche d'Information Distribuée.

1.4 Principe de Recherche d'Informations Distribuée

La Recherche d'Information Distribuée (RID) est un type de recherche visant à accéder de façon transparente à un ensemble de sources d'informations distribuées partir d'un point unique qu'on appelle serveur courtier. Ce dernier reçoit les requêtes des utilisateurs, choisit les sources adéquates pour satisfaire la requête posée et la transmet aux sources sélectionnées. Chaque source se charge de retourner une liste ordonnée de résultats (documents) selon sa propre stratégie de recherche. À ce stade, le serveur courtier doit restituer une liste combinée et ordonnée de résultats à l'utilisateur. La recherche distribuée soulève trois principales problématiques : la description des sources distribuées, leur sélection et la fusion de résultats [Callan 2000].

Processus de RID

Faisant référence aux problématiques de recherche distribuée citées, le processus de RID se compose de trois phases que nous décrivons ci-dessous :

- **Phase 1- Description des sources :** Comme en Recherche d'Information classique, retrouver une source pouvant satisfaire une requête repose sur la correspondance entre la représentation de la source et celle de la requête. La méthode de représentation la plus utilisée en RID consiste à considérer une source comme un ensemble de termes, qui figurent dans ses documents, accompagnés par des statistiques sur la fréquence de ces termes dans chaque document ainsi que la fréquence des documents contenant un terme pour la collection [Callan 1995, Si 2002b]. Généralement, les schémas d'indexation des différentes sources dans un système de RID sont hétérogènes.

- **Phase 2- Sélection de sources :** la sélection constitue l'un des problèmes clés pour la RID. Réussir un processus de RID revient à réussir tout d'abord le choix des bonnes sources susceptibles de satisfaire les besoins des utilisateurs. Cette sélection se base généralement sur un ensemble d'algorithmes dont les plus répandus sont ceux qui utilisent la similarité entre la source et le document étant donnés leurs descriptions [Callan 2000, Larkey 2000]. Le défi dans cette phase consiste à minimiser les sources à visiter sans pour autant perdre en qualité de réponse. Nous parlons également du concept de routage des requêtes vers les sources sélectionnées. Le système GloSS [Gravano 2000] par exemple, est lié à un modèle de RI (deux versions ont été proposées, l'une pour le modèle booléen, l'autre pour le modèle vectoriel). Le choix des meilleures sources se fait par construction d'un résumé de celles-ci. Ce résumé se base, d'une part, sur le nombre de documents par terme (pour une source donnée), et d'autre part, sur la somme des poids d'un terme dans tous les documents (pour une source donnée). La requête émise sur le méta-moteur est alors comparée à ces résumés pour choisir les meilleures sources. Les expérimentations ont montré l'efficacité de GloSS pour sélectionner les meilleures

sources tout en ayant un résumé compact (la taille des résumés est de l'ordre de 2% de celles des index totaux).

– **Phase 3- Fusion des résultats :** Le courtier qui achemine les requêtes vers les sources sélectionnées doit être en mesure de rendre une réponse agrégée à l'utilisateur. Le défi qu'impose cette phase, est de trouver un référentiel commun pour fusionner les résultats issus des ressources hétérogènes. Ces résultats sont incomparables, parce que premièrement, leurs schémas d'indexation sont hétérogènes. En effet, chaque source choisit son propre langage de représentation de ressources. En second lieu, les modèles de Recherche d'Information utilisés dans chaque source sélectionnée pour restituer les documents pertinents à la requête utilisateur sont différents. En effet, un document classé premier par une source peut se voir classé dans une autre position dans la logique d'une autre source.

Les deux dernières phases représentent les problèmes clés de la RID. Cependant, ces derniers peuvent être atténués lorsque le système distribué se compose d'un nombre réduit de sources. De plus, le serveur courtier peut réduire l'écart entre les sources hétérogènes, puisqu'en général, il acquiert certaines informations concernant les contenus des sources. Ainsi, ceci lui permettra une meilleure connaissance de leurs contenus et donc de proposer une meilleure recherche.
Le problème devient beaucoup plus complexe lorsqu'il s'agit d'un système disposant d'un nombre très important de sources. Dans ce cas, il est très difficile de mettre en place un courtier disposant d'informations sur les contenus de toutes les sources (les coûts de construction et de maintenance de ces informations deviennent prohibitifs). C'est le cas des systèmes Pair-à-Pair que nous présentons dans la prochaine section.

1.5 Les systèmes de Recherche d'Information en Pair-à-Pair RIP2P

Initialement destinés au partage de fichiers multimédia (audio, vidéo, ...) tels que Napster [Nap 2006], les systèmes P2P sont aujourd'hui utilisés dans beaucoup d'autres contextes. En effet, il y a des systèmes de partage de contenu de bureau tels que Beagle++ [Beagle2 2008], de partage de fichiers textuels tel que [Wang 2009], etc. Le système P2P est le premier contributeur au changement de rôle des utilisateurs en acteurs principaux de leurs applications.

Nous présentons dans ce qui suit, les caractéristiques des systèmes P2P ainsi que leurs architectures et nous abordons le sujet de la Recherche d'Information dans de tels systèmes.

1.5.1 Revue sur les systèmes P2P

Les systèmes P2P sont une famille de systèmes distribués qui ont modifié la perception des utilisateurs vis-à-vis d'Internet par leur facilité de mise en œuvre.

Plus besoin de disposer d'un serveur puissant, mais une connexion Internet et un ensemble de nœuds dits pairs (ordinateurs personnels) sont suffisants pour former un système Pair-à-Pair. C'est d'ailleurs la manière dont Internet a été conçu.

De manière plus précise, il s'agit de systèmes distribués constitués d'un grand nombre de pairs (plusieurs milliers, voire, plusieurs millions) et fonctionnant sans état global. Aucun pair n'a une connaissance complète de tous les autres, chaque pair ne connaît qu'une partie du système et n'a qu'une approximation (au mieux) du système complet. Chaque pair doit être, à la fois, contributeur et initiateur, sous peine de voir les autres, l'éjecter du système (ou tout simplement quitter un système qui ne leur apporte rien). De plus, le nombre de pairs implique une grande dynamicité (les pairs entrent et sortent avec une fréquence élevée). Ces caractéristiques changent complètement la philosophie des systèmes distribués classiques qui doivent être repensés complètement. Bien que ces systèmes préconisent la décentralisation, ils peuvent être bâtis autour d'architectures centralisées, totalement décentralisées ou hiérarchiques.

- **Architecture centralisée :** Les pairs sont organisés autour d'un serveur central qui détient un résumé de ce qui existe sur tous les autres pairs du système (une description de leurs fichiers partagés).
  Contrairement aux systèmes client-serveur où tout se passe à travers le serveur, dans une architecture P2P centralisée, les pairs communiquent directement entre eux et les ressources (données et/ou services) sont toujours localisées sur les pairs. Le système P2P représentatif de ce type d'architecture est Napster [Nap 2006].
  L'approche n'a, cependant de P2P, que le nom et, la plus grande partie des fonctionnalités est assurée de manière centralisée. Si le serveur central tombe en panne par exemple, plus aucune recherche ne peut être menée à bien.

- **Architecture décentralisée :** C'est le pur P2P. Les pairs sont des composants autonomes qui ne détiennent qu'une vision partielle du système. Aucun tiers coordinateur n'est nécessaire dans le réseau. Les relations sont purement symétriques et chaque pair demande et fournit des services aux autres. Les systèmes P2P purement décentralisés peuvent être non structurés tel que Gnutella [Gnu 2007] ou bien structurés, ce qui est le cas des systèmes à tables de hachage distribués (Distributed Hash Table (DHT)).

  Les systèmes non structurés s'adaptent naturellement à la dynamicité des pairs et respectent au mieux leur autonomie. Dans un système, basé sur le protocole Gnutella, chaque pair ne connaît que quelques autres pairs. Le routage des requêtes de recherche se fait de manière aléatoire. Chaque pair résout une requête entrante localement avant de la propager sur un sous-ensemble aléatoire de ses voisins. La terminaison est assurée par un mécanisme de contrôle de la profondeur de la recherche et une détection des cycles. Le principal dé-

faut de ces systèmes est leur faible niveau de performance, notamment, le nombre élevé de messages échangés pour créer et maintenir les tables de routages ainsi que le fait qu'ils ne peuvent garantir de retrouver une ressource présente sur un pair (à cause de la recherche aléatoire).

Dans les systèmes structurés, tels que, Chord [Stoica 2001], Can [Ratnasamy 2001] et Pastry [Rowstron 2001], on cherche à améliorer les performances de recherche en organisant les pairs selon une structure logique (ou overlay) qui permet à chaque pair de connaître un certain nombre de ses prédécesseurs et/ou successeurs. Par exemple dans Chord, une table de routage existe sur chaque pair donnant une liste de taille réduite (logarithmique relativement à la taille du réseau) des successeurs du pair. Cette table permet de router à un coût logarithmique une requête de recherche d'une clé. La structure du système est également maintenue malgré la dynamicité des pairs. Cela se paye, par contre, par une perte d'autonomie, tant, au niveau du stockage (un pair ne choisit pas ce qu'il stocke), que du routage (un pair ne choisit pas les pairs de sa table de routage).

– **Architecture hiérarchique :** Cette architecture a été proposée pour trouver un compromis entre l'efficacité des systèmes clients-serveurs et la dynamicité des systèmes P2P non structurés. Dans cette approche, les pairs sont séparés en deux catégories : les super-pairs et les pairs. Un super-pair gère un groupe (ou cluster) de pairs dont il est le représentant. Chaque super-pair indexe l'ensemble de son groupe selon une approche client-serveur classique et chaque pair connaît son super-pair. Seuls les super-pairs forment un système P2P (généralement non structuré)La résolution de requête se fait seulement entre les super-pairs, ce qui diminue fortement la complexité. Les super-pairs sont choisis généralement en fonction de la largeur de leurs bandes passantes et leurs puissances de calcul plus importante par rapport à celle des pairs normaux. Kazaa [Liang 2004] adopte ce type d'architecture. Les expérimentations montrent que ces systèmes offrent effectivement le meilleur compromis. Par contre, il faut gérer la tolérance aux fautes des super-pairs.

Dans la suite, nous allons nous intéresser essentiellement à la Recherche d'Information dans les systèmes P2P décentralisés non structurés, qui se placent naturellement dans la lignée des systèmes de RI distribuée.

1.5.2 La Recherche d'Information dans les systèmes P2P

Les systèmes P2P se présentent aujourd'hui comme des paradigmes forts de recherche de documents dans les bibliothèques numériques distribuées sur le réseau [Balke 2005]. Ils permettent une fédération massive de ressources d'informations indépendantes et possèdent des modèles riches pour l'extraction des contenus, des

annotations, des résumés, des images, du discours et l'indexation de textes.

Les défis d'une Recherche d'Information P2P (RIP2P) sont essentiellement de retrouver les bons pairs répondant à une requête d'un utilisateur (problème de sélection), puis d'agréger les résultats provenant de ces pairs afin de restituer seulement les résultats intéressants du point de vue utilisateur. Ces défis sont ceux de la RID. Cependant, le contexte change énormément, vu le facteur d'échelle dans ces réseaux, qui s'étendent naturellement à des milliers voire des millions de pairs. Ceci va impliquer notamment la suppression d'un serveur courtier inadéquat dans ce contexte.
Ainsi, vu ce changement de contexte, il convient de reposer des questions clés se rapportant aux contraintes de recherche dans les systèmes de RIP2P, notamment : Comment choisir les bons pairs pour satisfaire un besoin donné ? Comment rechercher de manière distribuée ? et comment agréger des résultats provenant de différents pairs hétérogènes dans une liste tout en reflétant les attentes de l'utilisateur ?

La Recherche d'Information dans les Systèmes Distribués à Large Échelle (RIDLE) s'avère beaucoup plus difficile qu'une simple RID. En RID, des descriptions des sources sont généralement utilisées pour calculer des statistiques globales servant comme référentiel pour le classement des résultats. Cependant, en RIDLE plusieurs facteurs font que les pairs n'ont qu'une vision très limitée du système et que l'hétérogénéité soit plus importante en schémas d'indexation, en taille de collections et en modèles de recherche.

En plus, un échange de statistiques ou d'une partie des index entre les pairs afin de faire des calculs globaux, est une solution qui pose d'énormes problèmes de coût, vu le risque de surcharger le réseau. Ces contraintes imposent le recours à une autre dimension pour améliorer la recherche dans ces systèmes. Nous parlons, alors, du P2P sémantique.

1.5.3 P2P sémantique

Afin d'améliorer la qualité de la recherche dans les systèmes de Recherche d'Information P2P non structurés, l'introduction de la sémantique est une piste intéressante.

Intégrer la sémantique dans un processus de RIP2P revient, comme présenté dans [Defude 2007], à agir envers plusieurs questions :

- **Quelle sémantique ?** : il s'agit de définir le type d'information à utiliser dans le processus de routage. Cela peut être l'information sur le contenu des pairs (les données/documents stockés), sur l'intérêt du pair (les requêtes déjà émises), sur les utilisateurs (profil d'un utilisateur ou de communautés d'utilisateurs).

- **Quelle représentation de la sémantique ?** : cela va d'une simple information temporelle (les pairs ayant répondus récemment) à des modèles non structurés (simple liste plate de concepts) ou à des modèles structurés (ontologies par exemple).

- **Comment construire la sémantique ?** : le processus de construction peut être manuel (par intervention de l'utilisateur) ou bien automatique (algorithmes d'apprentissage). On peut également trouver des approches mixtes où l'utilisateur intervient via des formes de feedbacks.

- **Qu'est ce qui est partagé entre les pairs ?** : a minima les pairs doivent partager des structures de données communes (représentation de la sémantique par exemple), mais cela peut aller au partage de fonctions (algorithmes d'apprentissage par exemple), voire au partage de connaissances (ontologie commune partagée).

- **Comment utiliser la sémantique ?** : le plus souvent la sémantique est utilisée pour sélectionner le sous ensemble des pairs les plus "pertinents" pour une requête donnée [Crespo 2002, Cuenca-Acuna 2003]. Cela peut aussi servir à organiser le réseau des pairs (classification des pairs selon leur contenu par exemple), [Voulgaris 2004], à modifier les requêtes [Nakauchi 2004] ou bien à interclasser les résultats retournés par les différents pairs.

- **Comment diffuser la sémantique** : la connaissance construite localement sur un pair peut être diffusée aux autres pairs pour qu'ils puissent améliorer leur connaissance du réseau. Cette diffusion peut être globale (à tous les pairs, [Cuenca-Acuna 2003]) ou partielle (à quelques uns, [Crespo 2002]) et la fraîcheur est également importante (diffusion lors de chaque modification ou bien périodique). Le coût de la diffusion en nombre de messages est par ailleurs crucial. Dans certaines propositions [Loser 2007], la sémantique n'est pas partagée (elle reste purement locale).

La notion de sémantique a été largement exploitée pour améliorer les algorithmes de routage dans les systèmes P2P. C'est dans ce contexte que PlanetP [Cuenca-Acuna 2003] a utilisé la notion de signature sémantique pour indexer le contenu des pairs et former un résumé à échanger entre les pairs afin d'en pouvoir sélectionner les meilleurs et en classer les meilleures ressources. La limitation majeure de PlanetP est que la sémantique doit être échangée entre tous les pairs (y compris lors de mises à jour). Pour limiter le coût de ces échanges, un algorithme épidémique a été proposé.

[Crespo 2002] étend les idées développées dans [Gravano 2000] à un contexte P2P. Chaque pair indexe ses propres documents à l'aide d'un vecteur thématique qui doit être utilisé par tous les pairs. Chaque pair diffuse ce vecteur à ses voisins directs de manière à ce que chacun puisse créer des vecteurs de chemins (ensemble

des documents qui peuvent être atteints à partir de chaque pair direct). Ces vecteurs sont ensuite utilisés pour sélectionner les meilleurs voisins vers lesquels on propage la requête. Là encore, le principal défaut de cette proposition est le coût de propagation des mises à jour des vecteurs de chemins, puisque celui-ci n'est pas borné (si les pairs sont fortement connectés, une mise à jour peut se diffuser vers tous les pairs).

INGA [Loser 2007] analyse les requêtes émises par chaque pair et les résultats retournés pour construire deux listes : une, associant pour chaque terme des requêtes émises, les pairs ayant retournés les meilleurs résultats (acquis grâce au feedback de l'utilisateur), et une autre associant aux termes les pairs émetteurs. L'intuition, est que, soit on connaît les pairs répondant à la requête, parce qu'on a déjà lancé une requête voisine dans le passé, soit on connaît les pairs qui ont déjà lancé une requête voisine dans le passé (et qui eux, vont connaître les pairs qui y répondent). L'intérêt majeur de INGA [Loser 2007] est que la construction de cette connaissance est purement locale (chaque pair analyse ses requêtes ainsi que celles qu'il voit passer via le routage) et ne nécessite aucun échange entre pairs.
Voulgaris et al., [Voulgaris 2004] de leurs côtés ont utilisé les intérêts communs entre les pairs appris de leurs résultats retournés comme approche pour créer un réseau sémantique entre les pairs proches. Nakauchi et al., [Nakauchi 2004] exploitent les corrélations sémantiques entre les mots clés pour effectuer une expansion des requêtes. Des bases de connaissances sont construites sur chaque pair à partir des ressources qu'il détient, puis enrichies, par des relations entre les mots clés ou par recours au feedback utilisateur.

La fonction d'interclassement n'a été que peu étudiée dans un contexte P2P. En effet, les travaux de [Chernov 2005, Witschel 2005, Witschel 2008] ont été recensés parmi le peu de travaux qui ont intégré la sémantique dans la phase d'interclassement. Dans ces travaux, nous parlons de profils utilisateurs (ceci sera analysé en détails dans le chapitre suivant).

1.6 Conclusion

Dans ce premier chapitre, nous avons étudié les concepts clés de la Recherche d'Information et discuté les principaux facteurs qui ont conduit à passer d'une Recherche d'Information dans un cadre centralisé à une Recherche Distribuée à Large Échelle. Après avoir comparé les architectures des systèmes P2P, nous avons retenu les systèmes P2P purs non structurés comme plate-forme pour notre recherche. Le choix de l'étude de ces systèmes a été motivé d'une part par la grande flexibilité qu'ils offrent spécialement dans l'implantation des solutions de RI à des coûts relativement faibles et d'autres part, pour leur capacité à supporter des systèmes hétérogènes (capacités de traitement, fonctionnalités, etc).
Également, l'utilisation de la sémantique peut être vue comme une piste intéres-

sante afin d'améliorer les processus de recherche dans les systèmes P2P quelque soit leurs architectures. En effet, même dans les architectures centralisées ou semi-centralisées, la sémantique contribue à contextualiser la recherche afin de restituer à l'utilisateur les résultats qui s'ajustent le plus possible à ses besoins. Dans cette thèse, nous nous focalisons sur la problématique de l'agrégation (i.e. la fusion et le classement) des résultats dans les systèmes distribués à large échelle (particulièrement les systèmes P2P non structurés). Une présentation de cette problématique ainsi qu'un état de l'art associé fera l'objet du chapitre suivant.

Chapitre 2

Agrégation de résultats dans un système de recherche distribué à large échelle

2.1 Introduction

Après avoir présenté les concepts de base de la recherche d'information et évoqué ses différentes problématiques, nous nous intéressons, dans le présent chapitre, à un problème clé de la Recherche d'Information Distribuée (RID), à savoir l'agrégation des résultats.
Cette phase de RID consiste à combiner des listes de résultats provenant de différentes sources, dans une liste finale classée par ordre de pertinence. Ce problème est d'autant plus difficile qu'il s'agit de fusionner des résultats provenant d'un nombre élevé de sources, à fortiori hétérogènes.
Dans les systèmes à large échelle tels que les systèmes Pair-à-Pair purs, la décentralisation fait qu'aucun pair ne possède une vue globale du système. C'est ainsi que le recours aux statistiques globales est une solution coûteuse qui risque de surcharger le réseau (pour maintenir ces statistiques à jour).
L'utilisation des statistiques locales est une solution possible mais à condition qu'un système contribuant à la réponse accepte de diffuser des informations sur lui-même. Souvent, un SRI résout une requête en retournant les liens vers les résultats sans donner aucune information ni sur sa méthode d'indexation, ni de recherche et classement. C'est le cas de Google et ses équivalents. En outre, même si on connaît la méthode, il est très difficile de savoir comment interpréter une statistique locale au niveau d'un système global. Plus l'anonymat et l'autonomie dans un système sont grands, plus nous perdons la signification des statistiques locales et plus l'agrégation (i.e, l'interclassement) des résultats devient difficile.

Nous reprenons dans ce chapitre, tous les éléments se rapportant au problème d'agrégation dans un système distribué. Nous présentons les origines terminologiques du problème et évoquons les problèmes voisins. Une définition de ce problème au sein de son contexte de base (la RID simple) sera donnée, puis nous mettons en exergue les problèmes qui s'ajoutent dans le cas du Pair-à-Pair (RIP2P). Pour ce faire, nous décrivons les propriétés requises d'un modèle d'agrégation en P2P. Nous proposons une catégorisation des modèles d'agrégation existants et analysons

particulièrement ces modèles à la base des propriétés requises afin de soulever leurs principales limites.

2.2 Agrégation : Origines et concepts

Une revue de la littérature, montre que la combinaison de résultats et leur classement sont une partie des problèmes clés dans les domaines de la méta-recherche et de la recherche d'informations distribuée [Aslam 2001]. En général, deux concepts ont été utilisés pour décrire ce problème : la fusion et l'agrégation de résultats.
Nous présentons dans ce qui suit l'origine de ces deux termes.

2.2.1 Agrégation vs Fusion

L'**agrégation** est un concept hérité du domaine des statistiques, où il a été utilisé pour combiner les préférences (ou les similarités) [Marcotorchino 1982], et en sciences sociales, où il a été exploité dans les élections politiques [Borda 1781]. Par la suite, le concept a été utilisé dans les domaines de la recherche multi-critères, la recherche des similarités et la classification [Fox 1994], puis, essentiellement, dans le domaine de la méta-recherche, pour définir la combinaison de résultats provenant de plusieurs moteurs de recherche dans une liste, en éliminant les doublons [Dwork 2001, Aslam 2002].

Le terme **fusion** a été utilisé dans plusieurs domaines d'application. Les algorithmes de fusion constituent une famille de techniques qui opèrent sur des listes en vue de produire un meilleur classement [Knuth 1997].

En Recherche d'Information Distribuée (ou méta-recherche), les termes fusion et agrégation ont été utilisés indistinctement pour désigner un concept unique qui est la combinaison et le classement de résultats [Mghirbi 2009].

2.2.2 Fusion de données vs fusion de collections

Dans le domaine de la méta-recherche, les chercheurs distinguent entre la fusion des données et celle des collections [Dwork 2001, Aslam 2001].
Nous parlons de fusion de données lorsqu'il s'agit de combiner des informations issues de plusieurs sources indexant effectivement le même ensemble de données [Aslam 2002]. Autrement dit, il s'agit de classer (dans une liste finale) les mêmes documents classés de différentes manières dans des listes locales (voir la partie gauche de la Figure 2.1). Dans cette figure, différents SRIs produisent le même ensemble de documents mais c'est seulement le classement qui change.
Le point fort de la fusion de données est de gérer différents classements d'un même

ensemble de documents pour donner un consensus final de classement. Ce dernier peut être basé sur un score global attribué à chaque document, combinant les scores locaux. Les méthodes de combinaison Comb(Somme, moyenne, min, max) de Fox et Shaw [Fox 1994] ont été largement utilisées à cette fin.

En fusion de collections, nous ne parlons plus de données identiques mais plutôt de résultats provenant de collections totalement ou partiellement disjointes, d'où l'hétérogénéité de ces collections (voir partie droite de la Figure 2.1). Dans notre contexte de Recherche d'Information Distribuée à Large Échelle (RIDLE), nous sommes clairement positionnés sur la fusion de collections mais cela n'empêche pas d'exploiter les principes de la fusion de données.

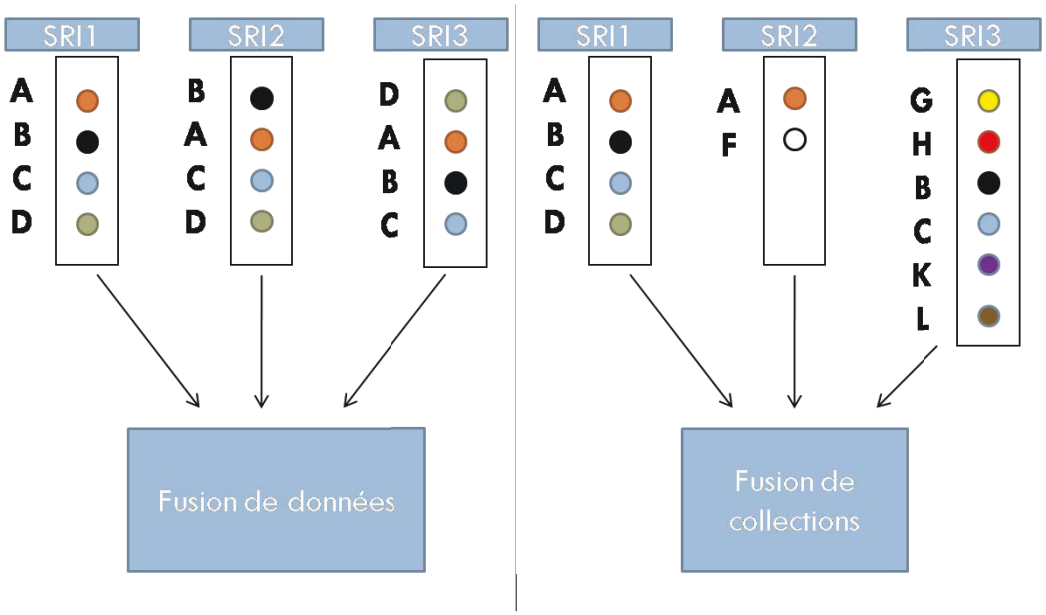


FIGURE 2.1 – Fusion de données vs Fusion de collections

Dans un souci d'uniformité, nous utilisons tout au long de cette thèse le terme agrégation pour désigner le problème de combinaison de résultats issus de plusieurs listes de résultats, quelque soit le degré de chevauchement entre ces listes.

2.3 Problème d'agrégation dans les systèmes à large échelle : cas des systèmes de RIP2P

2.3.1 Définition du problème d'agrégation

La phase d'agrégation représente l'un des maillons les plus importants dans une chaîne de recherche d'information distribuée (RID), et ce quelque soit l'échelle de la distribution. Elle représente une des clés du succès de tout le processus de la recherche. Le principe général de la RID peut se définir comme suit (voir Figure

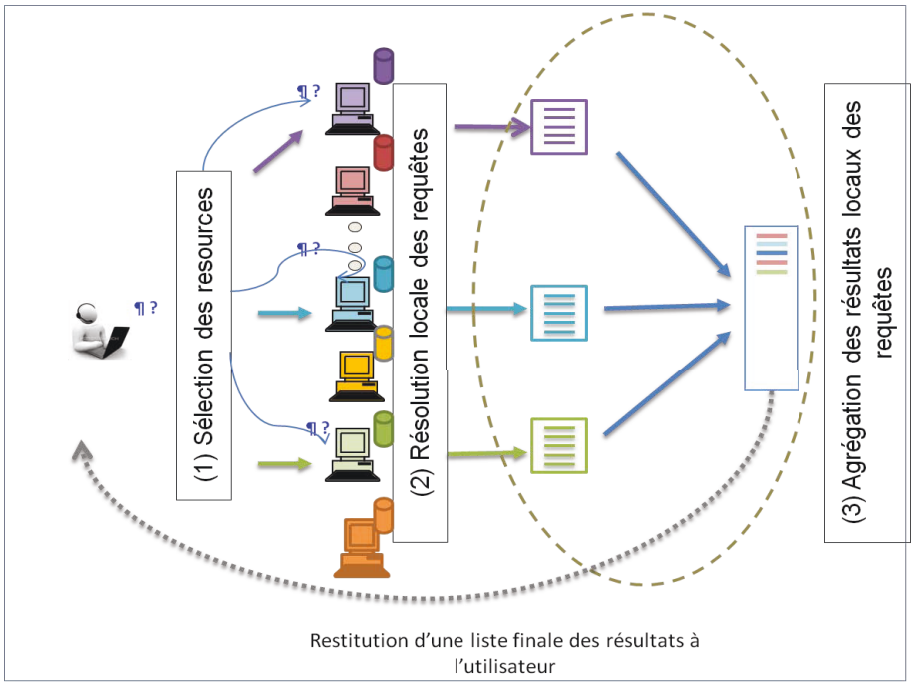


FIGURE 2.2 – Problème d'agrégation dans un processus de Recherche d'Information Distribuée

2.2) :
Lorsqu'un utilisateur soumet une requête à travers une interface d'interrogation, un serveur courtier ou médiateur lance une procédure de sélection des ressources (ou systèmes de recherche) potentiellement pertinentes à la requête : c'est la procédure de sélection représentée par la première phase de la Figure 2.2. Ce problème ne sera pas étudié dans le cadre de cette thèse.
Par la suite, chaque système se charge de résoudre localement la requête selon sa propre stratégie de recherche permettant ainsi de faire correspondre à la requête, une liste de réponses locales classées. Ceci correspond à la phase 2 de la Figure 2.2. Chaque liste retournée sera retournée au serveur courtier qui se charge cette fois-ci d'interclasser les résultats des différentes listes en veillant à placer les réponses les plus pertinentes dans les premières positions : c'est le problème d'**agrégation de résultats** (phase 3 de la Figure 2.2).
Généralement une pertinence utilisateur (validée par la satisfaction de l'utilisateur lui même) est préférée par rapport à une pertinence système (validée par des mesures d'évaluation, telles que la précision et le rappel).
La complexité du problème d'agrégation émane du fait que les collections à partir desquelles les résultats locaux sont fournis, sont souvent hétérogènes en termes de tailles, schémas d'indexation et modèles de recherche [Callan 1995, Shokouhi 2007]. Nous modélisons dans la suite, le problème d'agrégation.

Soit $\Re$ un SRI retournant une liste classée de résultats, nous symbolisons cette

liste par un ensemble d'identifiants de documents entre les symboles $\prec$ et $\succ$. Le résultat d'un SRI est écrit comme suit : $\Re_i \prec d_{i1}, ..., d_{in} \succ$. Le problème d'agrégation peut être modélisé comme suit :
Soit $\{\Re_1 \prec d_{11}, \ldots, d_{1n} \succ, \ldots, \Re_m \prec d_{m1}, \ldots, d_{mk} \succ\}$ un ensemble de listes de résultats locales en provenance de m différents systèmes de recherche en réponse à une requête soumise q. La tâche d'agrégation vise à trouver une fonction d'agrégation Ψ qui prend en entrée l'ensemble des listes de résultats locales et les classe dans une liste finale :

$$\Psi(\{\Re_{11}, \prec d_1, \ldots, d_{1n} \succ, \ldots, \Re_m \prec, d_{m1}, \ldots, d_{mk} \succ\}) \longrightarrow \Re_f \prec D \succ \tag{2.1}$$

où :

$$D = \bigcup_{i=1}^{i=m,j=n} d_{ij} \tag{2.2}$$

avec :

- m : nombre des SRI sélectionnés pour répondre à la requête q ;
- n, k : nombre de documents retourné par un ensemble d'SRIs ;
- $\bigcup$: est la fonction de fusion des différentes listes de résultats en supprimant les documents redondants (doublons).

Par rapport au contexte de RID, le problème d'agrégation dans les systèmes massivement distribués, et particulièrement les systèmes de RIP2P est un problème analogue auquel des contraintes relatives à la nature de ces systèmes viennent s'ajouter de manière significative et exigent d'explorer de nouvelles pistes qui tiennent compte de la nature de ces systèmes. En effet, les statistiques globales qui permettent une comparaison directe des réponses sont trop coûteuses à échanger et à maintenir, à cause de la taille de ces systèmes. De plus, l'hétérogénéité entre les pairs est beaucoup plus grande que dans les systèmes de RID classiques. En outre, aucun tiers médiateur n'est présent dans le système et chaque pair ne détient qu'une vision très partielle du reste du système. Cependant, nous pouvons définir deux rôles pour les pairs d'un système de RIP2P. En effet, un pair peut assurer à un instant donné : (i) **un rôle d'initiateur** qui consiste à sélectionner les pairs pertinents pour une requête, puis à fusionner les résultats de ces derniers pour les restituer de manière qui "correspond" le mieux possible aux attentes des utilisateurs. Le rôle initiateur est généralement détenu par le pair de l'utilisateur (celui qui a initié la requête) ; (ii) **un rôle de contributeur** , qui concerne tous les pairs sélectionnés pour contribuer à la résolution de la requête. Un pair initiateur peut toutefois contribuer à la résolution de la requête et jouer ainsi les deux rôles.

Ainsi, vu les contraintes liées à la nature des systèmes de RIDLE en général, et de RIP2P en particulier, un processus d'agrégation doit prendre en compte les caractéristiques intrinsèques de ces systèmes. Pour cette raison, nous discutons dans la section suivante les propriétés requises d'un modèle d'agrégation en RIP2P.

2.3.2 Propriétés requises d'un modèle d'agrégation en RIP2P

Dans un modèle d'agrégation en RIP2P, les propriétés requises sont très liées à celles du système P2P [Chernov 2005] et sont notamment la dynamicité, le passage à l'échelle, l'autonomie et la disponibilité. Nous présentons dans ce qui suit les propriétés que nous jugeons les plus importantes à respecter dans un modèle d'agrégation en RIP2P [Mghirbi 2011].

2.3.2.1 Autonomie

Dans un système distribué, chaque SRI repère l'ensemble de ses ressources (documents, enregistrements, etc), grâce à une structure locale (un index) permettant d'accéder rapidement à leurs contenus et d'accélérer les opérations de recherche. Dans les systèmes P2P centralisés, les index locaux peuvent être agrégés afin de localiser les ressources des SRIs distribués et d'en repérer, de manière centralisée, le contenu (ce qui est connu par l'utilisation d'un index global).
Cependant, les SRIs peuvent préserver leur autonomie en refusant d'exporter un résumé de leurs ressources dans un index global. Plus largement, ces SRIs peuvent préserver leurs méthodes de calcul ou de traitement, ou leurs modèles de RI. Généralement, un système qui décide d'être autonome doit présenter un ensemble de caractéristiques lui permettant de s'intégrer facilement à un système sans introduire de contraintes supplémentaires (pas de modification au niveau des composants d'un système) [Tanenbaum 2006].

On peut classifier la notion d'autonomie pour la méta-recherche en trois catégories. Nous parlons respectivement de systèmes :

- **coopératifs** : Quand ils envoient au serveur courtier, en plus des identifiants des documents répondant à une requête, des informations sur leurs index (des scores, des statistiques, du contenu, ...).
- **non coopératifs** : Nous parlons d'environnement semi coopératif quand les SRIs ne fournissent aucune information sur leurs contenus : seulement les rangs sont fournis
- **semi-coopératifs** : Quand les systèmes fournissent une information incomplète sur leurs index et qu'il s'agit le plus souvent de remédier ce manque d'information par recours à des procédures d'apprentissage. Dans tel environnement, le courtier détient une connaissance incomplète (partielle) sur les collections. Toutefois, il doit recevoir des informations supplémentaires notamment, les scores de documents [Shokouhi 2007].

Afin de respecter l'autonomie des pairs, un modèle d'agrégation doit être en mesure de fusionner les listes de résultats sans que le pair initiateur exige des informations sur les index des pairs contributeurs ni sur les modèles de RI utilisés. Ainsi, selon le degré de coopération des systèmes de recherche locaux, le module d'agrégation reçoit en entrée, entre autres les scores (et /ou) les rangs. Tout manque d'information d'entrée peut être satisfait par apprentissage.

2.3.2.2 Décentralisation

Un système P2P pur non structuré suppose une décentralisation totale. Pour cette raison, le modèle d'agrégation ne doit pas supporter un point de centralisation notamment l'utilisation d'un index global. Afin d'éviter le recours à un index global unique, souvent une replication de cet index est possible. Cependant, ceci est très coûteux lorsque le système est large, vu le coût de maintien et de mise à jour de l'index. De même, la mise à jour ne doit pas être basée sur un point central afin de respecter la règle de décentralisation.

2.3.2.3 Passage à l'échelle

C'est la capacité d'un système à livrer une même qualité de service avec un nombre croissant de ressources. En effet, un des objectifs essentiels d'un système distribué, et particulièrement d'un système P2P, est sa capacité de passer à l'échelle. C'est la propriété par laquelle un système peut essentiellement étendre de façon permanente le nombre de ses ressources sans causer des dégradations de sa performance (essentiellement les attentes supplémentaires pour répondre à un besoin d'utilisateur). Nous soulignons ici, que l'une des causes primordiales d'augmentation du temps de réponse dans un système distribué est dûe généralement au phénomène d'occupation de la bande passante suite aux accès simultanés à une ressource (utilisation excessive) ou à un phénomène d'échange d'information entre ressources. Ainsi, un système à large échelle, tel que le système P2P, doit être en mesure d'éviter les structures centralisées, d'éviter les calculs globaux et de supporter l'hétérogénéité des données et des supports techniques. L'une des principales raisons du non passage à l'échelle est l'occupation de la bande passante. Pour cette raison, un modèle d'agrégation doit minimiser les échanges excessifs d'information entre les pairs afin d'éviter toute surcharge du système.

2.3.2.4 Hétérogénéité

Un modèle d'agrégation dans un système de RIP2P doit prévoir l'hétérogénéité des pairs en terme de taille de collections sur chaque pair, de contenu (dans le cas où les pairs sont spécialisés dans une thématique précise par exemple) et de modèle de RI utilisé sur ce pair (vectoriel ou booléen par exemple). En d'autres termes, il ne doit pas faire d'hypothèse sur un certain type d'homogénéité des pairs.

2.3.2.5 Disponibilité

Le modèle d'agrégation doit prévoir la dynamicité des pairs et assurer la qualité d'agrégation même en cas de volatilité de pairs. Un modèle se basant sur des informations locales, peut échapper à l'effet de perte introduite par la déconnexion d'un ou de plusieurs pairs contributeurs. Une étude de la sensibilité du modèle d'agrégation à cette caractéristique devient nécessaire pour vérifier sa faisabilité dans un tel environnement.

2.3.2.6 Efficacité du modèle

L'objectif de tout système de recherche d'information est de prouver la pertinence des résultats fournis. Ceci dépend, en RIP2P, des résultats fournis par le modèle d'agrégation et qui se mesure en fonction des mesures d'évaluation de pertinence notamment la précision, le rappel et les métriques dérivées (MAP, etc). A noter que l'efficacité sera, ici, en général comparée à celle obtenue, en centralisé et non pas à l'optimal absolu.

Nous présentons dans ce qui suit les principaux modèles d'agrégation issus, essentiellement, des systèmes de méta-recherche ou de la recherche distribuée et, très peu, des systèmes P2P. Nous analysons notamment l'adéquation de ces modèles d'agrégation aux exigences des systèmes distribués à large échelle.

2.4 Catégorisation des principaux modèles d'agrégation

Une revue de la littérature nous a permis d'établir une catégorisation des principaux modèles d'agrégation. Bien que plusieurs catégorisations existent dans la littérature spécialisée [Le Calvé 2000, Aslam 2001, Montague 2002, Renda 2003, Liu 2007], nous avons opté pour un raffinement des classifications existantes en y ajoutant de nouvelles dimensions liées à notre problématique (voir Table 2.1). En effet, en respectant les classifications données dans différents travaux [Aslam 2001, Renda 2003], nous distinguons les modèles à base de scores et les modèles à base de rangs scindés eux mêmes en modèles à base d'apprentissage ou sans apprentissage [Aslam 2002]. Par contre, nous avons procédé à une définition plus large pour les modèles à base de scores. En effet, nous ne considérons plus cette catégorie de méthodes comme celle qui reçoit seulement des scores locaux de documents et procèdent à leurs normalisations mais nous regroupons dans cette catégorie toute méthode permettant de calculer un score global à partir d'un contenu fourni par des SRIs locaux contribuant à la réponse. Sur la base de cette définition, nous donnons des sous-catégories du modèle selon les entrées utilisées. De même, nous avons distingué pour les modèles à base d'apprentissage les méthodes supervisées et non supervisées qui peuvent intégrer une classe basée sur le comportement des utilisateurs (avec profils).

2.4.1 Modèles à base de scores

Nous considérons qu'un modèle d'agrégation à base de scores est un modèle qui est en mesure de calculer un score global de documents retournés en réponse à une requête. Ces documents sont issus de plusieurs systèmes contributeurs qui les classent selon leurs visions individuelles et les fournissent avec suffisamment ou peu d'information renseignant sur leurs contenus, leurs index et la nature de leurs modèles de recherche. Selon l'entrée fournie par le système contributeur, un modèle d'agrégation à base de score va proposer une formule de calcul de score global permettant de classer les documents les uns par rapport aux autres. Sur la base de cette

<table>
<tr><td colspan="2" rowspan="3"></td><td rowspan="3">Sans apprentissage</td><td colspan="3">Avec Apprentissage</td></tr>
<tr><td rowspan="2">Supervisé</td><td colspan="2">Non Supervisé</td></tr>
<tr><td>Sans profils</td><td>Avec Profils</td></tr>
<tr><td>Modèle à base de rangs</td><td></td><td>exp : RR, BC,
exp : RR, BC,
méthodes floues</td><td>exp :
RR pondéré,
BC pondéré</td><td rowspan="4">exp :

Clustering</td><td>-
-
-</td></tr>
<tr><td rowspan="3">Modèle à base de scores</td><td>N-Scores-Locaux</td><td>Approches de normalisation</td><td rowspan="3">exp Régression logique,

SSL</td><td rowspan="3">exp
LMPR
Approche de witchel</td></tr>
<tr><td>Scores-B-Statistiques</td><td>Exp VSM, BM25</td></tr>
<tr><td>Scores Combinés</td><td>exp CORInet, LM, PlanetP</td></tr>
</table>

TABLE 2.1 – Catégorisation des modèles d'agrégation

donnée, nous avons scindé les modèles à base de scores en trois sous-classes de méthodes (voir Modèle à base de score de la Table 2.1). La première sous-classe opère sur des scores locaux et permet de calculer des scores globaux basés sur la normalisation (N_Scores_Locaux), la deuxième opère sur des statistiques (qui peuvent être locales ou globales) et recalcule un score global à partir de ces statistiques. Nous appelons cette classe (Scores_B_statistiques) et la troisième sous-classe permet de calculer des scores de documents par une combinaison des poids des documents (ou de statistiques) avec celui des collections (ScoresCombinés). Ces trois sous-classes de méthodes sont décrites, respectivement, dans ce qui suit :

2.4.1.1 Scores à base de normalisation des scores locaux (N-Scores-Locaux)

Un SRI contributeur ne retourne que les scores locaux des documents. Loin d'être une information suffisante, les scores seuls sont considérés comme une boite noire et ne sont pas directement comparables. Aucune information sur leurs natures (statistiques, probabilistes), algorithmes de recherche, tailles d'index, etc ; n'est fournie. C'est par exemple le cas où il s'agit d'interclasser deux documents dont le premier est classé avec un score de 0.7 par un premier système de recherche et le deuxième est classé avec un score de 0.5 par un autre système de recherche. Rien ne permet d'affirmer que le document de score 0.7 est meilleur que celui de score 0.5. Une simple normalisation peut rendre ces scores comparables. Différents schémas de normalisation ont été proposés dans des travaux antérieurs notamment ceux décrits dans [Aslam 2001, Renda 2003, Grappy 2012].
Cependant, l'hétérogénéité des systèmes de recherche augmente les risques de non comparabilité entre les scores locaux d'un ou plusieurs documents. En effet, normaliser sur la base des positions, du score maximum ou minimum dans chaque liste de résultats n'est pas toujours en mesure de donner de bonnes performances en agrégation.

Afin de pallier ce manque d'information, des algorithmes **d'apprentissage** ont été proposés [Aslam 2001, Si 2002a]. Ils visent à aider à normaliser les scores locaux en leurs faisant correspondre des scores globaux.
Une solution proposée par [Le Calvé 2000] consiste à utiliser la régression logistique pour la définition d'une formule de conversion de scores locaux en scores globaux. Elle suppose l'utilisation d'un échantillon centralisé des documents provenant des différentes collections et l'exécution d'une requête aussi bien sur l'échantillon centralisé que sur les collections locales. Une normalisation des scores des différents documents est effectuée sur la base d'une analyse de regression logistique utilisant les scores de documents requis de l'échantillon centralisé qui est pris comme référentiel commun pour tous les systèmes.
Le même principe a été appliqué dans les travaux de [Si 2003] pour la fusion des résultats des moteurs de recherche en utilisant la méthode Semi Supervised Learning (SSL).
Dans les deux travaux cités ci-dessus, la normalisation consiste à : (i) construire une collection centralisée par un échantillon de documents, (ii) calculer le score global de chaque document sur la base d'une formule de score donnée, puis (iii) chercher, par apprentissage, une correspondance entre les scores locaux et globaux pour un ensemble de documents. Ainsi, une formule basée sur les scores locaux est produite pour le calcul d'un score global.

En guise de synthèse, nous pouvons dire, que dans un modèle de score global, la fonction d'agrégation prend comme entrée des scores partiels et essaie de les transformer en utilisant des schémas de normalisation ou procède par apprentissage à partir d'un échantillon de requêtes pour apprendre à convertir des scores locaux en globaux. Dans les systèmes distribués à large échelle, cette solution peut être critiquée pour deux raisons : (i) le facteur hétérogénéité des pairs est beaucoup plus important. Ainsi, normaliser des scores bruts ne peut pas être adopté pour la RIP2P ; (ii) la solution de recourir à l'apprentissage, qui a prouvé son efficacité en RID, exige une représentation résumée quasi-exhausive de toutes les collections d'un système distribué. Dans les systèmes distribués à large échelle en général, et pour les systèmes P2P en particulier, l'échantillonnage centralisé ne peut pas être appliqué vu les problèmes de disponibilité. En effet, face à des entités volatiles (i.e. ressources), nous ne pouvons pas récupérer une information centralisée. De plus, à chaque instant de nouveaux pairs peuvent s'ajouter au système alors qu'ils ne sont pas représentés dans l'échantillon. Ces raisons font que cette solution n'est pas applicable dans un contexte de RIP2P.

2.4.1.2 Scores à base de statistiques (Scores-B-statistiques)

En RID, les systèmes de recherche contribuant à la résolution d'une requête, peuvent collaborer aussi pour fournir, au SRI chargé d'agréger les résultats, un ensemble de statistiques sur le contenu de leurs collections [Shokouhi 2007]. Idéalement, les statistiques fournies par ces systèmes sont globales. Cela suppose que

soit on utilise un serveur centralisé afin d'avoir un index global, soit que chaque système de recherche dans le réseau, doit accepter de coopérer avec les autres pour construire et sauvegarder un index global répliqué. L'utilisation d'un index global facilite énormément le processus de classement. Une fonction de correspondance est utilisée comme dans une approche centralisée. Cette fonction peut être basée sur le modèle vectoriel [Salton 1975] ou probabiliste (tel que l'utilisation okapi BM25 [Robertson 1995]) ou autre afin de recalculer de manière homogène la similarité document-requête. Le deuxième cas peut avoir lieu quand les pairs fournissent une partie de leur index locaux (statistiques brutes). La normalisation des statistiques est jugée nécessaire afin de surmonter l'incomparabilité des statistiques brutes provenant des différents pairs.
L'adoption de cette classe de modèles pour l'agrégation des résultats en RIP2P se heurte à un double problème. En utilisant un index global pour un système de pairs, nous imposons d'avance une restriction sur l'autonomie des pairs. Ceci est d'ailleurs vrai même dans le cas de l'utilisation d'un échange partiel des index locaux. Le second problème concerne le passage à l'échelle, puisqu'il faut payer le coût d'échange rédhibitoire des informations locales vers tous les pairs.

2.4.1.3 Scores Combinés (ScoresCombinés)

D'autres solutions ont été proposées dans les travaux de [Callan 1995] et [Ponte 1998, Si 2002b] afin de pondérer les scores de documents par le poids de leurs collections. L'algorithme CORI de fusion de résultats, proposé par Callan [Callan 1995], est l'un des algorithmes les plus utilisés pour l'agrégation de scores. L'approche considère les collections des différents systèmes de recherche comme des super-documents. L'algorithme est basé sur des réseaux d'inférence, dans lesquels les feuilles représentent des collections de documents et les statistiques de nœuds représentent les termes de la collection. Le score final d'un document, selon l'algorithme, est basé sur une normalisation combinée entre les scores des collections et ceux des documents. La même logique est utilisée dans l'approche PlanetP [Cuenca-Acuna 2003] pour le classement global des documents. Ce classement est basé sur l'utilisation du poids du pair de façon analogue à la mesure idf, approximée ici par l'ipf (qui désigne la fréquence de pairs stockant le document). L'algorithme de classement fournit des capacités de recherche comparables à celles en centralisées et fonctionne indépendamment de la façon dont les documents sont distribués. Chaque pair sélectionné pour répondre à une requête retourne un ensemble d'URLs de documents ainsi que leur valeur de pertinence combinant le poids du pair avec celui du document. Cette combinaison est possible grâce à l'utilisation de deux types d'index : (i) un index local au niveau de chaque pair afin de représenter le contenu de ses ressources en se basant sur une représentation vectorielle et (ii) un index global pour décrire tous les pairs ainsi que l'information qu'ils partagent dans la communauté. L'index global est répliqué sur tous les pairs. Afin de réduire la bande passante utilisée, chaque pair résume l'ensemble des termes de son index local dans un filtre de Bloom [Broder 2002]. L'approche PlanetP s'avère consommatrice de

bande passante malgré la structure compacte de son filtre de Bloom puisqu'il faut échanger les résumés à chaque mise à jour. De plus, l'autonomie des pairs n'est pas respectée puisque chaque pair publie l'ensemble de son index partout dans le réseau.

Le Modèle de Langue [Ponte 1998] est un autre exemple de méthode de score des documents. Son application aux environnements distribués est directe dans la mesure où on considère les collections de pairs distants comme une concatenation des documents qui les constituent nommés "super-documents". Le Modèle de Langue attribue une probabilité à une séquence de m termes $P(w_1, ..., w_m)$ par le biais d'une distribution de probabilité. Lorsqu'il est utilisé en recherche d'information, un Modèle de Langue est associé à un document dans une collection. À la soumission d'une requête q, les documents recherchés sont classés sur la base de la probabilité que le Modèle de Langue du document puisse générer les termes de la requête. Cette classe de modèle exige que les systèmes de recherche échangent suffisamment d'information pour pouvoir donner de l'information aussi bien sur la collection que sur les documents. Cet échange se fait au détriment du respect de l'autonomie des pairs, si nous désirons l'appliquer dans le contexte d'agrégation en RIP2P. En outre la procédure d'agrégation s'avère très liée à celle de la sélection de ressources.

Dans ce qui suit, nous allons explorer une deuxième famille de modèles d'agrégation largement utilisés en méta-recherche où les SRIs, contribuant à la résolution d'une requête, ne fournissent rien sur leurs contenus. Nous parlons alors de modèles à base de rangs.

2.4.2 Modèle à base de rangs

Le modèle d'agrégation à base de rangs est aujourd'hui privilégié car il constitue une famille de méthodes à faible coût. En effet, il nécessite un simple rang pour interclasser, sans avoir aucune information sur les statistiques globales des documents retournés. Seuls les rangs sont considérés pour le calcul de poids finaux de documents. Plusieurs méthodes à base de rangs sont connues dans la littérature. Nous passons en revue trois exemples de méthodes de cette classe, à savoir la méthode Round Robin, Borda count et les méthodes floues.

2.4.2.1 Round Robin (RR)

La méthode " Round Robin " [Voorhees 1995a, Voorhees 1995b, Steidinger 2000] permet d'agréger des documents selon leurs rangs dans les listes de résultats retournées par les systèmes de recherche sélectionnés (les premiers puis les deuxièmes et ainsi de suite). Le premier document dans la liste finale sera le premier document de la première liste retournée et ainsi de suite. Le choix des listes se fait de manière aléatoire. Si et al affirment dans [Si 2003] que la méthode Round Robin (RR) est un choix à faire si les scores individuels des documents sont complètement in-

comparables (lorsque les systèmes de recherche utilisent différents algorithmes ou lorsque les collections utilisent différentes statistiques). Cependant, les travaux de [Savoy 2002] ont montré que les meilleurs résultats de la méthode RR sont observés lorsqu'il s'agit des mêmes stratégies de recherche. Le problème de la non compatibilité des listes de résultats peut être atténué par la normalisation des rangs des différents documents. La simplicité de l'approche Round Robin ne fait pas d'elle certainement la meilleure puisqu'elle ne tient pas compte de l'importance relative des systèmes de recherche. En utilisant cette méthode, chaque système de recherche a les mêmes chances d'être classé premier, même s'il est moins pertinent que d'autres relativement à une requête. Dans les travaux de [Steidinger 2000], l'efficacité de la méthode RR comparée à une collection centralisée excède la performance obtenue par une méthode à base d'*Idf*. Cette constatation nous a incité à utiliser la méthode RR comme l'une des méthodes d'agrégation de référence auxquelles nous souhaitons nous comparer.

2.4.2.2 Borda Count(BC)

Faisant partie de méthodes de votes positionnelles, la méthode BC sert à combiner les préférences de plusieurs experts. L'entrée d'un système de vote est dite profil. Afin d'adapter les algorithmes de vote au problème de la recherche d'information distribuée, l'ensemble des documents à classer sont considérés comme candidats. Les votants sont les systèmes de recherche se présentant en entrée de l'algorithme d'interclassement. Borda count se base sur le principe suivant [Borda 1781] : chaque votant classe un ensemble de "c" candidats selon un ordre de préférence [Farah 2007]. Pour chaque votant, le premier candidat acquièrt "c" points, le deuxième acquière "$c-1$" et ainsi de suite. S'il reste des candidats non classés par les votants, les points restants seront divisés équitablement entre les candidats non classés. L'agrégation se fait en triant la liste des documents par ordre décroissant du nombre de points cumulés. L'un des inconvénients de la méthode Borda Count est qu'elle se contente de diviser le nombre de points restant de façon équitable entre les documents manquants dans la liste. Ceci réduit considérablement l'importance des documents manquants, qui se voient toujours classés aux dernières positions. Le deuxième inconvénient de la méthode Borda Count est qu'elle attribue une importance égale à tous les systèmes de recherche, ce qui n'est pas réaliste. Une mesure de préférence peut améliorer l'efficacité de cette méthode.

2.4.2.3 Méthodes Floues

Les méthodes d'agrégation floues reposent sur le principe des ensembles flous introduit par Zadeh en 1965 [Zadeh 1965]. Elles fournissent un fondement mathématique pour estimer l'incertitude associée au processus cognitif humain. En recherche d'information, les approches floues ont été utilisées dans les différentes phases du processus de recherche puisqu'il s'agit d'un problème sujet à l'incertitude et à l'imprécision lexicale.

Basées sur les opérateurs d'agrégation flous, ces approches ont été appliquées pour la résolution des problèmes de l'agrégation des résultats. La pertinence d'un document peut être modélisée comme un concept relatif. Une approche d'agrégation floue se présente sous forme d'un problème de prise de décision multicritères [Fuller 1996]. Les critères sont les différents systèmes de recherche et leurs différentes alternatives sont les documents retournés. Les opérateurs flous permettent aux décideurs de s'impliquer dans le processus d'agrégation en imposant une logique de satisfaction plus ou moins rigoureuse des différents critères participant dans le classement des alternatives [Fuller 1996]. Ce jugement est généralement représenté via un quantifieur linguistique lié à ce modèle. Les quantifieurs linguistiques servent à calculer les jugements de pertinence globaux pour chaque document. Dans la littérature, plusieurs types de quantifieurs linguistiques sont cités et ils sont tous compris entre la satisfaction de tous les critères et la satisfaction d'au moins un critère. Un quantifieur "TOUS", par exemple, propose la fusion des seuls documents qui apparaissent dans toutes les listes de résultats. Il favorise l'intersection des listes. Ce quantifieur favorise le maximum " Andness " (ET). À l'autre extrême, nous retrouvons ceux qui considèrent tous les documents retournés, même ceux apparaissant dans une seule liste de résultats ou satisfaisant un seul critère : cette agrégation maximise le critère OU ou l' "Orness" des critères. Entre ces deux types de quantifieurs, plusieurs autres existent favorisant soit le "andness" ou le "orness" des critères. Le quantifieur considéré dans les travaux de [De 2007] est noté Regular Increasing Monotone (RIM) et il est considéré comme degré standard d'Orness [Chiclana 2007].

Ce que nous pouvons constater pour cette classe de méthodes, c'est qu'elle base son classement sur une notion de popularité des documents. Un document peut être considéré comme important s'il est retourné par toutes les listes, ou dans une autre logique s'il est retourné par au moins une liste. L'hypothèse de ces méthodes est basée sur une logique de fusion de données qui considère un chevauchement important entre les listes retournées par tous les systèmes de recherche. Dans une logique de RIP2P, les pairs présentent un taux de chevauchement [Chernov 2005] assez réduit, ce qui fait que cette classe de méthodes ne semble pas adaptée à notre choix d'agrégation.

2.4.3 Les Modèles à base d'apprentissage

L'apprentissage représente un concept de base pour un grand nombre de domaines d'application. En Recherche d'Information, il a été utilisé de différentes manières afin d'apporter des gains en qualité aux systèmes de recherche, notamment pour augmenter leur précision, rappel, etc. Généralement, le recours à l'apprentissage est expliqué par un manque de statistiques, ce qui est le cas lorsque les nœuds d'un système distribué sont autonomes et ne fournissent aucune information sur leurs contenus. Dans la classe des méthodes à base de score, nous avons déjà cité des exemples d'utilisation de l'apprentissage, pour convertir des scores locaux en globaux. Dans cette section, nous citons des exemples pour des approches basées

sur des méthodes d'apprentissage supervisés et non supervisés. Nous considérons dans le dernier type d'apprentissage les méthodes basées sur le comportement de l'utilisateur qui représente un résultat des interactions implicites ou explicites entre l'utilisateur et les systèmes de recherche.

2.4.3.1 Apprentissage supervisé pour la pondération des systèmes de recherche

En agrégation de résultats, l'apprentissage est typiquement utilisé pour le classement des systèmes de recherche. Le but derrière ce classement est d'attribuer des niveaux de confiance à ces systèmes et par conséquent aux documents qu'ils retournent. Il s'agit d'un apprentissage supervisé [Frécon 2009] qui se base sur les mesures d'évaluation les plus communément utilisées dans le contexte de la recherche d'information : le rappel ou/et la précision ou sur des méthodes qui en sont dérivées. Les données d'apprentissage prennent souvent la forme d'un ensemble de requêtes, leurs réponses, accompagnées de jugements de pertinence qui représentent une évaluation humaine de la pertinence d'un document par rapport à une requête. Cette famille de méthodes d'apprentissage qui permet d'affecter des poids aux systèmes de recherche, aide à améliorer les processus de sélection des collections et d'agrégation. Ceci a été appliqué à différentes méthodes notamment BC, RR ce qui contribue à des améliorations de leurs performances [Steidinger 2000].

2.4.3.2 Méthodes basées sur la classification non supervisée

La classification non supervisée, ou *clustering* est un processus qui essaie de détecter une certaine relation entre un ensemble d'objets afin de pouvoir les organiser dans des groupes homogènes non préalablement fixés [Lefebure 2001]. En Recherche d'Information, le clustering des documents a été utilisé aussi bien avant qu'après la récupération des résultats. La motivation essentielle derrière l'application du processus de clustering repose sur le fait que des documents similaires sont jugés plus pertinents pour répondre à une même requête que des documents non similaires. Dans [Zhang 2001], Zhang et al présentent un exemple d'application du clustering pour améliorer le processus d'agrégation de résultats. Cette méthode est basée sur deux hypothèses : les documents pertinents pour une même requête peuvent être regroupés ensemble. Ainsi, un algorithme de clustering efficace doit être capable de séparer facilement les documents pertinents des non pertinents.
Dans les listes de résultats retournés, un document est jugé pertinent s'il apparaît dans plusieurs listes. De telles listes (qui présentent des document en commun) sont dites chevauchées. En se basant sur cette hypothèse, on peut déduire qu'un cluster n'est jugé fiable que s'il présente un haut potentiel de chevauchement.
Le processus de clustering présenté dans [Zhang 2001] est déclenché dès que les systèmes de recherche locaux retournent leurs listes. Chaque liste de résultats est ainsi divisée en un nombre de clusters créés par l'algorithme *k-means* [MacQueen 1967]. Des mesures de similarité sont utilisées entre deux clusters et entre une requête et

un cluster. Ainsi, la pertinence d'un document dépend de sa pertinence par rapport à la requête tout en tenant compte de la fiabilité du cluster dans lequel il apparaît. La pertinence globale du document est déduite de la somme de ses scores dans chaque liste : nous parlons de la combinaison "CombSum" [Fox 1994].
Zhang et al [Zhang 2001] considèrent qu'une fusion efficace est obtenue en tenant compte du facteur chevauchement entre listes de résultats. En effet, ces hypothèses ont été confirmées par les expérimentations puisque les meilleures résultats de combinaison ont été obtenus au sein des clusters présentant 80% de chevauchement. Ces hypothèses ne peuvent être adoptées dans notre contexte d'agrégation en RIP2P. En effet, la première raison est que la formation des clusters exige une grande coopération entre les pairs, ce qui contredit notre hypothèse d'autonomie, la seconde raison est que les pairs sont des entités très dynamiques et que l'ensemble de ces ressources peuvent changer, ce qui impose de grandes contraintes sur le modèle. Enfin, l'hypothèse de fiabilité basée sur le chevauchement, ne peut pas être adoptée dans le cadre des pairs qui ne présentent qu'un pourcentage de chevauchement assez réduit [Chernov 2005].

2.4.3.3 Méthodes basées sur le comportement des utilisateurs

En se basant sur le principe qu'un utilisateur, qui soumet une requête à un système, est le seul qui soit apte à juger les réponses, les chercheurs se sont orientés vers des essais d'intégration du comportement utilisateur dans le processus de RI. Le bouclage de pertinence (Relevance Feedback) a été l'une des premières formes d'expression des préférences de l'utilisateur vis à vis de ce qu'il attend comme réponse d'un SRI, afin d'en améliorer la qualité des résultats [Rocchio 1971].
Un bouclage de pertinence consiste à prendre les résultats qui sont retournés initialement à partir d'une requête donnée et d'en indiquer la pertinence (du point de vue de l'utilisateur). Un bouclage de pertinence peut être soit donné explicitement par l'utilisateur, soit déduit implicitement à partir de ses interactions avec la réponse. Entre les deux modèles de pertinence, nous parlons de bouclage de pseudo pertinence.

Les modèles d'agrégation qui ont intégré un apprentissage basé sur un comportement humain ou une simulation de ce dernier, ne sont pas nombreux dans le cadre de la Recherche d'Information Distribuée à Large Echelle (RIDLE). Nous présentons dans cette section les deux principaux travaux qui ont essayé d'intégrer le comportement d'utilisateur dans la sélection des sources et en agrégation des résultats dans les réseaux P2P. Un comportement utilisateur est souvent intégré dans un concept appelé *profil utilisateur* [Chernov 2005, Witschel 2005, Witschel 2008].

Dans [Chernov 2005], l'auteur propose l'utilisation d'un modèle de préférence afin d'améliorer l'efficacité de la fusion des résultats dans un système P2P structuré. Ce travail rentre dans le cadre du projet Minerva, qui est un moteur de recherche Web P2P basé sur Chord [Stoica 2001]. Dans ce projet, les pages sont visitées

en exploitant les traces des utilisateurs, ce qui permet de refléter leurs intérêts spécifiques. En partant de ce principe, Chernov et al. [Chernov 2005], ont essayé d'exploiter ces intérêts en utilisant le feedback de pseudo-pertinence pour retrouver le modèle qu'ils appellent Modèle de Langue de préférence. Pour ce faire, ils ont simulé la préférence de l'utilisateur par *les top-k réponses retournées par le meilleur nœud*. Ce dernier est le résultat d'une procédure de sélection basée sur l'utilisation de statistiques globales (apprises à partir de statistiques locales collectées au niveau de chaque pair). Le Modèle de Langue [Ponte 1998] est appliqué, globalement, pour classer les pairs par ordre de pertinence, par rapport à la requête. Dès lors, le profil utilisateur ajouté à la requête sera propagé aux différents pairs sélectionnés (à partir du deuxième pair) et localement exécuté au niveau de chaque pair sélectionné, en se basant sur le Modèle de Langue qui cherche la probabilité qu'un pair sélectionné puisse générer le modèle de préférence de l'utilisateur ainsi que la requête d'origine.

Ce que nous pouvons remarquer dans le cadre de ce travail, est que le comportement utilisateur n'a pas été inféré d'une vraie interaction de l'utilisateur avec le système, mais a été déduit des résultats de la sélection du meilleur pair. Ceci fait que l'agrégation dans cette méthode dépend de la réussite de la sélection des pairs pertinents. En deuxième lieu la sélection même des pairs est basée sur un échange de statistiques locales, ce qui viole l'autonomie des pairs et impose une certaine surcharge au système. De plus, il suppose d'avance que les pairs sont suffisamment stables pour fournir des informations sur leurs contenus.

Une autre expérience d'utilisation des profils en RIP2P est citée dans [Witschel 2005, Witschel 2008]. Witschel et al., ont proposé une approche de sélection des meilleurs pairs d'un système de RIP2P en se basant sur une procédure d'apprentissage semi-supervisée. Il s'agit d'une approche visant à remplacer l'utilisation des statistiques globales par le développement d'un mécanisme qui permet de distinguer les pairs utiles des autres. Ce mécanisme est basé sur une conception d'un profil sur chaque pair. Le profil d'un pair est défini dans [Witschel 2005, Witschel 2008] comme étant l'ensemble de termes t contenus dans les documents de ce pair accompagnés par des statistiques locales relatives au pair. Le comportement de l'utilisateur a été exploité aussi bien dans la procédure de routage des requêtes que dans le renforcement des poids des termes contenus dans le profil d'un pair. Ces derniers sont préalablement calculés en se basant sur la méthode CORI [Callan 1995, Callan 2000] qui calcule la croyance qu'un terme t trouve sa réponse dans le pair p.

L'approche intervient, lors du processus d'agrégation, pour privilégier les documents ayant les termes à poids renforcés. La procédure de renforcement consiste à ajouter un bonus au poids des termes qui ont été sollicités pour répondre à une requête utilisateur dans ce pair. Les données concernant les termes sont directement déduites à partir de l'historique des requêtes.

L'auteur propose également l'utilisation d'un échantillon large de données contenues dans les collections des pairs afin d'estimer des mesures globales d'*idf* pour chaque terme du profil et rendre ces termes plus comparables. La question clé dans

les travaux de [Witschel 2005, Witschel 2008] est la capacité à compresser la taille des profils pour qu'ils contiennent un nombre réduit de termes représentatifs. Une fois le processus de sélection terminé, les profils renforcés peuvent guider le processus d'agrégation en favorisant les documents contenant des termes renforcés pour la requête. Le classement interne des documents est basé sur la fonction de similarité Okapi BM25 [Robertson 1995].

La proposition de Witschel est intéressante puisqu'elle utilise des informations locales comprises dans le profil d'un pair ce qui permet le passage à l'échelle de la solution proposée. De plus, les résultats empiriques ont prouvé que cette méthode donne des résultats meilleurs que les méthodes à base de scores (notamment LM et CORI). Cependant, l'utilisation de mesures globales basées sur un échantillon large suppose comme dans la proposition de [Chernov 2005] que les pairs ne sont pas autonomes. De plus, ils ne considèrent pas les statistiques des pairs qui se connectent ultérieurement au système (après échantillonnage).

2.4.4 Discussion

Les principales méthodes citées précédemment ont été développées dans le contexte de RID et ont donné des résultats acceptables. Cependant, leur utilisation dans un système distribué à large échelle, notamment les systèmes P2P, doit être étudiée en prenant en considération les caractéristiques de ces systèmes. Les différentes caractéristiques que nous avons utilisées comme critères de choix des modèles d'agrégation sont les propriétés requises des systèmes d'agrégation en RIP2P présentées dans la section 2.3.2. Ainsi, un modèle d'agrégation doit, selon l'étude réalisée dans 2.3.2, privilégier l'autonomie des pairs, la décentralisation totale, le passage à l'échelle, l'hétérogénéité des collections sur l'ensemble des pairs et l'efficacité en matière de RI. Nous essayons d'analyser, à travers la table 2.2, l'adaptabilité des modèles d'agrégation étudiés à de telles propriétés.

La Table 2.2 montre beaucoup de limitations des méthodes citées pour des systèmes P2P. En fait, nous pouvons voir ici que le score à base de statistiques (Scores-B-statistiques) qui est basé sur un index global peut donner une bonne précision en terme de classement et c'est pourquoi elles sont toujours considérées comme efficaces. Cependant, cette précision se paye en terme de centralisation où nous tombons dans le problème du goulot d'étranglement traditionnel. De plus, une grande coopération est requise afin d'échanger de l'information globale ce qui viole l'aspect autonomie des pairs et est très coûteux en terme de consommation de bande passante. En ce qui concerne l'échange des statistiques locales, le problème se transforme en un problème de violation de l'autonomie des pairs.

La méthode de score à base de normalisation (N-Scores-Locaux), qui se base sur des scores partiels, s'avère selon le bilan de la Table 2.2 moins consommatrice en terme de coopération explicite comparée à la méthode précédemment citée. La centralisation peut être évitée puisqu'on peut être basée seulement sur la norma-

	Modèles à base de scores			Modèles à base de rangs			Modèles à base d'apprentissage		
	Scores-B-Statistiques	N-Scores-Locaux	ScoresCombinés	RR	BC	M.floues	M.pond	M.Nor	M.profils
Autonomie	non	non	non	oui	oui	oui	non	non	non
Décentralisation	non	moy	oui	oui	oui	oui	non	non	moy
Le passage à l'échelle	non	non	non	oui	oui	oui	non	non	moy
Hétérogénéité	oui	non	moy	non	non	non	oui	moy	oui
Dynamicité	non	non	non	oui	moy	non	non	non	moy
Efficacité	oui	moy	oui	moy	moy	moy	oui	oui	oui

TABLE 2.2 – Etude de l'adaptation des modèles d'agrégation classiques à l'agrégation des résultats en RIP2P

Légende :

- M.floues : Méthodes floues
- M.pond : Méthodes de pondération des collections
- M.Nor : Méthodes de normalization basées sur l'apprentissage
- M.Profils : Méthodes basées sur l'utilisation du comportement de l'utilisateur.

Nous avons pris comme exemple la méthode proposée par Witshel [Witschel 2008] et citée dans la section 2.4.3.3

Les valeurs présentées dans le tableau sont les suivantes :

- oui : la méthode supporte le critère
- non : la méthode ne supporte le critère
- moy : la méthode supporte moyennement le critère

lisation. Cependant l'hétérogénéité des collections sur les différents pairs rend les scores partiels, de plus en plus, incomparables et réduit clairement l'efficacité du modèle.

Ce problème peut être surmonté lorsque de l'apprentissage est utilisé comme dans le montre les travaux de [Le Calvé 2000, Si 2002a] mais nous rappelons qu'un échantillon centralisé ne peut pas être représentatif pour l'ensemble des pairs à cause de la dynamicité (des pairs peuvent apparaître ou disparaître après construction de l'échantillon).

Les méthodes de Scores combinés (ScoresCombinés) sont basées sur des statistiques partielles où l'échange se fait dans des conditions acceptables. Toutefois, ces méthodes, comme présenté dans la Table 2.2, perdent en termes de qualité par rapport à l'utilisation d'un index global dans les méthodes dites (Scores-B-statistiques). Dans le contexte de méthodes de ScoresCombinés, la stratégie de fusion basée sur CORI par exemple n'est qu'une combinaison linéaire des scores des pairs sélectionnés avec ceux des documents. Cette approche combine heuristiquement les scores locaux avec les rangs des serveurs. Les rangs des serveurs sont déduits lors de l'étape de sélection. Cette approche exige de même que les pairs utilisent le même modèle d'indexation et de recherche.

Nous pouvons affirmer, comme présenté dans la Table 2.2, que les modèles à base de rangs semblent être plus adaptés aux propriétés des systèmes P2P. En effet, avoir des listes de résultats locales triées est suffisant pour avoir l'information "rang". Dans cette catégorie de méthodes, seul une faible coopération entre les SRIs contributeur est nécessaire. Le gain de la faible coopération est double : le passage à l'échelle et le respect de l'autonomie des pairs. Cependant, ces modèles s'avèrent moins efficaces (côté RI) lorsque les collections sur chaque pair présentent un haut niveau d'hétérogénéité. En effet, l'hétérogénéité des pairs augmente considérablement la non comparabilité des rangs. En outre, les rangs sont moins interprétables pour le processus d'agrégation que les scores. La méthode RR, par exemple, est basée sur l'hypothèse que toutes les collections choisies ont le même nombre de documents pertinents et fonctionne tout simplement en entrelaçant les documents provenant des collections individuelles, un par un.

Une autre hypothèse de la plupart des algorithmes est que les collections distantes retournent, non seulement de simples listes de classement des documents, mais aussi les scores assignés à ces documents. Par conséquent, ces approches utilisent généralement des scores ou rangs locaux afin de fusionner les résultats retournés. Nous pouvons voir aussi, que les modèles utilisant des scores, des rangs ou statistiques partielles sont subjectifs et que leur qualité est proche des performances des algorithmes locaux de RI. L'intérêt local de l'utilisateur qui est un axe central du processus de recherche n'est pas considéré par ces modèles.

En se basant sur les travaux de Witshel et al [Witschel 2005, Witschel 2008],

nous déduisons tel que dévoile la Table 2.2, que l'utilisation d'un profil implicite des pairs basé sur l'interaction avec les réponses des requêtes soumises sur ce pair peut améliorer la performance de l'algorithme d'agrégation. Ce profil généré localement sur chaque pair permet de fournir une information locale pertinente pour favoriser un document par rapport à un autre. L'utilisation des profils a été bien décentralisée, et assure le passage à l'échelle. Dans l'approche de Witschel, seul le recours aux statistiques globales n'est pas à notre avis bénéfique pour fournir une solution totalement décentralisée et qui respecte l'autonomie des pairs.

Cette analyse nous a permis de présumer que l'utilisation des rangs d'une part pour donner une idée sur les décisions locales d'un pair, et de l'intérêt estimatif de l'utilisateur (profil) d'autre part peuvent être un avantage pour améliorer la qualité de l'agrégation des résultats sans avoir besoin d'une coopération des nœuds du système. En effet, une partie importante des intérêts spécifiques des utilisateurs peut être construite à partir des requêtes passées de l'utilisateur. Nous pouvons exploiter ce fait, en utilisant une heuristique non supervisée pour trouver le profil le plus approprié à partir des requêtes passées les plus pertinentes.

2.5 Conclusion

Dans ce chapitre, nous avons présenté la problématique de notre thèse avec l'exposition de ses problèmes clés et des solutions proposées dans la littérature pour les résoudre. Nous avons commencé par la définition du problème d'agrégation en général. Puis, nous avons présenté l'état de l'art en montrant les problèmes posés par un environnement massivement distribué, tel que les systèmes P2P non structurés. Nous avons proposé une catégorisation des modèles d'agrégation existants en explorant leurs adaptabilités aux propriétés requises d'un modèle d'agrégation en RIP2P. Cependant, le manque de statistiques globales et le coût de leur acquisition dans un environnement massivement distribué, nous ont incités à chercher une dimension indépendante des statistiques des collections, pertinente et facile à construire sur chaque pair. Nous parlons de profil utilisateur. Partant de cette notion de profil, qui a prouvé son efficacité dans des travaux antérieurs, nous proposons, dans le chapitre suivant, un modèle d'agrégation adapté au contexte P2P et s'appuyant sur les profils.

Chapitre 3

Modèle d'agrégation hybride à base de profils

3.1 Introduction

Nous avons essayé d'introduire, dans les chapitres précédents, la problématique d'agréger les résultats dans un système distribué à large échelle, tels que les systèmes P2P. Plusieurs limites ont été recensées quant à l'adoption des approches de RID pour l'agrégation des résultats dans un système de RIP2P, vu l'hétérogénéité des informations à agréger (de point de vue taille, contenu et modèles de RI) d'une part, et l'absence d'une vision globale du système, d'autre part.
À la lumière de ces limites, nous avons essayé d'exploiter, dans le présent chapitre, une nouvelle piste pour agréger les résultats. Cette dernière vise à améliorer la pertinence des résultats, de point de vue de l'utilisateur, sans imposer des contraintes supplémentaires sur le système. Par ailleurs, nous faisons appel aux modèles de P2P sémantiques que nous avons introduit dans le chapitre 1. La sémantique introduite, consiste en un ensemble de profils d'utilisateurs qui vont nous guider à prédire ses intentions (lors de l'expression de ses nouveaux besoins), à partir de l'exploitation de l'historique de ses interactions, avec les réponses à des besoins passés.
Pour ce faire, nous définissons un modèle d'agrégation hybride qui se base sur les étapes suivantes :

1. La définition d'un modèle de profils permettant d'intégrer la connaissance dans l'approche d'agrégation.
2. La définition d'une méthode hybride de calcul de score global (de manière distribuée) qui intègre, à côté des rangs des documents, la connaissance issue de l'utilisateur.
3. Un algorithme d'agrégation à base de profils (PBA) qui sera suivi d'une étude de complexité.

Pour commencer, une architecture présentant les briques de base de notre modèle d'agrégation sera explicitée.

3.2 Architecture du modèle d'agrégation

Nous visons à proposer un modèle d'agrégation qui permet d'éviter, d'une part, le coût de l'utilisation des statistiques et d'améliorer, d'autre part, l'efficacité du

processus de recherche d'information dans un système distribué à large échelle, en se basant sur les profils des utilisateurs.
L'architecture du modèle d'agrégation a pour but de présenter l'approche d'agrégation proposée qui se base, aussi bien sur les rangs, que sur la connaissance, sous forme d'un ensemble de profils. Nous rappelons, à travers cette architecture, les différentes parties ou modules impliqués, dans la résolution d'une requête jusqu'à en agréger les résultats de manière globale. Il convient de rappeler, également, les rôles des pairs impliqués dans cette architecture : les pairs initiateurs qui lancent les requêtes sur le système et les pairs contributeurs qui contribuent à la résolution de ces requêtes (voir Section 2.3 du Chapitre 2).
Nous présentons, dans cette section, l'architecture fonctionnelle du modèle d'agrégation à base de profils, reproduite au niveau de chaque pair initiateur. La Figure 3.1 nous permet d'identifier trois principales couches indispensables pour l'agrégation. Nous citons respectivement :

- **La couche de résolution globale de requêtes** : utilisée par les pairs initiateurs et comprenant deux composants principaux : un composant pour la sélection des pairs et un deuxième, pour l'agrégation des résultats provenant des pairs sélectionnés.

- **La couche de résolution locale de requêtes** : exécutée sur les pairs contributeurs (à la résolution de requêtes). Chaque pair utilise sa propre méthode de RI pour rechercher et classer les résultats pertinents par rapport à la requête soumise.

- **La couche de gestion de profils** : permettant de fournir les intérêts des utilisateurs, lors de la résolution des requêtes similaires passées. Ces intérêts vont aider à orienter le système sur les attentes des utilisateurs, lors de la résolution de nouvelles requêtes. Les profils des utilisateurs sont générés localement sur tous les pairs et de manière hors ligne, mais seuls les profils d'un pair initiateur seront utilisés afin d'agréger les résultats des requêtes qui y sont lancées. La localité des profils favorise l'aspect de décentralisation qui favorise à son tour le passage à l'échelle, indispensable aux systèmes de RIDLE. L'aspect hors ligne évite quant à lui, à l'utilisateur le temps d'attente dû à la construction des profils lors de sa recherche.

La couche de résolution globale de requêtes vise à attribuer des scores globaux à toutes les réponses aux requêtes afin de les trier de façon uniforme. Le module de sélection des pairs dans cette couche, est responsable de la propagation des requêtes à tous les pairs pertinents dans le système. Cette propagation se fait selon différents algorithmes, qui sont hors de la portée de cette thèse.
Chaque pair, sélectionné au cours de cette étape, se charge de la résolution locale de la requête (dans la couche résolution locale des requêtes) visant à rechercher les meilleures réponses pertinentes correspondant à la requête. La couche de résolution

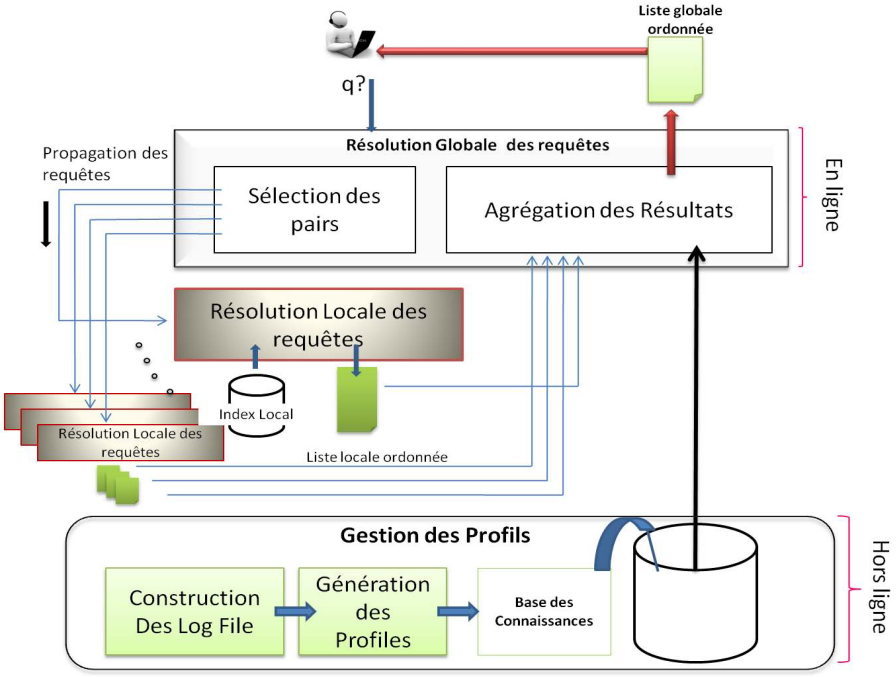


FIGURE 3.1 – Architecture du Modèle d'agrégation

locale de requêtes utilise l'index local et le modèle de RI du pair contributeur pour retourner une liste locale classée au pair initiateur.

Une fois les listes locales, classées issues de la résolution locale de la requête sont restituées, le pair initiateur exécute le module d'agrégation. Le module d'agrégation prend en entrée les rangs locaux des documents (fournis par les pairs contributeurs) d'une part et les connaissances du pair fournies par la couche de gestion de profils, d'autre part. Les connaissances produites sont basées sur une approche de log-mining appliquée de manière hors ligne par pair initiateur. Cette approche permet une analyse du comportement d'un pair, et reflète ainsi les intentions de l'utilisateur quand il interagit avec les pairs contributeurs.

Selon cette architecture, les pairs contributeurs sont chargés seulement, de la résolution locale des requêtes. Les pairs initiateurs peuvent effectuer, outre le processus de résolution globale, le processus local et ce, lorsque le module de sélection choisit ces pairs entre autres, pour résoudre la requête. Le pair initiateur utilise également la couche locale de gestion de profils. La connaissance est utilisée afin d'améliorer la performance du processus d'agrégation.

Les propriétés citées dans l'architecture actuelle confirment que notre modèle d'agrégation vise à satisfaire les contraintes suivantes :

– **Une vision entièrement décentralisée** : chaque pair produit localement les profils de ses utilisateurs, sans interagir avec d'autres pairs. Ces profils sont utilisés également d'une manière purement locale afin d'améliorer le processus

d'agrégation de résultats dans le pair initiateur.

- **Une autonomie totale** : il n'est pas nécessaire de partager des ressources ni de demander une coopération explicite entre pairs afin d'agréger le contenu des profils locaux dans une structure centralisée. Chaque pair est responsable des traces de ses utilisateurs.

- **Évolutivité** : L'absence d'un contrôle global fait que le système peut être étendu naturellement sans imposer des contraintes additionnelles, ni sur son contenu, ni sur son profil. A travers une approche de mise à jour que nous détaillerons par la suite dans le chapitre 6, les profils dans chaque pair évoluent pour intégrer des nouvelles informations concernant les nouveaux besoins sur un pair.

- **Disponibilité** : la connexion et la déconnexion des pairs contributeurs n'affecte pas le processus d'agrégation, car les résultats ne dépendent que de l'information locale.

Dans ce qui suit, nous présentons plus en détails la couche de gestion profils à travers la présentation du modèle du profils proposé. Une formalisation des différentes étapes nécessaires à l'intégration des profils dans notre modèle d'agrégation est envisagée. En effet, afin de manipuler les profils utilisateurs, nous avons opté pour la définition d'un ensemble d'opérateurs permettant la sélection des profils, la recherche de meilleurs services (documents) et de pairs.

3.3 Modèle de profils utilisateurs

L'intégration des préférences des utilisateurs nécessite, comme présenté dans [Bouzeghoub 2005, Defude 2007], un processus qui consiste, tout d'abord, à définir ce que représente un profil dans une application, puis à présenter la manière de l'acquérir et de le représenter et enfin à l'intégrer dans l'application en question.
Nous présentons, dans ce qui suit, les étapes nécessaires pour l'intégration du profil utilisateur dans notre modèle d'agrégation.

3.3.1 Définition du profil utilisateur

Les auteurs dans [Wahlster 1986] définissent le profil utilisateur comme suit :

Définition 3.1
"Un profil utilisateur" (ou encore modèle utilisateur) est un ensemble de données qui concernent l'utilisateur d'un service informatique. C'est une source de connaissances qui contient des acquisitions, sur tous les aspects de l'utilisateur qui peuvent être utiles pour le comportement du système.

Dans notre contexte de RIDLE, nous définissons un profil utilisateur comme étant l'ensemble des intérêts des utilisateurs déduits de leurs interactions avec un système de recherche (un pair), lorsque ce dernier répond à une requête passée. En d'autres termes, un profil utilisateur est l'expression d'un besoin en information et de ce qui découle de sa résolution, comme corrélations sémantiques reliant ce besoin à ses résultats. Il inclut l'historique des ressources (documents) téléchargées, consultées, et des pairs retournant ces ressources aux utilisateurs lorsqu'ils expriment un besoin en information. L'ensemble de ces profils constituent une base de connaissances locale détenue par un pair initiateur.
Nous présentons formellement, les différents concepts utilisés dans la définition de profils.

Définition 3.2
Un n-uplet est une collection d'éléments qui se rapportent les uns aux autres :
$u = (p_1 : type_1,\ p_2 : type_2,\ ...,\ p_n : type_n)$ où p_i est une propriété (i.e objet ou attribut) de u et $type_i$ est son type.

Exemple 1: Afin d'instancier cette définition à notre contexte, nous considérons les ensembles suivants $\mathcal{Q}$, $\mathcal{T}$, $\mathcal{P}$ et $\mathcal{D}$ qui dénotent respectivement : un ensemble des identifiants des requêtes posées par les utilisateurs, l'ensemble leurs termes relatifs, un ensemble des pairs interrogés et un ensemble de ressources recherchées (documents). Un n-uplet peut être décrit comme suit :
$u = (q : String, T_q : Set,\ P_q : Set,\ D_q : Set)$, avec :

- $q \in \mathcal{Q}$: une requête utilisateur (ou l'identifiant d'une demande).
- $T_q \subset \mathcal{T}$: une liste de thèmes (ou termes) de la requête utilisateur q.
- $P_q \subset \mathcal{P}$: une liste de pairs positifs (Pairs contributeurs) qui ont "bien" répondu à la demande de l'utilisateur.
- $D_q \subset \mathcal{D}$: une liste de ressources (documents) ou services accédés par un utilisateur (après téléchargement, consultation, etc.) suite à sa demande.

Définition 3.3
Une base de connaissances $\mathcal{KB}$ est un ensemble de profils utilisateurs, plus formellement : $\mathcal{KB}$=$\{pr_1, ..., pr_p\}$, où pr_i est un profil utilisateur, $i \in [1, |\mathcal{KB}|]$.

Définition 3.4
Un profil utilisateur $pr \in \mathcal{KB}$ peut être modélisé sous forme d'un triplet $pr(\mathcal{T}, \mathcal{P}, \mathcal{D})$, où :

- $\mathcal{T} = \bigcap_{q \in \mathcal{Q}} T_q$: est un ensemble d'intérêts communs (termes) exprimés par des requêtes des utilisateurs de l'ensemble $\mathcal{Q}$.
- $\mathcal{P} = \bigcap_{t \in \mathcal{T}} P_q$: est un sous-ensemble de pairs contributeurs qui ont été sollicités pour répondre à des requêtes de termes de $\mathcal{T}$.
- $\mathcal{D} = \bigcap_{t \in \mathcal{T}} D_q$ est un ensemble de ressources (dans notre cas documents ou en général de services) correspondant aux besoins des utilisateurs.

En guise d'exemple nous pouvons considérer le profil $pr = (\{t_1, t_2, t_3\}, \{p_1, p_2\}, \{d_1, d_2\})$ qui peut être interprété via les corrélations suivantes : les requêtes utilisant les

thèmes t_1, t_2 et t_3 ont, comme bonnes réponses, les documents d_1 et d_2 provenant des pairs p_1 et p_2.

3.3.2 Acquisition des profils

L'acquisition du profil utilisateur est la deuxième étape indispensable à son utilisation. Cette acquisition peut être faite aussi bien d'une manière implicite qu'explicite. Dans une acquisition explicite, un utilisateur doit exprimer ses données personnelles ou ses préférences à travers la spécification de ses centres d'intérêt, le marquage de services ou, la réponse à des questions sur ses intérêts. Ceci présente l'inconvénient d'obliger l'utilisateur de s'impliquer dans le processus de recommendation, ce qui lui impose une charge additionnelle à son utilisation du système [Bouzeghoub 2005]. En plus, souvent l'utilisateur, qui se trouve obligé à remplir un formulaire, ou à répondre à une question, peut introduire des informations erronées. Pour ces raisons, les systèmes préfèrent utiliser une approche implicite pour extraire les profils des utilisateurs.
Les données utilisateurs peuvent être, dans ce cas, soit renseignées par sélection de profils pré-existants créés par des experts du domaine (profil expert), soit apprises par le système au cours de l'utilisation (profil dynamique) [Wahlster 1986]. Les techniques de Data Mining sont généralement utilisés pour extraire le comportement des utilisateurs tout en tenant compte de ses actions passées ainsi que ses choix [Bouzeghoub 2005].

Dans notre cas, nous avons recouru vers une acquisition implicite des intentions de l'utilisateur. Afin d'adopter une approche implicite, de nombreuses sources peuvent être exploitées pour capturer les préférences de l'utilisateur. L'auteur dans [Kostadinov 2006] énumère une taxonomie des sources de données typiques à partir desquelles on peut présumer le comportement des utilisateurs et leurs centres d'intérêt. Nous citons respectivement : les données de l'utilisateur, les données de contenu et les données de structure. Les données d'utilisation visent à capter les traces de l'utilisateur lors de l'interaction avec les serveurs. La source de données principale permettant de faire ceci a été représentée pendant de nombreuses années par les fichiers journaux (log files).

Dans notre approche, l'acquisition du comportement de l'utilisateur repose sur une génération du fichier journal (log file). Nous considérons, ici un journal par pair de telle sorte que l'acquisition de profils se fait de manière décentralisée. Cette génération du fichier log tient compte seulement de l'information qui présente un intérêt pour l'utilisateur. Cependant, cette information peut être étendue par d'autres de manière à être exploitée pour divers besoins (sélection des ressources, agrégation de résultats, l'expansion de requêtes, etc.)

De manière formelle, l'extraction du comportement de l'utilisateur, qui repose l'utilisation du fichier journal, se traduit par les informations que nous modélisons comme suit :

Définition 3.5
$LogF = \{u_1, ..., u_p\}$, où u_i est un n-uplet.

Le recours à l'information passée est une expression formelle des données collectées lors des réponses à des requêtes passées, notamment, les termes des requêtes passées, les pairs qui ont été sollicités pour répondre à ces besoins, ainsi que les documents qui leur correspondaient. Ces données peuvent être stockées dans un fichier ou une base de données, etc. Cependant, le problème d'une telle représentation plate est qu'elle ne tient compte, ni de l'aspect de répétition de termes demandés par l'utilisateur, ni du nombre de fois qu'un pair a été sollicité pour répondre à un besoin. Ainsi, il n' y a aucune considération des corrélations implicites existantes entre ses différents besoins. Cette construction s'est reposée sur l'utilisation d'une technique de data mining permettant de construire un ensemble d'intérêts des utilisateurs, dits profils.
Un modèle de profils utilisateur décrit formellement la base de connaissances. A cet effet, nous avons recouru à construire une base de connaissances (KB) qui tient compte des corrélations sémantiques entre les besoins des utilisateurs, les ressources demandées (les documents) et les pairs positifs.
Nous tenons à décrire ultérieurement dans la Section 3.3.3, la technique utilisée pour la construction de la base des connaissances et plus précisément la technique qui nous a permis d'extraire des corrélations à partir du fichier journal.

3.3.3 Représentation du profil

L'utilisation des profils pour personnaliser les applications dépend de la façon dont ils sont représentés. Généralement les profils sont représentés sous forme d'un ensemble de mots-clés pondérés, de réseaux sémantiques ou de concepts pondérés [Wan 2010]. Différents modèles de représentation de profils existent. Nous citons, le modèle vectoriel, graphique, sémantique, etc [Bouzeghoub 2005].

Dans notre contexte, nous nous sommes plutôt intéressés, comme présenté dans la section précédente, à une représentation qui relève des correlations entre les informations du log. C'est ainsi que notre attention s'est tournée, à un modèle de profils qui représente des associations à partir des données.
L'extraction des associations à partir des données est une opération difficile qui demande une attention spéciale aux phénomènes qui peuvent surgir dans une base informative. Une revue de la littérature révèle que les techniques de fouille de données sont souvent utilisées pour extraire et analyser un ensemble de comportements à partir des données
[Lefebure 2001].

Choix d'une technique de représentation

La recherche d'un ensemble de corrélations dans un ensemble de données a été l'un des objectifs cibles de plusieurs techniques de la fouille de données [Lefebure 2001]. En effet, la fouille de données possède l'initiative de découvrir par elle-même les as-

sociations entre les données sans que l'utilisateur ait à lui dire de rechercher dans telle ou telle direction. Il s'agit d'un concept pluridisciplinaire qui repose sur un ensemble de techniques héritées de l'analyse de données et de l'intelligence artificielle [Lefebure 2001].
Les techniques basées sur l'analyse de données permettent de renforcer les méthodes statistiques fondées sur un ensemble d'hypothèses non toujours vérifiées [Lefebure 2001]. La technique la plus connue pour des fins de découvertes d'associations est sans doute l'extraction des règles associatives introduite par Agrawal et al [Agrawal 1993] et qui trouve son application immédiate dans l'analyse du panier de la ménagère. Cette analyse tend à identifier les corrélations qui peuvent exister entre les différents produits et/ou services.
Mathématiquement, une règle associative est une implication de la forme "si X alors Y" ou encore "$X \Rightarrow Y$". X et Y sont des ensembles d'attributs qui dénotent respectivement la prémisse et la conclusion de la règle associative. Afin d'évoquer la force de la relation extraite et sa signification, deux indicateurs issus des statistiques sont utilisés : la confiance qui mesure la force de la règle associative et le support qui mesure sa fréquence [Agrawal 1993]. Ainsi, le vrai problème d'extraction des règles associatives devient un problème de recherche des règles intéressantes. Ces dernières sont spécifiées par des niveaux de confiance et de support supérieurs ou égaux à des valeurs seuils pour ses indicateurs prédéfinis par l'utilisateur. De vue, la logique d'extraction des règles associatives semble simple à comprendre et à interpréter par un utilisateur métier. Le revers de la médaille, est que l'application de cette technique, dans un contexte large, comme dans notre cas, est assez complexe à prendre en main. En effet, un nombre surabondant de règles, souvent redondantes, sera généré au niveau de chaque pair initiateur. Ceci rend difficile, voire impossible, l'exploitation et l'interprétation dans cet espace de règles.
Des techniques d'élagages de règles ont été proposées à cette issue, entre autres, les techniques de la classification basées sur un ensemble de critères imposés par l'utilisateur. La frénésie que montraient les travaux d'élagage cachent derrière, une possibilité de perte d'information qui pourraient être intéressantes. À la lumière des problèmes recensés, quand on utilise les règles associatives, plusieurs travaux se sont intéressés à l'utilisation de l'Analyse Formelle de Concepts (AFC) [Ben Yahia 2009]. Cette technique s'impose, par rapport à la première, puisqu'elle contribue à une réduction du nombre de corrélations générées grâce à une agrégation des variables qui présentent les mêmes intérêts ensemble, sans conduire à une perte d'informations [Ben Yahia 2009]. L'AFC a été utilisée dans différents domaines, spécialement dans l'analyse des données, le génie logiciel, l'apprentissage machine et plus récemment dans la recherche d'information [Poelmans 2010]. Pour ces raisons, nous avons choisi d'extraire des corrélations, à partir du fichier journal, sur la base de l'Analyse Formelle de Concepts [Ganter 1997]. Un bref aperçu sur l'AFC est donné par la suite :

Analyse Formelle de Concepts (AFC)

L'Analyse Formelle de Concepts est basée sur une mathématisation de la nature du concept et d'hiérarchie de concepts. Elle exploite des méthodes mathématiques pour une analyse conceptuelle des données et un traitement des connaissances [Ganter 1997].

Définition 3.6
Un ***contexte formel*** $\mathcal{K} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$ est constitué de deux ensembles $\mathcal{O}$ et $\mathcal{A}$, et d'une relation $\mathcal{R}$ entre $\mathcal{O}$ et $\mathcal{A}$. Les éléments de $\mathcal{O}$ sont considérés comme **objets** alors que ceux de $\mathcal{A}$ sont dits **attributs**. Afin d'exprimer qu'un objet o est lié à un attribut a, on écrit $o\mathcal{R}a$. Ceci est lu : "l'objet o possède l'attribut a" [Ganter 1997].

Définition 3.7
Étant donné un ensemble d'objets $O \subseteq \mathcal{O}$ et un ensemble d'attributs $A \subseteq \mathcal{A}$, on définit l'ensemble d'attributs partagés par les objets dans O [Ganter 1997] :

$$O^{\blacktriangleright} = \{a \ \in A \mid o\mathcal{R}a \ \forall \ o \ \in O\} \tag{3.1}$$

L'ensemble des objets ayant tous les attributs dans A :

$$A^{\blacktriangleleft} = \{o \ \in O \mid o\mathcal{R}a \ \forall \ a \ \in A\} \tag{3.2}$$

Le couple des opérateurs, ($\blacktriangleright$,$\blacktriangleleft$), est une **connexion de Galois.**

Définition 3.8
Un ***Concept*** formel du contexte $(\mathcal{O}, \mathcal{A}, \mathcal{R})$ [Ganter 1997]

$$O \ \subseteq \ \mathcal{O}, A \ \subseteq \ \mathcal{A}, \ O^{\blacktriangleright} = A \ et \ A^{\blacktriangleleft} = O \tag{3.3}$$

O est dit **L'extension** et A est dit **L'intension** du concept (O, A).

Il s'avère évident à cette étape que la construction d'un profil utilisateur va être basée sur l'Analyse Formelle de Concepts (AFC) qui vise à identifier des groupes d'objets ayant des attributs communs. Cependant dans notre cas, un profil comporte comme présente sa définition dans la section 3.3.1 trois types d'information au lieu de deux et présente deux types de corrélations (de relations) qui relient les trois types d'information. Par ailleurs, la représentation du modèle de profils doit passer par une étape d'adaptation de AFC, en vue de générer les corrélations nécessaires.

3.3.3.1 Modélisation des profils à base de AFC

Dans cette section, nous présentons comment modéliser un profil en se basant sur l'Analyse Formelle des Concepts. Nous présentons ainsi notre contexte formel qui est le résultat d'une ou de plusieurs projections sur le fichier journal. Pour cette raison, nous commençons par proposer un opérateur de projection, noté π_{xy}, qui sera suivi d'un ensemble de définitions nécessaires à notre modélisation de profils. Avant de procéder à notre modélisation, nous proposons un ensemble de notations regroupées dans la Table 3.1 :

Symbole	Signification
$\mathcal{X}$, $\mathcal{Y}$	Deux ensembles d'objets
X, Y	Deux sous-ensembles de respectivement : $\mathcal{X}$, $\mathcal{Y}$
$\mathcal{K}$	Un contexte formel
(x,y)	Un couple de $\mathcal{K}$
$(x_i,\ y_j)$	n'importe quels objets de X et de Y
$\mathcal{R}$	une relation binaire
$c_i(X_i, Y_i)$	Un concept
$\mathfrak{C}$	Une couverture de concepts

TABLE 3.1 – Table des notations - Modèle de Profils

DÉFINITION 3.9
$\pi_{\mathcal{XY}}(LogF)$: opérateur de projection qui extrait les propriétés $\mathcal{X}$ et $\mathcal{Y}$ à partir du fichier log $LogF$.

DÉFINITION 3.10
Un context $\mathcal{K} = (\pi_{\mathcal{XY}}(LogF),\ \mathcal{R})$, est un couple où $\pi_{\mathcal{XY}}(LogF)$ est l'opérateur de projection et $\mathcal{R}$ est une relation entre $\mathcal{X}$ et $\mathcal{Y}$.

Un contexte peut être décrit comme une matrice binaire $|\mathcal{X}| \times |\mathcal{Y}|$, où les objets de $\mathcal{X}$ forment les titres de lignes et les objets de $\mathcal{Y}$ forment les titres de colonnes. Soit $mat(\mathcal{K})$, la représentation matricielle du contexte $\mathcal{K}$, nous devons spécifier complètement les entrées de cette matrice telles que :

$$mat(\mathcal{K})_{ij} = \begin{cases} 1 & si \quad x_i \mathcal{R} y_j \\ \\ 0 & sinon \end{cases} \tag{3.4}$$

DÉFINITION 3.11
Nous définissons un concept comme étant un couple formé de deux ensembles $(\{x\}, \{y\})$, $\forall\ \ x \in \mathcal{X}$, $\forall\ \ y \in \mathcal{Y}$, $x\mathcal{R}y$ selon l'équation 3.3

DÉFINITION 3.12
Une couverture du contexte $\mathcal{K} = (\mathcal{X}, \mathcal{Y}, \mathcal{R})$ est définie comme étant un ensemble de concepts formels : $\mathfrak{C} = \{c_i, i \in [1, n]|\ \forall (x, y)\ de \mathcal{X} \times \mathcal{Y} :\ (x, y) \in c_i\}$.

Afin d'avoir un tri-concept (concept augmenté) qui représente une corrélation entre différents objets, nous définissons l'opérateur de jointure, $\bowtie$.

DÉFINITION 3.13
Opérateur de jointure $\bowtie$: Soient deux concepts, c_1 et c_2, l'application de l'opérateur de jointure $\bowtie$ se fait comme suit :
$\bowtie\ (c_1, c_2) = (extension(c_1), intension(c_1), intension(c_2))$ crée un concept "augmenté" par l'addition (l'extension) de l'intension d'un concept à un autre. Les concepts doivent être compatibles i.e, partagent les mêmes objets.

Définition 3.14
On définit un nouvel opérateur ▶▶ tel que
$Y1^{\blacktriangleright\blacktriangleright^{\subseteq}}\ Y12$ avec :

$$Y1^{\blacktriangleright\blacktriangleright} = \{(y_1, y_2) \in Y1 \times Y2 \mid x_1\ \mathcal{R}(y_1, y_2),\ \forall\ x_1 \in X1\ et\ x_1R_1y_1\ et\ x_2R_2y_2\} \tag{3.5}$$

Lemme 3.1
Un concept augmenté est un n-uplet concept ou concept multidimensionnel (triplet, quadruplet concepts,...).

Preuve 3.1
Soient deux contextes $\mathcal{K}1 = (\mathcal{X}1, \mathcal{Y}1, \mathcal{R}1)$ et $\mathcal{K}2 = (\mathcal{X}2, \mathcal{Y}2, \mathcal{R}2)$ tels que $\mathcal{X}1 = \mathcal{X}2$. A partir de $\mathcal{K}1$ et $\mathcal{K}2$, nous générons respectivement, les couvertures de concepts associées $\mathfrak{C}_1$ et $\mathfrak{C}_2$ tel que, pour un ensemble $X1 \subseteq \mathcal{X}1$ et $Y1 \subseteq \mathcal{Y}1$

$$X1^{\blacktriangleright} = \{y1 \in Y1 \mid x1\mathcal{R}1y1, \quad \forall\ x1 \in X1\} \tag{3.6}$$

L'ensemble des objets qui ont tous les attributs dans $Y1$:

$$Y1^{\blacktriangleleft} = \{x1 \in X1 \mid x1\mathcal{R}1y1, \quad \forall\ y1 \in Y1\} \tag{3.7}$$

et aussi, pour tout ensemble $X2 \subseteq \mathcal{X}1$ ($\mathcal{X}1 = \mathcal{X}2$)

$$X2^{\blacktriangleright} = \{y2 \in Y2 \mid y2\mathcal{R}2y2, \quad \forall\ x2 \in X2\} \tag{3.8}$$

Pour un ensemble $Y2 \subseteq \mathcal{Y}2$, on aura :

$$Y2^{\blacktriangleleft} = \{x2 \in X2 \mid x2\mathcal{R}2y2, \quad \forall\ y2 \in Y2\} \tag{3.9}$$

Mais les deux contextes partagent le même ensemble d'objets $X1$, donc $Y2^{\blacktriangleleft}$ devient

$$Y2^{\blacktriangleleft} = \{x1 \in X1 \mid x1\mathcal{R}2y2, \quad \forall\ y2 \in Y2\} \tag{3.10}$$

Nous pouvons générer une relation ternaire $\mathcal{R} \subseteq \mathcal{X}1 \times \mathcal{Y}1 \times \mathcal{Y}2$, $Y1^{\blacktriangleright\blacktriangleright} \subseteq Y1 \times Y2$ où

$$X^{\blacktriangleright\blacktriangleright}1 = \{(y1, y2) \in Y1 \times Y2 \mid x1\ \mathcal{R}\ (y1, y2),\ \forall\ x1 \in Y1\} \tag{3.11}$$

$$Y1^{\blacktriangleleft} = \{x1 \in X1 \mid x1\ \mathcal{R}\ (y1, y2),\ \forall\ (y1, y2) \in Y1 \times Y2\} \tag{3.12}$$

D'où $(X1, Y1, Y2)$ est le concept augmenté correspondant aux concepts (X1,Y1) et (X1,Y2).

Définition 3.15
Couverture augmentée : A partir de deux contextes $\mathcal{K}1(\mathcal{X}1, \mathcal{Y}1, \mathcal{R}1)$ et $\mathcal{K}2(\mathcal{X}2, \mathcal{Y}2, \mathcal{R}2)$, nous générons respectivement les couvertures associées $\mathfrak{C}_1$ et $\mathfrak{C}_2$.
A partir de ces couvertures, nous définissons la couverture augmentée comme suit :
$\mathfrak{C}_{sim}(\mathfrak{C}_1, \mathfrak{C}_2) = \{\bowtie (c_x, c_y)\}$
tel que $\forall c_1 \in \mathfrak{C}_1$, $\forall c_2 \in \mathfrak{C}_2$, $Sim(c_1, c_2) = Max(Sim(c_i, c_k))$
$c_k \in \mathcal{C}_2$

La fonction Sim est définie comme suit :

Définition 3.16
Sim : La fonction de similarité $Sim : X \times Y \longmapsto [0, 1]$ fournit la similarité entre un objet $x \in X$ et un concept $y \in Y$ en se basant sur les thèmes communs. Une valeur de similarité est comprise entre 0 et 1 telle que si la valeur est égale à 0, x et y ne sont pas similaires, et si cette valeur est égale à 1, ils sont similaires.

Exemple 2: L'exemple de la Figure 3.2 représente la génération d'une couverture augmentée à partir de deux contextes $\mathcal{K}1$ et $\mathcal{K}2$ qui représentent respectivement les relations entre les termes de requêtes posées par l'utilisateur et leurs pairs contributeurs d'une part, et les termes des requêtes et les documents positifs, d'autre part. La Figure 3.2 illustre les différentes étapes nécessaires pour la construction d'une couverture augmentée à partir d'un Log File. Ainsi, nous considérons dans la première phase (1) un log File, constitué des traces de résolution des requêtes R1, R2 et R3. Ces traces considèrent, pour chaque requête, ses différents termes (Ti), les pairs qui ont contribué à la résoudre (Pi) et les documents qui les ont répondu à ces requêtes (Di). Les phases 2.2 et 2.3 permettent de construire les contextes formels en faisant les projections sur le fichier journal, et ce en considérant respectivement, la relation entre les termes des requêtes et les pairs contributeurs dans 2.2 et celle entre les termes et les documents dans 2.3.

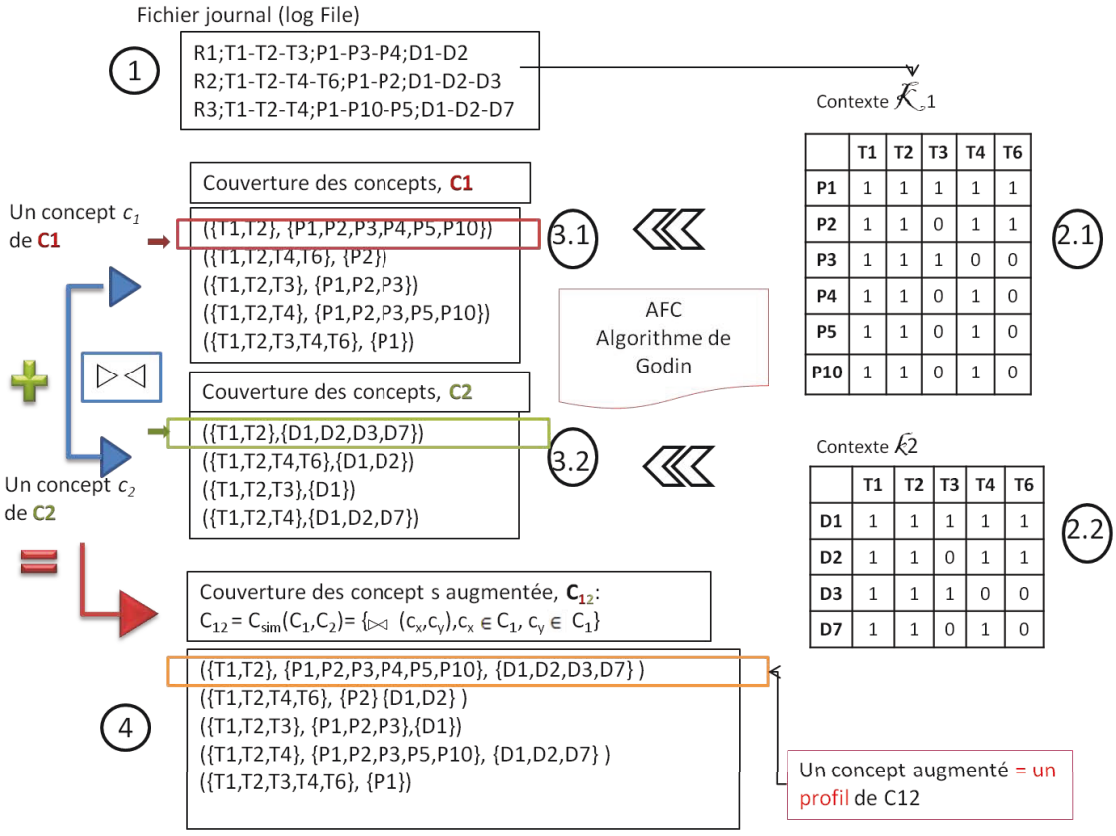
À partir des contextes $\mathcal{K}1$ et $\mathcal{K}2$, nous génèrons, comme présenté dans les phases 3.2 et 3.2, respectivement, les couvertures associées $\mathfrak{C}_1$ et $\mathfrak{C}_2$ en se basant sur l'AFC. Afin de générer **les profils** qui sont des concepts augmentés, nous appliquons dans la phase (4), la définition 3.15, ainsi que la similarité de Salton, en se basant sur les extensions des concepts générés [Salton 1989].
Par exemple, le concept le plus proche dans la couverture $\mathfrak{C}_2$ au premier concept de la couverture $\mathfrak{C}_1$ est le premier. Le concept augmenté correspondant est alors : $(\{T1, T2\}, \{P1, P2, P3, P4, P5, P10\}, \{D1, D2, D3, D7\})$.

Ainsi, nous définissons un profil utilisateur comme un concept augmenté. Une fois la base des connaissances (qui est un ensemble de profils) est construite, elle sera intégrée dans le processus de RI. Un ensemble d'opérateurs est nécessaire pour la manipuler.

Définition 3.17
$\sigma_{(c)}(\mathcal{KB})$: est l'opérateur de sélection utilisé en vue d'obtenir un sous ensemble de concepts augmentés (de profils) à partir de la couverture augmentée (la base de connaissance KB) selon la condition de sélection c.

Cet opérateur permet de sélectionner des profils similaires (i.e, concepts augmentés) à une requête donnée q.

Définition 3.18
$\pi_{(X)}(C(\mathcal{T}, \mathcal{P}, \mathcal{D}))$: est un opérateur de projection permettant d'extraire l'ensemble X à partir de n'importe quel concept augmenté $\mathcal{C}$. X peut être un sous-ensemble



	T1	T2	T3	T4	T6
P1	1	1	1	1	1
P2	1	1	0	1	1
P3	1	1	1	0	0
P4	1	1	0	1	0
P5	1	1	0	1	0
P10	1	1	0	1	0

	T1	T2	T3	T4	T6
D1	1	1	1	1	1
D2	1	1	0	1	1
D3	1	1	1	0	0
D7	1	1	0	1	0

FIGURE 3.2 – Exemple de génération d'une couverture augmentée

de $\mathcal{T}$, $\mathcal{P}$ ou $\mathcal{D}$.

Nous avons défini les principaux opérateurs et une fonction de similarité, qui seront utilisés pour sélectionner un ensemble de concepts augmentés pertinents (i.e, profils) par rapport à une requête $q \in Q$. Nous définissons la base de connaissances $\mathcal{KB}$ comme suit :

Proposition 1:

$\mathcal{KB} = \mathfrak{C}_{sim}(\mathfrak{C}_{sim}(\mathfrak{C}_1, \mathfrak{C}_2), \mathfrak{C}_3)$

PREUVE 3.2

La similarité entre un concept $c_i \in \mathfrak{C}_1$ et un concept $c_j \in \mathfrak{C}_2$ est calculée comme suit :

$$sim(c_i, c_j) = \frac{|\pi_{(X)}(c_i) \cap \pi_{(X)}(c_j)|}{|\pi_{(X)}(c_i) \cup \pi_{(X)}(c_j)|} \tag{3.13}$$

où : $\pi_{(X)}(c)$ représente la partie X du concept augmenté c.

$$\mathcal{C}_{sim}(\mathcal{C}_1,\ \mathcal{C}_2) = \{\bowtie (c_x, c_y) \mid \forall\ c_x \in \mathcal{C}_1, \forall c_y \in \mathcal{C}_2,\ Sim(c_x, c_y) = Max(Sim(c_i, c_k))\} \quad c_k \in \mathcal{C}_2$$

$$\equiv \mathcal{C}_{1,2} = \{c(T, P, D)\ T \subseteq \mathcal{T},\ P \subseteq \mathcal{P},\ D \subseteq \mathcal{D}\}$$

$$\mathcal{C}_{sim}(\mathcal{C}_{1,2},\, \mathcal{C}_3) = \{\bowtie (c_x, c_y) \mid \forall c_x \in \mathcal{C}_{1,2},\ \forall c_y \in \mathcal{C}_3,\ Sim(c_x, c_y) = Max(Sim(c_i, c_k)\} \quad c_k \in \mathcal{C}_3$$

D'où $C(T, P, D)$ est un concept augmenté et il peut être dénoté comme un profil $Pr(T, P, D)$

Dans ce qui suit, nous présentons les différentes étapes de construction d'une base des connaissances dans un système de RIDLE. La base des connaissances sera une instance des profils des utilisateurs tel que présenté ci-dessus. Nous visons à générer un ensemble de profils qui représentent la relation sémantique entre *les requêtes*, leurs *pairs contributeurs* et *les documents positifs*.

Tel que présenté dans notre modèle, un profil utilisateur est un triplet $Pr(T, P, D)$, avec une corrélation exprimée entre un sous-ensemble de termes de requêtes (i.e T) et les pairs associés (i.e P). Le même processus est fait afin d'avoir l'ensemble D des documents correspondants. La question qui se pose maintenant, est comment instancier ce modèle dans notre cas pour l'intégrer dans le processus d'agrégation ?

3.3.4 Intégration de profils utilisateurs dans un processus d'agrégation de résultats à large échelle

L'objectif de l'étape d'intégration est de présenter les démarches nécessaires pour instancier le modèle afin de construire une base des connaissances dans un Système Distribué à Large Échelle (SDLE).
Ainsi, notre objectif est de générer un ensemble de profils qui représentent la relation sémantique entre les termes, leurs réponses (les documents) et les pairs contributeurs. Un profil utilisateur est : $Pr\ (T, P, D)\ \in \mathcal{KB}$.

Pour appliquer l'approche AFC, nous avons utilisé deux contextes :
$\mathcal{K}1 = (\pi_{TP}(LogF), \mathcal{R}_1)$ et $\mathcal{K}2 = (\pi_{TD}(LogF), \mathcal{R}_2)$ où :

- T sont les termes correspondant aux requêtes soumises.
- P regroupe les pairs contributeurs à la résolution de ces requêtes.
- D constitue l'ensemble des services sélectionnés (documents dans notre cas).
- R_1 et R_2 sont deux relations binaires.

Pour chaque contexte, nous appliquons un algorithme de génération de concepts formels en vue d'avoir un ensemble de concepts correspondant aux deux couvertures, dénommées respectivement : $\mathfrak{C}_1$ et $\mathfrak{C}_2$. À partir des contextes $\mathcal{K}1$ et $\mathcal{K}2$. Nous avons généré une couverture augmentée $\mathfrak{C}_{1,2}$ qui représente les relations entre les Requêtes, des documents et des pairs positifs. Cette couverture augmentée représente notre base de connaissances. En outre, nous précisons qu'il n'y aura pas de connaissance globale partagée entre tous les pairs.

Dans cette section nous appliquons le modèle de profils au problème d'agrégation de résultats dans un contexte de RIP2P où :

- $\mathcal{P}$: est l'ensemble de pairs constituant le système P2P.

– $\mathcal{Q}$: est l'ensemble de requêtes envoyées par un pair initiateur $p \in \mathcal{P}$.

L'étape d'intégration de profils servira à l'action effective d'agrégation des résultats. Elle consiste à développer une solution d'agrégation à base de profils passant à l'échelle. Elle vise à utiliser une méthode de score hybride régie par un algorithme d'agrégation à base de profils (PBA) afin de classer les résultats de manière globale les uns par rapport aux autres. Ainsi, nous fusionnons et classons les résultats de plusieurs pairs contributeurs à la réponse en utilisant simplement les rangs des documents et les profils utilisateurs collectés localement sur chaque pair initiateur de manière à éviter toute surcharge.

Le processus d'intégration est détaillé dans la section suivante.

3.4 Agrégation hybride à base de profils

À cette étape, il est essentiel de rappeler notre besoin de trouver une méthode de score qui permet d'interclasser les résultats d'une requête soumise dans une liste finale. Cette fonction ne souhaite utiliser ni les scores classiques basés sur des statistiques locales, en raison des coûts de communication élevés qu'elles encourent lorsqu'elles sont échangées, ni les rangs seuls vu la faible efficacité que fournissent les modèles d'agrégation à base de rangs, en particulier lorsque les collections sont hétérogènes. L'idée principale de la proposition est d'améliorer l'efficacité des modèles d'agrégation à base de rangs avec des informations d'intérêt de l'utilisateur capturées sur les pairs initiateurs, sans imposer des coûts supplémentaires. Pour commencer, nous précisons, dans ce qui suit, le principe de notre méthode de score.

3.4.1 Score d'agrégation hybride

Nous proposons, dans le Tableau 3.2 les notations que nous adoptons respectivement, dans la méthode de score et l'algorithme proposé dans la prochaine section.

Nous présentons, dans cette section, notre méthode de calcul de score distribué. Ce score propose d'hybrider deux types d'informations : une information recueillie à partir du rang du document dans la liste de résultats retournée par un pair contributeur et deux informations calculées à partir de la base des connaissances sur chaque pair initiateur. Ainsi, trois paramètres sont mis en exergue pour calculer ce score.

1. **Valeur positionnelle du document** (PV) : Ce paramètre reflète l'importance récente du document dans son pair local. Elle permet de déduire l'importance d'un document de point de vue de sa classification (son rang) dans la liste de résultats retournée de la part de son pair contributeur. La valeur PV d'un document d_k retourné par un pair p_j est basée sur la formule suivante :

$$PV(d_{kj}) = \frac{|D_{qj}| - rang(d_{kj}) + 1}{|D_{qj}|} \tag{3.14}$$

Symbole	Signification
$\mathcal{KB}$	Base de connaissances (Knowledge Base)
q	requête utilisateur
$\mathcal{P}_q$	Liste de tous les pairs contributeurs à une requête q
p_j	Un pair de P_q
D_{qj}	Liste de tous les résultats (documents) de q et provenant du pair p_j
d_k	Un document de D_{qj}
$rang(d_{kj})$	le rang du document d_k dans D_{qj}
$\mathcal{PR}_q$	Liste de tous les profils de KB contenant au moins un terme de la requête q
pr_i	Un profil de PR_q
$\mathfrak{S}$	Seuil de décision de similarité requête-profil
$\mathcal{PR}_{pj}$	Tous les Profils de $\mathcal{PR}_q$ contenant le pair p_j
$\mathcal{PR}_{dk}$	Tous les Profils de $\mathcal{PR}_q$ contenant le document d_k
PV	Valeur Positionnelle d'un document
IPP	Importance Passée d'un pair
IPD	Importance Passée d'un document
GS	Score Global d'un document
LF	Liste finale agrégée

TABLE 3.2 – Table des notations - PBA

où $|D_qj|$, $rang(d_{kj})$ sont respectivement la taille de la liste de résultats locale relative à un pair contributeur p_j, et le rang du document dans la liste locale. Il convient de mentionner que le paramètre rang est déduit implicitement (liste ordonnée) à partir de la liste de résultats retournée par un pair contributeur p_j.

2. Scores à base de profils :

- 2.1 **Importance Passée du Pair** IPP : ce paramètre représente l'importance du pair par rapport à une requête, en puisant dans la base des connaissances. Il représente le degré de confiance que nous pouvons attribuer à ce pair puisqu'il a été déjà sollicité en réponse à une requête passée similaire à celle que l'utilisateur a soumis. Le calcul de la valeur de IPP, d'un pair contributeur p_j suit le fil directeur suivant :
  - La recherche des profils similaires à la requête.
  - La réduction du nombre de profils retournés sur la base de la méthode d'élagage choisie (en fonction d'un seuil de similarité).
  - La sélection à partir de l'ensemble élagué des profils similaires à la requête q, PR_q, de ceux qui ont sollicité le pair contributeur p_j dans la réponse à des requêtes similaires passées.

  La sélection des profils similaires à une requête q est effectuée via l'opérateur de sélection de profils σ, à partir de la base des connaissances $\mathcal{KB}$ en fonction de leurs similarités à la requête q.

  La condition de sélection proposée vise à réduire l'ensemble des profils si-

milaires à la requête en gardant ceux dont la similarité est supérieure à un seuil prédéfini $\mathfrak{S}$. L'avantage de cet élagage est double : tout d'abord, il permet de réduire l'espace de recherche pour des profils potentiels similaires à la requête, et améliore la qualité des profils participant dans le processus d'agrégation et donc il permet d'améliorer la qualité du processus d'agrégation lui-même.
La recherche de profils sélectionnés pour un pair p_j, contribuant à la réponse de la requête q, (PR_{p_j}) repose sur l'opérateur de sélection σ ayant comme objectif de sélectionner à partir des profils similaires élagués, la sous-liste contenant un pair contributeur.

– 2.2 **Importance passée du document** IPD : Ce poids définit le taux de présence d'un document résultat dans l'historique des requêtes similaires passées. Plus un document est présent, plus il a été téléchargé (ou consulté) et donc plus il est jugé intéressant pour l'utilisateur.

L'ensemble de ces paramètres nous a conduit à définir un nouveau score global $GS(d)$ pour un document d retourné à partir d'un ensemble de listes de résultats de plusieurs pairs contributeurs $\mathcal{P}_q$. Le nouveau score tient compte de l'importance du document dans la réponse locale au niveau de chaque pair , i.e., son rang est spécifié par le paramètre PV, ainsi que le taux de satisfaction des utilisateurs, calculé à partir de l'historique des requêtes.
Ainsi, la méthode de score utilise trois arguments qui sont respectivement PV, IPP et IPD et retourne un score global pour chaque document. Ce score est formulé comme suit :

$$GS(d) = \frac{\Sigma_{i=1}^{|P_{qd})|} IPP_i(\alpha * PV_i(d) + \beta * IPD(d))}{|\mathcal{P}_{qd}|} \tag{3.15}$$

où :
– $|P_{qd}|$: la cardinalité des pairs contributeurs à la requête q par le document d
– α et β : sont deux paramètres de lissage, $\alpha \leq 1$, $\beta \leq 1$ et $\alpha + \beta = 1$;
Le score global, $GS(d)$ est normalisé de sorte que $0 \leq GS(d) \leq 1$; où 1 est la meilleure valeur qu'un document peut avoir comme score global. Les étapes nécessaires au calcul de $GS(d)$ seront explicitées à travers l'Algorithme 3.4.2.

3.4.2 Algorithme d'agrégation à base de profils

L'algorithme d'agrégation à base de profils (PBA) est utilisé dans le module de résolution globale des requêtes à partir de chaque pair initiateur. Il est chargé de fournir une liste de classement finale triée sur la base du score global, déduit à partir des scores des profils, puis, filtré afin d'éliminer les doublons.

L'algorithme d'agrégation correspondant est présenté ci-dessous. Il intègre une méthode qui calcule les scores des documents et de ses différents paramètres.

[H] **Algorithm :** PBA 5pt

$\mathcal{KB}$ [*Base des connaissances du pair initiateur*]
q [*une requête utilisateur*]
PR_q [*les profils de $\mathcal{KB}$ similaires à q*]
P_q [*l'ensemble des pairs contributeurs à q*]
p_j [*un pair contributeur de P_q*]
D_{qj} [*l'ensemble des documents pertinents à q et provenant de p_j*]
$\mathfrak{S}$ [*un seuil de similarité*]

$LF \quad PR_q = \sigma_{Sim(q,pr_i)>\mathfrak{S}}(\mathcal{KB}) \backslash \ pr_i \in \mathcal{KB}$

$p_j \in P_q \quad PR_{pj} = \sigma_{(p_j)}(PR_q)$
$IPP = setPeerPastImportance\ (PR_{pj}, p_j, q)$
$d_k \in D_{qj} \quad PV = setDocPosValue\ (d_k)$
$PR_{dk} = \sigma_{(d_k)}(PR_q)$
$IPD = setDocPastImportance\ (PR_{dk}, d_k, q)$
$GS = ComputeGlobalScore\ (PV, IPP, IPD, d_k)$
$LF.add\ (d_k, Gs) \quad sort\_Filter(LF)$
retourner(LF) Algorithme "Profile-Based Aggregation (PBA)"

Soit P_q, l'ensemble de tous les pairs contributeurs à la résolution d'une requête q, lancée à partir d'un pair initiateur ; D_{qj}, une liste ordonnée de documents répondant à q et retournée par un pair contributeur p_j ; $d(k)$, un document retourné par p_j et $\mathcal{KB}$, la base des connaissances locale du pair initiateur.

Formellement l'algorithme PBA, utilisé sur chaque pair initiateur, se base sur un ensemble de méthodes pour : (i) accéder à la base des connaissances locale en vue de sélectionner les profils similaires à la requête q. À la Ligne 3 de l'Algorithme 3.4.2, la fonction *sim* est utilisée pour sélectionner les profils dont la similarité à q dépasse un seuil $\mathfrak{S}$. Pour tout pair contributeur p_j, PBA permet de sélectionner, parmi les profils similaires à la requête q, respectivement, ceux qui contiennent les pairs contributeurs (voir Ligne 5 de l'algorithme), et les documents réponses (Ligne 9 du même algorithme) ; (ii) calculer l'importance IPP d'un pair contributeur p_j (la méthode $setPeerPastImportance\ (PR_{pj}, P_j, q_i)$ de l'Algorithme 3.4.2, ligne 6), et l'importance IPD d'un document d_k qui résout la requête q (la méthode $setDocPastImportance\ (PR_{dk}, d_k, q_i)$ de l'Algorithme 3.4.2, Ligne 10).
(iii) évaluer ce que nous appelons, l'importance récente PV, d'un document d_k par rapport à la requête q, en fonction de son rang dans son pair local p_j (voir la méthode $setDocPosValue(d_k)$, la Ligne 8 de l'Algorithme 3.4.2).
Un score global est ainsi calculé en fonction des facteurs mentionnés ci-dessus (voir Ligne 11 de l'Algorithme 3.4.2).

La méthode $sort\_Filter$, telle que présentée à la Ligne 13 de l'algorithme PBA, permet de trier et de filter la liste finale agrégée selon la valeur des scores. Enfin, la liste finale est restituée (Ligne 14 du mêma algorithme) à l'utilisateur, par le

pair initiateur. Les étapes indiquées dans l'Algorithme 3.4.2 sont effectuées dès la récupération d'un document à partir d'un pair contributeur.

Dans ce qui suit, nous présentons un calcul de la complexité algorithmique, et plus précisément, temporelle de l'algorithme d'agrégation.

3.4.3 Complexité algorithmique

Nous étudions ici la complexité temporelle de PBA dans le pire des cas. Pour ce faire, nous commençons par identifier les paramètres suivants :

- $|\mathcal{P}_q|$: nombre de pairs contributeurs à une requête q, autrement dit c'est le nombre de listes de résultats retournées. Ce nombre est, évidemment, limité par la technique de routage utilisée, ce qui le limite à quelques dizaines de nœuds.
- $|\mathcal{PR}_q|$: La cardinalité du sous-ensemble de profils similaires à une requête donnée ;
- $|\mathcal{BK}|$: La cardinalité de la base de connaissances.
- $|D_{qmax}|$: La cardinalité maximale d'une liste de documents retournée.
- $|\mathcal{D}_{qp}|$: le nombre des documents positifs à la requête q retournés par p.

La complexité dans le pire des cas, correspond aux coûts de :

1. La sélection des profils similaires à la requête q : $\mathcal{O}(|\mathcal{BK}|)$. Il correspond au coût d'une sélection (voir Ligne 3 de l'Algorithme 3.4.2).
2. Le calcul de l'Importance Passée d'un Pair contributeur IPP : $\mathcal{O}(|\mathcal{P}_q| \times |\mathcal{PR}_q|)$: ce coût comprend la sélection des profils similaires à la requête et contenant un pair (voir Lignes 5 et 6).
3. Le calcul de la valeur positionnelle (Ligne 8 de l'Algorithme 3.4.2) se fait en $\mathcal{O}(1)$.
4. Le calcul de l'Importance Passée d'un document : est effectué avec une complexité de $\mathcal{O}(|\mathcal{D}_q max| \times |\mathcal{PR}_q|)$ (Ligne 9 de l'Algorithme 3.4.2). Ce calcul comprend la condition de sélection des profils à la Ligne 8.
5. L'algorithme *ComputeGlobalScore* est effectué avec une complexité $\mathcal{O}(1)$ qui est négligeable par rapport aux autres coûts.
6. L'algorithme Sort-Filter, correspond à $\mathcal{O}(|\mathcal{D}_{qp}| \log(|\mathcal{D}_{qp}|))$

Ainsi, le coût global (CG) est :
$CG = \mathcal{O}(\mathcal{BK}) + \mathcal{O}(|\mathcal{P}_q| \times |\mathcal{PR}_q| \times |D_q max|) + \mathcal{O}(|\mathcal{D}_{qp}| \log(|\mathcal{D}_{qp}|))$

Nous avons, ainsi, une complexité polynômiale pour l'algorithme PBA relativement à la taille de la réponse globale.

3.4.4 Exemple illustratif

L'exemple de la Figure 3.3 illustre le principe de calcul des scores selon l'approche PBA (Algorithme 3.4.2). Soit q une requête de termes $\{T1, T2\}$, envoyée

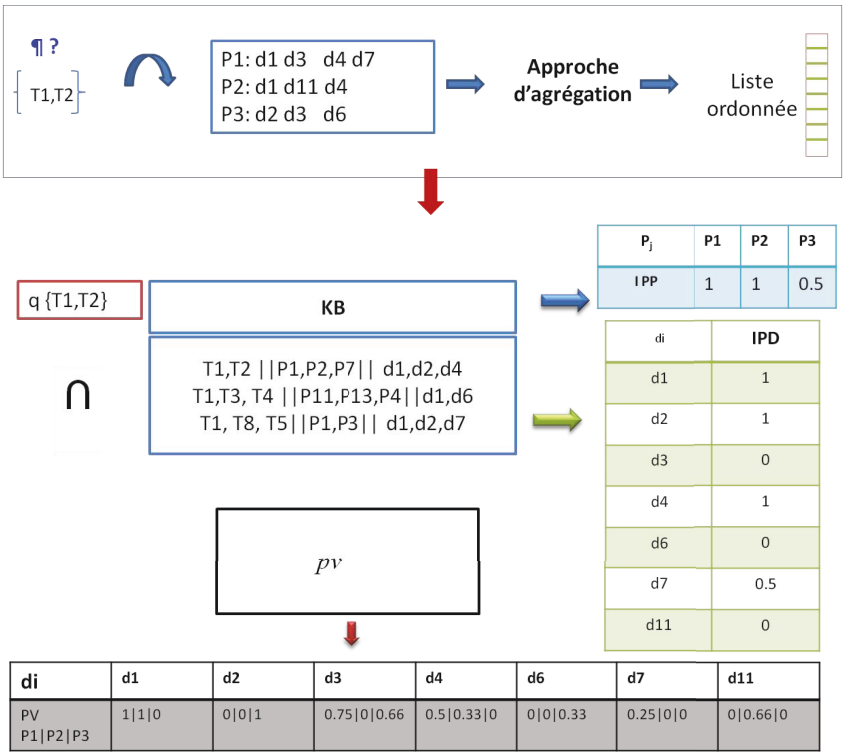


FIGURE 3.3 – Exemple de calcul de score

à partir d'un pair initiateur. Soient $P1$, $P2$ et $P3$, trois pairs contributeurs à la résolution de la requête q, et produisant trois listes ordonnées relatives : $P1 = \{d1, d3, d4, d7\}$; $P2 = \{d1, d11, d4\}$ et $P3 = \{d2, d3, d6\}$.
Soit $\mathcal{KB}$ la base des connaissances du pair initiateur. Nous calculons, respectivement, les valeurs de PV, IPP et IPD des documents retournés comme indiqué à la Figure 3.3.
En effet, en appliquant la méthode $setDocPosValue(d_k)$ et particulièrement l'équation 3.14 nous obtenons les valeurs de PV telles que indiquées dans la Figure 3.3. Si le document résultat de la requête n'a pas été retourné par le pair contributeur, sa valeur de PV prend 0.
Afin de calculer, respectivement, l'Importance d'un document par rapport à une requête IPD et celle d'un pair contributeur IPP, nous utilisons l'information sauvegardée dans la base des connaissances du pair initiateur de la requête.
L'opérateur $\sigma_{Sim(q,pr_i)>\mathfrak{S}}(\mathcal{KB})$ permet de retourner tous les profils de la base de connaissances dont l'extension contient des termes de la requête. Dans un souci de simplification, nous n'avons pas utilisé la méthode d'élagage citée dans l'Algorithme (le seuil de similarité, ce qui correspond à la condition $Sim(q, pr_i) > \mathfrak{S}$). Ensuite, nous avons sélectionné les profils positifs à nos documents retournés comme réponses à la requête, $(d1, d2, d3, d4, d6, d7, d11)$ et nos pairs contributeurs, soient (P1,P2,P3). Nous calculons l'importance IPP et IPD en se basant respectivement sur les mé-

thodes *setPeerPastImportance* et *setDocPastImportance* de l'algorithme PBA. Ces méthodes visent à évaluer l'importance d'un objet (document ou pair) par rapport à la requête. Le score global GS est obtenu dans un premier essai (qui a été validé par la suite empiriquement) en affectant les valeurs de 0.333 au paramètre de lissage α et 0.666 à β. Ce choix de valeurs a été motivé par la volonté de mettre plus de poids à IPP et IPD qu'au rang local du document.
Compte tenu de ces mesures, nous calculons, en guise d'exemple, le score global du document $d7$ comme suit :
$GS(d7) = \frac{1*(\alpha*0.25+\beta*0.5)}{1}$
Le score GS est normalisé par le nombre de pairs contributeurs (i.e. 3 dans notre cas). Il reste à trier les documents dans l'ordre décroissant des scores obtenus.

Il est intéressant de mentionner ici que, dans notre modèle d'agrégation de résultats, l'utilisation de profils répond le plus possible au critère de localité : le processus de gestion des profils est effectué hors-ligne que ce soit durant l'étape de sauvegarde ou de calcul des scores des profils. Cette dernière étape se fait d'une manière locale relativement à un pair initiateur.

3.5 Conclusion

Dans ce chapitre, nous avons proposé une approche d'agrégation de résultats à base de profils dans le contexte de Recherche d'Information dans les systèmes P2P. La motivation sous-jacente était de tenir compte des préférences des utilisateurs d'un pair afin d'améliorer l'efficacité du processus d'agrégation.
L'utilisation des profils, construits hors ligne, n'a été possible qu'à travers la définition d'un modèle générique de profils visant à résoudre des besoins de personnalisation à large échelle, tout en respectant, l'autonomie des pairs. Notre approche est complètement décentralisée, c'est-à-dire, que d'une part, l'autonomie des pairs est respectée (il n'y a pas de statistiques globales) et, d'autre part, la construction des profils utilisateurs est également locale.
Un algorithme d'agrégation à base des profils utilisateurs a été proposé. Il intègre une méthode de calcul de score qui hybride les rangs avec les scores provenant des profils. Le plus qu'apporte PBA, par rapport à une méthode à base de simples rangs, réside dans sa capacité à attribuer un cœfficient de préférence aux différents pairs, plutôt que de les considérer comme identiques.
Les approches de pondération de collections ont été déjà adoptées dans plusieurs travaux dans la littérature [Callan 1995, Ponte 1998, Si 2002a, Renda 2003]. Cependant, elles sont généralement basées sur des approches d'apprentissage utilisant des échantillons centralisés de statistiques, comme mentionné dans [Si 2003]. D'autres approches utilisent les jugements de pertinence ce qui est trop coûteux à mettre en place dans les systèmes P2P. L'algorithme PBA peut être vu comme une alternative aux approches de pondération citées, puisqu'il propose une approche d'apprentissage locale, et peut, donc étendre n'importe quel modèle à base de rangs, avec les poids calculés à partir de profils. L'algorithme PBA va bien au-delà de la pondé-

ration des SRIs. Il fournit un modèle d'agrégation de résultats qui intègre, outre les rangs des documents, qui sont des indicateurs subjectifs de pertinence, un autre indicateur basé sur l'historique des interactions de l'utilisateur avec les résultats. PBA est ainsi indépendant, aussi bien, du modèle à base de rangs, que de la technique de construction de profils (AFC).

Afin de valider l'efficacité de notre approche, il est important de procéder à son évaluation. Ceci doit passer par une étape indispensable, consistant à définir une démarche d'évaluation s'adaptant à la particularité de l'environnement P2P.

Chapitre 4

Démarche d'Evaluation

4.1 Introduction

L'évaluation des SRIs est l'une des pratiques les plus anciennes dans le domaine de la Recherche d'Information. Dans un cadre centralisé, l'évaluation se base sur l'utilisation des compagnes de test constituées autour d'un ensemble de documents, de requêtes et des jugements de pertinence. Évaluer une approche de RI, consiste donc à utiliser un ensemble de métriques permettant de la situer par rapport à ces jugements de pertinence, afin de dévoiler, selon les résultats trouvés, les limites de l'approche et les conditions favorisant sa réussite, ou son amélioration. Depuis des premiers test dirigés par les grandes campagnes d'évaluation, telles que Crandfield [Cleverdon 1991] et TREC [TREC 2008], la majorité des métriques consenties pour fournir une méthode d'évaluation sont quantitatives. Elles sont souvent centrées autour des indices précision/rappel.
Dans un contexte distribué à large échelle, les problèmes d'évaluation prennent plusieurs dimensions, que le cadre centralisé ne peut résoudre. En effet, tester de point de vue quantitatif, n'est plus suffisant pour analyser finement la qualité d'un classement, par exemple. Également, des aspects d'évaluation de performance liés à la distribution doivent être étudiés. Une autre dimension concerne la disponibilité des collections de test distribuées. En effet, ces dernières sont soit payantes soit incomplètes (pas de requêtes, des jugements de pertinence, pas d'indication sur le modèle de distribution, etc.).
La dernière dimension est liée particulièrement aux choix de plate-formes d'évaluation et à leurs adaptations aux contextes étudiés.
Afin de pallier ces problèmes, nous cernons la problématique d'évaluation dans un contexte de RIDLE dans la réponse à ces trois questions.

- Qu'est ce qu'on mesure pour classer les approches de RI de point de vue efficacité et performance ?
- Quelles collections utiliser afin de bien mener les évaluations ?
- Comment évaluer en reproduisant les conditions réelles ?

En se basant sur les réponses aux questions posées, nous proposons dans ce chapitre, un cadre d'évaluation qui décrit, dans un premier temps, les métriques quantitatives, qualitatives et de performance à utiliser pour l'évaluation d'un ensemble d'approches d'agrégation. Dans un second temps, nous proposons un protocole pour la définition des jeux de données appropriés à partir des collections centralisées et ce, en se basant sur des modèles de distribution et de réplication de données. Enfin,

nous présentons une plate-forme d'évaluation que nous concevons sur la base d'un simulateur.

4.2 Métriques d'évaluation pour les systèmes largement distribués

Le principal défi pour un système de RIDLE est de sélectionner pour une requête donnée les "meilleurs" pairs pouvant la satisfaire pour essayer d'obtenir des résultats similaires à ce qu'aurait donné une recherche centralisée (sur un seul pair). C'est l'essence même du problème de la recherche d'information distribuée. C'est ainsi qu'un processus de RIDLE doit être aussi bien *efficace* que *performant*. Le premier critère ou *efficacité* est généralement estimé en fonction des résultats pertinents retournés par rapport aux besoins de l'utilisateur, tandis que la *performance* en RIDLE se rapporte à un accès rapide incluant le temps de calcul et surtout le temps de communication.
Par conséquent, un processus d'agrégation, dans un contexte de RIDLE, devrait i) fournir une liste classée unique des résultats similaire à celle produite dans un contexte centralisé et ii) optimiser le temps de communication, l'utilisation de la bande passante, minimiser le nombre de messages échangés entre les pairs et ce en minimisant le nombre de pairs explorés. Malheureusement, les contraintes i) et ii) sont orthogonales.

L'efficacité et la performance de tout système de RIDLE par rapport à un ensemble de pairs donné dépend de l'efficacité et la performance de ses processus de sélection de pairs et d'agrégation de résultats. Nous concluons que l'efficacité doit se déterminer par le retour des résultats les plus pertinents et leurs classement aux premières positions et que cela doit se faire en minimisant les temps des communications (qui peut être atteint en minimisant l'échange de données entre les pairs).

4.2.1 Mesures d'efficacité

En général, en Recherche d'Information, les métriques d'évaluation sont des mesures d'efficacité des systèmes. L'objectif est de valider empiriquement les approches de recherche. Ainsi, tout SRI est évalué par un ensemble de mesures qui valident empiriquement ses approches d'indexation, de recherche ou de classement. Les mesures les plus importantes en Recherche d'Information sont la précision et le rappel. Cependant, valider l'efficacité quantitative des systèmes de RIDLE, ne semble pas suffisant pour examiner les degrés de pertinence des réponses retournées par chaque système, par rapport à un classement de référence, considéré comme "idéal".
Par ailleurs, nous avons procédé à scinder les métriques d'efficacité en quantitatives, ce qui représente les métriques classiques de RI utilisées dans l'état de l'art, et en métriques qualitatives, que nous avons définies dans le cadre de cette thèse.

4.2.1.1 Les métriques quantitatives

Dans cette famille de métriques, nous reprenons les principales métriques de l'état de l'art que nous utilisons dans notre validation empirique.

a- Les métriques de base

La précision et le rappel sont les métriques les plus communément utilisées dans l'évaluation de l'efficacité des SRIs.
Le rappel mesure à quel point un système est bon dans la recherche des documents pertinents à partir du corpus complet. En effet, étant donné qu'un système de recherche d'information renvoie les documents d dont r sont pertinents à la requête et sachant que dans le corpus il y a un nombre total de documents $R >= r$ qui sont pertinents à la requête. Le rappel est calculé ainsi :

$$Rappel = \frac{r}{R} \tag{4.1}$$

Alors que la précision est définie par la fraction :

$$Precision = \frac{r}{d} \tag{4.2}$$

Généralement, les utilisateurs ne sont pas intéressés par le classement de toutes les ressources, mais plutôt par le classement des meilleures d'où une utilisation large des métriques relatives (à une position donnée) que nous détaillons dans la prochaine section.

b- Les métriques relatives

Plusieurs auteurs ont testé l'efficacité de leurs SRIs sur la base de ces n premières réponses basées sur la précision et le rappel, les mesures relatives évaluent une «efficacité relative» du système. Plusieurs variantes de ces mesures existent, nous citons à titre d'exemple la précision à une position k, la précision moyenne (non-interpolée), etc. Nous donnons un intérêt spécial aux mesures suivantes qui servent à l'évaluation quantitatives de notre approche d'agrégation.

- $P@k$: Soit P l'ensemble des documents pertinents relatifs à la requête q de l'ensemble de toutes les requêtes Q. Soit R l'ensemble des documents retournés, k, un rang donné de la liste retournée. La précision pour les k premières positions et sa moyenne sont formulées comme suit :

$$P@k = \frac{|R \bigcap P|@k}{k} \tag{4.3}$$

$$MP@k = \frac{1}{|Q|} \sum_{q \in Q} P@k \tag{4.4}$$

Toutefois, les auteurs dans [Voorhees 2005] ont indiqué que la précision moyenne à une position reste insuffisante pour la comparaison des systèmes. En fait, elle sera dominée par des requêtes avec les documents pertinents [Witschel 2008]. Pour cette raison, une mesure orientée système a été définie. Il s'agit de la précision moyenne.

– Precision moyenne (Mean Average Precision MAP) : La MAP fournit une vision plus générale pour évaluer l'efficacité d'un système. La précision moyenne est considérée comme un exemple de mesures orientées système [Witschel 2008]. Elle est calculée comme suit :

$$AvP = \frac{1}{|R|} \sum Precision@k(d) \tag{4.5}$$

$$MAP = \frac{1}{|Q|} \sum_{q \in Q} AvP@k \tag{4.6}$$

– La précision relative (RP) : est la mesure proposée dans les travaux de [Witschel 2008] afin de pallier le problème de manque de jugements de pertinence. La mesure vise à comparer un classement donné D fourni par un SRI avec un classement de référence C. Cette mesure traduit la probabilité de pertinence d'un document estimée comme l'inverse de son rang dans le classement de référence. $RP@k$ mesure la probabilité moyenne de la pertinence parmi les k premiers documents d'un classement D.

$$RP@k(d) = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{r_C(d)} \tag{4.7}$$

où $r_C(d)$ est le rang du document d de D, dans le classement de référence C .

4.2.1.2 Synthèse

Les mesures de précision et de rappel ont été largement utilisées comme mesures fondamentales pour tester l'efficacité des SRIs [Renda 2003]. Toutefois, le recours aux courbes populaires de rappel-précision peut conduire à des interpolations qui sont parfois sources d'erreurs, en particulier pour des niveaux plus élevés de rappel [Voorhees 2005]. En outre, ils sont jugés insuffisants pour évaluer les classements distribués [Witschel 2008].
Les métriques d'évaluation devraient révéler sensiblement les différences qui existent entre le classement attendu des documents et celui fourni par le SRI.
Les courbes traditionnelles de précision-rappel ne s'avèrent pas assez complètes pour les considérer comme indicateurs d'évaluation des systèmes de RIDLE. En effet, elles gardent une même distance envers tous les documents jugés pertinents sans tenir compte de ce degré de pertinence ou de leurs classements.

Généralement, les métriques d'évaluation citées ont été largement utilisées dans l'évaluation des systèmes. Toutefois, lorsque l'on compare un classement donné avec

un classement de référence, nous avons la nécessité d'évaluer si les positions des résultats dans le classement fourni par un SRI ont été conservées par rapport au classement de référence ou si elles ont été fortement perturbées.
Il serait également intéressant d'évaluer le taux avec lequel le classement a été perturbé. Un classement donné serait préféré par rapport à un autre quand il présente un moindre décalage par rapport au classement de référence.

Exemple 1: Soit le scénario suivant :
Le système centralisé retourne une liste de documents classés comme suit (D1, D2, ..., D10) et deux systèmes distribués $S1$ et $S2$ donnent les classements suivants : $S1$ (D4, D3, D1, D2, D5) et $S2$ renvoie (D6, D2, D3, D4, D5). Il en résulte pour la métrique quantitative $P@5$ une valeur de 0,9 pour le système $S2$ et de 1 pour le système de $S1$, c.à.d que l'évaluation prétend que le système $S1$ fournit un meilleur classement par rapport à celui de $S2$.
Or qualitativement, le système $S2$ a réussi à obtenir le même classement pour les documents (D2, D3, D4, D5) que celui du classement centralisé. En d'autres termes les documents (D2, D3, D4, D5) ont les mêmes positions que le système centralisé.

Par conséquent, le problème d'évaluation fourni par une métrique quantitative revient au fait qu'elle néglige l'ordre des résultats fourni par un classement de référence, et elle néglige également le degré de perturbation de ce classement.
Pour cette raison nous avons proposé dans le cadre de cette thèse deux mesures d'évaluation qui se rapportent à une évaluation plutôt qualitative de la nature du classement fourni par rapport à un classement de référence.
Pour ce faire, nous avons défini deux nouvelles métriques qualitatives orientées utilisateur qui sont présentées dans la section suivante.

4.2.1.3 Mesures qualitatives

L'idéal dans un contexte distribué est de pouvoir classer les documents comme s'ils appartenaient à une seule collection en évitant le bruit dû à l'hétérogénéité des modèles de RI utilisés par les différents pairs.
Ainsi, un modèle de RID est jugé efficace, s'il se rapproche le plus possible du classement de référence. De cette manière, les documents fournis par un classement distribué, ne sont considérés pertinents que lorsqu'ils figurent dans le classement centralisé. Le degré de pertinence dépend plus particulièrement de leur présence parmi les premières positions du classement de référence. Nous parlons ici d'une évaluation qualitative des systèmes.

Nous définissons deux métriques dont la première permet l'estimation du taux de positions similaires ($SimPos@k$) et la deuxième s'intéresse au taux de perturbation, ($DeltaNoise@k$), par rapport au classement de référence.

a- Taux de Positions Similaires SimPos@k

Pour obtenir une évaluation qualitative plutôt que quantitative de notre classement, nous avons commencé par définir le pourcentage de documents gardant les mêmes positions par rapport à la liste de classement de référence, sachant le nombre de positions similaires, dénoté par, $SameNp$. Cette mesure est définie comme suit (par rapport aux k premiers documents retournés) :

$$SimPos@k = \frac{\sum_{q \in Q} SameNp}{k} \tag{4.8}$$

Cette première mesure, SimPos@k, bien que souhaitable, elle semble être un peu idéaliste. En effet, dans un environnement distribué à large échelle, il est souvent difficile de garder les mêmes positions par rapport au classement centralisé. C'est la raison pour laquelle, nous optons pour définir une deuxième métrique, qui vise à qualifier les systèmes par le taux de perturbation par rapport au classement idéal. Nous définissons la métrique DeltaNoise@k. **b- Bruit de classement DeltaNoise@k**

Un second critère que nous proposons, évalue le taux de bruit de classement, $DeltaNoise@k$, par rapport à la liste de classement de référence. Il est basé sur le décalage des positions entre deux classements donnés, $Np\Delta k$. Cette mesure est définie comme suit :

$$DeltaNoise@k = \frac{1}{k} \sum_{q \in Q} Np\Delta k \tag{4.9}$$

où Np Δ k est le décalage d'un document donné. A ce niveau, nous avons défini et présenté les mesures d'efficacité qui serviront à évaluer notre approche d'agrégation, à savoir, les mesures quantitatives et qualitatives. Cependant, une évaluation d'un système distribué ne doit pas passer à côté d'une évaluation de sa performance.

4.2.2 Étude de la performance d'un système de RID

Les mesures de performance sont généralement liées à des caractéristiques physiques du système, telles que, la largeur de la bande passante, la vitesse des processeurs utilisés, etc. La performance concerne les cinq éléments suivants :

- Le nombre de messages : c'est le nombre de messages requis pour agréger. Il dépend du nombre de pairs participant dans le processus d'agrégation (soit directement, soit indirectement, par exemple pour propager des mises à jour ou construire des statistiques globales) ;
- La taille d'un message : c'est la taille des chaînes transmises au pair initiateur par les pairs contributeurs (l'identifiant du document en octet).
- Le volume des messages : c'est le nombre de messages multiplié par la taille du message.
- La communication : elle est estimée par le volume des messages par la largeur de la bande passante.

– La synchronisation : c'est le maximum de temps d'attente nécessaire pour commencer l'agrégation.

Généralement, nous considérons qu'une approche d'agrégation proposée peut améliorer la performance si elle arrive à réduire les coûts de communication et de synchronisation.

4.3 Baseline

L'objectif ultime de notre approche d'agrégations à base de profils était particulièrement de pallier les problèmes dus à l'échange des statistiques et de ce qui en découle de problème de surcharge réseaux et de passage à l'échelle.
L'utilisation des rangs aurait dû être la solution si son efficacité était un peu meilleure d'autant plus qu'il s'agit d'un environnement hétérogène de point contenu, taille et modèles de RI. Côté performance, les modèles à base de rangs sont les seuls qui n'entrainent aucune charge aux réseaux et n'imposent aucune contrainte sur l'autonomie des systèmes. Partant de ce point de vue, nous avons pensé à hybrider les rangs avec les profils afin d'étendre cette approche avec une information qui peut améliorer nettement son efficacité. C'est la raison pour laquelle, il est important de comparer l'efficacité de quelques méthodes à base de rangs avec notre approche dans différents environnements. Ainsi, nous insistons dans un premier temps à comparer PBA à RR [Voorhees 1995a, Voorhees 1995b] et BC [Borda 1781, Renda 2003] qui sont des méthodes à base de rangs. La première, RR, étant issue des statistiques et malgré l'utilisation d'un pur hasard a prouvé son efficacité dans plusieurs cas de test en RI. La seconde, BC, est par contre spécifique à la famille de fusion de données et a été largement utilisée en tant que méthode de vote [Farah 2007].
Nous rappelons que notre approche demande un effort particulier pour la construction des profils en se basant sur l'approche de l'AFC. Ainsi nous avons vu le besoin de justifier cet effort de construction de profils en comparant notre approche à une qui utilise les informations plates du log file sans chercher à construire des groupements. Cette motivation nous a incités à définir la méthode Light Profile (LP), qui sera décrite plus tard dans le chapitre.
Enfin, la méthode PBA est une méthode qui se base sur l'apprentissage pour la construction des profils, il était possible de la comparer avec une méthode similaire telle que celle de Witchel [Witschel 2008]. Cependant, cette dernière utilise d'une part les statistiques globales (IDF global) pour calculer le score global des documents (en se basant sur la formule BM25 [Robertson 1995], et d'autre part, renforce les termes des documents selon leurs demandes dans les requêtes. Ainsi se comparer à l'approche de Witchel revient à se comparer par rapport à une méthode à base de score, dont on prévoit le résultat (plus, on utilise des statistiques globales, plus l'efficacité est meilleure). D'où nous avons procédé à un test avec la méthode cosinus (cos) [Salton 1975].

4.4 Collections de test

La collection de test constitue une composante essentielle dans l'évaluation d'un SRI. Elle est souvent constituée d'un ensemble de documents, d'un ensemble de requêtes, et d'une série de jugements de pertinence, permettant d'identifier les documents qui sont pertinents relativement à la requête. Plusieurs collections standards existent pour un cadre centralisé, notamment TREC [TREC 2008]. Les algorithmes de RI sont évalués selon leur capacité à retrouver les documents pertinents.

L'évaluation des systèmes de RIDLE nécessite le recours à des collections distribuées pour tester les algorithmes proposés. Cependant, il n'y a pas de collections distribuées standards comme dans le cadre centralisé. En effet, elles sont plutôt payantes et/ou non disponibles, ou souvent incomplètes (sans jugements de pertinence ou sans requêtes) [Bawa 2003, Lu 2003, Neumann 2006]. De plus, il n'y a pas, en général d'indication sur la méthode de distribution des données et des requêtes [Lu 2003]. C'est la raison pour laquelle, plusieurs travaux se sont orientés vers l'étude de la distribution d'une collection centralisée pour construire plusieurs collections de test distribuées [Zammali 2010]. Ceci consiste, dans les systèmes de RIP2P, à placer dans les différents pairs, des sous-collections, disjointes ou non, à partir de la collection centralisée. Le problème majeur de telles collections distribuées est que les scénarii de distribution ne sont pas souvent réalistes et surtout sont différents d'une étude à l'autre. Il en résulte que la comparaison d'approches distribuées est pratiquement impossible à faire. Disposer de plusieurs distributions de documents réalistes, selon différents scénarios devient une exigence pour la construction et l'expérimentation avec des environnements de RIDLE réalistes.

Dans nos stratégies d'évaluation, différentes méthodes de recherche de documents pertinents dans les pairs ont été testées. En effet, en se basant sur différentes méthodes de filtrage issues du modèle vectoriel, le système peut choisir, aléatoirement, ou imposer une des mesures pour décider de la pertinence des documents par rapport aux requêtes. De plus, nous faisons bien la différence entre une similarité globale et une autre locale retournée par un pair contributeur. En effet, la pertinence d'un document par rapport à une requête dans une collection qui renferme 500 documents, par exemple, doit être différente de sa pertinence dans une collection à 20 documents.
Ce qui change aussi, c'est l'ensemble des documents disponibles sur chaque pair (c'est à dire selon la méthode de distribution) et ainsi la méthode de replication des documents.

Afin d'adresser tous ces défis, nous proposons un certain nombre de critères, pour évaluer les systèmes de RIP2P. Nous commençons par présenter les principaux modèles de distribution et de réplication que nous avons utilisés.

4.4.1 Modèles de distribution

Dans la littérature, les auteurs n'accordent pas une grande importance au modèle de distribution adopté pour distribuer leurs données. La majorité appliquent des approches statistiques afin de distribuer la collection centralisée sur un ensemble de pairs. Les approches statistiques peuvent décider d'une distribution équitable des données entre les pairs, ou d'une distribution aléatoire ou autres. Ce qui importe sur ce plan est que chacune des approches de distribution peut influencer les applications qui exploitent ces données. C'est dans ce contexte, que notre problème d'agrégation en RIP2P doit utiliser un modèle de distribution qui convient aux hypothèses sous-jacentes (notamment l'hétérogénéité des pairs tant en taille de collection, qu'en contenu). Un modèle de distribution uniforme n'est pas suffisant, puisqu'il va garantir l'homogénéité des pairs (au moins en termes de taille de collections), ce qui ne satisfait pas nos hypothèses.
C'est dans ce cadre, que la recherche d'un modèle de distribution, tenant compte de la sémantique des documents est une voie intéressante à explorer. Nous présentons, respectivement dans ce qui suit, les principaux modèles de distribution statistiques et sémantiques que nous avons appliqués sur nos données afin de simuler les scénarii réels et possibles.

4.4.1.1 Modèles statistiques

La distribution des collections peut suivre plusieurs lois. Nous parlons de distribution uniforme, aléatoire et zipF.

1. **Distribution uniforme** : La distribution uniforme, comme son nom l'indique, distribue de façon équitable les documents de la collection centralisée sur le nombre de pairs constituant le système distribué. Ainsi, on crée plusieurs collections de même taille. Dans certains cas, on peut également utiliser le contenu comme critère de distribution (chaque sous-collection présente alors la même distribution en contenu que la collection centralisée). La distribution uniforme, bien que peu réaliste, a été largement utilisée dans plusieurs scénarios de distribution, telles que la distribution des capacités des pairs, de la charge des pairs ou de la popularité des clés dans les tables de hachage.
2. **Distribution aléatoire** : Une distribution aléatoire quant à elle, exploite le pur hasard pour la distribution de documents. Aucune corrélation n'existe entre les documents d'un pair, et la taille des pairs est également hétérogène. En effet, le nombre de documents et de requêtes affecté à chaque pair se fait de façon purement aléatoire. De même, la nature des documents dans une collection n'est pas homogène, ce qui contribue à créer des collections non spécialisées relativement à un thème donné. La distribution aléatoire permet d'avoir des sous-collections qui regroupent des thèmes hétérogènes avec un nombre différents de documents. En guise d'exemple, le système PAST applique une distribution aléatoire pour créer son système P2P [Druschel 2001].
3. **Distribution de ZipF** : Faisant référence à son auteur, le linguiste américain

George Zipf Kingsley, la loi de zipF apparaît dans les années (1935, 1949) [Adamic 2002] comme une loi empirique formulée en utilisant la statistique mathématique. Selon la loi de Zipf, étant donné un corpus documentaire écrit en langage naturel, la fréquence de retrouver un mot est inversement proportionnelle à son rang dans un tableau des fréquences des différents mots utilisés. Ainsi, le mot le plus fréquent se produira environ deux fois plus souvent que le deuxième mot le plus fréquent, trois fois plus que le troisième mot le plus fréquent, dix fois que le dixième mot et cent fois que le centième mot le plus fréquent, etc.
Par exemple, dans le corpus Brown [Adamic 2002], le mot «the» est le mot le plus fréquent , et représente à lui seul près de 7% de toutes les occurrences de mots (69 971 sur un peu plus de 1 million). Le deuxième mot "of" représente quant à lui un peu plus de 3,5% des mots (36,411 occurrences), suivi de «and» (28 852).
La loi Zipf estime que, dans un texte donné, la fréquence d'occurrences f(n) d'un terme est liée à son rang n dans l'ordre des fréquences par une loi de la forme :

$$f(n) = \frac{K}{n} \tag{4.10}$$

où K est une constante. Ces observations ont été généralisées à d'autres domaines tels que les sciences sociales, où cette même relation a été observée entre la population d'une ville et son rang en terme de population, les classements sur le revenu, et ainsi de suite.

Dans notre cas, une distribution zipF des occurrences des documents (respectivement des requêtes) sur les pairs a été appliquée.

4.4.1.2 Modèles Sémantiques

La spécialisation des sous-collections des documents à des thèmes bien spécifiques correspond bien à la réalité. En effet, un pair peut être assimilé à un utilisateur et donc, va avoir en général, un ensemble de documents liés à un thème donné. En effet, il parait peu probable qu'un utilisateur s'intéresse à tout.
La prise en compte de cette hypothèse peut amener plus d'efficacité pour la recherche distribuée (on peut rechercher les pairs pertinents). C'est pour cela que nous avons créé des collections spécialisées à partir de la collection globale et des requêtes. Nous rappelons que la collection de référence est constituée d'une part des documents et d'autre part des requêtes et de leurs réponses selon une mesure de similarité globale. Étant donné que la requête représente un thème recherché, nous avons cherché à regrouper les requêtes similaires, en se basant sur leurs termes communs. Nous avons alors regroupé les documents sur la base de leur contribution à la réponse à des requêtes similaires. Cependant, vu que la collection que nous essayons de constituer n'est autre que le contenu d'un pair, nous avons été amené à introduire un peu de bruit sur le thème général de la collection. En effet, rien n'in-

terdit un pair qui a beaucoup de documents sur un thème donné d'avoir d'autres documents sur un autre thème.
Ainsi, pour chaque cluster de requêtes similaires, la part la plus grande (X %) des documents contribuant à la réponse à ces requêtes sera affecté à un pair choisi aléatoirement et le reste, (100-X) %, sera affecté aléatoirement à un ensemble de pairs dont le nombre ne dépasse pas le nombre de réponses. Une distribution sémantique des données sur les pairs a pour but de définir des pairs sémantiquement homogènes et bien étiquetés.
La construction des collections décentralisées qui reposent sur un principe de spécialisation sémantique, constitue donc un cadre réaliste pour évaluer notre approche d'agrégation.

4.4.2 Modèles de réplication

La réplication est une variante largement utilisée lors de la construction des collections de test. Comme la distribution, la réplication concerne aussi bien les requêtes que les documents. Elle est également appliquée afin d'augmenter le nombre de requêtes et de documents dans une collection. En simulation, même si la réplication des documents n'est pas appliquée, celle des requêtes s'avère, en général, nécessaire afin d'exprimer des besoins similaires à partir de plus qu'un pair. De plus cette réplication semble naturelle puisque souvent les utilisateurs peuvent exprimer des besoins similaires. Google utilise ce principe par exemple pour proposer des termes à rechercher à partir de la saisie partielle d'un utilisateur.
La réplication des documents dans un système de recherche consiste à dupliquer un document qui se trouve sur une collection dans d'autres collections (une ou plusieurs fois), ce qui contribue à la formation de collections non disjointes.
Les réplications des documents et des requêtes peuvent être faites de manières différentes. La replication peut décider de dupliquer aléatoirement les ressources (requêtes et/ou documents) parmi les différentes collections. La réplication peut être aussi effectuée uniformément sur un ensemble de collections. En guise d'exemple, nous pouvons décider de répliquer les requêtes 5 fois sur les collections de test. Ainsi, pour 5000 requêtes initiales, nous allons nous retrouver avec un total de 25000 après la réplication. Une réplication peut être réalisée, sur les requêtes les plus populaires, ou les ressources les plus demandées, d'où le recours à la loi de zipF [Adamic 2002] pour les déterminer.

4.5 Environnement

4.5.1 Choix de la plate-forme d'évaluation

Etant donné que la mise en place d'un vrai environnement distribué à large échelle est une solution quasiment impossible, vu les coûts qu'elle engendre et la difficulté de maintenance et de synchronisation, l'évaluation de notre approche nous a amené à choisir entre deux solutions :

- Utiliser une plate-forme d'évaluation distribuée et reproduire les conditions réelles du système.
- Simuler le système distribué avec ses différents nœuds en considérant une seule machine.

Le recours aux plate-formes de test distribuées permet d'émuler les applications distribuées dans des conditions réelles puisqu'elles se basent sur des machines géographiquement distantes dont chacune émule un ensemble de machines virtuelles. Par cette stratégie, ces plate-formes de test assurent une bonne montée en charge et reproduisent la condition de distribution entre les machines physiques utilisées. C'est la raison pour laquelle plusieurs plate-formes de test réparties ont vu le jour telles que PlanetLab [Pla 2002a] et Grid5000 [Cappello 2005] pour la communauté française. Cependant, ces dernières souffrent encore de plusieurs problèmes se rapportant au ralentissement de la vitesse de communication entre les machines d'un même site physique au détriment de la simulation du réseau. De plus, une difficulté de prise en main et un manque de performance des systèmes de déploiement sont généralement remarqués [Lübke 2012]. En outre, généralement l'utilisation effective de ces systèmes est réservée aux établissements qui partagent des sites du système distribué, c'est ainsi qu'un accès sécurisé est généralement nécessaire [Pla 2002b]. L'évaluation via de telles plate-formes doit passer également par une étape de réservation préalable des machines pour un temps limité ce qui n'est pas très pratique pour réaliser plusieurs expériences.

La deuxième solution, quant à elle, consiste à simuler un ensemble de nœuds sur une seule machine physique. De cette manière nous perdons un élément important qui est la reproduction de la distribution. Cependant, cela amène une simplification importante dans la mise en oeuvre de la simulation et évite de devoir construire et/ou utiliser des ressources physiques importantes. Une simulation permet également de tester des configurations difficiles à mettre en oeuvre autrement (plusieurs milliers de noeuds par exemple). D'autre part, la simulation peut commencer et s'arrêter à tout moment pour sauvegarder l'état du système ou vérifier un bug ce qui très important pour la maintenance du système et la compréhension de ses problèmes.
Toutefois, les différentes facilités qu'offre un simulateur de système distribué se paient au détriment d'un ensemble d'abstractions du système réel telles que l'omission de détails protocolaires fins du réseau informatique par exemple. De plus, plusieurs types de mesures de performance notamment la question du temps de réponse peuvent être difficiles à simuler.

Dans notre cas d'application, nous avons choisi d'évaluer la performance de notre algorithme d'agrégation de manière analytique et empirique. En reposant sur l'ensemble de ces hypothèses, nous avons opté pour le choix du simulateur générique Peersim étendu dans le cadre du projet de Routage optimisé par Apprentissage de Requêtes (RARE) [RARE 2007]. Ce dernier permet des recherches sur le thème du

routage sémantique de requêtes dans les systèmes P2P. Le simulateur utilisé est basé sur Peersim [Jelasity 2007] et auquel nous ajoutons des modules spécifiques à notre problématique.

4.5.2 Simulateur Peersim-Rare

PeerSim [Jelasity 2007], est un outil Opensource écrit en Java qui présente l'avantage d'être déjà spécialisé dans l'étude des systèmes P2P. Il dispose d'une architecture ouverte et modulaire, ce qui lui permet d'être adapté à des besoins spécifiques.
Pour la recherche d'information dans les systèmes pair-à-pair, nous avons opté pour le simulateur générique de Recherche d'Information réalisé dans le cadre du projet RARE [RARE 2007]. Ce simulateur est construit au dessus de PeerSim et peut être vu comme une spécialisation de ce dernier pour la recherche d'information. Pour cette raison nous avons choisi de l'appeler PeerSim-RARE.
La description de PeerSim-RARE ne peut être faite en ignorant ses composants de base hérités de son ascendant Peersim [Jelasity 2007]. Ces composants seront utilisés comme paramètres pour la configuration du simulateur (voir plus tard dans le chapitre -Exemple 5.4.2-).

4.5.2.1 Composants de base de Peersim

Héritant de toutes les fonctionnalités de base de PeerSim, PeerSim RARE repose sur quatre types de composants :

- **Les Protocoles** : Le simulateur Peersim est basé sur l'utilisation des protocoles qui sont des fonctionnalités associées à chaque nœud du système distribué. Ainsi, pour conduire l'ensemble des simulations nécessaires dans le système, une suite de protocoles doit être activée sur chaque nœud et dans chaque cycle de simulation.
- **Les Initialisateurs** (initializer) : Comme leurs noms l'indiquent, les initialisateurs sont les modules qu'utilise Peersim afin d'initialiser les protocoles en début de simulation, notamment la définition initiale du voisinage de chaque nœud et l'injection des fichiers de données à utiliser.
- **Les Dynamiques** (Dynamics) : Ce sont des modules qui permettent d'introduire une dynamique dans le réseau, notamment activer ou désactiver certains nœuds afin de simuler les scénarii de connexion et de déconnexion de pairs, l'ajout et la suppression des documents, etc.
- **Les Observateurs** (observers) : Ils servent à recueillir les données nécessaires afin de présenter le résultat du processus de simulation. Ils sont généralement lancés à la fin de chaque cycle de simulation.

Après avoir présenté les composants de base de PeerSim, il est important de présenter comment Peersim-RARE organise ses briques de base pour la recherche d'information dans les systèmes P2P. Nous parlons d'une architecture applicative de

ce simulateur. Cette architecture est très importante à décrire puisque elle sert de noyau à notre simulateur d'agrégation présenté ultérieurement dans ce chapitre

4.5.2.2 Architecture applicative de Peersim-RARE

FIGURE 4.1 – Architecture applicative de Peersim-RARE

Afin de tester nos expérimentations au sein de Peersim-RARE, nous avons été amenés à examiner son architecture applicative. Cette dernière explique la logique utilisée lors de la recherche d'information en P2P. Elle met en œuvre les couches suivantes (voir Figure 4.1).

- **La couche de Réseau Physique** : c'est une couche de bas niveau qui simule le concept de machines et l'attribution des adresses IP. C'est la couche qui prend en charge les paramètres de configuration du simulateur.
- **La couche de Réseau Logique** : c'est la couche consacrée aux relations de voisinage des pairs et des mécanismes de leur gestion (adressage, la distribution des algorithmes, des requêtes P2P, et propagation de requêtes P2P) tout en reposant sur la première couche. Elle définit également les opérations sur les pairs notamment l'ajout, la suppression de pairs, etc.
- **La couche de Gestion de Documents** : cette couche représente les concepts de gestion des documents par les pairs. Elle permet essentiellement le stockage des documents sur l'ensemble des pairs du système en considérant les fichiers de définition et de distribution de documents.
- **La couche requête/réponse** : c'est la couche de gestion de requêtes, elle permet de représenter une requête sous forme d'un ensemble de termes, de réaliser l'appariement avec les documents, de la stocker et de la distribuer sur les différents pairs du système.
- **La couche de recherche d'information** : Dans cette couche, le simulateur exploite un ensemble de modèles de routage de requêtes. Les modèles déjà implémentés au niveau de cette couche sont essentiellement "Gnutella" [Gnu 2007] et "PlanetP" [Cuenca-Acuna 2003]. Il s'agit essentiellement de modèles pour la sélection des pairs pertinents à la requête selon les algorithmes utilisés, puis d'agrégation de résultats de ces pairs.

4.5.2.3 Discussion

L'architecture applicative, telle que présentée dans PeerSim-Rare, souffre de certaines limites, quant à l'évaluation de notre approche d'agrégation à base de profils. En effet, aucune prise en compte de la base des connaissances n'est envisagée dans ce simulateur. De plus, si la couche de recherche d'information permet d'implémenter des algorithmes de routage (Planet P et Gnutella), rien n'est proposé pour l'agrégation. Or, dans notre cas nous avons besoin d'une couche d'agrégation indépendante de la couche de routage afin d'implémenter les approches d'agrégation indépendamment de cette dernière. En outre, l'utilisation d'un algorithme de routage aléatoire tel que Gnutella [Gnu 2007] peut impacter les résultats d'agrégation. C'est la raison pour laquelle nous avons cherché à neutraliser l'effet du routage en proposant une méthode de "routage idéal" dont le principe sera explicité plus tard dans le chapitre.
Tous ces points nous ont amené à ajouter de nouvelles couches dans Peersim-RARE et à proposer quelques autres modifications. Nous appelons notre simulateur d'agrégation "PeerSim-RARE+" qui fera l'objet de la section suivante.

4.5.3 Simulateur d'agrégation : PeerSim-RARE +

PeerSim-RARE + est une spécialisation de PeerSim-RARE. Il intègre trois nouvelles couches : la première est relative à la création et la gestion des profils des utilisateurs (partant de la construction des fichiers log jusqu'à la création et l'évolution des bases de connaissance), et une deuxième couche permettant d'agréger les résultats de la phase de routage selon différents algorithmes, entre autres notre l'algorithme PBA qui utilise les profils. Une troisième couche d'évaluation est consacrée à l'implantation d'une panoplie de mesures d'efficacité quantitatives et qualitatives. La couche de recherche d'information a également été modifiée pour intégrer une démarche de "routage idéal" afin de ne pas subir l'effet des différentes méthodes de routage sur les approches d'agrégation.

Nous présentons dans ce qui suit les couches modifiées et/ou ajoutées par Peersim-RaRE+

4.5.3.1 Couche de recherche d'information

Au niveau de cette couche, nous proposons une méthode de recherche qui nous permet de nous focaliser exclusivement sur l'efficacité des algorithmes d'agrégation sans subir l'effet du routage. Pour cela, nous partons d'une hypothèse de connaissance préalable des pairs pertinents à la requête.
Pour connaître ces pairs, nous nous basons particulièrement sur le fichier de définition de requêtes qui présente pour chaque requête, l'ensemble de ses réponses (les identifiants de requêtes). À partir du fichier de distribution de documents, nous pouvons énumérer les différents pairs qui contiennent les documents. Le détail de ces fichiers sera donné plus tard dans ce chapitre. Ainsi, notre méthode de recherche

consiste à envoyer les requêtes des utilisateurs directement aux pairs pertinents dans le système, c'est ce que nous appelons "routage idéal". Par la suite, n'importe quelle méthode d'agrégation peut être utilisée et évaluée conjointement à cette méthode de routage.

4.5.3.2 Couche de gestion de Profils

La couche de gestion de profils est un préalable à notre algorithme d'agrégation PBA et à toute application utilisant les profils comme décrit dans le chapitre 3, un exemple étant le routage sémantique présenté dans [Yeferny 2009].
La couche de gestion de profils permet les fonctionnalités suivantes :

- La mise en place d'un protocole d'initialisation des profils dans le composant *initializer*. Le protocole se déclenche au début des cycles de simulation et fait partie des protocoles qui configurent le simulateur.
- La création des logs files en se basant sur un processus d'apprentissage se basant lui même sur la validation croisée (2/3 des données sont réservées à la phase d'apprentissage et le 1/3 restant est réservé pour le test).
- La création d'une base de connaissance par nœud : La génération des profils a été faite en utilisant l'algorithme de Godin [Godin 1995], implémenté dans la plate-forme Galicia V 3 [Valtchev 2003]. La contribution principale à ce niveau est la décentralisation totale des connaissances entre les différents noeuds du réseau ce qui améliore le potentiel de passage à l'échelle dans l'approche proposée.
- L'évolution contrôlée de la base de connaissances en utilisant les détecteurs de défaut d'apprentissage que nous allons décrire ultérieurement, dans le chapitre 6.

4.5.3.3 Couche d'agrégation

La couche d'agrégation permet d'implémenter des algorithmes d'interclassement de résultats. Un ensemble de méthodes a été implanté notamment :

- **Des algorithmes à base de rangs :** Les méthodes *Round Robin RR* [Voorhees 1995a] et *Borda Count BC* [Borda 1781] présentées dans le chapitre 2 ;

- **Des algorithmes à base de score** : *la méthode cosinus* [Salton 1975, Egghe 2010] calculée de manière distribuée. Également, nous avons implémenté les méthodes Dice overlap, jaccard.

- **Un algorithme d'agrégation à base de profils *(PBA)*** : utilise une base de connaissances par nœud, et nécessite l'intégration de la couche de gestion de profils détaillée dans la section 4.5.3.2.

- **Un algorithme Light Profile *LP*** : il s'agit d'un algorithme que nous avons défini pour créer une variante d'algorithme à base de profils qui ne nécessite

pas de mécanismes d'apprentissage sophistiqués.
L'intuition derrière l'implantation de cet algorithme était de justifier l'effort de construction de connaissance à partir d'un fichier log plat puisque LP exploite directement ce log.
Le principe de LP est de privilégier les documents provenant des nœuds qui ont bien répondu aux requêtes passées. Cette information est déduite directement du log file qui détient une trace de la requête émise, des nœuds contributeurs et des documents qui ont bien servi à satisfaire la requête. Le poids d'un nœud est ainsi déduit du taux des requêtes auxquelles il a bien répondu par rapport à toutes les requêtes soumises. Ce poids est calculé comme suit :

$$poids\_n = \frac{NRservies(n)}{NR} \tag{4.11}$$

avec :
- NRservies(n) : est le nombre de requêtes servies par le nœud "n" (i.e, dont "n" est un pair contributeur).
- NR : le nombre de requêtes lancées sur le réseau.

En se basant sur le poids de ce nœud, $poids\_n$, le poids du document, $poids\_d$, est donné par la formule suivante :

$$poids\_d = poids\_n \times PV(d) \tag{4.12}$$

tel que PV (d) est la valeur positionnelle du document dans la liste de réponse calculée comme indiqué dans le chapitre 3.

4.5.3.4 Couche d'évaluation

Contrairement à PeerSim-RARE qui intègre, dans la couche de RI, un calcul implicite de la performance et efficacité des modèles de RI, nous avons choisi d'intégrer une couche spécifique pour le calcul des métriques de RI. Les métriques développées à ce niveau sont celles présentées dans ce chapitre 4 :
- La précision à une position donnée : P@k
- La précision moyenne non interpolée : MAP
- La précision relative : RP@k
- Les positions similaires au classement idéal à un rang donné : SimPos@k
- Le bruit de classement à un rang donné : DeltaNoise@k

4.5.4 Scénarii de simulation

La simulation doit prendre en considération tous les paramètres dans un environnement d'application. C'est ainsi que dans un système de RIP2P, nous devons être en mesure de simuler la distribution des données et des requêtes et leurs schémas de réplication. Pour cela plusieurs modèles doivent être sélectionnés et appliqués sur notre collection de test. D'autres paramètres doivent être pris en considération,

notamment, l'hétérogénéité des pairs de point de vue taille et modèles de RI. De même, vu que la dynamique est un principe constant dans un environnement P2P, nous sommes appelés à simuler les situations pour lesquelles soit de nouveaux pairs joignent ou se déconnectent du réseau, soit de nouveaux documents sont rajoutés sur certains pairs ou sont supprimés. De plus, la simulation doit prévoir l'arrivée de nouvelles requêtes sur les pairs et le changement du besoin de l'utilisateur. Enfin, la modélisation des intensions des utilisateurs et leur choix de télécharger ou de consulter des documents suite à sa requête doivent être également simulés. Ainsi plusieurs scénarii de test peuvent être réalisés afin de détecter l'impact de ces différents paramètres. Nous citons respectivement :

- l'impact de l'utilisation d'une distribution statistique des jeux de données ;
- l'impact de l'utilisation d'une distribution sémantique des jeux de données ;
- l'impact de la réplication des documents sur les différents pairs, notamment dans le cas de chevauchement de collections distribuées ;
- l'impact de l'utilisation de collections hétérogènes en taille, en contenu et en modèle de RI ;
- l'impact de la dynamicité du système, tant via l'émergence de nouveaux besoins utilisateurs (nouvelles requêtes) que de l'évolution du système (apparition/disparition de pairs, ajout/suppression de documents).

4.6 Conclusion

Nous avons proposé dans ce chapitre une démarche d'évaluation du système, en soulignant les principales hypothèses à se poser afin de procéder à l'évaluation effective du modèle. Nous avons présenté les principales mesures à exploiter à cette fin. Nous avons défini trois niveaux d'évaluation : une évaluation quantitative de l'efficacité du système, une évaluation qualitative et une évaluation de la performance du système. Nous avons également choisi la plate-forme d'évaluation de notre système en se basant sur la simulation. Afin de mener à bien notre simulation, nous avons défini notre propre simulateur qui peut être vu comme une spécialisation du simulateur PeerSim-RARE.
Dans le chapitre suivant, une étude expérimentale des différents scénarii de test sera menée.

Chapitre 5

Validation et expérimentation

5.1 Introduction

Afin de valider notre approche, une partie importante de notre travail doit être réservée à l'évaluation. Une démarche d'évaluation tenant compte de la difficulté d'évaluer des systèmes de Recherche d'Information à large échelle a été déjà introduite dans le chapitre précédent. Nous avons présenté les modèles de distribution et de réplication de données, les métriques d'évaluation de performance et d'efficacité quantitative et qualitative pour les systèmes de Recherche d'Information Distribuée à Large Échelle, en général, et les systèmes de RIP2P en particulier.
Dans le présent chapitre, nous menons une étude analytique et expérimentale visant à valider la performance et l'efficacité de notre approche d'agrégation proposée au chapitre 3, dédiée aux systèmes de RIP2P purs non structurés.
Pour estimer la performance de notre approche une étude analytique de PBA est proposée dans une première section. L'étude expérimentale utilise un simulateur d'agrégation basé sur le simulateur générique Peersim [Jelasity 2007] étendu dans le cadre du projet de Routage optimisé par Apprentissage de Requêtes (RARE) [RARE 2007] que nous avons introduit dans le chapitre 4. Notre simulateur permet d'observer les résultats des différentes métriques quantitatives et qualitatives par rapport à un classement de référence (le classement de la collection centralisée).
Les principales métriques que nous avons utilisées sont la précision relative par rapport à une position donnée ($p@k$), la précision moyenne non interpolée (MAP), la précision relative RP, le taux de positions similaires par rapport au classement de référence ($SimPos@k$) et le bruit de classement à un rang donné ($deltaNoise@k$).
Nous présentons brièvement les corpus centralisés que nous avons utilisés et la démarche nécessaire à la construction des collections distribuées à grande échelle.
Différents scénarii de simulation ont été appliqués en vue d'étudier l'impact de différents paramètres sur l'efficacité des algorithmes d'agrégation. Ces paramètres concernent la manière de distribuer les données, l'impact de la réplication, l'impact de l'hétérogénéité des collections (particulièrement des modèles de RI utilisés) et la prise en compte ou non de la fraîcheur des bases de connaissances. Un bilan des résultats observés est donné à la fin de ce chapitre.

5.2 Etude de la performance de l'algorithme PBA

La validation de l'algorithme PBA nécessite une validation de sa performance et de son efficacité. Dans cette partie, nous nous intéressons particulièrement à l'aspect analytique de PBA. Ainsi, nous optons à quantifier le temps nécessaire à l'agrégation, par rapport au temps de synchronisation et de communication.
Pour cela, nous utilisons le tableau de notations suivant :

Symbole	Signification
VG	Volume global de messages échangés
$\|P_q\|$	nombre de pairs contributeurs à la réponse d'une requête q
$\|D_{qp}\|$	nombre de documents répondant à une requête q et provenant d'un pair p
$TMsg$	Taille d'un message donné
$tIPP$	temps de calcul de l'importance d'un pair selon l'historique
$tIPD$	temps de calcul de l'importance d'un document (analogue à tIPP)
$\|\mathcal{PR}_q\|$	nombre de profils similaires à une requête q
α	coût de l'opération d'intersection et d'union
$Tcalcul$	temps de calcul pour la génération de la liste finale agrégée

TABLE 5.1 – Table des notations - Mesure de performance

Contrairement aux méthodes d'agrégation de l'état de l'art, une *synchronisation* n'est pas nécessaire pour démarrer le calcul du score final de chaque réponse retournée. En effet, PBA commence son calcul de score dès la réception de la première réponse du premier pair contributeur. Ainsi, il évite un temps nécessaire chez certains algorithmes, tels que BC et RR, qui nécessitent toutes les contributions des pairs avant d'entamer la tâche d'agrégation.
De même, dans notre cas, les données échangées sont de faibles tailles : seulement l'identifiant du document est nécessaire, ce qui réduit la taille d'un message, à la taille (en octet) de cet identifiant. De plus, aucune donnée statistique n'est demandée.
Ainsi, le volume global de messages échangés est calculé comme suit :

$$VG \ = \ |P_q| \ \times |D_{qp}| \times TMsg \tag{5.1}$$

où la « taille d'un message », TMsg : correspond à la taille de l'identifiant d'un document retourné et la taille de son rang (en octet), si la liste retournée n'est pas ordonnée (le rang est fourni). Souvent, cette information (i.e le rang) est implicitement fournie par le retour d'une liste ordonnée de documents par un pair contributeur.

Sachant que la sélection des profils similaires se réalise au moment de la recherche, le temps associé sera entrelacé au temps de la recherche. Le temps de calcul est estimé à :

$$Tcalcul = |P_q|.|D_{qp}|.(tIPP + tIPD) \tag{5.2}$$

Or, $tIPP$ et $tIPD$ sont les temps correspondant respectivement, aux calculs de l'importance d'un pair et de celle d'un document selon l'historique des requêtes. Ce calcul nécessite de chercher dans les profils sélectionnés le degré d'apparence du pair contributeur ou de document. Alors tIPP, et tIPD sont respectivement :

$$tIPP = \alpha|\mathcal{PR}_q| \tag{5.3}$$

et

$$tIPD = \alpha|\mathcal{PR}_q| \tag{5.4}$$

Ainsi, le temps de calcul global est donné par l'équation suivante :

$$Tcalcul = 2\alpha|P_q| \times |D_{qp}| \times |\mathcal{PR}_q| \tag{5.5}$$

L'exemple ci-dessous donne un aperçu sur le calcul du nombre de messages nécessaires à l'agrégation des résultats.

Exemple 1:

Soit S, un système P2P constitué de 800 pairs, où chaque pair héberge 500 documents. Soit q, une requête telle que le nombre de pairs qui contribuent à sa réponse est estimé à 10 pairs

Si nous supposons que chaque pair contributeur, p_q, retourne une liste de 11 documents, en réponse à q, le calcul du nombre de messages pour l'agrégation des résultats de la requête sera effectué comme suit :

- $|P_q| = 10$.
- $|D_{qp}| = 11$.
- Taille d'un message (taille de l'identifiant d'un document) = 2 octets.
- Le nombre de messages nécessaires à l'agrégation des résultats de $q : VG = 10 \times 11 \times 2$ ce qui correspond à 220 octets.

Si on suppose le lancement de 1000 requêtes par heure, on atteindra un volume de 200 Ko/h, ce qui est tout à fait raisonnable. Si au lieu de notre méthode, nous utilisons une méthode à base de score, où nous avons besoin d'échanger une partie des index locaux, nous aurons besoin d'échanger des statistiques, telles que, pour chaque document, TF, IDF et taille de document, on doit pouvoir multiplier le volume de message au moins par 3 (3 entiers sur 4 octets = 12 octets en plus par messages).

5.3 Jeux de données utilisés

Dans cette section, nous présentons les collections de données que nous avons utilisées dans notre simulation. Il s'agit de trois types de collections : deux collections construites au sein du projet RARE : une première générale (BigDataSet) et la deuxième spécialisée (MusicDataSet) et une troisième collectée à partir de l'annuaire DMOZ. En effet, par ce choix, nous avons essayé de voir, de près, l'impact de la nature de la collection sur les paramètres évalués.

5.3.1 Le jeu de données BigDataSet

Produit dans le cadre du projet RARE [RARE 2007], le jeu de données "BigDataSet" a été obtenu à partir d'une analyse statistique sur des données collectées à partir du système Gnutella [Gnu 2007] et des données de la collection TREC [TREC 2008], ce qui nous permet de réaliser des simulations dans des conditions réelles. BigDataSet est composé de 25000 documents et de 5000 requêtes répartis sur 500 pairs [Mghirbi 2010b].

Les données(documents) et les requêtes sont fournies sous forme de deux fichiers, de définition de données et de requêtes, au format XML, dont les structures seront détaillées dans la section 5.4.1.

5.3.2 Le jeu de données Music DataSet

En vue d'avoir une collection spécialisée, dans un domaine de recherche d'information spécifique pour une communauté particulière, nous avons opté d'utiliser un corpus spécialisé (Musique MP3). Les données de ce corpus ont été collectées à partir du site web allmusic [allmusic 2012] qui se compose de différents genres musicaux notamment, le rap, le JAZZ, le POP et autres. Chaque genre musical contient des chanteurs, leurs albums et les paroles de leurs chansons. Nous avons dans notre corpus plus de 17000 documents et 2300 requêtes, distribués sur le réseau.

5.3.3 Le jeu de données Dmoz

DMOZ est l'un des plus grands et plus complets annuaires internationaux du web construit au sein du projet Open Directory [DMOZ 2011]. Il est construit et maintenu par une vaste communauté mondiale d'éditeurs bénévoles. Fondé en Californie en juin 1998 par Rich Skrenta et Bob Truel sous le nom de Newhoo, il a été racheté par Netscape en 1998 qui continue de faire fonctionner le site gratuitement. Nous avons sélectionné un sous-ensemble de l'annuaire web DMOZ [DMOZ 2011]. Il contient de nombreux sujets (4246) regroupés par catégories qui classent les sites web des liens avec leurs descriptions.
L'ensemble des données utilisées est constitué de 28128 documents, 3375 requêtes distribuées sur 810 pairs.

5.4 Processus d'évaluation expérimentale

Notre démarche d'évaluation expérimentale doit prendre en compte les points suivants :

1. Utiliser une collection centralisée et observer l'impact des différentes méthodes de distribution et de réplication sur les résultats d'évaluation.
2. Utiliser deux ou plusieurs collections de test (spécialisée ou générale) et observer l'impact des données sur les résultats.

3. S'assurer que les conditions réelles sont prises en considération. L'étude expérimentale peut adapter les ressources à une grande variété de scénarios de recherche.
4. Procéder à des essais multiples afin de valider expérimentalement les résultats obtenus sans être sujet au hasard.

Tenant compte de ces différents points, nous avons conçu un prototype de simulateur qui reprend les différentes étapes nécessaires à l'évaluation de nos approches à travers la simulation. Ce prototype est décrit par la Figure 5.1.

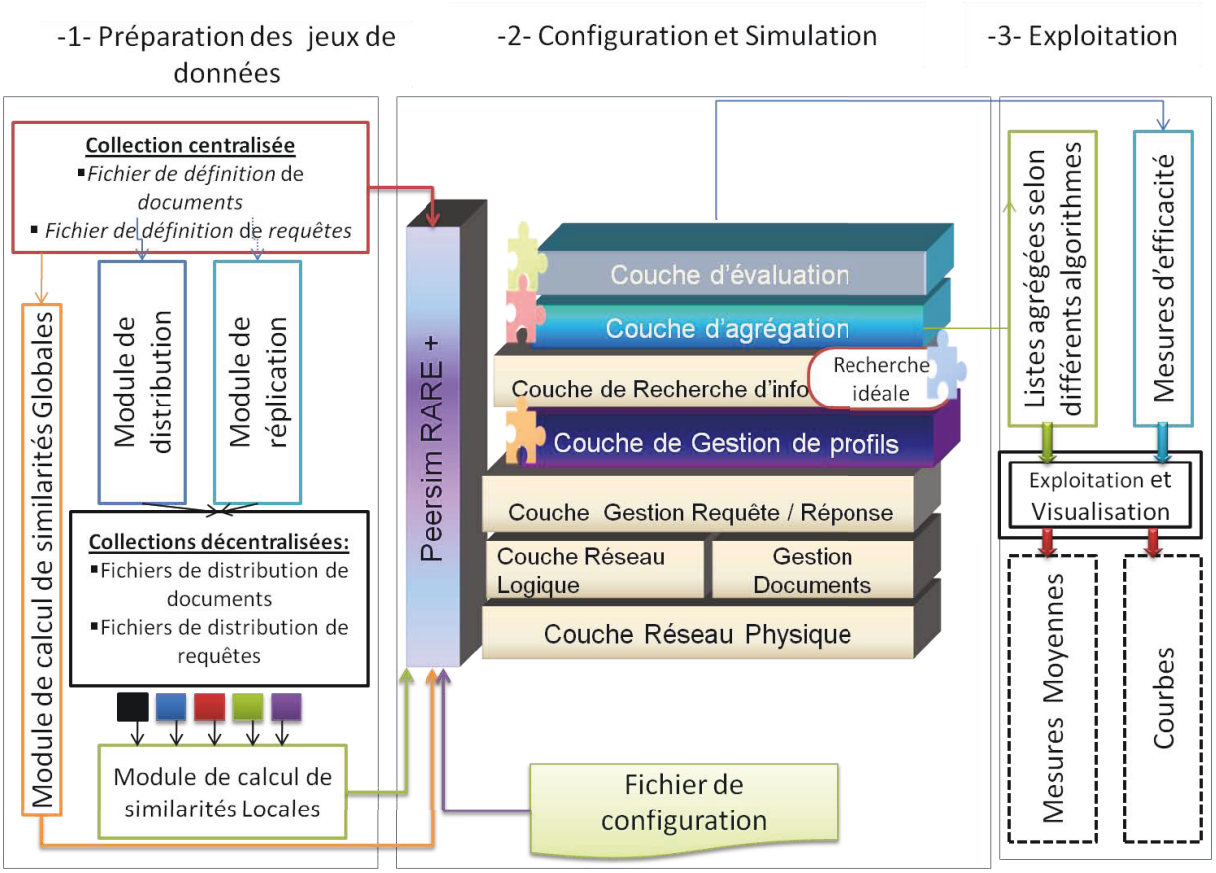


FIGURE 5.1 – Processus d'évaluation

Nous identifions sur la Figure 5.1 trois phases principales de notre processus d'évaluation :

- La préparation des jeux de données : qui consiste à mettre les collections dans un format accepté par le simulateur, à les distribuer et enfin à calculer les similarités entre les documents et les requêtes pour remplacer les jugements de pertinence.
- La configuration du simulateur et la simulation proprement dite avec la génération des listes agrégées selon les différents algorithmes et les mesures

d'efficacité par requête.
– L'exploitation des résultats de simulation par la génération des mesures agrégées par plusieurs requêtes dans des fichiers textes et de courbes comparatives.

L'ensemble de ces phases est détaillé dans les sections suivantes.

5.4.1 Préparation des jeux de données

Le simulateur utilise un jeu de données centralisé basé sur deux fichiers de définition de documents et de requêtes, de format XML. La préparation des jeux de données exige l'utilisation d'un générateur de fichier XML si les données utilisées sont dans un format texte.

Chaque fichier XML, utilisé à l'entrée du simulateur, contient l'identifiant des documents (respectivement des requêtes) et l'ensemble de leurs termes (voir Figure 5.2). Pour les requêtes, le fichier comporte également comme le montre la Figure 5.3 un ensemble de réponses identifiées à partir du fichier de documents de la collection centralisée avec leurs similarités. Le calcul des similarités entre les réponses et les requêtes repose sur un module de calcul de similarité (voir Figure 5.1). La fonction que nous avons utilisée pour classer les réponses à une requête d'une façon globale est la fonction cosinus.

Afin de construire des collections décentralisées à partir de la collection centralisée, nous avons intégré un module pour la distribution et la réplication des données (documents et requêtes). Ce module est configurable, il offre à l'utilisateur la possibilité de définir certains paramètres tels que la taille du système, les méthodes de distribution des documents et des requêtes et les taux de réplication, etc.

Il suffit d'introduire le nombre de pairs sur lequel nous désirons répartir ou répliquer les données, et les modèles de distribution ou de réplication, pour obtenir des collections décentralisées. Les méthodes de distribution, que nous avons utilisées à ce niveau, sont respectivement : les méthodes uniforme, aléatoire et zipF de nature statistique et la méthode par spécialisation des nœuds de nature sémantique. En ce qui concerne les méthodes de réplication, les méthodes uniformes, aléatoires et ZipF ont été développées selon différentes constantes de replication.

En ce qui concerne le calcul de similarité au niveau de chaque pair, Peersim-RARE utilise seulement des similarités globales pour faire la recherche d'information et le routage et par conséquent l'agrégation des résultats. Or, en réalité, ces similarités ne reflètent pas la vraie distance entre les requêtes et les réponses au niveau des collections locales. Partant de cette limite, nous avons intégré un module de calcul de similarités locales, qui tient compte du contenu des collections sur chaque pair. Nous avons également pensé à utiliser différentes méthodes de recherche d'information pour rechercher dans les collections locales. Actuellement, quatre méthodes de similarité ont été implémentées, notamment, les méthodes cosinus, Jaccard, overlap et dice [Egghe 2010]. Nous avons étudié le cas où la même méthode de RI est utilisée sur toutes les collections locales, et le cas où différentes méthodes sont utilisées,

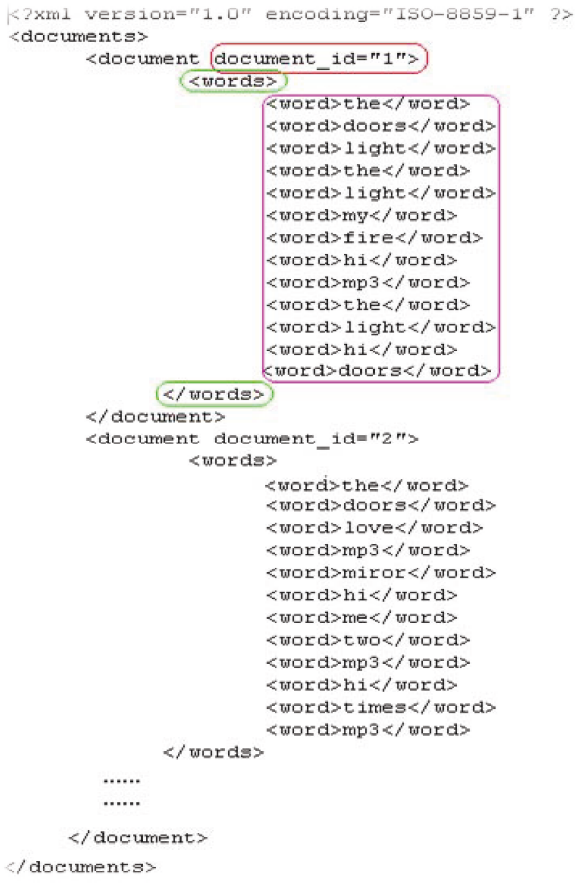


FIGURE 5.2 – Aperçu sur la structure de documents

```
<queries>
  <query query_id="1">
    <words>
      <word>ann</word>
      <word>her</word>
      <word>lee</word>
      <word>hate</word>
      <word>womack</word>
      <word>i</word>
    </words>
    <responses>
      <response document_id="6" similari="0.03480423" />
      <response document_id="1125" similari="0.104110725" />
      <response document_id="3528" similari="0.045775957" />
      <response document_id="3560" similari="0.07617463" />
      <response document_id="7032" similari="0.25667638" />
      <response document_id="7063" similari="0.09809306" />
      <response document_id="7073" similari="0.09809306" />
      <response document_id="7076" similari="0.09809306" />
      <response document_id="7079" similari="0.19612817" />
      <response document_id="7082" similari="0.079022005" />
      <response document_id="7085" similari="0.09809306" />
      <response document_id="7088" similari="0.09809306" />
      <response document_id="7091" similari="0.09809306" />
      <response document_id="7349" similari="0.04569112" />
      <response document_id="7368" similari="0.04965409" />
      <response document_id="7371" similari="0.04355295" />
      <response document_id="7373" similari="0.07114095" />
      <response document_id="7377" similari="0.07390752" />
      <response document_id="7381" similari="0.0691758" />
      <response document_id="7393" similari="0.0317729" />
      <response document_id="7441" similari="0.045899857" />
      <response document_id="7463" similari="0.0329377" />
      <response document_id="7466" similari="0.056829054" />
```

FIGURE 5.3 – Aperçu sur la structure de requêtes

selon les collections, afin d'introduire de l'hétérogénéité dans le système.

5.4.2 Configuration et Simulation

Le fichier de configuration est une composante indispensable au démarrage des simulations. Cette section sert d'exemple pour détailler le contenu d'un fichier de configuration et expliquer le fonctionnement de PeerSim-RARE +. Afin de démarrer une simulation, un nombre de paramètres relatifs au réseau et à la simulation doivent être fixés notamment, le nombre de pairs et la durée de la simulation matérialisée par le nombre de cycles par simulation. Ensuite, il y a lieu de décrire l'ensemble des protocoles à utiliser (voir Exemple de la section5.4.2, les Lignes 10–29), à savoir les protocoles de gestion du réseau logique, des documents, de la résolution des requêtes (ou protocole de gestion requête réponse), de gestion de la base des connaissances (profils), le protocole de Recherche d'Information (RI) et le protocole d'agrégation de résultats. Chaque protocole peut appeler d'autres protocoles pour exercer une partie de sa fonction.

L'initialisation de l'ensemble de ces protocoles s'effectue dans la partie *initializers* (Exemple de la section 5.4.2, les Lignes 31–48). Les initialisateurs permettent également d'injecter dans le simulateur, les jeux de données nécessaires à la simulation, notamment le fichier de définition (Exemple, sec 5.4.2, les lignes 37,42), et de distribution de documents (respectivement des requêtes) (Exemple, sec, 5.4.2 les Lignes 38 et 43) et les fichiers contenant la base de connaissances nécessaire à l'agrégation des résultats (Exemple, sec 5.4.2, ligne 47). Le simulateur utilise de même les composantes "Dynamics" afin d'introduire des événements qui rendent la simulation dynamique et ce, à n'importe quel cycle.
À la fin de chaque cycle de simulation, des "observateurs" sont déclenchés afin de visualiser les résultats. Dans notre cas, il s'agit de présenter les résultats d'agrégation en appliquant les différentes méthodes indiquées dans le fichier de configuration, notamment le résultat de l'agrégation selon PBA, BC, RR, Dcos (cosinus distribué) et LP, comme présenté dans les protocoles d'agrégation (Exemple de la section 5.4.2, Lignes 23–27).
L'observateur qui permet de visualiser ce résultat s'intitule *AggregationObserver* (Exemple, sec 5.4.2, la Ligne 51).

Une instance du fichier de configuration que nous avons utilisé, est explicité dans l'exemple suivant :

Exemple 2: Exemple d'un fichier de configuration

```
#_____ Paramètres du réseau _____
1.simulation.cycles 1000
2.overlay.size 810
3.random.seed 18
4.requete.classe com.get.peersim.architectureGenerique.
coucheRequeteReponse.GeneraleRequete
6.reponse.classe com.get.peersim.architectureGenerique.
coucheRequeteReponse.GeneraleReponse
#_____ Protocoles _____
# Gestion du Reseau Logique : gestionResauLogique
10.protocol.0 com.get.peersim.architectureGenerique.coucheReseauLogique.
ProtocolReseauLogique
# Gestion des documents : gestionDocuments
12.protocol.1 com.get.peersim.architectureGenerique.coucheGestionDocuments.
ProtocolPairDocument
# Gestion des requêtes et des réponses : gestionRequeteReponse
14.protocol.2 com.get.peersim.architectureGenerique.coucheRequeteReponse.
ProtocolGestionRequeteReponse
# Gestion des profils : gestionProfils
16.protocol.3 com.get.peersim.architectureGenerique.coucheGestionProcols.
ProtocolPairProfile
# Recherche d'Information : RI
18.protocol.4 com.get.peersim.SystemesRechercheP2P.IdealRouting
19.protocol.4.gestionRequeteReponse 2
20.protocol.4.gestionDocuments 1
21.protocol.4.gestionReseauLogique 0
```

#_____ **Protocoles** _____

 # Agrégation de résultats
23.protocol.5 com.get.peersim.aggregationLayer.PBA
24.protocol.5 com.get.peersim.aggregationLayer.BC
25.protocol.5 com.get.peersim.aggregationLayer.RR
26.protocol.5 com.get.peersim.aggregationLayer.Dcos
27.protocol.5 com.get.peersim.aggregationLayer.LP
28.protocol.5.gestionProfil 3
29.protocol.5.RI 4
#_____ **Initialisateurs** _____
31.init.0 peersim.dynamics.WireRegularRandom
32.init.0.degree 10
33.init.0.protocol 0
34.init.0.pack
#Initialiser les documents
36.init.1 com.get.peersim.architectureGenerique.coucheGestionDocuments.
InitPairDocument
37.init.1.documentsDefinitionFile *src/com/get/peersim/config/document*
_definition.xml
38.init.1.documentsDistributionFile *src/com/get/peersim/config/document*
_distribution.xml
39.init.1.protocol 1
#Initialiser les requêtes
41.init.2 com.get.peersim.architectureGenerique.coucheRequeteReponse.
InitRequeteReponse
42.init.2.requetesDefinitionFile *src/com/get/peersim/config/*
query$_d$efinition.xml
43.init.2.requetesDistributionFile *src/com/get/peersim/config/*
query$_d$istribution.xml
44.init.2.protocol 2
#Initialiser les profils
46.init.3 com.get.peersim.architectureGenerique.coucheGestionProfils.
InitPairProcol
47.init.3.queriesTermsConceptsDirectory *src/com/get/BK*
48.init.3.protocol 3
#_____**Observateurs** _____
#afficher des informations sur le routage des requêtes
51.observer.0 com.get.peersim.architectureGenerique.AggregationLayer.
AggregationObserver
52.observer.0.protocol 5
53.observer.0.from 1001

Suite à cette configuration, la simulation doit concerner les niveaux suivants :

- **La Recherche d'Information :** Dans PeerSim-RARE, la simulation au niveau de la couche de RI concerne l'aspect "routage des requêtes". Cependant, l'un des objectifs du simulateur PeerSim-RARE + est d'évaluer les algorithmes d'agrégation en neutralisant la phase de sélection de pairs pertinents. C'est d'ailleurs l'objectif de la méthode *idleRouting* que nous avons intégrée aux systèmes de recherche P2P. Ainsi, PeerSim-RARE+ simule un routage "idéal" qui permet de router les requêtes aux pairs pertinents qui sont préalablement connus à partir de fichier de définition de la requête (là où on a les bonnes réponses aux requêtes) et du fichier de distribution de documents (où on a les pairs qui hébergeant les documents pertinents).
- **la Gestion de profils :** Au niveau de cette couche, PeerSim-RARE+ permet la recherche des profils similaires à la requête, et qui contiennent les documents et les pairs contributeurs à la réponse. Le calcul des scores des documents et des pairs en se basant sur les profils sélectionnés est également effectuée au sein de cette couche.
- **L'agrégation des résultats :** par simulation, nous sommes en mesure d'appliquer différents algorithmes d'agrégation. Ainsi, nous pouvons obtenir les listes finales ordonnées à partir des résultats des pairs sélectionnés dans la phase de routage.
- **L'évaluation de l'efficacité des algorithmes d'agrégation :** Le simulateur permet également de calculer les différentes métriques pour évaluer l'efficacité des algorithmes d'agrégation. Les mesures que calcule le simulateur PeerSim-RARE sont le rappel et le nombre de messages. Cependant, étant donné que nous agrégeons suite à une stratégie de routage qui ramène tous les documents pertinents, le recours à la mesure rappel n'aura plus de sens. En effet, nous sommes plutôt orientés précision que rappel. Pour ces raisons, nous avons intégré dans le simulateur les métriques d'efficacité suivantes : la précision à un rang donné, la précision moyenne, la précision relative et les mesures qualitatives que nous avons définies (voir Chapitre 4). Par la suite, le simulateur permet d'exporter, dans un fichier spécifique, les résultats des métriques calculées dans un fichier .csv.

5.4.3 Visualisation et exploitation des résultats

Le problème avec les mesures calculées par simulation, est qu'elles ne sont pas forcément interprétables puisqu'il s'agit de mesures individuelles (par requête). Pour pallier ce problème, nous avons intégré un module pour calculer de manière globale ces mesures (pour 100, 200, 300 requêtes, etc), de manière à comparer simultanément un ensemble de mesures. Ces mesures globales sont générées dans un fichier spécifique et visualisées par des courbes (tracées avec Gnuplot) comparatives.
Ainsi, nous avons proposé un module implémenté en Java pour pouvoir exploiter les résultats, métrique par métrique, et pour différentes valeurs de rang de précision

(k de 1 à 6). Il suffit de choisir le chemin des fichiers .csv générés par le simulateur, de choisir la métrique d'évaluation à utiliser et à quel rang, puis de valider pour générer le fichier d'évaluation synthétique et les courbes de performance. Ces courbes seront présentées dans les résultats des scénarii réalisés dans la section suivante.

5.5 Etude expérimentale

Dans les tests que nous effectuons, nous cherchons à valider les défis annoncés dans cette thèse :

1. Prendre en compte l'hétérogénéité des pairs du système :
   (a) Selon la taille des collections ;
   (b) Selon leur contenu (thématique spécialisée ou aléatoire) ;
   (c) Selon les modèles de Recherche d'Information (différents modèles selon les pairs).
2. Construire une base des connaissances par pair en adoptant des techniques d'apprentissage
3. Maintenir la qualité de l'agrégation même en cas d'évolution du système.

Aussi, nous avons défini un ensemble de scénarii expérimentaux construits à partir des cinq paramètres suivants :

- 1- Jeux de données utilisé : spécialisé ou non spécialisé.
- 2- Modèle de distribution utilisé : taille des collections sur les pairs uniforme ou non, contenu des collections spécialisé ou non.
- 3- Réplication : oui, non.
- 4- Modèle de recherche d'informations sur les pairs : même pour tous ou différents.
- 5- Evolution des bases de connaissances : oui, non.

Dans ce chapitre, nous ne tenons pas compte du dernier point cité, à savoir la question de la fraîcheur des bases. En effet, ceci fera l'objet du chapitre suivant.

Il convient de mentionner que dans cette série d'expériences, nous avons essayé de tester l'efficacité de l'algorithme d'agrégation à base de profils (PBA) [Mghirbi 2010a] par rapport à un ensemble d'algorithmes (essentiellement à base de rangs et à base de profils) et ce en comparant par rapport à un référentiel unique qui est le classement de la collection centralisée. Ainsi, les résultats considérés pertinents sont ceux obtenus par un système centralisé.
Ceci implique que les valeurs que nous obtenons pour les métriques de RI (P@k, MAP, RP, etc.) sont relatives par rapport à ce système. Ainsi, si un système d'agrégation atteint une valeur de P@k égale à 1, ceci indique que le système donne le même résultat qu'un système centralisé.

Chaque scénario essaiera de répondre simultanément aux cinq questions posées (hormis la question de fraîcheur de la base des connaissances). Ainsi, pour analyser

l'impact de l'hétérogénéité des collections (qui est une condition naturelle dans les systèmes P2P) sur les algorithmes d'agrégation. Pour ce faire, nous appliquons différentes méthodes de distribution pour introduire une hétérogénéité de contenu et de taille sur les collections. De même, nous procédons à introduire un cas d'hétérogénéité de méthodes de RI par utilisation de plusieurs méthodes de filtrage sur les collections.

Avant de procéder aux tests, nous tenons à présenter une idée sur la mise en place du processus d'apprentissage qui est indispensable à l'exécution de PBA.

5.5.1 Apprentissage

Pour évaluer l'algorithme PBA, nous avons appris les profils utilisateurs selon la stratégie d'apprentissage basée sur la validation croisée selon les proportions suivantes : 2/3 des requêtes sur chaque pair sont utilisées pour l'apprentissage et la 1/3 restant pour l'évaluation. Cette stratégie consiste à faire varier l'ensemble des requêtes d'apprentissage servant de base pour la construction de l'ensemble des profils utilisateurs en alternant avec l'ensemble des requêtes de test.

La phase d'apprentissage est basée sur les étapes décrites dans la couche de gestion de profils (voir Chapitre 3). Les profils sont extraits à partir de fichiers journaux en utilisant l'Analyse de Formelle de Concepts, et plus précisément, en utilisant l'algorithme de Godin [Godin 1995], implémenté dans la plate-forme Galicia V-3. [Valtchev 2003]. Nous rappelons que la base des connaissances d'un pair est créée à partir de son fichier journal local sans coopération explicite avec d'autres pairs dans le système. Elle dépend des requêtes passées et de leurs réponses relatives. Ces connaissances locales contiennent des informations implicites des pairs qui ont été sollicités pour répondre aux requêtes passées du pair initiateur.

Nous présentons dans ce qui suit l'ensemble des scénarii réalisés et les conclusions associées.

5.5.2 Premier scénario de test : distribution aléatoire

Dans ce premier scénario, nous avons utilisé le jeu de données Dmoz avec les paramètres associés (Voir Tableau 5.2) :

Dans tous les tests réalisés, le scénario d'apprentissage est appliqué de la façon indiquée dans la partie 5.5.1. Le nombre de requêtes apprises correspond au paramètre "taille_KB" du tableau 5.2 alors que le nombre de requêtes de test correspond à "taille-test".

Ce scénario correspond aux conditions suivantes :

- Jeux de données : Dmoz qui est une collection spécialisée constituée de 16 grandes catégories comportant plus de 42000 sous catégories.
- Distribution du nombre de documents sur les pairs : nous avons choisi la distribution ZipF, ainsi, nous allons aboutir à des tailles hétérogènes sur des groupes de pairs.

Collection	Dmoz : collection générale à plusieurs catégories spécialisées [DMOZ 2011]
Nbre_documents	28128
Nbre_requêtes	3375/avec replication 5400
Nbre_pairs	810
taille_KB en requêtes	3600
Taille_test en requêtes	1800
Modèle de RI du référentiel (calcul de similarité dans la collection centralisée)	Cos
Modèles de RI dans les collections locales	Cos

TABLE 5.2 – Paramètres relatifs au jeu de données Dmoz

- Distribution de contenu : Nous avons appliqué la méthode aléatoire pour répartir le contenu sur les pairs. Ainsi, aucune sémantique ne réunit les documents d'un même pair.
- Méthodes de RI : Dans cette première série d'expériences, nous avons utilisé la même méthode de RI sur toutes les collections. C'est d'ailleurs, la même méthode (i.e, la méthode cosinus), appliquée au niveau de la collection globale, considérée comme référentiel pour le classement.
- Réplication de documents : Il convient de mentionner qu'aucune réplication n'a été appliquée sur les documents. En revanche, les requêtes sont répliquées selon la loi zipF.

Ce scénario (S1) correspond à un cas peu favorable pour PBA, car il ne considère aucune hétérogénéité entre les collections des différents pairs (même modèle de RI, même pertinence des pairs relativement à une requête). Les résultats attendus correspondent donc à une borne inférieure des performances de PBA.
Les Figures 5.4, 5.5 et 5.6 montrent que PBA a une efficacité quantitative qui est, au pire des cas, au même niveau que les approches à base de rangs. Cependant, l'approche LP dont le principe est expliqué dans la Section 4.5.3.3 du chapitre 4, donne de bons résultats face à PBA. En efficacité qualitative, nous constatons, à la lecture de la Figure 5.7, que PBA a un comportement très proche de celui des méthodes à base de rangs. Ceci est prévisible, puisque, si les collections, ne sont pas suffisamment spécialisées, PBA va recevoir des réponses qui proviennent potentiellement de tous les pairs et il va recommender sans préférence tous les pairs qui ont bien répondu. Ainsi, il aura un comportement comparable à celui des méthodes à base de rangs.

5.5.3 Deuxième scénario : Impact de la réplication sur PBA

Les mêmes conditions du premier scénario sont reproduites dans le présent scénario (S2), mais avec une réplication de données selon la loi ZipF. Nous avons appliqué un très faible taux de réplication sur les documents selon leur pertinence

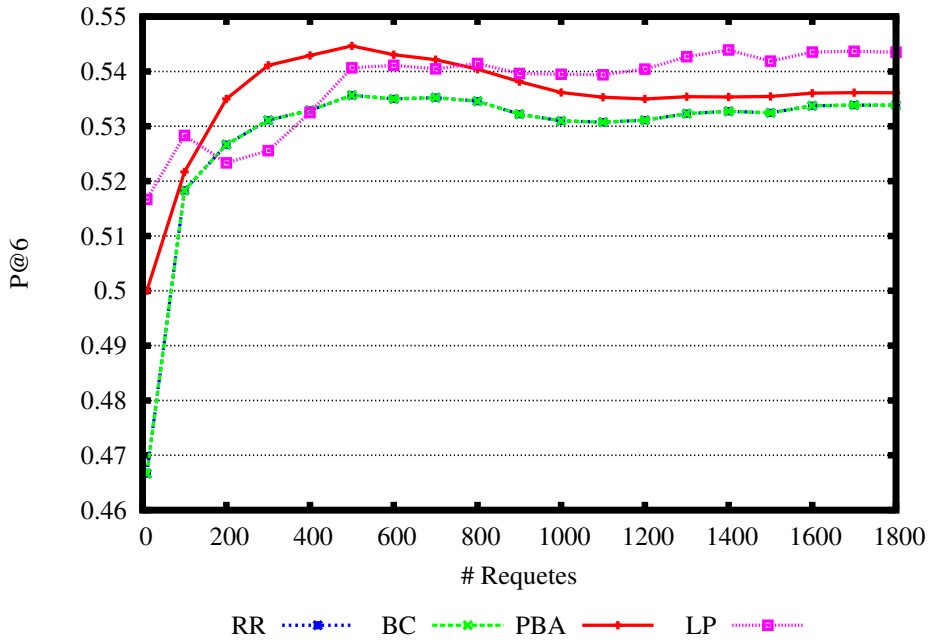


FIGURE 5.4 – S1 : D-Aléatoire Precision moyenne @ 6

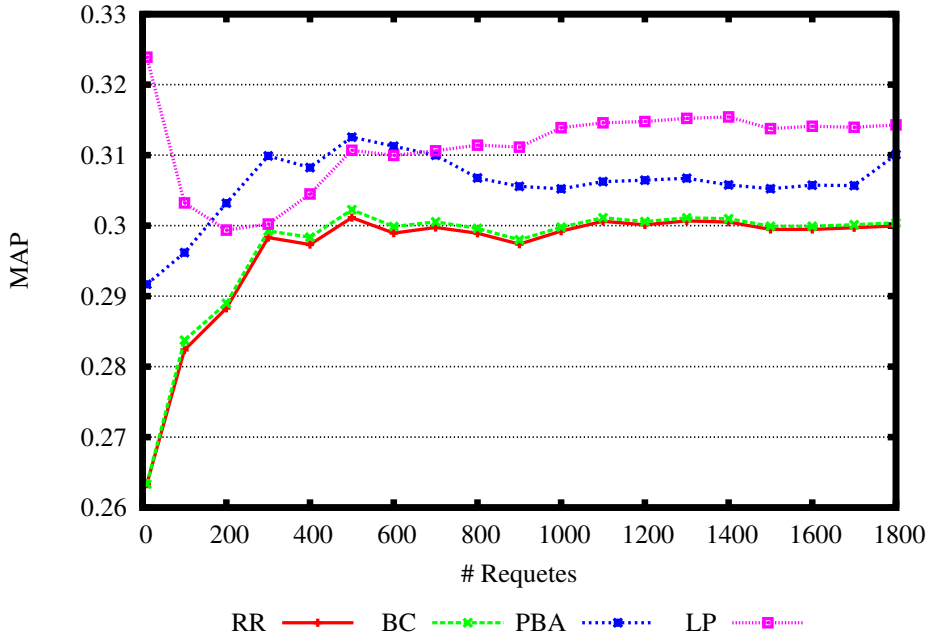


FIGURE 5.5 – S1 :D-Aléatoire :Mean Average Precision

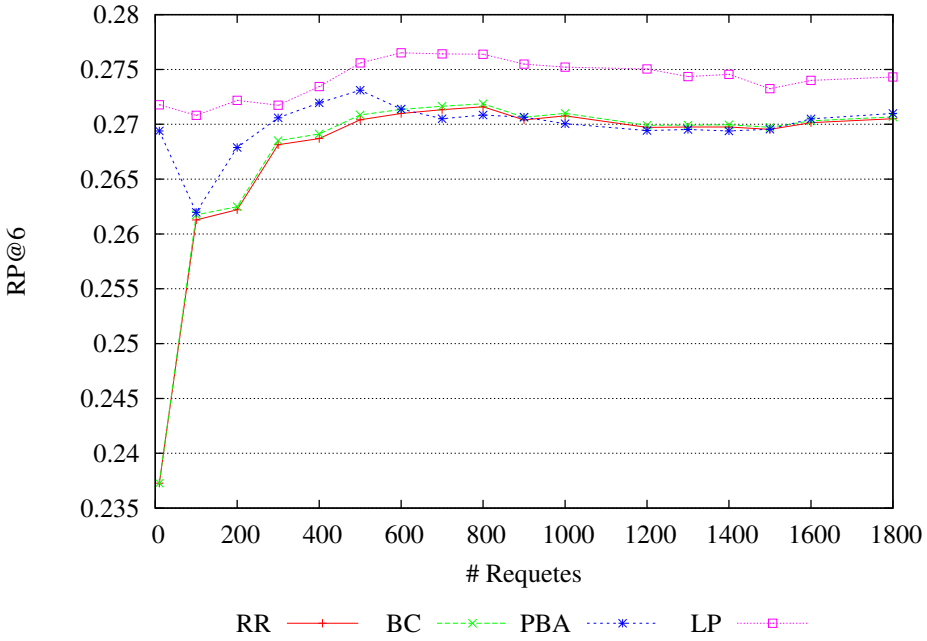


FIGURE 5.6 – S1 :D-Aléatoire :Mean Average Precision

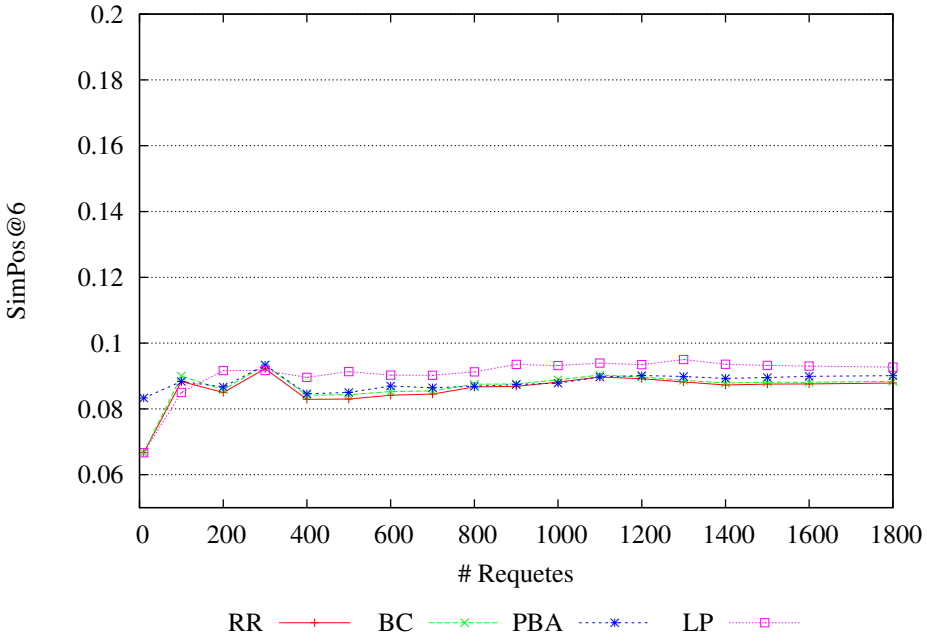


FIGURE 5.7 – S1 : D-Aléatoire :taux de positions similaires

relativement aux requêtes. Cette pertinence est calculée en se basant sur la similarité cosinus entre documents et requêtes.
Un aperçu sur les résultats de ce scénario est donné à travers la Figure 5.8. Il se base sur la métrique MAP.

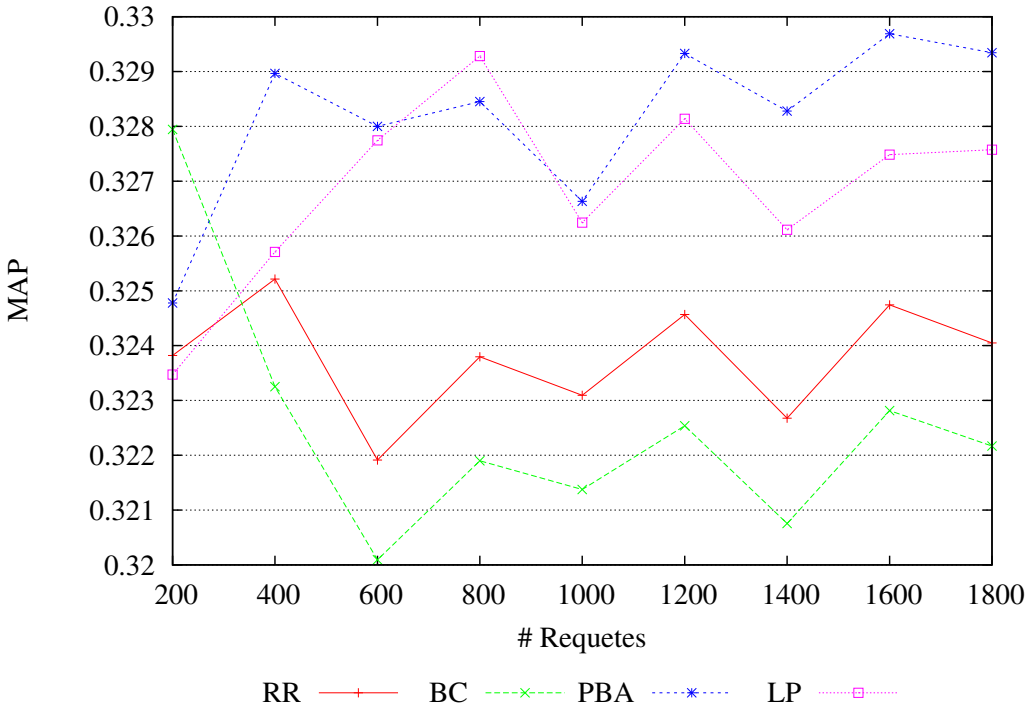


FIGURE 5.8 – S2 :D-Aléatoire-Replication : Mean Average Precision

Les résultats présentés dans la Figure 5.8 montrent que la réplication ne change pas véritablement le comportement des algorithmes. Cependant, si on compare la qualité globale des algorithmes par rapport aux résultats du modèle distribué aléatoire sans réplication (voir la Figure 5.5), nous remarquons qu'en absolu, une amélioration des valeurs des différentes courbes est détectée.
Théoriquement, la réplication est une condition favorable aux méthodes d'agrégation à base de rangs tels que BC, ce qui explique pratiquement l'amélioration au niveau de leur efficacité en termes de MAP. Notre algorithme a vu également une amélioration légère de son efficacité et excède LP bien que nous sommes toujours situés dans un cadre défavorable.

5.5.4 Troisième scénario : distribution sémantique

Dans les scénarii précédents, nous avons utilisé des collections hétérogènes en taille, la même méthode de RI et une distribution statistique des documents. Dans le présent scénario (S3), nous comptons créer des collections en se basant sur une distribution sémantique des documents. En effet, étant basé sur la collection DMOZ

[DMOZ 2011], nous avons remarqué que cette dernière fournit des documents réunis en 16 catégories comportant plus de 42 000 sous catégories. Nous avons considéré un sous ensemble de ces catégories pour construire des pairs spécialisés mais en thèmes un peu généraux (une catégorie art peut grouper plusieurs sous catégories qui peuvent être différentes).
Plus de 10 documents sont affectés à chaque pair en se basant sur cette distribution sémantique. Par conséquent, notre benchmark respecte le critère d'hétérogénéité du contenu entre pairs.

Aucune réplication n'est utilisée pour les documents. Les requêtes sont, au contraire, aléatoirement distribuées et répliquées. Par réplication, nous arrivons à un nombre total de 5400 requêtes.

Les algorithmes d'agrégation ont été comparés en se basant sur les métriques de RI. Les Figures 5.9 et 5.10 présentent les résultats de la métrique $P@k$ pour les valeurs de k=1 et k=6.
Tester la précision pour une valeur de $k = 1$ ($p@1$) n'est pas la finalité en elle-même, puisqu'elle ne ne fournit pas une évaluation absolue (par rapport à un jugement de pertinence), mais une évaluation relative par rapport à notre classement de référence qui est le classement de la collection centralisée. A ce niveau, il est essentiel de montrer que notre approche donne la meilleure efficacité, même pour le premier rang et maintient ce niveau de compétitivité jusqu'à la sixième position ou rang ($k = 6$), ce qui représente en moyenne la moitié des réponses à une requête dans notre benchmark.

L'oscillation que nous voyons dans la courbe de précision pour k=1 ($P@1$) était prévisible puisque cette métrique donne un résultat de pertinence binaire (soit pertinent en première position ou non pertinent).

Comme le présente la Figure, les courbes des autres méthodes sont presque entièrement au dessous des courbes de l'algorithme PBA pour toutes les métriques testées.

Dans la courbe de précision $P@6$ de la Figure 5.10, nous avons ajouté à la comparaison, les deux algorithmes LP et cos brut (sur chaque pair donc calculé de manière distribuée) ou Dcos. Les courbes d'efficacité montrent la réussite de PBA par rapport à LP qui, comme présenté dans la section 4.5.3.3, est une méthode statistique basée sur le contenu des logs files.
La courbe de *Cos* a montré plus d'efficacité par rapport à PBA. Ce résultat était prévisible puisque toutes les collections utilisent le même modèle de RI basé sur cosinus et c'est le modèle utilisé par le référentiel de comparaison.

Les résultats des métriques $SimPos@k$ et $DeltaNoise@k$ sont représentées sur les figures 5.13 et 5.14. La métrique $SimPos@k$ montre que le taux de résultats qui ont gardé leur position par rapport au classement de référence (le classement centralisé) donne de meilleures performances dans notre algorithme par rapport aux autres.
Le gain introduit par notre méthode en lui-même n'est pas très important (max rang 11 % centralisée) mais c'est un indicateur d'efficacité qualitative par rapport

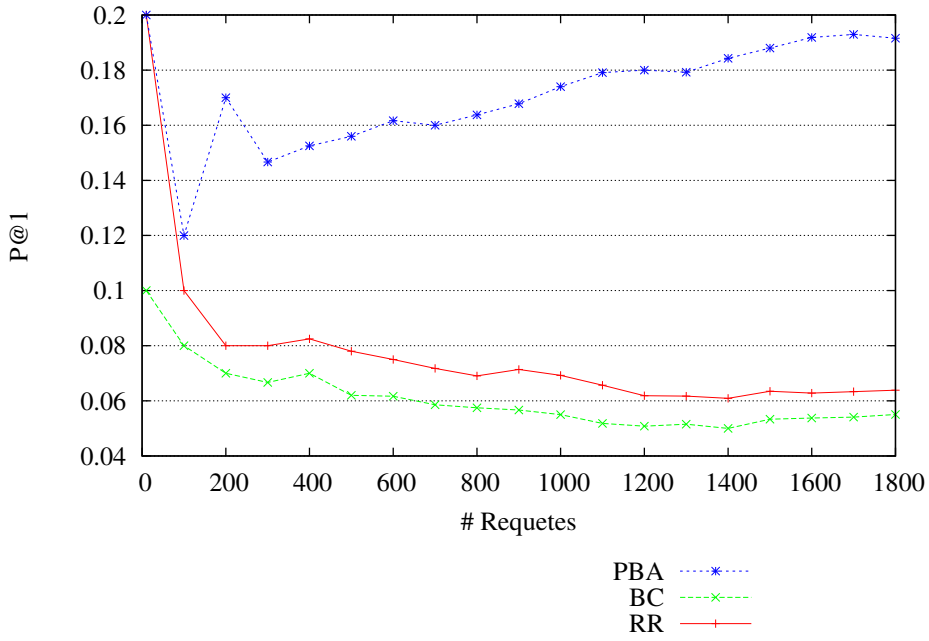


FIGURE 5.9 – S3 :Précision moyenne($k = 1$)

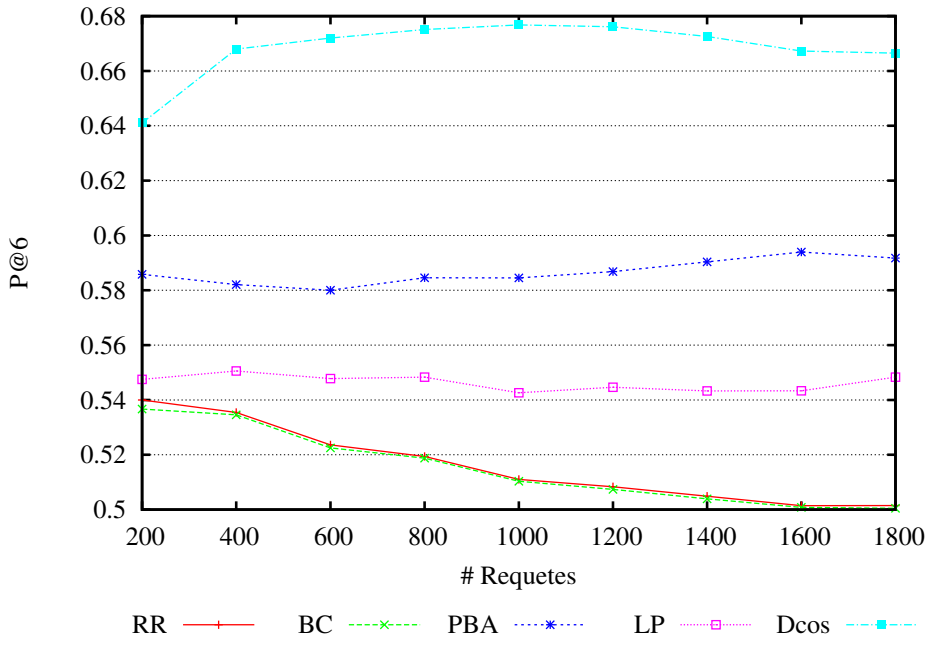


FIGURE 5.10 – S3 : Précision moyenne (k=6)

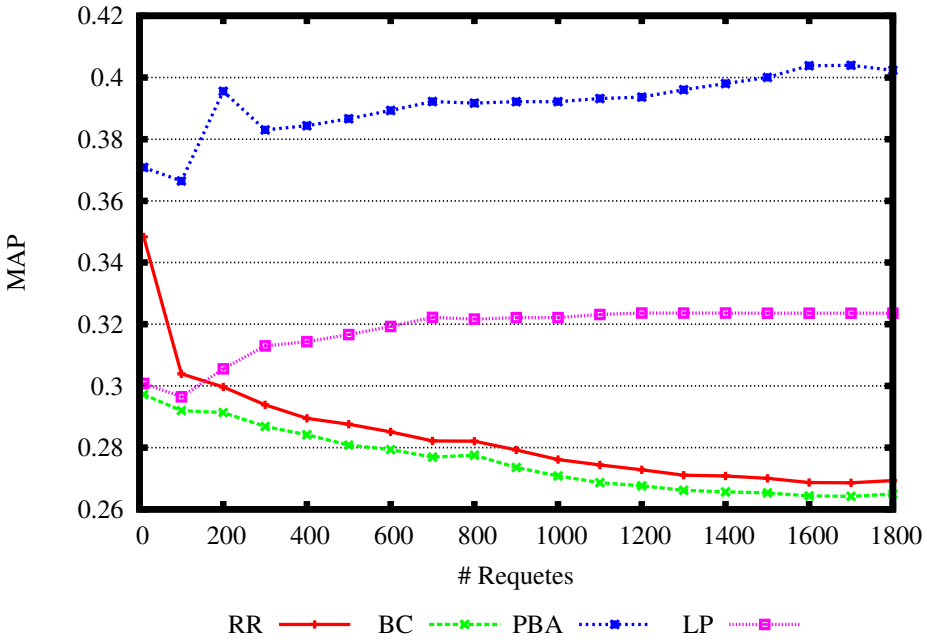


FIGURE 5.11 – S3 : Mean Average Precision

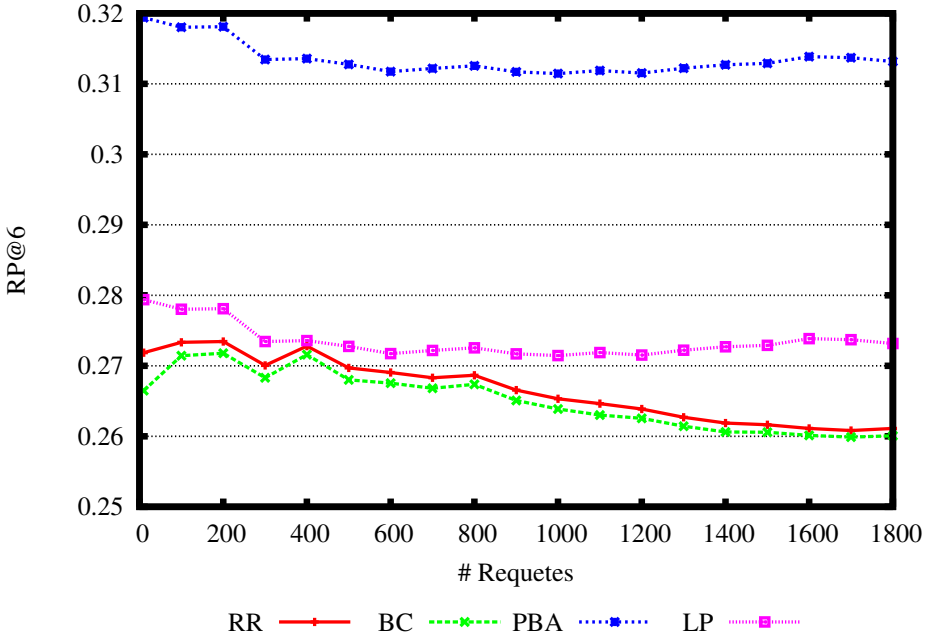


FIGURE 5.12 – S3 : Relative Precision (k=6)

aux autres méthodes.
La divergence de notre algorithme par rapport au classement de référence était prévisible parce que, dans de nombreux cas, lorsque les scores calculés avec notre méthode de score sont égaux, notre algorithme classe les résultats selon la stratégie "premier arrivé, premier servi" par rapport à l'arrivée des pairs.

La métrique $DeltaNoise@k$ (voir Figure 5.14) qui vise à trouver le bruit de rangs des résultats par rapport au classement de référence, montre pour les cinq premiers résultats que l'algorithme PBA donne le minimum de bruit comparé à l'état de l'art.

Les résultats de nos expérimentations assurent que l'approche, que nous avons proposée, donne une bonne performance pour l'ensemble du système. Toutefois, aucune idée n'est donnée sur les comportements individuels des pairs. Pour cette raison, nous avons ajouté une nouvelle expérimentation pour la métrique $p@k$ afin de déceler les comportements des pairs locaux. La Figure 5.15 compare sur la base de la métrique citée, notre approche à l'approche RR (qui donne les meilleurs résultats pour l'état de l'art).
Ainsi, nous avons pu montrer que les résultats que donne notre approche sont mieux que ceux de RR, malgré la contrainte de n'avoir qu'une vision très partielle du système. En fait, le pourcentage de pairs où RR donne les meilleurs résultats ne dépasse pas le 1%. Ce pourcentage peut être facilement expliqué par l'état des bases des connaissances des différents pairs, dans lesquelles, les pairs n'ont pas le même potentiel d'apprentissage.

Les résultats de nos expérimentations par rapport aux approches classiques à base de rangs peuvent être expliqués par la capacité de l'algorithme PBA à attribuer un coefficient de pondération privilégiant des pairs par rapport à d'autres. Ce facteur est important vu qu'il donne une préférence aux pairs pertinents plutôt que de les traiter d'une manière uniforme. Dans la littérature, les auteurs dans [Renda 2003] assurent que l'introduction des poids des collections peut améliorer nettement les résultats de ces méthodes d'agrégation ce que confirme nos résultats. Ils proposent de calculer ce poids en se basant soit sur un échantillon centralisé des statistiques soit sur un système d'évaluation de collections basé sur les jugements de pertinence. Ces stratégies ne correspondent pas à nos critères de conception initialement fixés (voir Chapitre 2), à savoir la décentralisation, l'autonomie, l'hétérogénéité et le passage à l'échelle, d'autant plus qu'il est difficile d'avoir des jugement de pertinence. En fait, PBA peut être vu comme un algorithme de classement autonome qui va au-delà d'une stratégie de pondération de pairs. Il fournit un moyen efficace d'évaluer l'importance d'un document par rapport à une requête sachant l'historique des requêtes similaires passées. Cette information est très importante car elle reflète les intentions de l'utilisateur quand il interagit avec des réponses retournées.

Ce que nous remarquons aussi à la lecture des résultats des différentes courbes, est que la sémantique introduite dans le contenu des collections n'a pas seulement

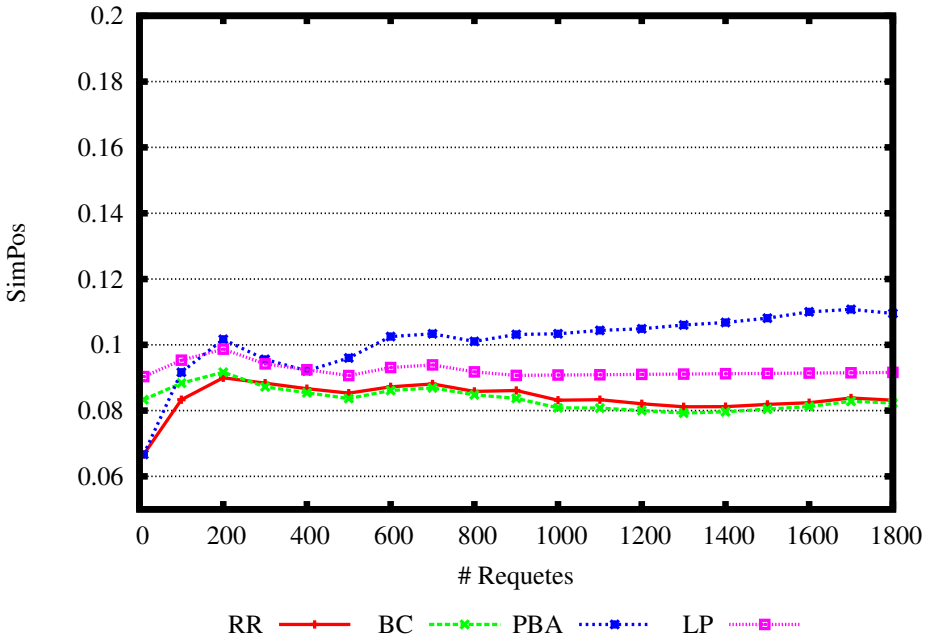


FIGURE 5.13 – S3 : taux de positions similaires-k=6

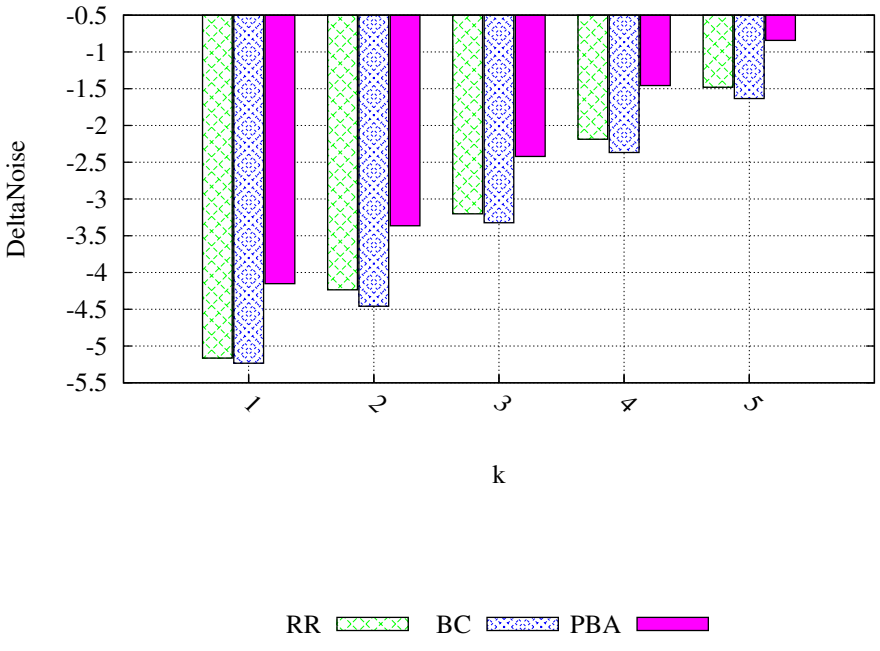


FIGURE 5.14 – S3 : $\Delta DeltaNoise@k$ (k=6)

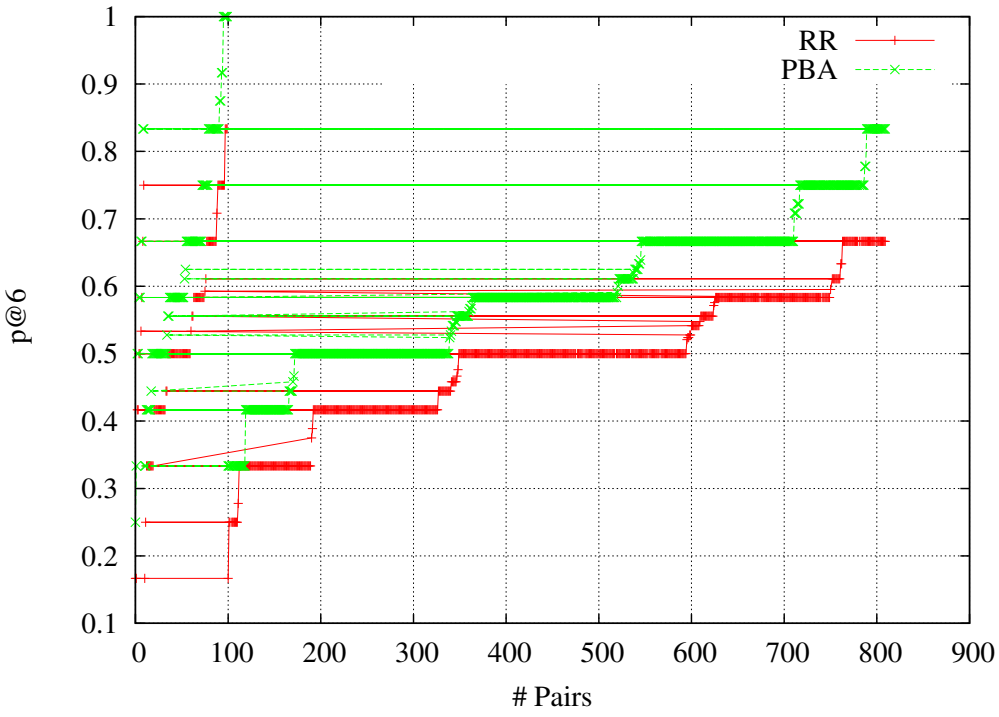


FIGURE 5.15 – taux de p@k par pair(k=6)

contribué à l'amélioration des valeurs d'efficacité de PBA par rapport aux scénarii précédents, mais a introduit une dégradation au niveau des approches classiques qui, comme prévu, ne s'adaptent pas bien à des contextes d'hétérogénéité.

5.5.5 Quatrième scénario de test : utilisation d'une collection de contenu général

Le but de cette expérimentation est de confirmer que notre algorithme peut donner des résultats acceptables par rapport aux méthodes d'agrégation classiques, même en étant dans un cadre défavorable. C'est la raison pour laquelle, nous présentons ces expérimentations pour le jeu de données BigDataSet, qui est très général en contenu et offre des conditions d'homogénéité des collections, côté taille, et modèle de recherche, et avec une distribution statistique des documents sur les différents pairs. Cela permet aussi de tester un autre jeu de données et montrer la reproductibilité de nos résultats sur des jeux de données différents. Les résultats de ces expérimentations sont illustrés sur les Figures 5.16 pour la mesure de précision et 5.17 pour la mesure qualitative relative au pourcentage de positions similaires par rapport au référentiel. Les résultats de ces expérimentations confirment le comportement de PBA dans des conditions défavorables et ne montrent pas un gain des autres méthodes par rapport à la nôtre.

5.5.6 Cinquième scénario de test : impact des modèles de RI sur les résultats

Après avoir testé les approches d'agrégation dans le cadre d'une collection centralisée générale, il est important de tester l'impact de l'utilisation d'une collection spécialisée (l'ensemble de thèmes musicaux). Les paramètres du jeu de données sont présentés dans la Table 5.3. Cette expérimentation (S5), est caractérisée par l'hété-

Collection	MusicDataSet
Nbre_documents	17916
Nbre_requêtes	2329/avec réplication 4020
Nbre_pairs	810
taille_KB en requêtes	2678
Taille_test en requêtes	1340
Modèle RI référentiel	Cos
Modèles RI locaux	hétérogènes : cos/dice/jaccard/overlap choisis aléatoirement par pair

TABLE 5.3 – Paramètres relatifs au jeu de données MusicDataSet

rogénéité des collections sur les pairs, d'un point de vue taille, contenus et modèles de RI. Ceci nous semble indispensable à la validation de notre approche vu que cela reflète la situation typique et la plus générale dans le cadre de la RIP2P. En effet, le défi dans les systèmes de RIP2P, est de pouvoir supporter l'hétérogénéité des pairs avec la plus grande efficacité tout en gardant une bonne performance.

5.5.6.1 Cas d'une distribution aléatoire

Dans ce scénario (S5.1), nous testons l'apport de la spécialisation de la collection, même dans un modèle de distribution défavorable à la logique de notre algorithme. Les résultats obtenus sont illustrés sur les Figures 5.18 et 5.19

Une première conclusion tirée des résultats présentés dans les différentes figures, est que PBA l'emporte sur les approches à base de rangs et l'approche statistique à base de profils. Le gain introduit par l'utilisation de cette collection peut être expliqué par l'utilisation d'une collection de thème spécialisé et confirme que PBA peut donner de bons résultats dans un cas d'hétérogénéité entre les collections locales. Le comportement de BC et RR assure, par contre, qu'ils donnent les meilleurs résultats, lorsqu'il s'agit de collections homogènes, ce qui n'est pas le cas de cette collection. Même avec une distribution statistique, l'efficacité de PBA excède celle des autres méthodes.

5.5.6.2 Distribution sémantique par spécialisation des pairs (S2.5)

A travers ce scénario, on se met réellement dans des conditions favorables à notre approche : une collection spécialisée avec une distribution par spécialisation des pairs aux réponses aux requêtes et un cadre défavorable aux autres approches

avec une grande hétérogénéité côté taille et modèles de RI. Toutes ces conditions expliquent les résultats observés au niveau des courbes des Figures 5.20 et 5.19. En effet, l'écart entre PBA, d'aune part, et BC et RR, d'autre part, tend à augmenter. LP se comporte également de manière favorable mais avec une efficacité toujours inférieure à celle de PBA.

5.6 Synthèse

Il est important de mentionner que PBA surpasse les approches classiques étudiées (à base de rangs) sur le plan de la synchronisation qui correspond, comme présenté dans le chapitre 4, au maximum de temps d'attente nécessaire pour commencer l'agrégation. Ceci comprend le temps de la récupération des réponses des différents pairs. En effet, dans l'approche PBA, l'agrégation commence immédiatement quand un pair contributeur fournit une réponse.
Dans d'autres méthodes, notamment *BC* et *RR*, la synchronisation correspond au temps de la réception, au moins, de la première réponse de tous les pairs, ce qui correspond à un temps important. De plus, l'étude analytique montre que PBA s'avère moins cher que les approches à base de score, en volumes de messages échangés. En effet, pas besoin d'échanger des statistiques sur le réseau. Un message correspond uniquement à l'identifiant du document fourni, comme réponse à une requête.
Analytiquement, le seul inconvénient de PBA est le temps passé au calcul de ses scores basés sur les profils, qui nécessite une recherche dans la base de profils (voir 5.2). Cependant, l'effort de la construction des bases des connaissances est un effort hors ligne.

Du point de vue empirique, les différents scénarii réalisés ont prouvé que notre approche améliore l'état de l'art. En effet, PBA peut être déployé dans toutes les circonstances sans engendrer des pertes significatives d'efficacité par rapport aux approches testées. Reste à signaler, que dans un cadre défavorable, l'adoption d'approches classiques peut être moins chère, vu le coût de recherche des profils dans notre modèle.
Cependant, il important de rappeler que ce coût est local (au pair) et qu'il reste inférieur au coût engendré par l'échange de statistiques globales ou même locales.
La série d'expérimentations réalisées nous a permis de déduire que dans les situations d'hétérogénéité, PBA n'éprouve aucune sensibilité contrairement aux modèles à base de rangs et même les méthodes à base de score.
Nous avons vu également un bon comportement de LP dans le cadre d'une distribution sémantique, ce qui prouve que les modèles à base de sémantique peuvent appréhender l'efficacité des méthodes à base de score. Cependant, la méthode LP n'a pas un rendement fixe vis à vis des situations testées. En effet, ce comportement est des fois antagoniste comme le montre par exemple les figures 5.19 et 5.18 d'une part et 5.21 et 5.20 d'autre part. Ce genre de comportements montrés par cette méthode simple justifie le recours à un vrai effort de construction de profils dans

l'approche PBA. En plus, il convient de rappeler les gains en performance introduits par PBA en évitant tout échange explicite de données avec les autres pairs et l'exploitation de connaissance purement locale pour gagner de l'expertise.

5.7 Conclusion

Dans ce chapitre, nous avons commencé par proposer une analyse de performance analytique de notre algorithme d'agrégation PBA. Un cadre favorable à l'expérimentation des modèles de RIP2P en se basant sur la simulation a été proposé. Un processus complet de simulation a été défini. Il concerne les étapes de préparation de données, la configuration du simulateur, la simulation proprement dite et l'exploitation des résultats de la simulation. Le simulateur PeerSim-RARE+ a été utilisé pour réaliser un ensemble de scénarii indispensables à la validation de l'approche d'agrégation proposée. Ces scénarii concernent respectivement les modèles de distribution, de réplication, le type de collection, l'hétérogénéité des modèles de RI. Les différentes expérimentations ont montré l'amélioration apportée par PBA à l'état de l'art ; notamment lorsqu'il est utilisé dans un contexte hétérogène. Cependant, les scénarii réalisés supposent que l'état du système est stable et que les recommendations restent toujours valables, ce qui n'est vrai. Afin d'assurer d'intégrité des connaissances utilisées dans PBA, une stratégie de maintenance doit être mise en place pour faire évoluer les bases de connaissances vers l'état courant du système. Ceci fera l'objet du chapitre suivant.

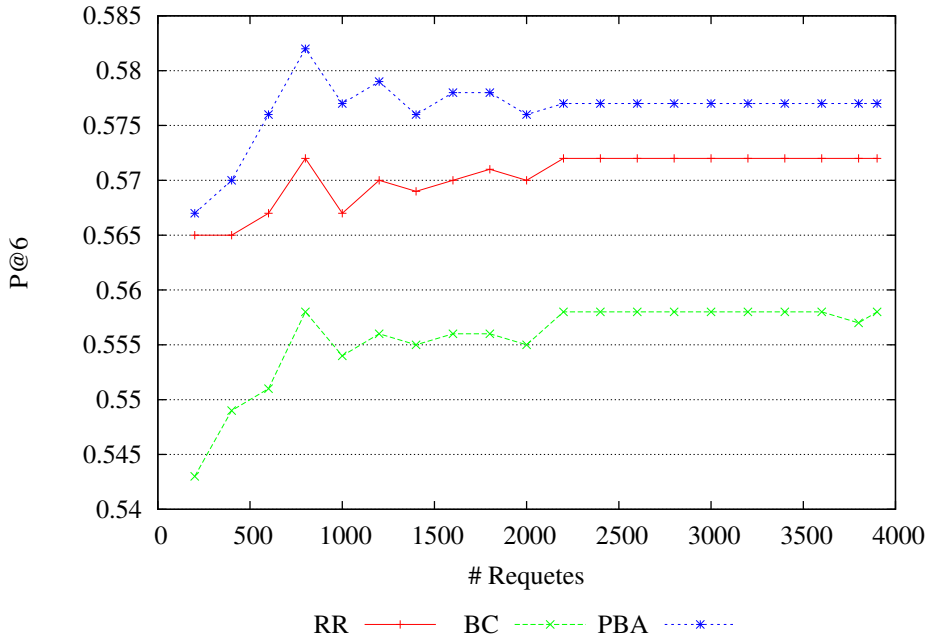


FIGURE 5.16 – S4 : Mean Average Precision

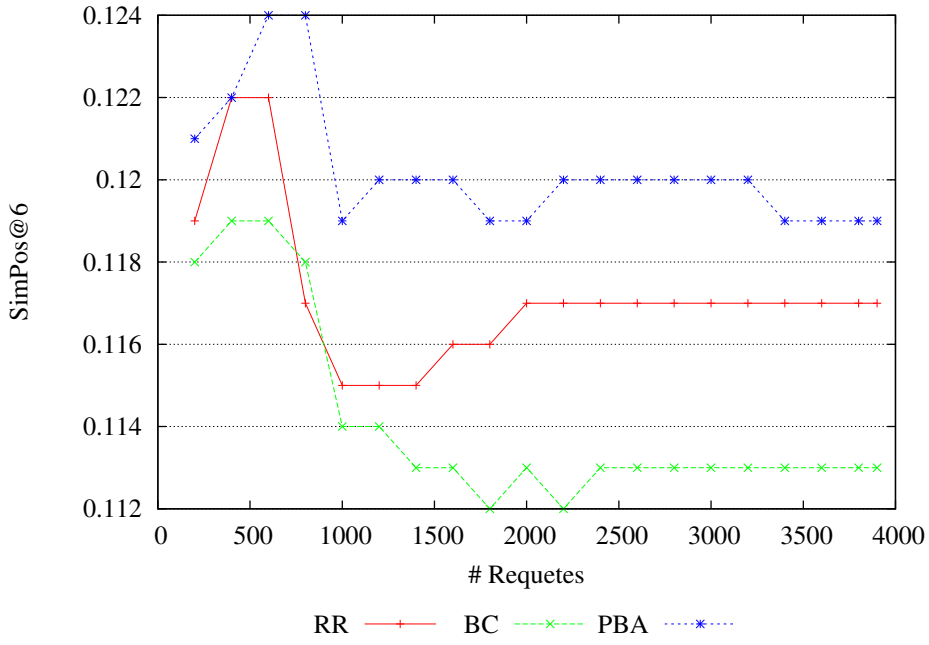


FIGURE 5.17 – S4 : taux de positions similaires (k=6)

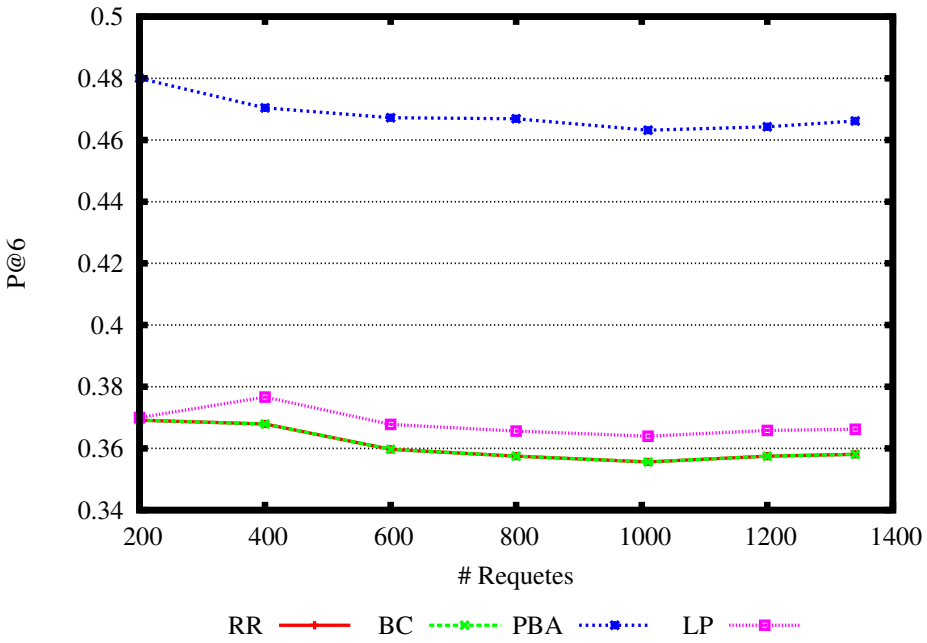


FIGURE 5.18 – S5.1 : MusicDataSet-D Aleatoire- précision moyenne@6

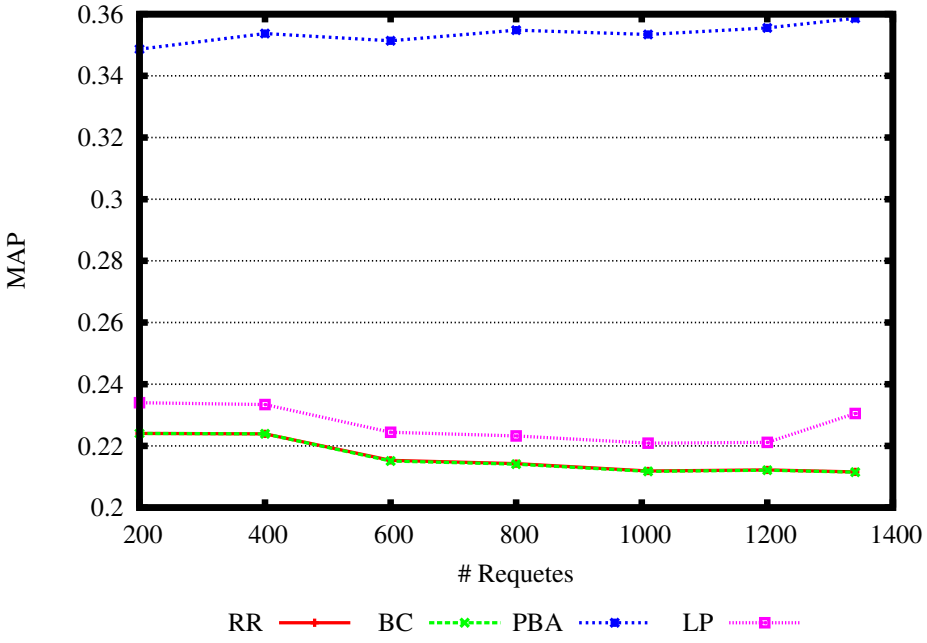


FIGURE 5.19 – S5.1 : MusicDataSet-D Mean Average Precision

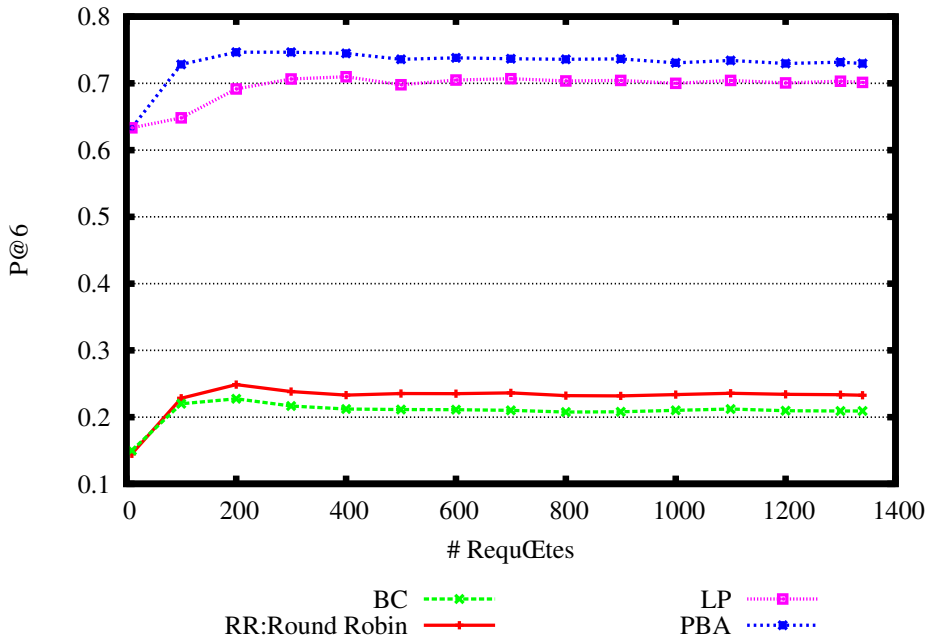


FIGURE 5.20 – S2.5 : MusicDataSet D-specialisée- Précision moyenne @6

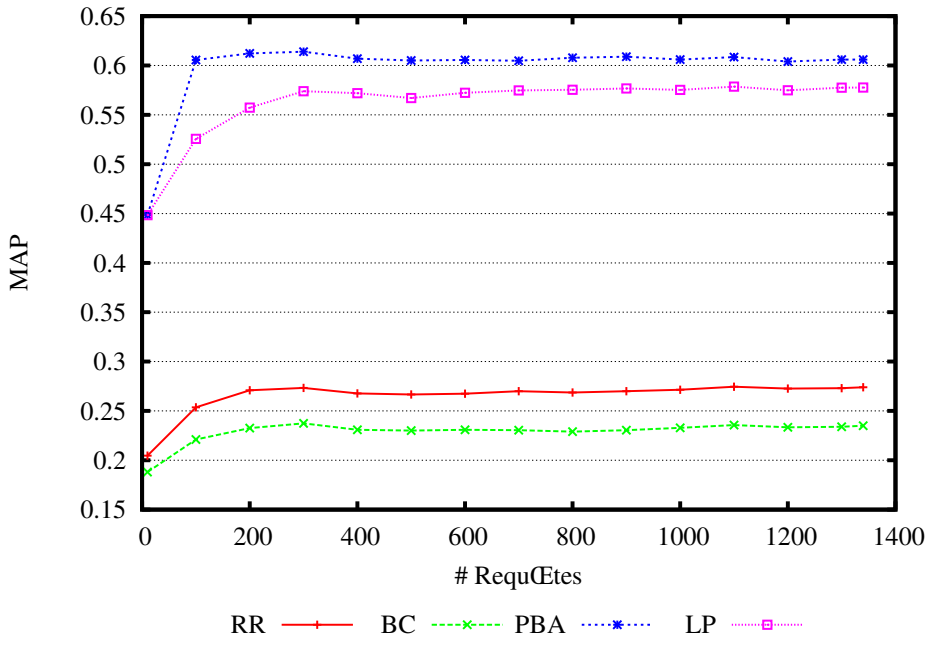


FIGURE 5.21 – S2.5 : MusicDataSet D-spécialisée- M.A.P

Chapitre 6

Evolution de la base des connaissances

6.1 Introduction

Le but derrière l'utilisation d'une base des connaissances construite par apprentissage, sur le passé, est certainement l'amélioration du service dédié par une application donnée, vis à vis de l'utilisateur. Cependant, si le système évolue, cette connaissance devient obsolète et cela va réduire la qualité du système. C'est la raison pour laquelle une stratégie d'évolution de la base des connaissances ($\mathcal{KB}$) doit accompagner toute intégration de connaissance, construite par apprentissage, dans les systèmes. Pour prendre en compte l'évolution, plusieurs questions doivent être traitées :

- Comment estimer l'écart entre l'état courant du système et celui de la base des connaissances ?
- Comment mettre en place des mécanismes d'évolution réactifs mais assurant la convergence de l'état de la base des connaissances ?
- Qui décide de l'évolution de la base des connaissances : une autorité centrale ou un mécanisme distribué ?

La réussite de notre modèle d'agrégation à base de profils dépend de sa capacité à profiter le mieux possible de ce que le système a acquis au cours de la résolution des requêtes passées. Théoriquement, cela suppose que notre base des connaissances soit toujours cohérente par rapport à l'état courant du système. Pourtant, le système évolue et les besoins des utilisateurs également. Cela remet en cause notre algorithme basé sur les profils qui risque de prendre de mauvaises décisions relativement à l'état courant du système et des besoins utilisateurs. Ainsi, un problème important dans ce contexte est de garder à jour les bases de profils afin d'améliorer l'efficacité de la recherche et ce, sans avoir besoin de payer des coûts de maintenance élevés.

6.2 Motivations

La plupart des approches proposées pour mettre à jour une base des connaissances sont périodiques. Le problème avec une mise à jour périodique est double : (i) la base peut être mise à jour alors que le système n'a pas encore évolué, ou (ii)

la base peut ne pas être mise à jour alors que le système a, lui, évolué. De plus, la majorité des approches utilisent un serveur centralisé pour gérer les actions de mise à jour. Intuitivement, il est évident que cela ne peut pas être efficace, dans les systèmes à grande échelle, en raison de la difficulté à synchroniser un grand nombre de pairs.

Il s'avère ainsi insistant de contrôler la fraîcheur de la base des connaisances en procédant à son évolution seulement lorsque nécessaire. Dans un système de RIP2P, les évolutions peuvent concerner :

- la suppression de ressources du système vu la déconnexion d'un ou de plusieurs pairs du système (ou la suppression de documents sur un pair existant),
- l'ajout de nouveaux contenus par l'hébergement de nouveaux documents sur les pairs existants ou la connexion de nouveaux pairs au système P2P.
- ou finalement le changement de comportement de l'utilisateur ou l'évolution de ses centres d'intérêts.

Une façon naturelle de rafraîchir la $\mathcal{KB}$ est de mettre en place des indicateurs qui vont nous informer chaque fois que des nouveaux besoins par rapport à la $\mathcal{KB}$ émergent. Il s'agit alors de mettre en œuvre des procédures de synchronisation de la base par rapport à ces nouveaux besoins détectés.
Pour ces raisons, nous cherchons à définir un mécanisme d'évolution contrôlé basé sur la détection de l'évolution du système ou des utilisateurs. Les indicateurs d'évolution seront évalués sur chaque pair, en observant les requêtes des utilisateurs, d'une part, et les réponses du système, d'autre part. Ces indicateurs sont capables de décider quand une base locale doit être mise à jour.

6.3 Stratégie d'évolution contrôlée

Dans cette partie, nous mettons en exergue, à partir de scénarii réalistes, les principales causes d'incohérence d'une base des connaissances, ($\mathcal{KB}$), stockée au niveau d'un pair donné par rapport à l'état d'un système de RIP2P. Nous dégageons, par ailleurs, les principaux indicateurs d'évolution (construits par la détection de défaut d'apprentissage) qui sont évalués comme un ensemble d'applications tournant en arrière plan pour contrôler la fraîcheur de la base et déclencher des actions de mise à jour au besoin.
Nous présentons, dans ce qui suit, un exemple de situations (cas) où il y aura une divergence entre l'état du système (ou les besoins de l'utilisateur) et les connaissances apprises (présentes dans la $\mathcal{KB}$). Nous présentons, de même, l'impact de ces situations sur le processus d'agrégation.

Cas1 : Un utilisateur qui exprime des besoins relatifs aux thèmes 'RID', 'P2P', peut s'intéresser également à un thème plus général tel que le 'processus de RI', ou plus spécifique tel que les problèmes de 'distribution', les 'systèmes P2P', les

'architectures', etc.
La base construite à partir des interactions de l'utilisateur avec les réponses fournies par le système, renferme donc des thèmes proches et complémentaires. Les documents figurant dans la base sont les plus intéressants de point de vue utilisateur par rapport à ses requêtes.
Le système démarre alors, à froid, avec les connaissances qu'il a acquises au bout d'une certaine période. Si un jour, l'utilisateur repose des questions sur un thème déjà vu, le processus d'agrégation va ordonner les résultats retournés sur cette base en favorisant, d'une part, les pairs qui ont contribué à la réponse au besoin exprimé, et d'autre part, les documents qui l'ont intéressés, et ce en tenant compte de l'importance du document par rapport aux SRIs locaux.
Cas2 : Une fois que l'utilisateur soumet une requête "achat voiture" par exemple, on remarque que les thèmes recherchés ont brutalement changé par rapport à ce qui a été déjà appris dans la base. Là, sur le plan du processus d'agrégation, aucune connaissance préalable n'existe pour orienter les décisions du classement. Ce dernier est effectué alors aléatoirement.
Cas3 : Si cette requête (ou une variante) n'est pas re-soumise par l'utilisateur, il n'y a pas besoin de mettre à jour la base. Par contre, s'il y a plusieurs re-soumissions, il va falloir faire évoluer la $\mathcal{KB}$.
Le scénario de changement de besoin exprimé par l'utilisateur (la requête), et cité dans ces différentes situations, n'est pas le seul indicateur d'incohérence de la base.
Cas4 : Soit une requête q de termes {t1,t2,t3} figurant dans la base des connaissances, telle que, sa réponse est construite à partir des ressources suivantes {N1{D1,D2}, N2{D2,D4}}. Si l'utilisateur re-soumet ultérieurement la même requête et que le système, lui, restitue les ressources suivantes : {{N1{D1,D5},N10{D2,D4,D6}}, nous remarquons que, d'une part le système a restitué une partie des résultats du pair N1 sauf D2, et d'autre part, il fournit, en plus, le document D5. De même, le pair N2 ne figure plus parmi les réponses, comme il figure dans l'historique de la requête. Finalement, le pair N10 vient s'ajouter à la liste des contributeurs aux réponses à la requête q.
L'absence du document D2 de N1 peut être interprétée par la suppression de ce document du pair N1 alors que la présence de D5 confirme l'ajout de ce document au pair par rapport à l'ancienne recherche. Quant à la présence et l'absence des pairs N10 et N2, nous interprétons ça par une simple connexion ou déconnexion des pairs au systèmes (nous ne discutons pas les cas où un noeud est atteint par routage (Time To Live ou TTL) ou un changement de chemin).
Ces deux exemples illustrent les cas que nous voulons détecter grâce à nos indicateurs.

6.4 Architecture proposée

La Figure 6.1 présente l'architecture du modèle d'évolution contrôlée de la base des connaissances d'un pair donné. Cette architecture est décrite à travers un dia-

gramme d'activité présentant les principales étapes réalisées afin de rafraîchir une base des connaissances.
Deux types de détecteurs (un pour les requêtes, un pour les ressources), fonctionnant en arrière plan, essayent de comparer trois types de besoins (thèmes, documents et pairs) par rapport aux informations stockées dans la base des connaissances.
Il s'avère évident, d'après la Figure 6.1, que le processus est exécuté dès que l'utilisateur reçoit les résultats de sa requête à partir de plusieurs pairs contributeurs. Les différents détecteurs utilisent un indicateur de présence ou d'absence de changement(i.e, Flag). Si la ressource est déjà apprise par le système, l'indicateur est mis à 0, tel que présenté, plus tard dans les algorithmes 6.5.1 et 6.5.2, respectivement, pour les requêtes et les ressources. Ceci correspond dans l'architecture proposée à déterminer la "valeur de détecteur".

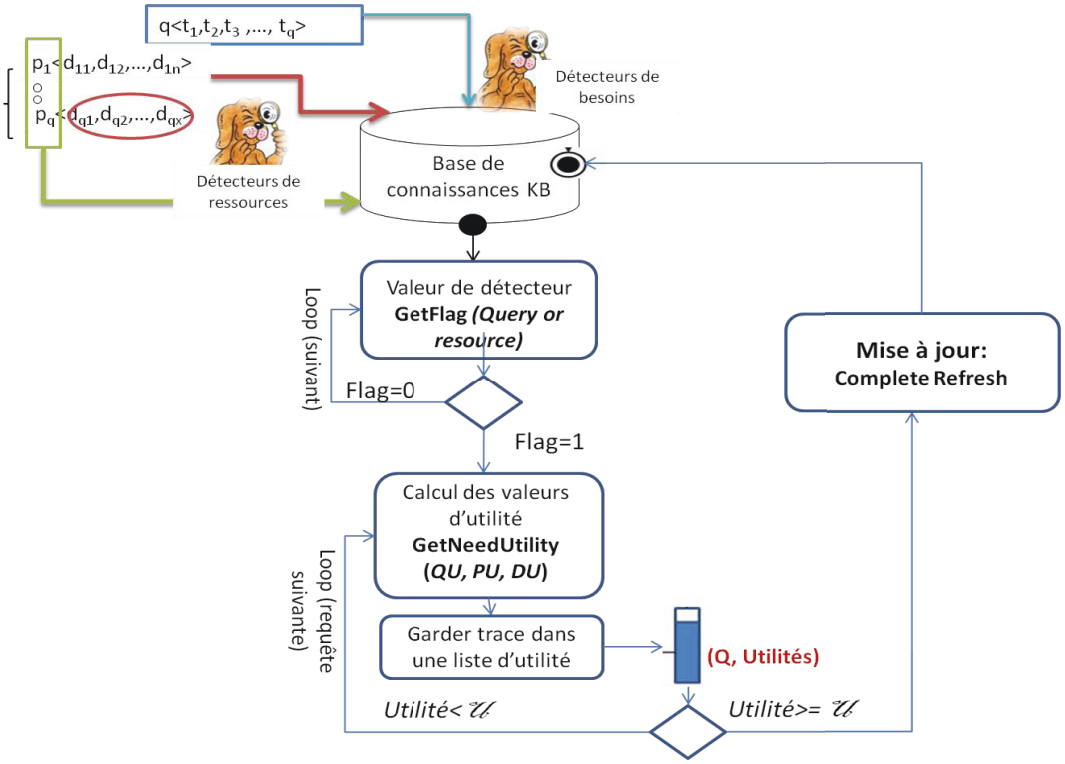


FIGURE 6.1 – Architecture du module de mise à jour des bases des connaissances

Rappelons qu'une mise à jour de $\mathcal{KB}$ peut s'appuyer sur une autorité centralisée [Theodoratos 1999] qui est en mesure de décider quelles sont les données à rafraîchir et quand. Cependant, le recours à une autorité centralisée dans le contexte de systèmes décentralisés à grande échelle est inadéquate. C'est la raison pour laquelle, nous avons opté pour une approche de mise à jour décentralisée. Par conséquent, chaque pair décide localement de mettre à jour sa base des connaissances, à partir des indicateurs.
Cela conduit le système à être plus cohérent à chaque instant. En effet, des chan-

gements peuvent avoir été observés par certains pairs et pas par d'autres et donc les bases peuvent ne plus être cohérentes les unes envers les autres. Cela n'est pas vraiment un problème parce que toutes les décisions liées à l'agrégation des résultats retournés sont prises simplement par un pair initiateur donné, (et donc à l'aide de sa base, seule). De plus, à terme, les changements vont être observés par tous les pairs concernés. Ainsi, la contribution principale de cette approche de mise à jour est son aspect incrémental de la mise à jour du système, chaque base se mettant à jour indépendamment des autres.

L'autre question à régler, est de savoir, quand faire la mise à jour de la base en fonction des changements observés ? S'agit-il d'un processus instantané (mise à jour dès qu'un changement est observé) ou différé en attente d'un nombre "suffisant" de changements ?
Il est ici naturel de différer la mise à jour, car il faut pouvoir s'assurer de la pérennité du changement observé. Dans le cas de la détection d'un nouveau besoin utilisateur, il faut s'assurer qu'il s'agit d'un réel changement des intérêts de l'utilisateur. Pour cela, il faut réaliser un cumul de changements de même type pour décider du déclenchement de la mise à jour. De même, l'arrivée ou la disparition d'un pair ne doit pas mené au déclenchement automatique de la mise à jour.

6.5 Processus de mise à jour

Nous avons présenté dans la sous section précédente les principales causes d'incohérence de la base dans le système de RIP2P. Ces dernières sont soit basées sur l'émergence de nouveaux besoins exprimés par l'utilisateur ou sur des modifications liées aux ressources du système. En se basant sur ces causes, notre processus de mise à jour contrôle l'état de la base en reposant sur deux types d'indicateurs [Mghirbi 2012] : un indicateur d'émergence de nouveaux besoins et un indicateur de changements de ressources. Les différentes notations utilisées dans les algorithmes proposés sont préalablement présentées dans le tableau 6.1.

6.5.1 Indicateur d'émergence de nouveaux besoins

Cet indicateur se base sur la détection de nouvelles requêtes par rapport à celles présentes dans la $\mathcal{KB}$. Pour ce faire, il analyse chaque soumission d'une nouvelle requête pour comparer ses termes aux termes appris dans les profils de la $\mathcal{KB}$ locale. Un score de satisfaction est calculé pour juger de la cohérence de la base par rapport au besoin exprimé. Si ce score est inférieur à un seuil $\Im$, la requête est jugée comme étant un nouveau besoin (on peut également considérer que c'est un défaut d'apprentissage) à prendre en compte pour une éventuelle mise jour. Cette action est répétée pour chaque nouvelle requête. Si en cours de traitement, le nombre de requêtes considérées comme nouvelles par rapport au nombre des requêtes soumises excède un deuxième seuil $\mathfrak{U}$, la mise à jour doit être déclenchée.

Symboles	Notations
q	Une requête utilisateur
N	Pair Initiateur
Q	L'ensemble de requêtes soumises à partir d'un pair initiateur
$\mathcal{KB}$	base des connaissances
T_q	l'ensemble des termes d'une requête q
$\Im$	a similarity threshold between queries and profiles termes qui vérifie la cohérence entre l'état du système et celui de la $\mathcal{KB}$
$\mathcal{PR}_q$	The set of profiles in $\mathcal{KB}$ similaires à une requête q
R	ressource (pair ou document)
$Flag$	un détecteur de nouveau besoin ou ressource
QU	Utilité des nouvelles requêtes
PU	Utilité des nouveaux pairs
DU	Utilité des nouveaux documents
$\mathfrak{U}$	Seuil d'Utilité nécessaire pour déclencher une M.A.J
$MaxUtility$	max(QU,PU,DU)
max	le score de similarité maximal entre une requête q et les profils de la requête $\mathcal{PR}_q$
score	score de cohérence
Nbq, Nbd	respectivement, le nombre de requêtes dans Q et, le nombre de documents

Table 6.1 – Table des notations - Evolution de la base des connaissances

[H] NN KBKB InputDonnéesOutputRésultat $\mathcal{KB}$: base des connaissances
$\Im$: Seuil de cohérence du système avec la $\mathcal{KB}$
q : requête utilisateur
N : Pair initiateur Flag : Valeur d'indicateur précisant le nouveau besoin $Flag = 0$
$Score(q) = GetNeedProfileSimilarity(q, \mathcal{KB})$
$Score(q) < \Im Flag = 1$ return $Flag$
GetQueryFlag

L'Algorithme 6.5.1 résume les principales étapes nécessaires pour détecter si une requête q est un nouveau besoin par rapport à l'état de la $\mathcal{KB}$. Par conséquent, un score de cohérence entre le système et l'état de la $\mathcal{KB}$ est calculé sur la base de l'algorithme 6.5.1. Si le score est inférieur à un seuil de similarité $\Im$, le détecteur de défaut d'apprentissage (Flag) relatif à la requête q prend la valeur 1, la requête est alors considérée comme un nouveau besoin.

[H] KBKB qq PRPR GetQuerySimilarProfilesGetQuerySimilarProfiles Input-DonnéesOutputRésultat $\mathcal{KB}$: base des connaissances
q : requête utilisateur
T_q : Les termes d'une requête q ScoreScore Score : score de cohérence max=0

$\mathcal{PR}_q = \sigma_{Sim(q,pr_i)}(\mathcal{KB}) \ / pr_i \in \mathcal{KB}$
$pr_j \in \mathcal{PR}_q = \frac{|T_q \cap \pi_{(T)}(pr_j)|}{|T_q \cup \pi_{(T)}(pr_j)|}$
>= maxmax = return Score GetNeedProfileSimilarity

6.5.2 Indicateurs de changement de ressources

Deux indicateurs sont utilisés ici afin de détecter un changement au niveau des ressources (les pairs et les documents). Si le système retourne en réponse à une requête présente dans la $\mathcal{KB}$ de nouveaux documents par rapport à ce qui a été appris, ou des nouveaux pairs ou encore restitue des documents en moins, alors la requête est sauvegardée dans la liste d'utilité. Un facteur d'utilité des nouveaux documents et pairs est calculé afin de voir le degré de sollicitation de la ressource vis à vis de la requête. L'utilité d'un document est calculée sur la base du degré de sa sollicitation dans la réponse aux requêtes, alors que celle d'un pair est calculée en fonction du nombre de nouveaux documents qu'il fournit en moyenne pour répondre à la requête.

L'objectif principal de cette détection est de vérifier si de nouvelles ressources (R) (pairs ou des documents) ont rejoint le système. Cette détection est basée sur une comparaison progressive entre les réponses fournies actuellement aux requêtes et les réponses apprises à partir de la base pour les requêtes similaires. La notion de similarité entre les termes des nouvelles requêtes et des termes dans les profils PR à partir $\mathcal{KB}$ est basée sur la similarité sémantique de Salton, explicitée dans l'algorithme 6.5.1.

L'Algorithme 6.5.2 affecte la valeur 1 à chaque détecteur de ressource (Flag) qui n'existe pas dans les profils de la requête q, $\mathcal{PR}_q$ et décide que R est un nouveau besoin par rapport à l'état de la $\mathcal{KB}$.

[H] KBKBQQ ExistExist InputDonnéesOutputRésultat $\mathcal{PR}_q$:Profils similaires à une requête q
q : requête utilisateur
R : une ressource (Pair contributeur ou document)
Flag : Indicateur d'arrivée de nouveaux pairs ; $Flag = 0$
!(R , $\mathcal{PR}_q$) $Flag = 1$
return $Flag$
GetResourceFlag

6.5.3 Estimation de l'utilité de mise à jour

Une requête est considérée comme défaut d'apprentissage, si elle correspond à un nouveau besoin, ou, si sa réponse a permis de détecter un changement (apparition ou

disparition) de ressources. La détection est donc effectuée au moyens de deux types de détecteurs : un détecteur de besoins exprimés par l'utilisateur (ou requêtes), exprimé au niveau de l'algorithme 6.5.1 et des détecteurs de ressources (documents et pairs) décrits par l'algorithme 6.5.2. Cependant, pour déclencher une mise à jour, il ne suffit pas de détecter un besoin, mais il faut essayer d'estimer l'importance ou 'utilité' de ce besoin. En fait, la décision du déclenchement est liée au contenu de l'utilité de chaque type de besoin.
Selon le type de détecteur utilisé, nous définissons trois compteurs d'utilité :

- QU : ou utilité des requêtes. Cette valeur s'incrémente à chaque fois qu'une requête est détectée comme nouvelle (si son Flag =1) par rapport à l'état de la $\mathcal{KB}$. Le calcul de QU est donné par la ligne 9 de l'Algorithme 6.5.3.
- PU : désigne l'utilité des pairs. Cette variable change de valeur à chaque détection d'un pair qui propose de nouvelles ressources (documents au système). L'utilité de ce pair est estimée par le nombre des nouvelles ressources qu'il introduit (voir la Ligne 14 de l'Algorithme 6.5.3).
- DU : désigne l'utilité de nouveaux documents (voir Ligne 15 de l'Algorithme 6.5.3).

[h] GetQueryFlagGetQueryFlagGetPeerFlagGetPeerFlagGetDocFlagGetDocFlag EntreesDonnéesOutputRésultat $\mathcal{KB}$: Base de connaissance ;
q : requête utilisateur ;
Q : ensemble de requêtes d'un pair initiateur NeedUtility : liste contenant les requêtes considérées comme nouveaux besoins Nbq=0
Nbd=0
QU=0
PU=0
DU=0
q in Q Nbq++
$QU = QU + ()$
p in q.P d in P.D Nbd++
$=1PU = PU + () DU = DU + ()$
() !=0 || () !=0 || () !=0 NeedUtility.add(q,QU,PU,DU,Nbq,Nbd) Return NeedUtility GetNeedUtility

Il faut noter que ces trois indicateurs ne sont pas forcément aussi importants les uns que les autres. DU a une utilité notamment moins forte que PU (un pair apporte généralement plusieurs documents et pas un seul). Il serait donc bon de pondérer l'importance relative de ces indicateurs. La détection des évolutions n'est également pas toujours fiable, notamment l'apparition/disparition de pairs qui peut être due à un problème de connectivité réseaux. Il faudrait notamment éviter des situations où un pair est trop dynamique (qui apparaît puis disparaît) et génère beaucoup de défauts forçant au rafraîchissement de la base des connaissances alors que dans ce cas, il faut l'ignorer.
Une trace complète des requêtes présentant, pour l'une des utilités citées, une valeur différente de 0 est gardée avec ces différents indicateurs, dans une liste appelée NeedUtility (voir l'Algorithme 6.5.3).

6.5.4 Mise à jour de la base des connaissances

Le calcul de la valeur d'utilité pour chaque type de besoin, tel que présenté dans l'algorithme 6.5.3, est la condition nécessaire pour effectuer la mise à jour. En effet, lorsque la valeur d'utilité atteint un seuil $\mathfrak{U}$ (voir l'algorithme 6.5.4), la mise à jour peut être réalisée. Nous considérons ici une valeur d'utilité (de la requête ou des ressources) en fonction du type d'indicateur utilisé. Lorsque tous les détecteurs sont utilisés, l'utilité maximale des trois est comparée au seuil d'utilité.

Pour actualiser la $\mathcal{KB}$, deux approaches peuvent être utilisées : la mise à jour complète qui consiste à régénérer la totalité de la base ou la mise à jour incrémentale qui consiste à apprendre seulement les requêtes qui ont déclenché le défaut d'apprentissage et de considérer après une connaissance sur un ensemble de bases. Dans notre cas, nous avons choisi de réaliser une mise à jour complète de notre base par recours à la régénération de l'information recueillie à partir de la liste de besoins dégagée dans l'Algorithme 6.5.1 et de la base des connaissances, tel que présenté dans l'Algorithme 6.5.4. Il est important de noter ici, que même un rafraîchissement complet se fait d'une manière, purement locale, par rapport à un pair donné.

[H] ListOQListOQListNQListNQ KBKBTailleTaille NeedsListNeedsList GetMaxUtilityGetMaxUtilityNeedNeed MaxUtilityMaxUtility NbqNbq GetNeedUtilityGetNeedUtility GetQueriesGetQueries GenerateBaseGenerateBase getLearnedQueriesgetLearnedQueries InputDonnéesOutputRésultat $\mathcal{KB}$:base des connaissances $\mathfrak{U}$:seuil d'utilité nécessaire pour déclencher la M.A.J $\mathcal{KB}$: KB mise à jour $= (())$ $> \mathfrak{U}(\mathcal{KB})$
Complete Refresh

Ainsi, l'Algorithme 6.5.4 présente les différentes étapes pour le rafraîchissement complet de la base des connaissances et détermine les conditions du déclenchement. La condition du déclenchement de la mise à jour dépend de la méthode $GetMaxUtility()$ qui récupère les dernières valeurs d'utilités (QU, DU et PU) (qui sont les plus grandes) et les divisent respectivement par (le nombre de requêtes traitées, *nbq*, et le nombre de documents passés comme réponses à une requête, *nbd*), puis récupère la valeur maximale des trois ratios dans MaxUtility. Si le ratio dépasse la valeur du seuil d'utilité considéré comme nécessaire pour déclencher la mise à jour, la base sera régénérée en tenant compte des requêtes contenues dans la liste NeedUtility.

6.6 Etude expérimentale

Nous proposons, dans cette partie, une évaluation présentant l'impact de la mise à jour des connaissances sur notre algorithme d'agrégation selon la stratégie de mise

Apprentissage et test		#requêtes
B0	base intitiale	2700
T0	test initial sur l'ensemble de pairs du système (450)	1100
B1	# Nouvelles requêtes détectées à partir de T0	275
T0z	Zoom test sur T0 (10 pairs)	———
T1	nouvelles requêtes pour test	4300
B2	base mise à jour	2700+275
tests ultérieurs	T0/B2 , T0z/B2, T1/B2	———

TABLE 6.2 – Scénarii de test

à jour proposée dans ce chapitre.
Pour commencer, nous pouvons noter que, même sans évolution, les bases ne sont pas toutes d'égale qualité dans le système. Une vision globale de la capacité d'apprentissage sur chaque pair est donnée dans la Figure 6.2. Un point (x, y) sur le graphique signifie qu'un pair x a appris y termes à partir des réponses des requêtes passées. Ce type d'information permet de donner une idée sur la qualité de la $\mathcal{KB}$ sur chaque pair et par conséquent une idée sur l'efficacité de l'algorithme de PBA sur ces pairs. En effet, plus on apprend, plus l'algorithme est en mesure de recommander dans les premières positions des documents pertinents pour les requêtes similaires aux requêtes apprises.
Nous notons que le nombre de termes appris par pair varie entre 2 et 458 termes. Cette différence de capacité d'apprentissage entre les différents pairs a un impact sur la qualité du notre approche d'agrégation quand il s'agit d'agréger les résultats des requêtes pour les pairs ayant une $\mathcal{KB}$ relativement pauvre.
Pour nos simulations associées aux différentes évolutions de la base, nous avons utilisé un sous-ensemble de données centralisé de l'annuaire DMOZ [DMOZ 2011]. Pendant les simulations, chaque URL est considéré comme un pair.
Les requêtes suivent une distribution aléatoire [Adamic 2002] et répliquées trois fois. Les documents sont distribués en se basant sur une méthode de classification qui jaillit naturellement de la collection et ce en se basant sur les URLs des sites [Lu 2003]. Aucune réplication n'est considérée pour les documents. L'évaluation de l'approche proposée pour l'évolution de la base des connaissances est basée sur l'utilisation de la l'algorithme d'agrégation PBA avec et sans la mise à jour de la base des connaissance. Afin d'évaluer notre approche de mise à jour de l'algorithme PBA, nous avons d'abord mis l'accent sur l'étape de l'apprentissage. Nous avons réservé 1/3 des requêtes sur chaque pair pour l'apprentissage. Par ailleurs, nous considérons l'apprentissage seulement au niveau de 450 pairs et non pour l'ensemble des 810 pairs. Cela permet au simulateur de ramener des réponses qu'il n'a pas déjà apprises pour se situer dans le pire des cas.

Deux conditions sont considérées pour la mise à jour :

- Le seuil $\Im$=50% est la condition pour décider qu'une requête est un nouveau besoin.
- L'utilité de la mise à jour est avérée si l'ensemble de nouveaux besoins détectés

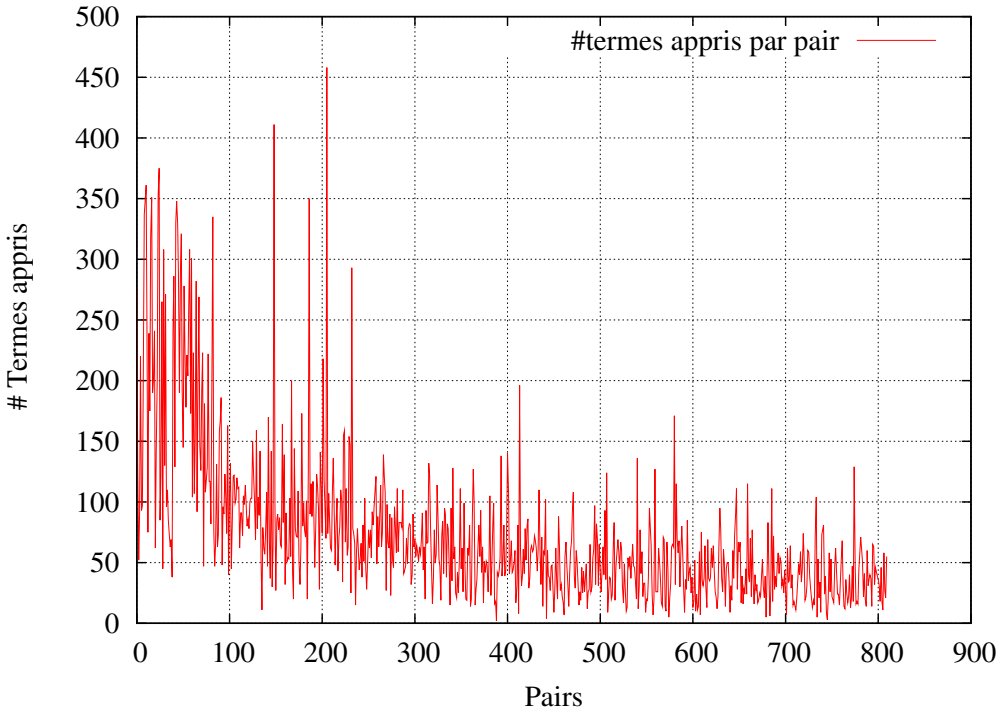


FIGURE 6.2 – Vision Globale sur les bases des pairs d'un système

au niveau d'un nœud excède 30 % de l'ensemble des requêtes lancées sur ce pair ou si la requête même est réclamée de nouveau par l'utilisateur pour plus que $> 30\%$ du nombre total des requêtes lancées à partir d'un pair initiateur L'utilité des besoins est ainsi la condition pour recommander la mise à jour.

Résultats expérimentaux

Nous commençons par le test $T0$ décrit dans le tableau 6.2 où nous lançons un nombre de 1100 requêtes à partir de 450 pairs. Le but de cette expérimentation est d'évaluer la capacité de notre approche à détecter correctement les défauts d'apprentissage des requêtes.

L'algorithme 6.5.1, pour la détection de défaut d'apprentissage, est lancé à base des paramètres de $T0$ et $T0z$ et fournit les résultats représentés dans les figures 6.3, 6.4 et 6.5. La figure 6.3 donne un aperçu sur le taux de la non-satisfaction moyenne des requêtes agrégées au niveau du système entier, et montre que le nombre de requêtes présentant un défaut d'apprentissage (qui présentent un taux de satisfaction inférieur au seuil) est croissant au cours du temps.

Afin de comprendre la raison de ce défaut d'apprentissage, on a fait des tests réduits sur certains pairs. Plus précisément, nous nous sommes concentrés sur dix pairs du système (voir Figure 6.4). Nous remarquons d'après les résultats présentés dans la Figure 6.4 que 6 pairs parmi les dix présentent un défaut d'apprentissage

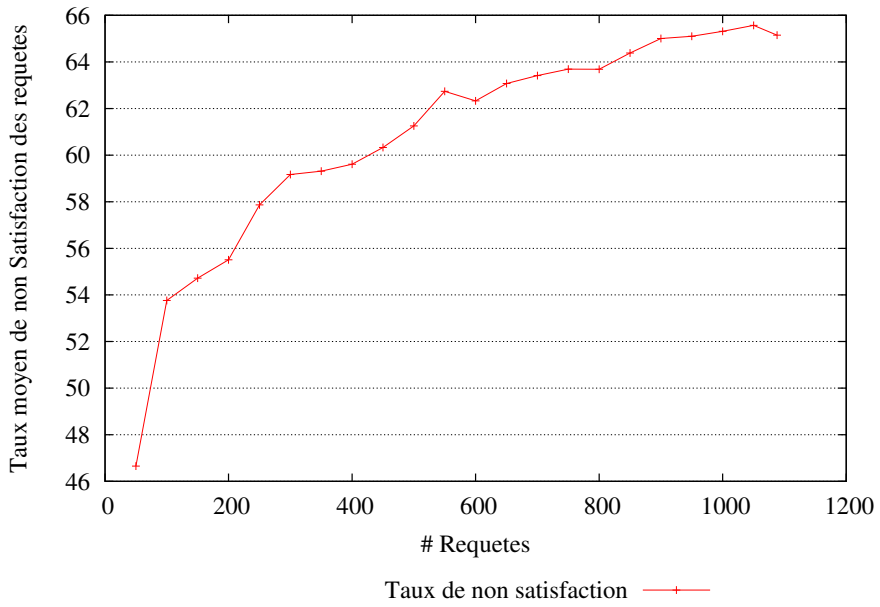


FIGURE 6.3 – Taux moyen de non satisfaction des requêtes

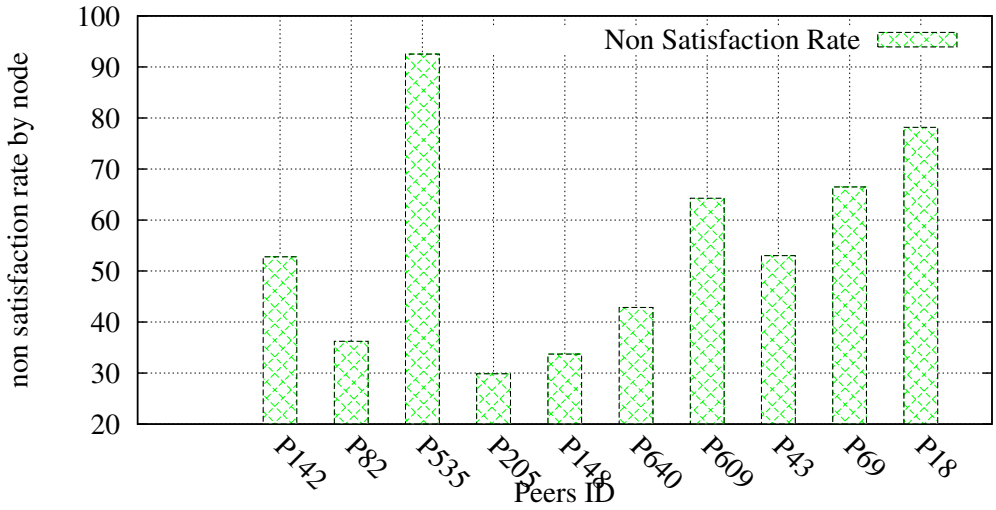


FIGURE 6.4 – Taux de non satisfaction des requêtes par pair

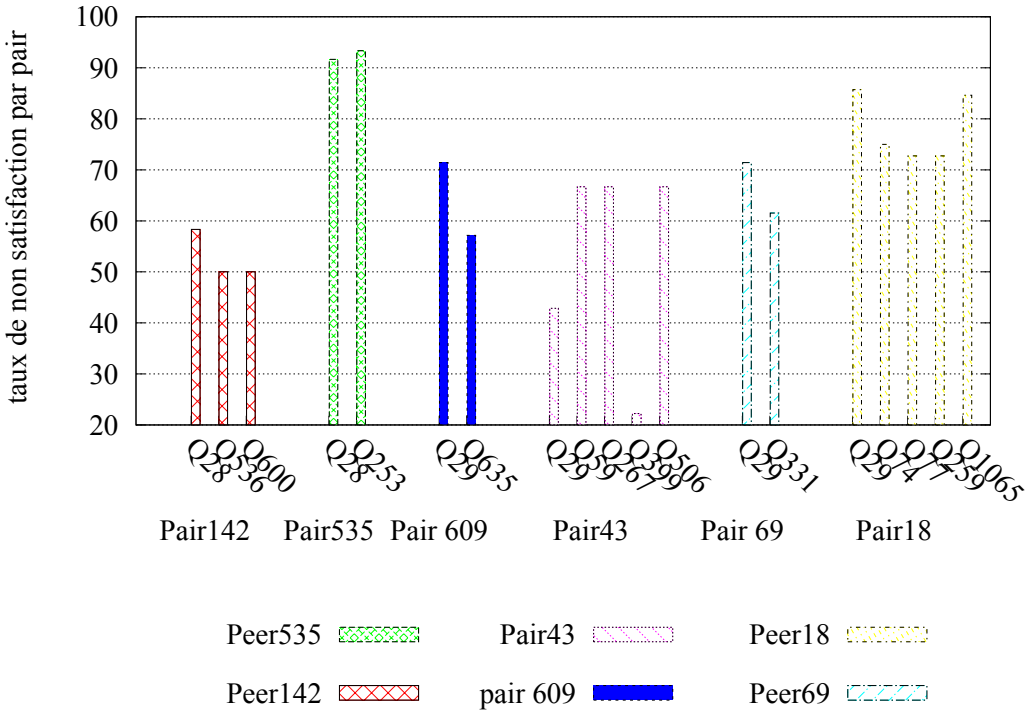


FIGURE 6.5 – Focalisation sur des requêtes spécifiques

puisque leurs taux de non satisfaction dépassent le seuil fixé. Nous avons suivi la même stratégie pour voir plus précisément la raison de l'échec de ces pairs et nous examinons cette fois-ci les requêtes lancées à partir de ces pairs. À la lecture de la Figure 6.5, nous remarquons que dans chaque pair, le nombre de requêtes qui présentent un défaut d'apprentissage dépasse 30% du nombre total des requêtes lancées sur le pair.

Afin de voir l'impact de la mise à jour sur le résultat de PBA, nous avons conduit le test $T0z$ avant et après la mise à jour, soit sur B0 et B2 qui représente la $\mathcal{KB}$ actualisée en tenant compte de B1 (l'ensemble des requêtes qui ont présenté un défaut d'apprentissage).

Le test a pour objectif de voir l'amélioration du système lorsque nous soumettons entre autres, des requêtes apprises. Les résultats de ce test sont illustrés sur les figures 6.6, 6.8 et 6.9. Toutes les métriques montrent une nette amélioration quand PBA est lancé avec une $\mathcal{KB}$ mise à jour. Cependant, ces résultats sont individuels (par rapport à leurs pairs), ainsi nous avons essayé d'avoir une vue d'ensemble sur l'efficacité du système lors du calcul de la MAP sur tous les pairs tels que présenté dans la figure 6.10. Le résultat présenté dans cette figure confirme que les décisions individuelles de mise à jour des connaissances prises sur chaque pair ne sont pas antagonistes et contribuent en revanche à améliorer l'efficacité globale du système.

La dernière question à laquelle nous avons voulu répondre à travers cette expé-

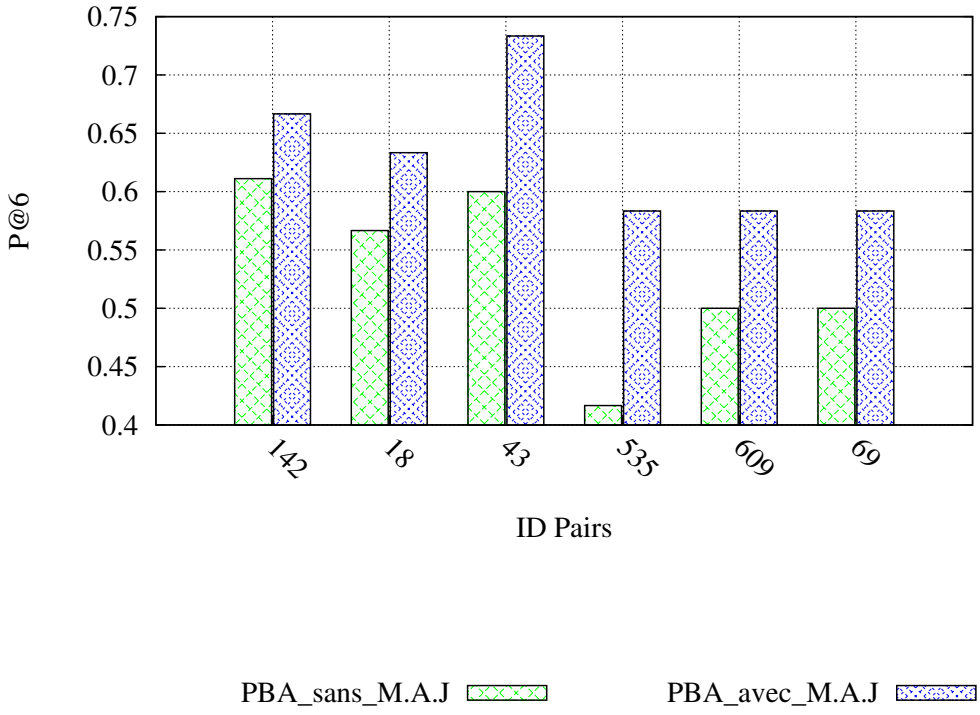


FIGURE 6.6 – Impact de la mise à jour des $\mathcal{KB}$ sur la Precision@k

rimentation est de voir l'impact de la mise à jour des connaissances sur les performances des nouvelles requêtes dans le système. C'est le but du test $T1$. Le résultat de ce test, pour la métrique MAP, est illustré sur la figure 6.11 qui présente l'efficacité du système initial (sur la $\mathcal{KB}$ initiale B0) et l'amélioration de cette efficacité après la mise à jour de $\mathcal{KB}$.

En résumé, les résultats de ces expériences sont très satisfaisants et montrent d'une part que nos détecteurs fonctionnent bien et d'autre part que la régénération de la base des connaissances (même effectuée seulement sur quelques pairs) améliore sensiblement les performances de PBA sans engendrer de coûts additionnels élevés.

6.7 Conclusion

Dans ce chapitre nous avons essayé de mettre l'accent sur l'aspect de maintenance de la base des connaissances. Les principales contributions de l'approche de mise à jour présentée, sont (i) une architecture décentralisée de l'évolution des connaissances, (ii) la définition des indicateurs de défaut de la $\mathcal{KB}$ et des algorithmes de décision associés.

Les résultats des expériences menées sont avérés satisfaisants. Il fournissent une idée sur le fonctionnement des détecteurs et confirment le besoin d'apporter des mises à jour, quand nécessaire. Ceci a confirmé l'impact de la qualité d'une $\mathcal{KB}$

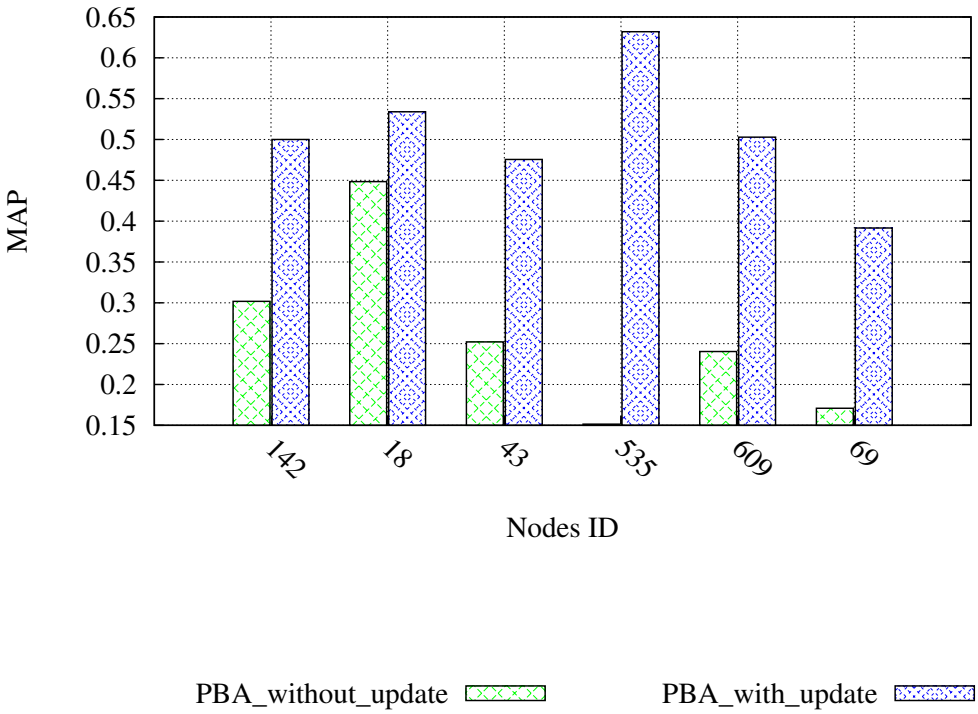


FIGURE 6.7 – Impact de la mise à jour des bases sur MAP

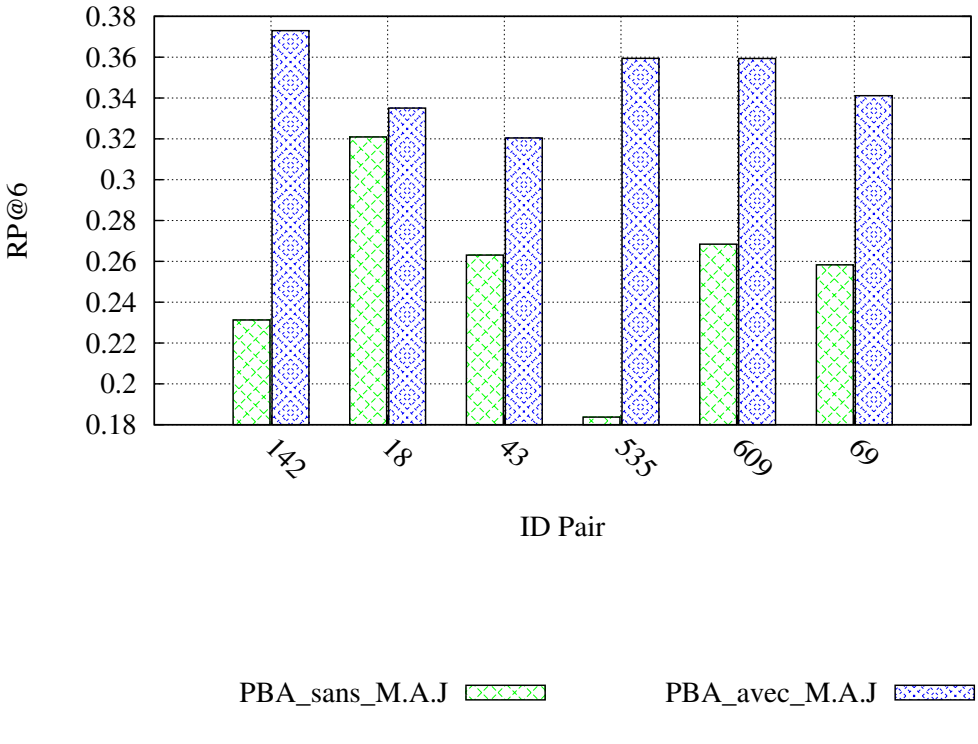


FIGURE 6.8 – Impact de la mise à jour des bases sur RP@k

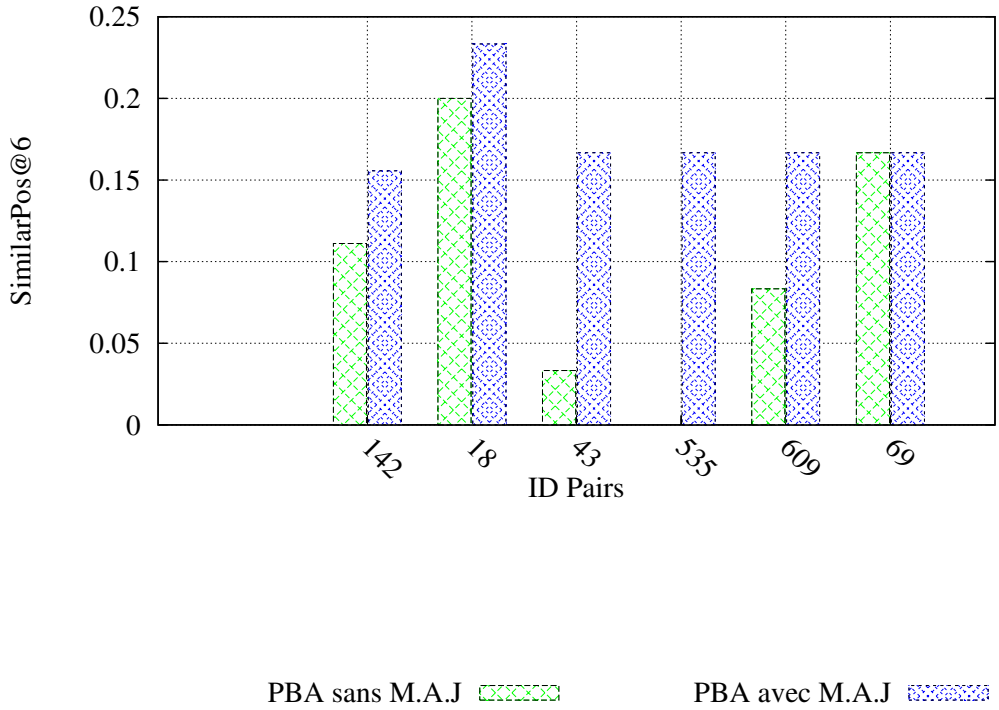


FIGURE 6.9 – Impact de la mise à jour des bases sur le taux SimilarPosition

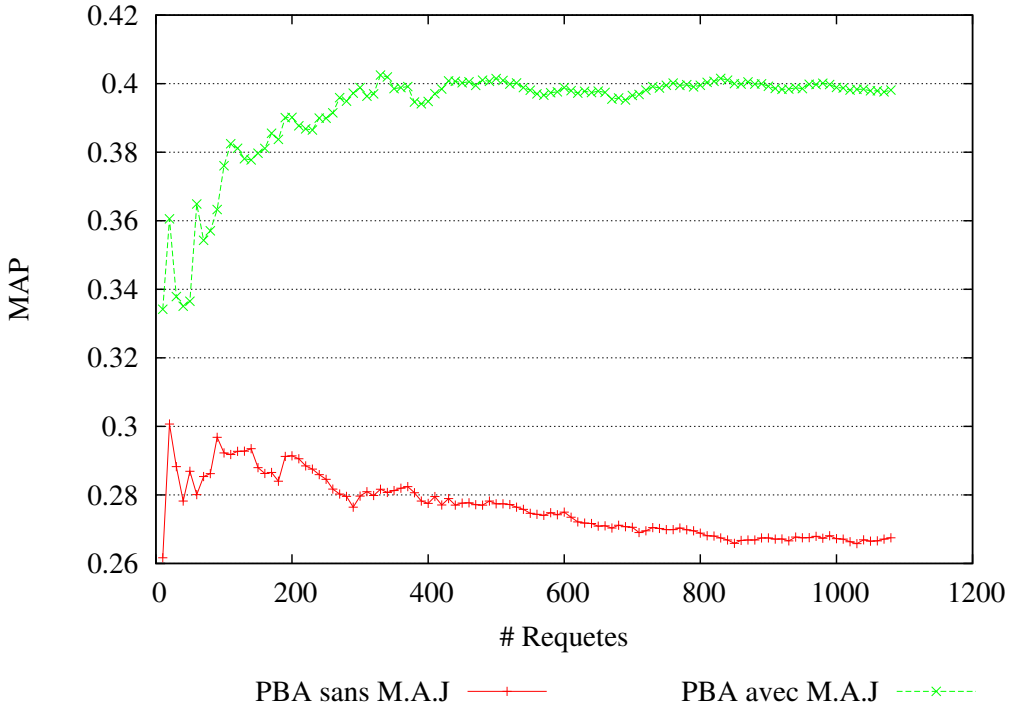


FIGURE 6.10 – Vue globale du système (amélioration de MAP)

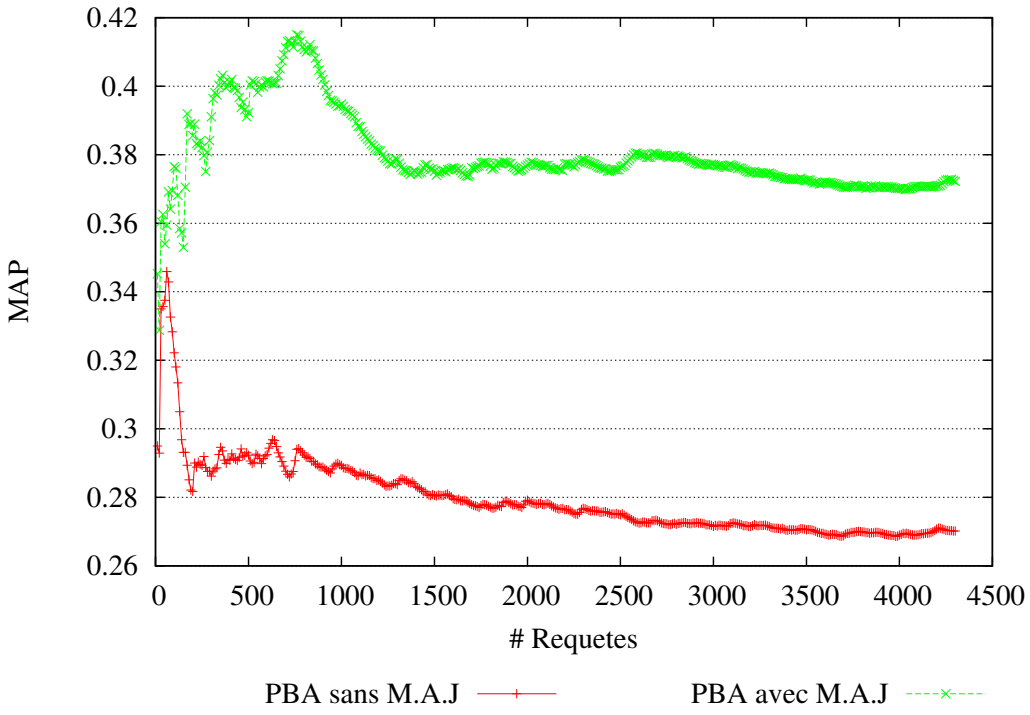


FIGURE 6.11 – Vue globale du système (amélioration de MAP) pour des requêtes nouvelles

évoluée sur l'amélioration de l'efficacité de PBA et ce, sans engendrer de coûts additionnels élevés.

Conclusion Générale

Synthèse

Les travaux présentés dans cette thèse s'inscrivent dans le contexte général de la Recherche d'Information Distribuée à Large Échelle (RIDLE). Plus particulièrement, nous avons abordé la problématique d'agrégation de résultats dans les systèmes de Recherche d'Information organisés sous forme d'un système Pair-à-Pair non structuré.
Dans de tels systèmes, aucune vision globale n'est détenue par les pairs du système, qui sont souvent des entités hétérogènes, autonomes et dynamiques, du point de vue de leurs connexions et déconnexions du système.
L'objectif principal recherché à travers cette thèse était d'apporter des contributions sur deux axes complémentaires : d'abord, proposer une nouvelle vision d'agrégation de résultats dans un contexte large échelle. Ensuite, il s'agit d'améliorer le cadre d'évaluation dans ce même contexte, à savoir les systèmes distribués à large échelle.
C'est ainsi, que nous avons proposé *une approche de score de classement distribuée* qui repose sur *l'hybridation* des poids des documents (basés sur leurs rangs dans les listes de réponse des pairs contributeurs), avec deux types de scores (de documents et de pairs) déduits des profils des utilisateurs. L'intuition derrière cette hybridation était de concilier différents types de confiance :

- La confiance que nous donnons à un document suite à son classement à un rang donné par un pair contributeur.
- La confiance que nous accordons à un document suite à sa consultation ou téléchargement lors de la réponse à une requête passée.
- La confiance que nous attribuons à un pair suite à sa capacité à bien répondre à des requêtes similaires dans le passé (d'un point de vue utilisateur).

L'utilisation des scores basés sur les profils nous a amené à concevoir *un modèle de profils* qui repose sur l'Analyse Formelle des Concepts (AFC). Des concepts augmentés ont été construits à partir de la jointure d'un ensemble de concepts formels. Ainsi, un profil a été représenté sous forme d'un triplet (T, P, D), regroupant l'ensemble T des termes d'un ensemble de requêtes soumises, les identifiants de l'ensemble P des pairs sollicités pour la résolution des termes de T, et les identifiants de documents D utilisés pour répondre aux requêtes soumises.
Un ensemble d'opérateurs permettant de sélectionner des parties d'un profil, de joindre des concepts, de l'augmenter et de calculer sa similarité avec un objet ou attribut ont été également proposés.

Nous rappelons à ce niveau que la génération des profils se réalise de *manière purement locale* au niveau d'un pair du système alors que l'utilisation des profils est faite également purement localement par *chaque pair initiateur* qui a soumis des requêtes.

Il convient de noter dans ce contexte, que réussir une recommandation basée sur des connaissances acquises dépend de la cohérence entre l'état courant du système (les besoins des utilisateurs, l'ensemble des pairs et des documents associés) et l'état de la base de connaissances.
Or, l'état du système évolue dans le temps (nouveaux besoins, pairs entrants ou sortants du système, ajouts/retraits de documents). Nous avons proposé une approche d'évolution des connaissances qui utilise un ensemble de détecteurs déclenchant des alertes lorsqu'il n'y a plus adéquation entre l'état de la base de connaissances et l'état observé du système. Un système de contrôle, tournant en arrière plan des opérations d'agrégation, va agréger les alertes produites par les détecteurs pour estimer la divergence par rapport à l'état de la base de connaissances. Au delà d'un certain niveau de divergence, une régénération de la base de connaissances sera alors mise en oeuvre.

Pour valider ces propositions, un prototype d'évaluation, basé sur le simulateur PeerSim-RARE, a été implementé en intégrant de nouvelles couches nécessaires à notre application et une démarche de validation par simulation a été définie. Cette démarche comprend les étapes suivantes :

- La préparation des jeux de données, qui nécessite la distribution des données.
- La configuration d'un fichier pour la simulation.
- La simulation.
- L'exploitation des résultats.

L'utilisation des modèles de distribution vise à créer des collections de tests distribuées qui reproduisent les caractéristiques liées aux systèmes P2P cibles, à savoir, l'hétérogénéité sur les collections du point de vue contenu (thème spécifique au pair ou général), la taille (uniforme ou hétérogène), la méthode de recherche (même méthode de calcul de similarité ou méthodes différentes).

Une série d'expérimentations sur trois collections distribuées de test, selon différents modèles, a été réalisée. La démarche d'évaluation que nous avons suivie respecte un ensemble de scénarii réalistes, notamment :

- L'utilisation des collections hétérogènes vs des collections homogènes.
- L'utilisation de distributions spécialisant les pairs vs des distributions statistiques.
- L'introduction de nouveaux besoins utilisateurs pour détecter les évolutions de la base de connaissances.

Les différents scénarii réalisés ont prouvé la validité de notre approche. En effet, elle peut être déployée dans toutes les circonstances, sans engendrer des pertes significatives d'efficacité par rapport aux approches classiques testées.
La série d'expérimentations réalisées nous a permis de déduire que, dans les situations d'hétérogénéité, notre approche se comporte mieux que les modèles à base de rangs et même ceux à base de score.
De plus, il convient de rappeler le coût minimum de notre approche, puisqu'elle

évite tout échange explicite de données avec les autres pairs durant la recherche et que la construction de la base de connaissances est également purement locale.

Perspectives

Sur la base des résultats obtenus, nous proposons de poursuivre nos tableaux dans les directions suivantes :

1. En ce qui concerne l'utilisation des profils utilisateurs dans les systèmes de RIDLE, nous estimons que plus les profils sont riches, plus l'efficacité du système sera bonne. Ainsi, l'approche de construction de profils peut être étudiée d'avantage, afin d'y introduire plus d'informations. Des informations supplémentaires, comme les préférences négatives de l'utilisateur, (par exemple les documents qui n'ont pas été sollicités par lui) pourraient être intégrées pour définir plus finement ses intérêts.
2. La formule de choix de score a été définie de manière linéaire essayant ainsi de renforcer les poids des documents par la confiance que nous pouvons accorder au pair contributeur. Cependant, en réalité plusieurs autres formules de scores sont possibles [Egghe 2010, Robertson 1995]. Dans [Koza 1992], Koza propose d'utiliser une approche d'apprentissage basée sur la programmation génétique pour définir une fonction de scoring (fonction de similarité document-requête) à base de paramètres tels que TF et IDF. Par analogie, nous pensons qu'il serait intéressant d'utiliser les paramètres de notre formule (le score du pair, du document et le rang) pour chercher la meilleure formule envisageable possible avec différents types d'opérateurs (+,-, Log, etc.) pour classer le plus efficacement possible les résultats agrégés.
3. Dans la partie gestion de la connaissance sur les comportements des utilisateurs, nous avons proposé de travailler sur un ou plusieurs détecteurs de défauts. Cependant, dans notre étude expérimentale, nous avons testé un seul détecteur à la fois. Ainsi, nous visons, dans de futurs travaux ,de combiner les différents détecteurs pour étudier leurs impacts sur la qualité de la base de connaissances.
4. Nous envisageons aussi de proposer, à court terme une autre approche de maintenance de la base de connaissances. En effet, l'objectif principal est de prédire les intentions futures de recherches des utilisateurs afin de décider de l'utilité de la maintenance à un instant donné. Nous avons déjà commencé à utiliser les modèles de markov cachés (MMC) [Baum 1966] pour la prédiction de ces intentions.

Bibliographie

[Adamic 2002] Lada A. Adamic and Bernardo A. Huberman. *Zipf's law and the Internet*. Glottometrics, vol. 3, pages 143–150, 2002. (Cité en pages 74, 75 et 122.)

[add online 2009] add online. *L'histoire d'Internet*. http ://www.addonline.fr/creation-site-web-lyon–271.html, 2009. (Cité en pages 1, 9 et 10.)

[Agrawal 1993] Rakesh Agrawal, Tomasz Imieliński and Arun Swami. *Mining association rules between sets of items in large databases*. SIGMOD Rec., vol. 22, no. 2, pages 207–216, Juin 1993. (Cité en page 50.)

[allmusic 2012] allmusic. *rovi*. http ://www.allmusic.com/, 2012. (Cité en page 86.)

[Andrieu 2011] Olivier Andrieu. *Moteurs de recherche web - Google, Bing et leurs challengers*. http ://www.techniques-ingenieur.fr/base-documentaire/technologies-de-l-information-th9/gestion-de-contenus-numeriques-42311210/moteurs-de-recherche-web-h7240/, mai 2011. (Cité en pages 2 et 8.)

[Aslam 2001] Javed A. Aslam and Mark Montague. *Models for metasearch*. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '01, pages 276–284, New York, NY, USA, September, 2001. ACM. (Cité en pages xviii, 22, 28, 29 et 30.)

[Aslam 2002] Javed Aslam, Jamie Callan, R Manmatha, Mark Sanderson and Ellen Voorhees. *MetaSearch : Data Fusion and Distributed Retrieval*. In Workshop on Challenges in Information. Amherst. Massachusetts. USA, 2002. (Cité en pages 22 et 28.)

[Baeza-Yates 1999] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. Modern information retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. (Cité en pages 2 et 3.)

[Balke 2005] Wolf-Tilo Balke, W Nejdl, W Siberski and U Thaden. *DL meets P2P - Distributed Document Retrieval based on Classification and Content*. In European Conference on Digital Libraries (ECDL), Vienna, Austria„ September, 2005. (Cité en page 15.)

[Baum 1966] Leonard E. Baum and Ted Petrie. *Statistical Inference for Probabilistic Functions of Finite State Markov Chains*. The Annals of Mathematical Statistics, vol. 37, no. 6, pages 1554–1563, 1966. (Cité en page 133.)

[Bawa 2003] Mayank Bawa, Gurmeet Singh Manku and Prabhakar Raghavan. *SETS : search enhanced by topic segmentation*. In SIGIR '03 : Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, Toronto, Canada, pages 306–313. ACM, July 28 2003. (Cité en page 72.)

[Beagle2 2008] Beagle2. *Beagle++*. http ://beagle2.kbs.uni-hannover.de/, 2008. (Cité en page 13.)

[Ben Yahia 2009] Sadok Ben Yahia, Ghada GASMI and Engelbert Mephu Nguifo. *A new generic basis of factual and implicative association rules*. Intelligent Data Analysis - An International Journal(IDA), vol. 13, no. 4, pages 633–656, jul 2009. DOI 10.3233/IDA-2009-0384. (Cité en page 50.)

[Bergman 2001] Michael K. Bergman. *The Deep Web : Surfacing Hidden Value*. Journal of Electronic Publishing, vol. 7, no. 1, August 2001. (Cité en page 8.)

[Boole 1848] George Boole. *The Calculus of Logic*. The Cambridge and Dublin Mathematical Journal, vol. 3, 1848. (Cité en page 5.)

[Borda 1781] Jean-Charles Borda. *Mémoire sur les elections au scrutin*. In Histoire de l'academie Royale des Sciences, 1781. (Cité en pages 22, 33, 71 et 80.)

[Bouzeghoub 2005] Mokrane Bouzeghoub and Dimitre Kostadinov. *Personnalisation de l'information : aperçu de l'état de l'art et définition d'un modèle flexible de profils*. In CORIA, pages 201–218, 2005. (Cité en pages 46, 48 et 49.)

[Brin 1998] Sergey Brin and Lawrence Page. *The anatomy of a large-scale hypertextual Web search engine*. In Proceedings of the seventh international conference on World Wide Web 7, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V. (Cité en page 8.)

[Broder 2002] Andrei Broder and Michael Mitzenmacher. *Network Applications of Bloom Filters : A Survey*. In Internet Mathematics, pages 636–646, 2002. (Cité en page 31.)

[Callan 1995] James P. Callan, Zhihong Lu and W. Bruce Croft. *Searching Distributed Collections with Inference Networks*. In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 21–28. ACM Press, 1995. (Cité en pages 12, 24, 31, 37 et 63.)

[Callan 2000] Jamie Callan. *Distributed information retrieval*. In W. Bruce Croft, editeur, Advances in information retrieval, pages 127–150. Kluwer, 2000. (Cité en pages 12 et 37.)

[Cappello 2005] Franck Cappello, Eddy Caron, Michel Dayde, Frederic Desprez, Emmanuel Jeannot, Yvon Jegou, Stephane Lanteri, Julien Leduc, Nouredine Melab, Guillaume Mornet, Raymond Namyst, Pascale Primet and Olivier Richard. *Grid'5000 : a large scale, reconfigurable, controlable and monitorable Grid platform*. In SAC'05 : Proc. The 6th IEEE/ACM International Workshop on Grid Computing Grid'2005, pages 99–106, Seattle, USA, Novembre 2005. IEEE/ACM. (Cité en page 76.)

[center 2012] Microsoft News center. *Bing Newsroom*. http ://www.microsoft.com/en-us/news/presskits/bing/Default.aspx, September 2012. (Cité en page 2.)

[Chernov 2005] Sergey Chernov, Pavel Serdyukov, Matthias Bender, Sebastian Michel, Gerhard Weikum and Christian Zimmer. *Database selection and result merging in P2P web search.* In Proceedings of the 3rd International Workshop on Databases, Information Systems, and Peer-to-Peer Computing (DBISP2P 2005), volume 4125 of *Lecture Notes in Computer Science*, pages 26–37, Heidelberg, Germany, September 2005. Springer Verlag. (Cité en pages 18, 26, 34, 36, 37 et 38.)

[Chiclana 2007] Francisco Chiclana, Enrique Herrera-Viedma, Francisco Herrera and Sergio Alonso. *Some induced ordered weighted averaging operators and their use for solving group decision-making problems based on fuzzy preference relations.* European Journal of Operational Research, vol. 182, pages 383–399, 2007. (Cité en page 34.)

[Cleverdon 1991] Cyril W. Cleverdon. *The significance of the Cranfield tests on index languages.* In Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '91, pages 3–12, New York, NY, USA, 1991. ACM. (Cité en page 65.)

[Crespo 2002] Arturo Crespo and Hector Garcia-Molina. *Routing Indices For Peer-to-Peer Systems.* In Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02), ICDCS '02, pages 23–, Washington, DC, USA, 2002. IEEE Computer Society. (Cité en page 17.)

[Cuenca-Acuna 2003] Francisco Matias Cuenca-Acuna, Christopher Peery, Richard P. Martin and Thu D. Nguyen. *PlanetP : Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities.* In HPDC, pages 236–249. IEEE Computer Society, 2003. (Cité en pages 17, 31 et 78.)

[De 2007] Arijit De, Elizabeth D. Diaz and Vijay V. Raghavan. *On Fuzzy Result Merging for Metasearch.* In FUZZ-IEEE, pages 1–6, 2007. (Cité en page 34.)

[Dean 2009] Jeffrey Dean. *Challenges in building large-scale information retrieval systems : invited talk.* In WSDM '09 : Proceedings of the Second ACM International Conference on Web Search and Data Mining, pages 1–1, New York, NY, USA, 2009. ACM. (Cité en pages xvii et 8.)

[Defude 2007] Bruno Defude. *Organisation et routage sémantiques dans les systèmes pair-à-pair.* In Actes du XXVème Congrès INFORSID, May 22-25, Perros-Guirec, France, pages 12–8, 2007. (Cité en pages 1, 16 et 46.)

[DMOZ 2011] DMOZ. *DMOZ Open Directory Project .* `http://olc.ijs.si/dmozReadme.html`, 2011. (Cité en pages 86, 97, 101 et 122.)

[Druschel 2001] Peter Druschel and Antony Rowstron. *PAST : A Large-Scale, Persistent Peer-to-Peer Storage Utility.* pages 75–80, 2001. (Cité en page 73.)

[Dwork 2001] Cynthia Dwork, Ravi Kumar, Moni Naor and D. Sivakumar. *Rank aggregation methods for the Web.* In World Wide Web Conference Series, pages 613–622, May, 2001. (Cité en page 22.)

[Egghe 2010] Leo Egghe. *Good properties of similarity measures and their complementarity.* J. Am. Soc. Inf. Sci. Technol., vol. 61, pages 2151–2160, October 2010. (Cité en pages 4, 6, 80, 88 et 133.)

[Farah 2007] Mohamed Farah and Daniel Vanderpooten. *An outranking approach for rank aggregation in information retrieval.* In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07, pages 591–598, New York, NY, USA, 2007. ACM. (Cité en pages 33 et 71.)

[Fox 1994] Edward A Fox and Joseph A Shaw. *Combination of Multiple Searches.* In The 2nd Text Retrieval Conference TREC2 NIST SP 500215 (1994), volume 500-215, pages 243–252. National Institute for Standards and Technology, NIST, 1994. Available from www-nlpir.nist.gov. (Cité en pages xviii, 22, 23 et 36.)

[Frécon 2009] Louis Frécon and Okba Kazar. Manuel d'intelligence artificielle. METIS LyonTech. Presses polytechniques et universitaires romandes, 2009. (Cité en page 35.)

[Fuhr 1992] Norbert Fuhr. *Probabilistic Models in Information Retrieval.* The Computer Journal, vol. 35, pages 243–255, 1992. (Cité en page 6.)

[Fuller 1996] Robert Fuller. *OWA Operators in Decision Making.* In Exploring the Limits of Support Systems, volume 3, pages 85–104. TUCS General Publications, 1996. (Cité en page 34.)

[Ganter 1997] Bernhard Ganter and Rudolf Wille. Formal concept analysis : Mathematical foundations. Springer-Verlag New York, Inc, Secaucus, NJ, USA, 1st édition, 1997. (Cité en pages 50 et 51.)

[Gnu 2007] *Gnutella website.* http ://www.gnutella.com/, Janvier 2007. (Cité en pages 14, 78, 79 et 86.)

[Godin 1995] Robert Godin, Rokia Missaoui and Hassan Alaoui. *Incremental concept formation algorithms based on Galois (concept) lattices.* Computational Intelligence, vol. 11, no. 2, pages 246–267, Mai 1995. (Cité en pages 80 et 96.)

[Grappy 2012] Arnaud Grappy, Brigitte Grau and Sophie Rosset. *Fusion des réponses de systèmes de question-réponses.* In 9th French Information Retrieval Conference : Conférence en Recherche d'Infomations et Applications - CORIA 2012, pages 99–110, Bordeaux, France, March 21-23 2012. (Cité en page 29.)

[Gravano 2000] Luis Gravano, Hector Garcia-Molina and Anthony Tomasic. *Gloss : Text-Source Discovery over the Internet.* Technical Report 2000-11, Stanford InfoLab, 2000. (Cité en pages 12 et 17.)

[Greengrass 2000] Ed Greengrass. *Information Retrieval : A Survey*, 2000. (Cité en pages 5 et 6.)

[Hennion 2011] Nicolas Hennion. *Introduction aux technologies cloud*, Janvier 2011. (Cité en page xvii.)

[Informatica 2010] Informatica. *Le volume des données numériques augmente exponentiellement.* http ://evollia.com/2011/02/le-volume-des-donnees-numeriques-augmente-exponentiellement/, 2010. (Cité en pages 1, 9 et 10.)

[Jelasity 2007] Mark Jelasity, Alberto Montresor, Gian Paolo Jesi and Spyros Voulgaris. *The Peersim Simulator.* `http://peersim.sf.net`, 2007. (Cité en pages 77 et 83.)

[Johnson 1972] Phillip. E. Johnson. A history of set theory. Prindle, Weber & Schmidt complementary series in mathematics. Prindle, Weber & Schmidt, 1972. (Cité en page 5.)

[Knuth 1997] Donald Knuth. *Sorting and Searching.* The Art of Computer Programming, vol. 3, pages 158–160, 1997. (Cité en page 22.)

[Köhler 2006] Jacob Köhler, Stephan Philippi, Michael Specht and Alexander Rüegg. *Ontology based text indexing and querying for the semantic web.* Know.-Based Syst., vol. 19, pages 744–754, December 2006. (Cité en page 3.)

[Kostadinov 2006] Dimitre Kostadinov. *Data Personalization : an approach for profile management and query reformulation.* 2006. (Cité en page 48.)

[Koza 1992] John R. Koza. Genetic programming : on the programming of computers by means of natural selection. MIT Press, Cambridge, MA, USA, 1992. (Cité en page 133.)

[Lan 2005] Man Lan, Sam yuan Sung, Hwee boon Low and Chew lim Tan. *A comprehensive comparative study on term weighting schemes for text categorization with support vector machines.* In Posters Proc. 14th International World Wide Web Conference, pages 1032–1033, 2005. (Cité en page 6.)

[Lancaster 1973] F.Wilfrid Lancaster and Emily Gallup. Information retrieval : online. Information sciences series. Melville Pub. Co., 1973. (Cité en page 9.)

[Larkey 2000] Leah S. Larkey, Margaret E. Connell and Jamie Callan. *Collection selection and results merging with topically organized U.S. patents and TREC data.* In Proceedings of the ninth international conference on Information and knowledge management, CIKM '00, pages 282–289, New York, NY, USA, 2000. ACM. (Cité en page 12.)

[Le Calvé 2000] Anne Le Calvé and Jacques Savoy. *Database merging strategy based on logistic regression.* Inf. Process. Manage., vol. 36, pages 341–359, May 2000. (Cité en pages 28, 30 et 40.)

[Lefebure 2001] René Lefebure and Gilles Venturi. Data mining. gestion de la relation client, personnalisation des sites web. deuxième édition, 2001. (Cité en pages 35, 49 et 50.)

[Leuf 2002] Bo Leuf. Peer to peer : Collaboration and sharing over the internet. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002. (Cité en page xvii.)

[Liang 2004] Jian Liang, Rakesh Kumar and Keith W Ross. *Understanding KaZaA*, 2004. (Cité en page 15.)

[Liu 2007] Yu-Ting Liu, Tie-Yan Liu, Tao Qin, Zhi-Ming Ma and Hang Li. *Supervised rank aggregation*. In Proceedings of the 16th international conference on World Wide Web, WWW '07, pages 481–490, New York, NY, USA, 2007. ACM. (Cité en page 28.)

[Loser 2007] Alexander Loser, Steffen Staab and Christoph Tempich. *Semantic social overlay networks*. IEEE Journal on Selected Areas in Communications (ISACEM) : Special Issue Peer-to-Peer Communications and Applications, vol. 25, pages 5–14, January , 2007. (Cité en pages 17 et 18.)

[Lu 2003] Jie Lu and James P. Callan. *Content-based retrieval in hybrid peer-to-peer networks*. In CIKM, pages 199–206, New Orleans, Louisiana, USA., November 3-8 2003. (Cité en pages 72 et 122.)

[Lu 2005] Yiyao Lu, Weiyi Meng, Liangcai Shu, Clement Yu and King-Lup Liu. *Evaluation of result merging strategies for metasearch engines*. In Proceedings of the 6th international conference on Web Information Systems Engineering, WISE'05, pages 53–66, Berlin, Heidelberg, 2005. Springer-Verlag. (Cité en page 7.)

[Lübke 2012] Robert Lübke, Robin Lungwitz, Daniel Schuster and Alexander Schill. *Emulation of Complex Network Infrastructures for Large-Scale Testing of Distributed Systems*. In IADIS WWW/Internet 2012 Conference, Madrid, Spain, 10 2012. (Cité en page 76.)

[MacQueen 1967] James B. MacQueen. *Some Methods for Classification and Analysis of MultiVariate Observations*. In L. M. Le Cam and J. Neyman, editeurs, Proceeding of the fifth Berkeley Symposium on Mathematical Statistics and Probability, volume 1, pages 281–297, Berkeley, California, Jannuary 1967. University of California Press. (Cité en page 35.)

[Marcotorchino 1982] Jean-François Marcotorchino and Pierre Michaud. *Agregation de similarites en clasiification automatique*. Revue des statistiques appliquees, vol. tome 30, 2, pages 21–44, 1982. (Cité en page 22.)

[Mghirbi 2009] Rim Mghirbi. *Recherche d'information distribuée : Fusion des résultats dans les systèmes P2P*. In Inforsid, pages 493–494. 2009. (Cité en page 22.)

[Mghirbi 2010a] Rim Mghirbi, Khedija Arour, Yahya Slimani and Bruno Defude. *A profile-based aggregation model in a peer-to-peer information retrieval system*. In Proceedings of the Third international conference on Data management in grid and peer-to-peer systems, Globe'10, pages 148–159, Bilbao, Spain, August 2010. Springer-Verlag. (Cité en page 95.)

[Mghirbi 2010b] Rim Mghirbi, khedija Arour, Yahya Slimani and Bruno Defude. *Modèle d'interclassement de résultats basé sur les profils des utilisateurs dans un SRI-P2P*. In CARI, Yamoussoukrou, Côte d'Ivoire, October 2010. (Cité en page 86.)

[Mghirbi 2011] Rim Mghirbi, Khedija Arour, Yahya Slimani and Bruno Defude. *An Hybrid Results Merging Model for P2PIR Systems.* Knowledge And Information Systems (KAIS), soumis december, 2011. (Cité en page 26.)

[Mghirbi 2012] Rim Mghirbi, Hanen Majdoub andKhedija Arour and Bruno Defude. *A Controlled knowledge base evolution approach for query hits merging in P2P systems.* International Journal of Advanced Science and Technology (IJAST), October 2012. (Cité en page 117.)

[Montague 2002] Mark Montague. Metasearch : data fusion for document retrieval. Dartmouth College, 2002. (Cité en page 28.)

[Mooers 1950] Calvin N. Mooers. The theory of digital handling of non-numerical information and its implications to machine economics, volume 48 of *Zator technical bulletin*. Zator Co., 1950. (Cité en page 1.)

[Nakauchi 2004] Kiyohide Nakauchi, Yuichi Ishikawa, Hiroyuki Morikawa and Tomonori Aoyama. *Exploiting semantics in unstructured P2P systems.* IEICE Trans. Comm, vol. 87, pages 1806–1817, 2004. (Cité en pages 17 et 18.)

[Nap 2006] *Napster*. http ://www.napster.com/, 2006. (Cité en pages xvii, 11, 13 et 14.)

[Neumann 2006] Thomas Neumann, Matthias Bender, Sebastian Michel and Gerhard Weikum. *A Reproducible Benchmark for P2P Retrieval.* In Proc. of the First International Workshop on Performance and Evaluation of Data Management Systems, ExpDB 2006 at ACM SIGMOD, Chicago, pages 1–8, June 30 2006. (Cité en page 72.)

[Page 1998] Larry Page and Sergey Brin. *Google.* http ://www.google.com/about/company/, september 1998. google sociÃ©tÃ©. (Cité en pages 2 et 8.)

[Pla 2002a] *PlanetLab Phase 0 : Technical Specification.* Rapport technique PDN–02–002, PlanetLab Consortium, August 2002. (Cité en page 76.)

[Pla 2002b] *PlanetLab website.* http ://www.planet-lab.org/, Mars 2002. (Cité en page 76.)

[Poelmans 2010] Jonas Poelmans, Paul Elzinga, Stijn Viaene and Guido Dedene. *Formal Concept Analysis in Knowledge Discovery : A Survey.* In Madalina Croitoru, Sebastien Ferre and Dickson Lukose, editeurs, ICCS, volume 6208 of *Lecture Notes in Computer Science*, pages 139–153. Springer, 2010. (Cité en page 50.)

[Ponte 1998] Jay M. Ponte and Bruce W. Croft. *A language modeling approach to information retrieval.* In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '98, pages 275–281, New York, NY, USA, 1998. ACM. (Cité en pages 3, 31, 32, 37 et 63.)

[PubMedline 2012] PubMedline. *U.S. National Library of Medicine National Institutes of Health.* http ://www.nlm.nih.gov/, AoÃ»t 2012. (Cité en page 2.)

[RARE 2007] RARE. *Le projet RARE (Routage optimisé par Apprentissage de REquêtes)*. http://www-inf.int-evry.fr/defude/RARE/, 2007. (Cité en pages 76, 77, 83 et 86.)

[Ratnasamy 2001] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp and Scott Shenker. *A scalable content-addressable network*. SIGCOMM Comput. Commun. Rev., vol. 31, pages 161–172, August 2001. (Cité en page 15.)

[Renda 2003] Elena M. Renda and Umberto Straccia. *Web metasearch : rank vs. score based rank aggregation methods*. In Proceedings of the 2003 ACM symposium on Applied computing, SAC 03, pages 841–846, New York, NY, USA, March, 2003. ACM. (Cité en pages 28, 29, 63, 68, 71 et 104.)

[Robertson 1976] Stephen E. Robertson and Karen Sparck Jones. *Relevance weighting of search terms*. Journal of the American Society for Information Science, vol. 27, no. 3, pages 129–146, 1976. (Cité en pages 6 et 7.)

[Robertson 1995] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Mike Gatford and A. Payne. *Okapi at TREC-4*. In The Fourth Text REtrieval Conference (TREC4), pages 73–96, 1995. (Cité en pages 4, 7, 31, 38, 71 et 133.)

[Rocchio 1971] J. J. Rocchio. *Relevance feedback in information retrieval*. In G. Salton, editeur, The Smart retrieval system - experiments in automatic document processing, pages 313–323. Englewood Cliffs, NJ : Prentice-Hall, 1971. (Cité en pages 4 et 36.)

[Rowstron 2001] Antony Rowstron and Peter Druschel. *Pastry : Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems*. Lecture Notes in Computer Science, vol. 2218, pages 329–350, 2001. (Cité en page 15.)

[Salton 1975] Gerard. Salton, A. Wang and C. S. Yang. *A vector space model for automatic indexing*. In Communication of the ACM, pages 613–620, 1975. (Cité en pages 3, 6, 9, 31, 71 et 80.)

[Salton 1989] Gerard Salton. Automatic text processing – the transformation, analysis, and retrieval of information by computer. Addison–Wesley, 1989. (Cité en pages 2, 4 et 54.)

[Savoy 2001] Jacques Savoy, Yves Rasolofo and Faiza Abbaci. *Recherche d'informations dans des sources distribuées*. In INFORSID, pages 237–252, 2001. (Cité en pages 3 et 7.)

[Savoy 2002] Jacques Savoy, Yves Rasolofo and Faiza Abbaci. *Fusion de collections dans les metamoteurs*. In JADT2002 : 6es Journees internationales d'Analyse statistique des Donnees Textuelle, 2002. (Cité en pages 7 et 33.)

[Shokouhi 2007] Milad Shokouhi, Justin Zobel and Yaniv Bernstein. *Distributed text retrieval from overlapping collections*. In Proceedings of the eighteenth conference on Australasian database - Volume 63, ADC '07, pages 141–150,

Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc. (Cité en pages 24, 26 et 30.)

[Si 2002a] Luo Si and Jamie Callan. *Using sampled data and regression to merge search engine results.* In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '02, pages 19–26, New York, NY, USA, 2002. ACM. (Cité en pages 30, 40 et 63.)

[Si 2002b] Luo Si, Rong Jin, James P. Callan and Paul Ogilvie. *A language modeling framework for resource selection and results merging.* In Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management, McLean, VA, USA, November 4-9, pages 391–397, 2002. (Cité en pages 12 et 31.)

[Si 2003] Luo Si and Jamie Callan. *A semisupervised learning method to merge search engine results.* ACM Trans. Inf. Syst., vol. 21, pages 457–491, October 2003. (Cité en pages 30, 32 et 63.)

[Steidinger 2000] Alexander Steidinger. *Comparison of different Collection Fusion Models in Distributed Information Retrieval.* In DELOS Workshop : Information Seeking, Searching and Querying in Digital Libraries, 2000. (Cité en pages 32, 33 et 35.)

[Stoica 2001] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek and Hari Balakrishnan. *Chord : A scalable peer-to-peer lookup service for internet applications.* SIGCOMM Comput. Commun. Rev., vol. 31, pages 149–160, August 2001. (Cité en pages 15 et 36.)

[Tanenbaum 2006] Andrew S. Tanenbaum and Maarten van Steen. Distributed systems : Principles and paradigms (2nd edition). Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006. (Cité en pages xvii et 26.)

[Theodoratos 1999] Dimitri Theodoratos and Mokrane Bouzeghoub. *Data Currency Quality Factors in Data Warehouse Design.* In In Proc. of the Int. Workshop on Design and Management of Data Warehouses (DMDW'99), Heidelberg, Germany., Heidelberg, Germany, 1999. (Cité en page 116.)

[TREC 2008] TREC. *Text REtrival Conference.* `http://trec.nist.gov/`, 2008. (Cité en pages 65, 72 et 86.)

[Valtchev 2003] Petko Valtchev, David Grosser, Cyril Roume and Mohamed Rouane Hacene. *Galicia : an open platform for lattices.* In Using Conceptual Structures : Contributions to the 11th Conference on Conceptual Structures (ICCS'03, pages 241–254. Shaker Verlag, 2003. (Cité en pages 80 et 96.)

[Voorhees 1995a] Ellen M. Voorhees, Narendra K. Gupta and Ben Johnson-Laird. *Learning collection fusion strategies.* In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '95, pages 172–179, Seattle, Washington, United States, 1995. ACM. (Cité en pages 32, 71 et 80.)

[Voorhees 1995b] Ellen M. Voorhees, Narendra K. Gupta and Johnson B. Laird. *The Collection Fusion Problem.* In Proceedings of the Third Text REtrieval Conference (TREC-3), pages 95–104, Gaithersburg, MD, 1995. (Cité en pages 32 et 71.)

[Voorhees 2005] Ellen M. Voorhees and Donna K. Harman. Trec : experiment and evaluation in information retrieval. The MIT Press, 2005. (Cité en pages xxi et 68.)

[Voulgaris 2004] Spyros Voulgaris, Anne-Marie. Kermarrec, Laurent Massoulie and Maarten van Steen. *Exploiting Semantic Proximity in Peer-to-peer Content Searching.* In Proc. 10th IEEE Int'l Workshop on Future Trends in Distributed Computing Systems (FTDCS 2004), Suzhou, China, Mai 2004. (Cité en pages 17 et 18.)

[Wahlster 1986] Wolfgang Wahlster and Alfred Kobsa. *Dialog-Based User Models.* In Proceedings of the IEEE, Special Issue on Natural Language Processing, pages 948–960, 1986. (Cité en pages 46 et 48.)

[Wan 2010] Xin Wan, Qimanguli Jamaliding, Fumihiko Anma and Toshio Okamoto. *Applying Keyword Map Based Learner Profile to a Recommender System for Group Learning Support.* volume 1, pages 3–6, Los Alamitos, CA, USA, 2010. IEEE Computer Society. (Cité en page 49.)

[Wang 2009] Long Wang and Zhenkai Wan. *Secure P2P File Sharing System Based on Full-Text Retrieval.* In Proceedings of the 2009 First IEEE International Conference on Information Science and Engineering, ICISE '09, pages 380–383, Washington, DC, USA, 2009. IEEE Computer Society. (Cité en page 13.)

[Witschel 2005] Hans Friedrich Witschel and Thomas Böhme. *Evaluating profiling and query expansion methods for P2P information retrieval.* In Proceedings of the 2005 ACM workshop on Information retrieval in peer-to-peer networks, P2PIR '05, pages 1–8, New York, NY, USA, 2005. ACM. (Cité en pages 18, 36, 37, 38 et 40.)

[Witschel 2008] Hans Friedrich Witschel. *Global and Local Resources for Peer-to-Peer Text Retrieval.* Phd. thesis, Mathematik und Informatik. Leipzig eingereichte, 2008. (Cité en pages 18, 36, 37, 38, 39, 40, 68 et 71.)

[Xu 1996] Jinxi Xu and Bruce W. Croft. *Query Expansion Using Local and Global Document Analysis.* In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 4–11, 1996. (Cité en page 4.)

[Yee 2005] Wai Gen Yee and Ophir Frieder. *On search in peer-to-peer file sharing systems.* In Proceedings of the 2005 ACM symposium on Applied computing, SAC '05, pages 1023–1030, New York, NY, USA, March, 2005. ACM. (Cité en page xvii.)

[Yeferny 2009] Taoufik Yeferny, Khedija Arour and Yahya Slimani. *Routage sémantique des requêtes dans les systèmes pair-à-pair.* In CORIA'09, pages 131–147, 2009. (Cité en page 80.)

[Zadeh 1965] Lofti Zadeh. *Fuzzy sets.* Information and Control, vol. 8, no. 3, pages 338–353, Juin 1965. (Cité en page 33.)

[Zammali 2010] Saloua Zammali and Khedija Arour. *P2PIRB : Benchmarking Framework for P2PIR.* In Proceedings of the Third international conference on Data management in grid and peer-to-peer systems, Globe'10, pages 100–111, Berlin, Heidelberg, 2010. Springer-Verlag. (Cité en page 72.)

[Zhang 2001] Jian Zhang, Jianfeng Gao, Ming Zhou and Jiaxing Wang. *Improving the Effectiveness of Information Retrieval with Clustering and Fusion.* Computational Linguistics and Chinese Language Processing, vol. 6, no. 1, pages 1–18, 2001. (Cité en pages 35 et 36.)

Résumé : Une grande partie de l'impulsion de diverses technologies d'Internet par les systèmes Pair-à-Pair (Peer-to-Peer ou P2P) peut être vue comme une réaction au détriment du centrage de contenu sur les serveurs devant des clients passifs. Une des caractéristiques distinctives de tout système P2P est ce que nous appelons souvent connectivité directe de bout en bout entre pairs égaux. Le Pair-à-Pair a augmenté les débits des échanges entre des communautés dynamiques des utilisateurs qui tendent à augmenter rapidement. Nous parlons donc de systèmes distribués à large échelle dans lesquels l'information échangée, partagée et recherchée atteint des volumes de plus en plus impressionnants. Dans le cadre de cette thèse, nous nous intéressons essentiellement à la Recherche d'Information dans les systèmes de Recherche d'Information P2P (RIP2P) et plus précisément au problème d'agrégation des résultats dans de tels systèmes. Résoudre le problème d'agrégation en RIP2P de la même manière que sa résolution dans un cadre de Recherche d'Information Distribuée (RID) va manquer beaucoup d'intelligibilité. En effet, ça fait perdre de vue tout un contexte qui a changé en RIP2P, vu le facteur d'échelle et l'absence d'une vision globale sur le système, dans ces réseaux qui s'étendent naturellement à des milliers voire des millions de pairs. Ceci va impliquer notamment la suppression d'un serveur courtier inadéquat dans ce contexte et va soulever le problème de retrouver de nouvelles politiques pour agréger des résultats provenant de pairs hétérogènes dans une liste unique tout en reflétant les attentes de l'utilisateur. Toutes ces raisons nous ont incités à explorer un mécanisme d'agrégation basé sur les profils des utilisateurs déduits de leurs comportements passés suite à leurs interactions avec les résultats d'une requête. Dans cette thèse nos contributions portent sur deux axes complémentaires. D'abord, nous proposons une nouvelle vision d'agrégation de résultats dans un contexte large échelle. Dans ce cadre un modèle de profils et une approche de score hybride à base de profils sont proposés. Ensuite nous avons mis l'accent sur la mise en place d'un cadre d'évaluation de notre approche dans les systèmes à large échelle.

Abstract : A huge part of the impetus of various internet technologies through the Peer-to-Peer (Peer-to-Peer or P2P) system can be seen as a reaction to the content centering detriment on the servers in front of passive clients. One of the distinctive features of any P2P system is what we often call direct connectivity between equal peers. The Peer-to-Peer increased the exchange flows between dynamic communities of users which tend to grow rapidly. We talk, therefore, about large-scale distributed systems in which the exchanged, shared and sought information reaches a more and more impressive volumes. Solving the aggregation problem in P2PIR systems the same way as its resolution in Distributed Information Retrieval (DIR) will miss a lot of intelligibility. In fact, the context has changed in RIP2P, given the scale factor and the lack of a global vision of the system in these networks that extend naturally to thousands or even millions peers. This will involve the removal of a broker server that is inadequate in this context and will raise the problem of finding new policies to aggregate results coming from heterogeneous peers in a single list while reflecting the user's expectations. All these reasons prompted us to explore an aggregation mechanism based on user profiles deduced from their past behavior due to their interaction with query results. Our contributions, in this thesis, focus on two complementary axes. First, we propose a new vision of results aggregation in a large scale system. In this context, a profiles model and a hybrid score profiles-based approach are proposed. Second, we focused on the development of an evaluation framework of our approach in large-scale systems. In this thesis, we are mainly interested in the Information Retrieval problem in P2P systems (P2PIR) and focusing more specifically on the problem of results' aggregation in such systems.