Gestion de méga-données : passage à l'échelle et qualité pour divers cas d'usage

Hubert Naacke

SORBONNE UNIVERSITÉ

Laboratoire LIP6

Habilitation à diriger des recherches

Discipline : Informatique

Gestion de méga-données : passage à l'échelle et qualité pour divers cas d'usage

Hubert Naacke

Rapporteurs :	Marta Rukoz	Professeure des Universités	Université Paris Nanterre
	Florent Masseglia	Directeur de recherche	INRIA - Université de Montpellier 2
	Pascal Molli	Professeur des Universités	Université de Nantes
Examinateurs :	Karine Zeitouni	Professeure des Universités	UVSQ, Université Paris Saclay
	Gabriel Antoniu	Directeur de recherche	INRIA - IRISA
	Pierre Sens	Professeur des Universités	Sorbonne Université

date de soutenance : 28 janvier 2022

document mis à jour le 16 février 2022

Table des matières

1 Introduction

1.1 Contexte

Maîtriser la donnée est un des grands défis sociétaux de cette décennie avec l'objectif de disposer d'une donnée plus accessible, propre, intelligible, *etc*.. La donnée est considérée comme une matière première indispensable pour prendre de meilleures décisions à tous les niveaux de la société. Un monde maîtrisant ses données de bout en bout, détiendrait un atout majeur pour relever les nombreux défis humains, environnementaux et économiques fixés par l'ONU pour l'horizon 2030[1]. La communauté des chercheurs en bases de données, c'est-à-dire en modélisation et gestion de données, mûrit une réflexion approfondie et régulièrement actualisée sur la traduction de ces défis sociétaux en défis scientifiques [4, 3, 19] qui considèrent l'impact sur la gestion de données, des évolutions récentes du matériel informatique (*e.g.*, processeurs, mémoire, stockage persistent), des usages (*e.g.*, la démocratisation de la science des données), des modes de déploiement (*e.g.*, services cloud). Pour la plupart des défis scientifiques identifiés, la notion de masse de données est évoquée de manière transversale et récurrente.

Dans ce document d'habilitation à diriger des recherches, je présente mes travaux effectués entre 2005 et 2021 et se situant dans le domaine du Big Data, ou *méga-données*, avec un accent mis sur la manipulation efficaces de grandes masses de données. Il s'agit de concevoir ou améliorer les éléments d'un système de gestion de bases de données (SGBD) considéré au sens large, c'est à dire incluant d'une part les SGBD relationnels et d'autre part les systèmes de gestion tenant compte des contraintes inhérentes aux méga-données : les systèmes dits NOSQL et NewSQL dont une définition plus précise est indiquée ci-dessous.

Les SGBD relationnels ont été conçus, dans les années 1980-2000, pour gérer des données dont le volume et la charge des requêtes étaient supposés stables. L'hypothèse admise étant

[1] `https://sdgs.un.org/fr/goals`

que l'ordre de grandeur du volume et de la charge sont connus initialement et fluctuent peu. Sous cette hypothèse, des solutions de gestion de données ont été conçues puis éprouvées à travers de nombreux cas d'usages conformes à l'hypothèse de départ. Cela confère un certain degré de généralité à ces solutions qui ont été qualifiées de *one size fits all* [80] bien qu'elles restent circonscrites aux situations où les données et la charge sont stables.

Puis l'avènement du big data est venu remettre en question cette hypothèse de stabilité. De plus en plus de cas d'usage présentent des spécificités non prises en compte par les SGBD relationnels qui ne parviennent plus à exécuter efficacement les traitements demandés, ou qui ne permettent pas d'exprimer ces nouveaux traitements. Par exemple, dans les réseaux sociaux, les utilisateurs interagissent en groupe et cela partitionne les requêtes de manière dynamique. Un autre exemple dans les bases de documents scientifiques, des chaînes d'analyse nécessitent des pré-traitements complexes pour extraire des sujets. Ces analyses ne peuvent pas s'exprimer directement en SQL. Les solutions existantes ne sont pas adaptées pour de tels volumes de données.

Les contraintes rendant plus difficile la gestion des méga-données peuvent se résumer ainsi :

- **Volume** : la quantité totale de données gérées peut croître fortement de plusieurs ordres de grandeur. La charge moyenne générée par les requêtes peut aussi croître très fortement lorsque les requêtes sont plus nombreuses. Les scénarios inverses sont également considérés, *i.e.* lorsque la quantité totale de données et la charge de requêtes diminuent fortement.

- **Vélocité** ou dynamicité des données et des requêtes : (i) Pour les flux de données, de nouvelles données peuvent arriver rapidement et en continu, indépendamment de leur volume. (ii) Pour les bases de données sollicitées par un très grand nombre d'utilisateurs, les requêtes peuvent également arriver rapidement et de manière très irrégulière avec des pics de charge importants.

- **Diversité et complexité** des données et des requêtes : (i) La structure des données devient plus complexe. (ii) Les besoins de manipulation se diversifient : les requêtes sont plus variées et demandent des analyses plus complexes.

Par ailleurs, les **infrastructures** informatiques ont fortement évolué avec l'essor du *cloud computing*. Une grappe de machines appelée *cluster* de calcul est devenue accessible simplement et à moindre coût. Il est devenu possible de louer rapidement un cluster servant

d'infrastructure pour gérer des méga-données. Ainsi des solutions logicielles parallèles et distribuées sont conçues pour offrir des solutions de gestion de données.

1.2 Enjeux et défis

La gestion de méga-données, au moyen d'une infrastructure distribuée de type cluster ou fédération de machines, soulève de nombreux défis, parmi lesquels nous avons abordé les suivants :

1.2.1 Défi du passage à l'échelle

Ce défi est tout d'abord posé par la contrainte de volume. Il s'agit de garantir que la solution de gestion de données continue de fonctionner lorsque la quantité devient très élevée (de l'ordre du téraoctet). Autrement dit, la solution doit être viable quelle que soit la quantité de données considérée. De plus, la contrainte de vélocité rend plus difficile le passage à l'échelle car l'adaptation doit être dynamique lorsque le volume des données fluctue. Cela nécessite de concevoir une solution dite *élastique* car conçue pour passer à l'échelle en utilisant un nombre variable de machines ; une machine étant une ressource unitaire de stockage et/ou de traitement.

Dans un environnement distribué, il s'agit de concevoir des algorithmes pour contrôler la distribution des données et des traitements :

- **Distribuer les données** sur plusieurs machines et adapter cette distribution en fonction du volume des données et du nombre de machines, et de la capacité de stockage des machines.
- **Distribuer les traitements** entre les machines. Pour faciliter le passage à l'échelle, les algorithmes doivent être *décentralisés* ce qui permet à chaque machine de traiter des données indépendamment. Cela nécessite également de coordonner les traitements des différentes machines pour être capable de traiter des requêtes plus complexes car accédant aux données de plusieurs machines.

Contrôler la distribution des données et des traitements permet d'atteindre l'objectif d' **équilibrage de charge** essentiel pour obtenir de bonnes performances. La mise en œuvre

d'une solution de gestion de données doit aboutir à une situation où la charge des traitements est équilibrée entre les machines, ou au moins, qu'aucune machine ne subisse de surcharge pouvant nuire aux performances globales de la solution.

1.2.2 Défi de la représentation des données

Ce défi est posé par les contraintes de diversité et de complexité des données. Les données peuvent provenir de contextes très divers. Le contexte correspond ici à la logique applicative, *i.e.* l'ensemble des notions et des concepts qui apportent plus de signification aux données et à leurs manipulations. Or le contexte contient souvent des informations implicites difficiles à exprimer, et il n'y a pas de formalisme général pour « injecter » automatiquement les informations du contexte dans un système de gestion de données. Le défi est de définir ou compléter la représentation des données avec des informations spécifiques issues du contexte des données ; l'objectif étant de tirer profit d'une représentation enrichie des données pour améliorer la qualité du résultat des requêtes et/ou leur performance. Nous détaillons la qualité et/ou la performance de la représentation des données dans les trois points suivants.

Premièrement, concernant la **qualité de la représentation** des données. Nous avons besoin de définir une métrique de qualité pour mesurer l'amélioration apportée par une représentation enrichie des données. Cet objectif est particulièrement pertinent pour les contextes applicatifs où le traitement à faire sur les données ne se limite pas aux requêtes des utilisateurs mais contient aussi une tâche préliminaire de préparation des données servant à calculer une représentation enrichie des données appelée modèle. Ce modèle est ensuite interrogé pour répondre aux requêtes des utilisateurs (cela concerne par exemple la recommandation de produits ou les chaînes d'analyse en science de données). Or il existe de nombreuses alternatives pour intégrer des informations du contexte applicatif dans un modèle et il est nécessaire d'en choisir une. Autrement dit, il s'agit de fixer les paramètres du modèle. Disposer d'une métrique de qualité permet donc d'instancier ou optimiser le modèle de telle sorte que cette métrique soit maximisée. Nous avons étudié deux types de métrique de qualité : (i) la qualité du modèle lorsqu'elle peut être définie indépendamment des requêtes, et (ii) la qualité des réponses fournies aux utilisateurs lorsque le modèle ne peut pas être qualifié directement mais que son usage peut l'être. Le modèle est évalué indirectement à travers le résultat des requêtes qui est qualifié par rapport à une vérité connue (*e.g.*, les produits à recommander pour des requêtes de recommandation).

Deuxièmement, lorsque la vélocité des données est forte de nombreuses données nouvellement arrivées doivent être préparées pour intégrer le modèle avant d'être prises en compte dans les réponses fournies aux utilisateurs. Cela pose un problème de performance car la durée de calcul du modèle retarde la prise en compte des nouvelles données dans le résultat des requêtes. Dans cette situation, le défi de la représentation des données est abordé à travers le **compromis qualité/performance**. Ce compromis permet de gagner en performance en optant pour un modèle plus rapide à calculer mais de moindre qualité. L'objectif est de fournir à l'utilisateur le moyen de « régler le curseur » de ce compromis selon ses exigences de qualité ou de performance.

Troisièmement, la représentation des données peut servir à améliorer exclusivement la performance des requêtes sans considérer la qualité. C'est le cas lorsque les requêtes sont exprimées dans un langage standard (*e.g.*, SQL, SPARQL) et ont un résultat exact donc de qualité parfaite. Enrichir la représentation des données en intégrant des caractéristiques implicites permet d'optimiser les requêtes en définissant des opérateurs élémentaires (*e.g.*, sélection, jointure, fermeture transitive) plus efficaces.

1.3 Cas d'usages

La gestion de données n'est pas spécifique à un domaine applicatif et est destinée à être générale pour de nombreux types de données et de traitements. Cependant, les domaines applicatifs comportent souvent des spécificités offrant des opportunités pour optimiser les solutions de gestion de données du domaine.

Nous avons choisi de couvrir plusieurs domaines applicatifs afin de caractériser plus précisément divers problèmes de gestion de méga-données et d'exploiter leurs spécificités pour résoudre d'une manière originale les défis mentionnés plus haut. Le tableau 1.1 introduit les cas d'usage que nous avons explorés dans différents domaines.

OLTP. Nous avons étudié les applications transactionnelles dites OLTP (Online Transaction Processing). Ces applications, tels que les sites de vente en ligne, sont très répandues. Les traitements sont constitués principalement de transactions courtes ciblant un panier d'achat. Ces applications peuvent contenir des fonctionnalités faisant appel à un SGBD centralisé, ce qui limite les performances en nombre de transactions par minute. L'infrastructure visée est

Domaine	Données	Traitements	Infrastruture cible	Catég.
OLTP	relationnelles	transactions	cluster	newSQL
Réseaux sociaux	relationnelles centrées sur l'utilisateur	transactions collaboratives	cluster	newSQL
Biodiversité	multi-dimensionnelles	analyse multi-dimensionnelles	fédération de machines	NOSQL
Recommandation	flux d'avis / traces d'utilisateurs	modèles de recommandation	multicore	NOSQL
Web sémantique	graphes RDF avec types hiérarchiques	requêtes SPARQL avec raisonnement	cluster	NOSQL
Web des sciences	corpus de publications scientifiques	extraction de topics, évolution de topics	cluster	NOSQL

Table 1.1 : Données, traitements, infrastructure ciblée et catégorie d'application par domaine d'étude.

un cluster de machines. Il s'agit d'améliorer les performances en exploitant plus de machines. La solution doit être non intrusive, *i.e.* transparente pour les applications existantes.

Biodiversité. Les données relationnelles sont multi-dimensionnelles : elles possèdent deux dimensions hiérarchiques. Les requêtes contiennent une condition de sélection sur ces dimensions. L'infrastructure visée est une fédération de machines faiblement couplées, peu fiables et connectées par le réseau Internet. Il s'agit d'améliorer la disponibilité et l'expressivité du service fourni aux utilisateurs.

Réseaux sociaux. Les données sont centrées sur les utilisateurs. Les transactions sont simples et ciblent des petits groupes d'utilisateurs qui évoluent lentement. Dans une infrastructure cluster, il s'agit d'exploiter la notion de groupes d'activité entre utilisateurs pour améliorer les performances des transactions.

Recommandation. Lorsque les données sont des avis d'utilisateurs, il s'agit d'améliorer la qualité de la recommandation en capturant certaines spécificités des utilisateurs (le biais de notation) ou des articles (catégories). Les données peuvent arriver en flux : il s'agit alors de

prendre en compte rapidement les données les plus récentes.

Lorsque les données sont des traces de déplacement, des spécificités géographiques et temporelles sont prises en compte pour améliorer la recommandation. Une infrastructure cluster doit accélérer le temps de calcul du modèle. Voir le chapitre 3 pour une présentation plus détaillée des défis et problèmes sous-jacents abordés.

Web sémantique. Les données sont des grand graphes de connaissance RDF décrivant des entités. La hiérarchie du type des entités est connue. Les requêtes sont exprimées en SPARQL et nécessitent de nombreuses jointures. Il s'agit optimiser les requêtes en exploitant les informations sur les types et une plateforme de calcul parallèle.

Web des sciences. Les données sont des corpus de documents. Les requêtes sont des questions de haut niveau explorant l'évolution temporelle de topics. Une étape de préparation des données extrait des topics dont la qualité doit être maîtrisée. Il s'agit de fournir un service interactif pour explorer les topics et leur évolution. Un cluster Spark [2] sert d'infrastructure parallèle et distribuée pour accélérer les analyses préparatoires.

1.4 Méthodologie des travaux de recherche effectués

D'un point de vue méthodologique, mes travaux de recherche suivent une démarche expérimentale en trois étapes :

- (i) une étude approfondie des données et traitements associés permet d'identifier avec précision une situation - cas d'usage - posant un problème de performance,
- (ii) Pour améliorer le cas identifié, une solution nouvelle est conçue avec des algorithmes de gestion démontrant leur efficacité,
- (iii) l'applicabilité de la solution à un ensemble plus général de cas est étudiée.

J'applique ces principes pour optimiser divers types de manipulations (requête, recommandation, transaction) sur divers types de données (relationnelles, graphes, textuelles) Cette approche présente l'avantage d'offrir une bonne transversalité vis-à-vis des domaines

[2] `https://spark.apache.org/`

délimitant tel type de donnée et tel type de manipulation, tout en préservant une démarche scientifique cohérente avec son objectif : caractériser un couple (donnée, manipulation) et l'optimiser.

Une part importante de mes travaux est consacrée à la validation expérimentale des algorithmes proposés. Un effort est investi pour réaliser des scénarios réalistes manipulant des données réelles ou à l'aide de benchmarks reproduisant des situations similaires à la réalité [3]. Lorsque l'infrastructure ciblée est distribuée (*e.g.*, un cluster) la complexité de gestion induite est prise en compte et maîtrisée en profondeur afin de garantir la validité des mesures de performances obtenues. L'objectif est d'apporter une meilleure reproductibilité aux résultats expérimentaux [27], ce qui inclut le partage des réalisations logicielles effectuées.

Par ailleurs, pour mener un travail de recherche à son terme, notamment dans le domaine des bases de données, il est nécessaire de rassembler des personnes apportant des compétences complémentaires formant un socle minimal requis. Ainsi, mes travaux se sont déroulés à travers de nombreuses et diverses collaborations : locales avec plusieurs équipes du LIP6, nationales avec Telecom Paristech et l'UPEM, et internationales avec l'Université Cheik Anta Diop de Dakar (UCAD).

1.5 Classement et résumé des contributions

Les contributions sont classées par défis puis type de problème. Le domaine d'étude et la référence de la section détaillant la contribution sont indiqués entre parenthèses.

1.5.1 Contributions liées au défi du passage à l'échelle

Distribution des traitements

- Algorithme de routage décentralisé de transactions (OLTP, 2.2)
- Optimisation de requêtes SPARQL sur une plateforme parallèle et distribuée (Web sémantique, 4.2)
- Parallélisation du calcul d'ensembles fréquents maximaux pour le calcul des topics (Web des Sciences, 5.2)

[3] `http://tpc.org`

- Calcul distribué des similarités entre topics (Web des Sciences, 5.4)
- Calcul distribué d'agrégats dans un graphe de topics (Web des Sciences, 5.3)

Distribution des données et équilibrage de charge

- Placement de données basé sur la charge des requêtes d'analyse (Biodiversité, 2.4)
- Optimisation de jointure pour des données biaisées (Web sémantique, 4.4.1)
- Redistribution de données en fonction des groupes actifs (Réseaux sociaux, 2.3)

1.5.2 Contributions liées au défi de la représentation des données

Représentation enrichie pour plus de qualité :

- Inférence d'un graphe de voisinage à partir de données spatio-temporelles et intégration dans un modèle adapté pour des données à large échelle (Recommandation, 3.3.2)
- Niveau de mobilité d'un voyageur, granularité des zones d'intérêt en fonction du niveau de mobilité (Recommandation, 3.3.3)
- Métrique de qualité des topics extraits par un modèle Latent Dirichlet Allocation (LDA). Notion de graphe pivot pour représenter l'évolution d'un topic (Web des sciences, 5.3)

Compromis qualité/performance :
Mise à jour transitoire et incrémentale d'un modèle de factorisation de matrices, basée sur le biais de notation d'un utilisateur (Recommandation, 3.2.1)

Représentation enrichie pour plus de performance : Les contributions correspondent à la définition des notions suivantes :

- Triangles inférables (Web Sémantique, 4.4.2)
- Impact du biais sur les jointures dans un grand graphe sémantique (Web Sémantique 4.4.1)
- Groupes d'utilisateurs actifs (Réseaux sociaux, 2.3)

1.6 Organisation du manuscrit

Les chapitres 2 à 5 s'intéressent aux défis soulevés par la gestion de méga-données, pour différents types de données fortement présents dans différents domaines. Chaque chapitre repose sur au moins un encadrement doctoral ou une collaboration indiqués dans le tableau 1.2.

Chapitre	Titre et Problématique	Doctorant	Collaborateur
2	Elasticité des transactions et requêtes SQL	I. Sarr, I. Gueye, N. Bame	S. Gançarski
3	Recommandation à large échelle	M. Gueye, J.-B. Griesner	T. Abdessalem
4	Requêtes sémantiques à large échelle		O. Curé, B. Amann
5	Optimisation de workflow en science des données	K. Li	B. Amann

Table 1.2 : Encadrements doctoraux par problématique

Le chapitre 2 rassemble les travaux concernant les données relationnelles manipulées avec des transactions ou requêtes SQL. L'objectif commun est d'améliorer l'élasticité de la couche de manipulation des données. Les problèmes abordés sont le routage des transactions OLTP, la distribution des données pour les applications de réseaux sociaux, et l'équilibrage de charge pour les requêtes de biodiversité. Ces travaux correspondent aux encadrements doctoraux d'Idrissa Sarr, Ibrahima Gueye et Ndiouma Bame.

Le chapitre 3 concerne les données centrées sur les utilisateurs et étudie des traitements permettant de fournir des recommandations aux utilisateurs. Il présente des modèles et algorithmes de recommandation conçus pour fonctionner à large échelle. Ces travaux correspondent aux encadrements doctoraux de Modou Gueye et Jean-Benoit Griesner.

Les deux chapitres suivants sont davantage développés afin de mieux mettre en évidence des travaux représentatifs de contributions personnelles récentes. Le chapitre 4 étudie les grands graphes de données sémantiques. Il présente l'optimisation de requêtes sémantiques à large échelle. Le travail appelé *Triag*, sur la prise en compte de motifs triangulaires est plus particulièrement détaillé. Il découle d'une collaboration avec Olivier Curé (Université Paris Est - UPEM).

Le chapitre 5 s'intéresse aux workflow d'analyse ; ce sont des chaînes de traitement très répandues en science des données. Des données textuelles, en particulier des documents scientifiques, sont transformées pour extraire des domaines sous la forme de graphe d'évolution. Les problèmes abordés sont la qualité l'extraction et le passage à l'échelle des transformations. Les travaux sur la parallélisation des ensembles fréquents maximaux et sur le calcul de similarité à large échelle sont plus particulièrement détaillés. Ces travaux correspondent à l'encadrement doctoral de Ke Li.

Le chapitre 6 dresse un bilan de mes travaux avec l'accent mis sur les récents travaux dans les domaines de la science des données et du Web sémantique. Les publications significatives balisant l'ensemble de mon activité de recherche sont récapitulées, puis des perspectives à moyen et long terme sont présentées.

2 Gestion élastique de méga-données relationnelles

Le traitement de données à large échelle pose des problèmes difficiles pour gérer la fluctuation des demandes. Pour faire face à cette fluctuation. Les ressources allouées au traitement peuvent être contrôlées dynamiquement avec la possibilité d'ajouter ou retirer rapidement des machines. Cette élasticité des ressources existe par exemple dans les offres locatives du cloud. L'élasticité ouvre de nouvelles opportunités pour économiser les ressources consommées, en optimisant l'allocation au plus près des besoins applicatifs. Cependant, cette élasticité focalisée sur l'ajout/suppression de machines, ne suffit pas pour gérer efficacement des méga-données. L'élasticité doit aussi être considérée et mise en place dans les couches successives du traitement des données, depuis la couche frontale pour exprimer des requêtes applicatives, jusqu'à la couche de stockage des données.

L'objectif de ce travail est de contribuer à la conception d'un service élastique de gestion de données relationnelles à large échelle. Il s'agit d'organiser dynamiquement le contenu et l'activité des machines allouées au traitement des transactions et requêtes SQL en tenant compte de la charge courante soumise par les utilisateurs.

2.1 Travaux connexes

De nombreux travaux ont étudié le problème de traiter des transactions SQL à large échelle. Nous détaillons dans cette section des travaux antérieurs aux nôtres, et d'autres plus récents viennent confirmer la pertinence des pistes de solutions que nous avions proposées.

Lorsque l'infrastructure sous-jacente est distribuée, l'occurrence de panne devient plus fréquente avec le nombre de machines qui augmente. Il devient plus difficile de concevoir une solution qui soit hautement disponible, *i.e.* qui puisse continuer de traiter des transactions

bien que certaines machines soient en panne. Pour améliorer la disponibilité, il est possible d'introduire de la redondance en répliquant les données. Cependant, cela nécessite de coordonner les transactions effectuées sur les diverses répliques pour garantir que l'état des données soit cohérent et corresponde à l'état obtenu après avoir traité toutes les transactions en séquence. Des travaux ont démontré de manière théorique qu'une situation de panne peut empêcher de maintenir à la fois la disponibilité du système et la cohérence des répliques manipulées [31]. Ce résultat appelé *théorème CAP* formalise un mode de fonctionnement assez intuitif dans les systèmes composés de plusieurs machines : les écritures sur deux répliques d'une donnée ont besoin de communiquer entre elles pour apporter la garantie que les deux répliques sont toujours identiques.

2.1.1 Traitement déterministe des transactions avec Calvin

La limite énoncée par le théorème CAP n'est pas un frein pour le traitement des transactions à large échelle dans la mesure où il est possible de délimiter plus spécifiquement le contexte applicatif. D'une part concernant l'accès aux répliques, il est possible de dissocier la réplication du traitement des transactions. D'autre part, il est possible d'exiger que l'ensemble des opérations constituant une transaction soit connu avant le traitement de celle-ci. L'idée est de poser des hypothèses simplifiant la notion de transaction dans le but de gagner en efficacité sur leur traitement à l'aide d'algorithmes nécessitant moins de coordination et passant mieux à l'échelle. Ce type de compromis s'avère intéressant en pratique car il correspond à la notion de transactions existant dans de nombreux cas d'usage.

Ainsi, des solutions ont été proposées dans cette direction : il s'agit des systèmes de traitement de transactions déterministe [2] dont un des plus aboutis est le système Calvin [82]. Dans Calvin, les données sont partitionnées sur une grappe de machines. Cela permet de dimensionner le nombre de machines selon le volume des données à gérer. Une transaction est entièrement déterminée à l'instant où elle est posée : elle est prédéfinie au moyen d'un programme appelé procédure stockée. Une transaction peut manipuler les données stockées dans plusieurs partitions. La connaissance *a priori* des données lues et écrites par les transactions permet de fixer l'ordre de traitement des transactions avant même de les exécuter. Chaque machine détermine l'ordre des transactions qu'elle doit traiter, à l'aide d'un protocole décentralisé efficace. Puis chaque machine transmet, aux seules machines qui en ont besoin, une copie temporaire des données utiles à une transaction. Cela permet de substituer le traitement d'une transaction distribuée par le traitement d'une transaction locale sur chaque

machine qui modifie au moins une des données de la transaction.

De plus, pour apporter de la disponibilité en cas de pannes, les données partitionnées sont entièrement répliquées sur une ou plusieurs autres grappes. Chaque machine (y compris les répliques) est un point d'entrée pouvant recevoir des transactions. Les machines stockant les répliques d'une même partition se coordonnent périodiquement pour partager entre elles les demandes de transactions. Chaque grappe peut ensuite traiter les transactions en toute indépendance car l'ordre des transactions est déterminé de manière identique dans chaque grappe. En résumé, Calvin propose une solution décentralisée efficace et hautement disponible pour gérer des transactions à large échelle. Un point commun avec nos travaux présentés en section 2.2 est de déterminer l'ordre des transactions avant de les exécuter. Une différence avec nos travaux, est que dans Calvin les transactions sont propagées sur les répliques le plus tôt possible tandis que nous avons opté pour une propagation à la demande en fonction des exigences des requêtes : cela permet de mieux lisser la charge lorsque les demandes arrivent de manière irrégulière et qu'il y a des pics de requêtes ou de transactions.

2.1.2 Transactions géo-distribuées avec Spanner

Spanner [9] est un système issu d'une décennie de travaux recherche à Google. Il permet de traiter des transactions distribuées manipulant des données géo-répliquées à l'échelle mondiale. Spanner s'appuie sur un réseau d'interconnexion fiable et très performant entre les machines. Ce réseau rend possible la réalisation de plusieurs services essentiels pour le traitement des transactions : un service d'horodatage offre une horloge globale très précise pour ordonner les transactions ; un service de consensus sert à valider les transaction et accéder aux répliques d'une donnée. Spanner s'appuie sur des mécanismes éprouvés issus des SGBD relationnels pour garantir l'isolation des transactions (verrouillage et estampillage pour contrôler les accès concurrents) et l'atomicité des transactions distribuées (validation en deux étapes). Une différence importante avec Calvin est que Spanner ne détermine pas l'ordre des transactions avant de les exécuter. Ainsi Spanner peut traiter des transactions interactives pour lesquelles les données accédées ne sont pas connues au début de la transaction [1].

Concernant la distribution des données dans Spanner, chaque table est fragmentée horizontalement par intervalle sur la valeur de la clé primaire. Pour améliorer les performances de certaines requêtes de jointure, les données de deux tables peuvent être co-localisées dans un fragment commun lorsque la clé de la première table est référencée dans les n-uplets de la

deuxième table. Par exemple, si on considère deux tables Personne et Commande (contenant les commandes d'une personne), un fragment commun peut contenir des personnes avec leurs commandes. Dans nos travaux sur les requêtes géo-distribuées, présentés en section 2.4, nous avons également proposé de fragmenter et co-localiser les données. Cependant la définition des fragments diffère : les intervalles de valeur ne sont pas définis sur la clé primaire mais sur des attributs représentant les dimensions des données analysées. Cela apporte plus de performances lorsque le modèle des analyses à traiter est connu à l'avance. En revanche la fragmentation de Spanner est bénéfique pour le traitement des transactions et l'ajout de fragments lorsque le volume des données augmente.

2.1.3 Transactions élastiques avec Elastras

Elastras [25] a proposé de séparer d'une part la gestion du stockage des données et d'autre part la gestion des transactions. Elastras ne supporte pas les transactions distribuées : une transaction est traitée localement sur une seule machine. Cela n'est pas un frein dans le contexte considéré qualifié de *multi-tenants* : l'objectif est de gérer les données de millions d'applications, chacune manipulant des données qui tiennent sur une machine. Elastras peut redistribuer dynamiquement les données sans interrompre le traitement des transactions sur les données en cours de déplacement. Cette propriété de migration en *live* rend la solution élastique car des machines peuvent être ajoutées ou supprimées de manière transparente pour les applications. Notons que ce choix n'a pas été retenu ni dans Spanner ni dans Calvin où une transaction échoue en cas de réorganisation des données et doit être re-exécutée. Spanner propose cependant de re-exécuter une transaction automatiquement en cas d'échec. Nos travaux présentés en section 2.3 ont un objectif commun avec Elastras : migrer les données de manière transparente sans interrompre les transactions. La différence est que les données auxquelles une transaction accède sont connues *a priori* (comme dans Calvin). Cela nous permet, lorsqu'un gain de performance est escompté, de migrer les données à la demande dans une étape préliminaire au traitement d'une transaction.

2.1.4 Architecture désagrégée

Un constat largement partagé dans ces travaux est la nécessité de concevoir des solutions ayant une architecture dite *désagrégée* qui dissocie la couche de stockage des données et la couche de gestion des transactions et des requêtes. Chaque couche est conçue comme un

sous-système pouvant être (re)dimensionné séparément en lui allouant des ressources de manière indépendante. Cela nécessite de concevoir des algorithmes pour gérer le passage à l'échelle dans chaque couche et pour coordonner les traitements entre les couches.

Spanner repose sur une telle architecture avec d'une part un service de stockage reposant sur le système distribué BigTable et d'autre part un service de traitement des requêtes et transactions. La désagrégation est encore plus fine dans CumuloNimbo [50] où la gestion des transactions a été divisée en trois services indépendants : estampillage, gestion des journaux et résolution des conflits. Notons que cette séparation en couches a été facilitée par le choix d'une gestion optimiste de transactions : une transaction est traitée sans verrouiller les données puis une coordination *a posteriori* permet de détecter un éventuel conflit et d'annuler la transaction, le cas échéant. Ce principe de désagrégation a été appliqué par d'autres travaux tels que CloudTPS [84] qui propose un intergiciel pour traiter des transactions au dessus d'un système de stockage distribué, ou [6] qui propose un protocole décentralisé pour traiter des transactions manipulant certains types de données géo-répliquées (*e.g.*, compteur, séquence) au dessus d'un système de stockage de type clé-valeur [5].

2.2 Routage de transactions à large échelle (thèse d'Idrissa Sarr)

Dans le cadre de la collaboration internationale que j'ai mise en place entre l'UCAD (Sénégal) et le LIP6, nous avons constaté que la problématique de la gestion de données distribuées à large échelle était un défi majeur au Sénégal dans un environnement où les ressources (machines, réseau) sont limitées et fréquemment indisponibles. Le problème principal est d'agréger un grand nombre de machines de manière transparente pour les applications, ce qui est difficile pour deux raisons : les ressources sont nombreuses et doivent être coordonnées efficacement, les situations de pannes sont plus probables et cela nécessite d'introduire de la redondance dans les données et les traitements. Ainsi, nous avons proposé des solutions (routage des requêtes et des transactions de mise à jour) pour des plateformes dites *clusters* composées de plusieurs dizaines de bases de données. Après avoir encadré des stagiaires de DEA/M2 à l'UCAD, j'ai proposé en 2007 à l'un d'entre eux, Idrissa Sarr, de démarrer une thèse sur le routage de transactions à large échelle dans le cadre du projet ANR Respires coordonné par S. Gançarski membre de l'équipe Bases de Données.

Le résultat de cette thèse (2007-2010) est un intergiciel distribué pour la gestion des transactions et des méta-données permettant de 1) réduire le temps de réponse des transactions

en équilibrant la charge d'accès aux répliques et en tenant compte de la disponibilité des ressources ; 2) contrôler la cohérence des accès aux données réparties et répliquées en accord avec les exigences des applications ; 3) garantir l'autonomie des applications et des SGBD. Ces travaux de thèse ont fait l'objet de cinq publications (une revue RSTI [78] et quatre conférences HPDGrid [75], ACM SAC DADS [79], DBKDA [76], BDA [77],). Afin de poursuivre ces travaux, j'ai participé avec 3 collègues, au montage d'un projet interne au LIP6 (2 ans et ayant financé deux stages de M2) pour étudier les algorithmes de synchronisation répartis adaptés aux systèmes de BD à très grande échelle. Le résultat a été publié dans l'atelier P2PDEP [63].

2.3 Transactions dans les réseaux sociaux (thèse d'Ibrahima Gueye)

Les systèmes de gestion de données ont évolué pour intégrer deux avancées technologiques importantes : d'une part, la mise à disposition du grand public des infrastructures en nuage et dématérialisées appelées infrastructures cloud. D'autre part, l'omniprésence des applications sociales dans le monde numérique. Elles mettent l'utilisateur au centre des applications et permettent de nouveaux types d'usages dits sociaux : partage de contenus, échanges de messages, annotation ou édition collaborative, et plus généralement tout échange de données et de services au sein d'un large réseau d'utilisateurs. Fort de ce constat, j'ai initié une activité de recherche abordant les problèmes liés au traitement de transactions dans les réseaux sociaux. Cette activité s'est inscrite dans la collaboration internationale avec l'UCAD à travers laquelle j'ai co-encadré la thèse d'Ibrahima Gueye détaillée ci-dessous.

Les transactions distribuées s'exécutant dans un système de gestion de données sur un cluster génèrent un surcoût de contrôle et de synchronisation qui s'avère être un obstacle majeur au passage à l'échelle car le temps de réponse d'une transaction dépend du nombre de machines participant à la transaction. Nous avons proposé de surmonter cet obstacle pour les applications centrées sur les utilisateurs telles que les applications de gestion de réseaux sociaux. Ce type d'application présente les spécificités suivantes que nous proposons d'exploiter :

- Les données sont centrées sur les utilisateurs. On suppose que le schéma des données est formé d'une table principale contenant des utilisateurs, et d'un ensemble de tables faisant toutes référence à un utilisateur. Ainsi, les données peuvent être fragmentées par utilisateur, ce qui permet de connaître les fragments auxquels accéder à partir d'un identifiant d'utilisateur.

- Chaque transaction fixe *a priori* un ensemble d'utilisateurs auquel accéder. Les fragments accédés sont donc connus à l'avance ce qui offre des opportunités pour déplacer les données afin de favoriser les accès locaux.
- La charge des transactions (*workload*) est dynamique. La fréquence d'accès à un fragment fluctue dans le temps et la distribution des fréquences d'accès est irrégulière.

Pour optimiser le traitement des transactions, la solution proposée consiste à déplacer les données de l'utilisateur vers un seul nœud ce qui permet de traiter une transaction localement donc efficacement. Ainsi, en supposant que les utilisateurs interagissent dans des *cercles d'activité*, il est possible de détecter les groupes de données concernées par les transactions issues d'un même cercle et d'adapter graduellement le placement des données et les ressources du cluster en fonction de l'apparition/disparition des cercles d'activité. Ce travail a fait l'objet de publications dans trois conférences : CARI [40], DBKDA [39] et DEXA [38]

2.4 Requêtes géo-distribuées (thèse de Ndiouma Bame)

Ce travail est issu d'une collaboration informelle avec le laboratoire LIS du Muséum National d'Histoire Naturelle (MNHN) dirigé par Régine Vignes, autour des problématiques de la gestion des données de biodiversité. Pour ce type de données, le GBIF (Global Biodiversity Information Facility) a conçu un portail qui fédère les données d'observation des espèces à un niveau mondial. Ce portail s'avère largement utilisé dans la communauté comme source principale de données servant à diverses analyses et modélisations. A la suite d'entretiens avec des chercheurs du MNHN, nous avons identifié deux limitations du portail GBIF : il n'accepte pas de requêtes déclaratives suffisamment expressives et manque de disponibilité.

Pour trouver les moyens humains permettant d'aborder ce problème, j'ai répondu à l'appel du programme doctoral international PDI MSC, ce qui m'a permis de démarrer la thèse en co-tutelle (UCAD/SU) de Ndiouma Bame intitulée *Gestion de données complexes pour la modélisation de niche écologique* pour laquelle j'ai été l'encadrant principal. L'objectif de cette thèse était de concevoir un système distribué répliquant les données de la base initiale et capable de traiter les requêtes des utilisateurs dans un délai imparti.

Solution. Pour surmonter le manque d'expressivité du portail, nous avons proposé un intergiciel distribué offrant une interface d'accès au GBIF de plus haut niveau. Dans l'intergiciel, chaque machine est un système de gestion de données capable d'accéder au portail du GBIF, de répliquer certains fragments de données et de traiter des requêtes SQL quelconques. L'architecture proposée a été publiée dans les conférences CARI [11] et CNRIA [13]. Les données sont fragmentées et répliquées à travers un réseau de machines participantes afin de mutualiser les données et les ressources de calcul et d'accélérer ainsi le temps de réponse des requêtes. La fragmentation des données tire profit de la forme spécifique des requêtes des utilisateurs : les analyses portent principalement sur deux dimensions hiérarchiques. La dimension géographique précise la zone, plus ou moins vaste, à analyser, et la dimension taxinomique précise le taxon (*i.e.* la famille, le genre ou l'espèce) à étudier. Or peu d'analyses sont globales sur toute la base mais sont pour la plupart interactives de telle sorte qu'un utilisateur soumet une série de requêtes pour analyser progressivement des données ciblant des zones géographiques et des taxons de plus en plus spécifiques. Cela permet d'une part de réutiliser le même fragment pour traiter plusieurs requêtes successives, et d'autre part, de décomposer les requêtes complexes en des sous-requêtes indépendantes pouvant être traitées en parallèle.

Nous avons proposé un algorithme décentralisé pour le traitement en parallèle de requêtes et une stratégie de placement et de réplication, basée sur un modèle de coût, qui maintient l'utilisation (stockage et calcul) des machines en dessous d'un certain seuil. Ils ont été publiés dans la revue ARIMA [10]. L'expérimentation avec un grand volume de données réelles de biodiversité montre le bénéfice de la solution proposée par rapport à des heuristiques de distribution et de réplication plus simples ; ce travail est paru dans la revue ARIMA [12].

3 Modèles et algorithmes de recommandation à large échelle

3.1 Enjeux de la recommandation

La recommandation consiste à prédire, pour un utilisateur, les notes ou les préférences qu'il attribuerait à un élément ou *item* qui peut être un document, un lieu, etc. Ces prédictions sont inférées à partir des données des utilisateurs (avis, préférences, liens sociaux, etc.). Deux principaux types d'approches existent :

- Définir un *modèle* représentant les caractéristiques latentes des utilisateurs et des items. Cela permet de prédire l'affinité d'un utilisateur pour un item quelconque. Ces approches s'appuient sur des méthodes de *factorisation de matrices*. Il est possible d'inférer la matrice latente des utilisateurs et celles des items de telle sorte qu'une prédiction pour un utilisateur et un item corresponde le plus précisément possible au produit scalaire de la représentation latente de cet utilisateur avec celle de l'item. Cette tâche d'inférence est appelée entraînement du modèle. Une fois que le modèle est entraîné son utilisation est immédiate et rapide car une prédiction se limite à calculer un produit scalaire. Ces approches basées sur un modèle ont l'avantage de fournir une recommandation de qualité relativement élevée lorsqu'on ne dispose pas d'information décrivant les utilisateurs autres que les avis qu'ils ont émis.
- Définir un *graphe de similarité* entre utilisateurs en se basant sur les informations décrivant un utilisateur. Pour cela, il existe de nombreuses fonctions de similarité selon les divers types de contenu dont on dispose sur les utilisateurs (ses préférences, les avis qu'il a émis, *etc*.). Cela permet de calculer un voisinage pour chaque utilisateur. Ensuite, la recommandation consiste à proposer les items les mieux notés (ou préférés) parmi les voisins les plus proches d'une personne. Un avantage de ce type d'approche est de pouvoir prendre en compte la diversité des informations contextuelles (*e.g.*, profil, âge,

lieu) décrivant les utilisateurs lorsqu'elles sont connues.

Ces deux types d'approches sont souvent combinées pour définir des modèles enrichis pouvant cumuler les atouts de chaque type. Nous avons suivi cette démarche avec l'objectif d'appréhender trois dimensions spécifiques des méga-données : la diversité, le volume et la vélocité.

- *Diversité* : Tenir compte d'informations explicites pouvant être connues sur les utilisateurs, les items, ou les relations existant entre les utilisateurs ou les items. En particulier les relations entre utilisateurs peuvent traduire des aspects sociaux et les relations entre items, lorsqu'ils sont géo-localisés, peuvent exprimer une proximité géographique ou une facilité pour transiter d'un item à un autre.
- *Volume* : Proposer des solutions qui passent à l'échelle, c'est-à-dire capables de faire face à un très grand nombre d'utilisateurs et d'items.
- *Vélocité* : Tenir compte de la dimension dynamique des données d'entrée sur lesquelles s'appuient la recommandation, qui peuvent arriver en continu.

3.1.1 Comparaison des approches proposées

Le tableau 3.1 compare nos différentes contributions selon les dimensions prises en compte (diversité, volume, vélocité) et les méthodes sous-jacentes (factorisation de matrice ou graphe de similarité).

Concernant la *diversité*, nous avons pris en compte les types d'attributs existant dans les jeux de données de référence utilisés dans la littérature : date à laquelle un utilisateur émet un avis, position GPS de l'événement avec éventuellement la catégorie du lieu s'il s'agit d'un point d'intérêt (POI), tag associé à un événement ayant un contenu textuel. La dimension *volume* indique la capacité de la solution à demeurer performante avec des grands jeux de données ; les valeurs vont de - (non scalable) à +++ (très scalable) avec l'indication *Poisson* lorsque la scalabilité est principalement apportée par la factorisation de Poisson, *cf.* section 3.3. La dimension *vélocité* indique quelle solution est capable de prendre en compte efficacement, de manière incrémentale, l'arrivée de nouvelles données. Pour les solutions basées sur un modèle de factorisation de matrice, nous indiquons les informations qui étendent le modèle sous-jacent (modélisation du biais des utilisateurs, de l'influence géo-temporelle ou sociale, des transitions entre points d'intérêt ou du niveau de mobilité). Pour celles basées sur un

Contrib.	**Diversité**	**Volume**	**Véloc.** (flux)	**Factorisation de matrice**	**Graphe de similarité**
[47, 43]	date	+	oui	biais utilisateur	
[34, 33]	GPS, date	-		influence géo-temporelle	items × géo-zones
[42, 46]	tag	+++	oui		user × user
[35]	GPS, date, catégorie POI	++ Poisson		influence sociale, transition entre items	user × user, item × item
[36]	GPS, date catégorie POI	+++ Poisson		social, transitions, niveau de mobilité	user × user, géo-zones hiérarchiques

Table 3.1 : Comparaison des approches de recommandation

graphe de similarité, le type des nœuds du graphe (utilisateur, item, zone géographique) est précisé.

Les travaux présentés dans le tableau 3.1 constituent les deux thèses successives que j'ai co-encadrées avec Talel Abdessalem (Télécom Paris) et sont détaillés dans les sections suivantes. Durant ce travail à l'intersection entre la gestion de mégadonnées et la recherche d'information, j'ai eu un rôle moteur important pour identifier les « niches » non encore explorées et apporter aux doctorants la méthodologie rigoureuse nécessaire à la validation expérimentale des algorithmes proposés. Ce travail se poursuit actuellement à travers une collaboration avec l'Université de Thiès, voir section 3.4.

3.1.2 Positionnement dans l'état de l'art

Nous résumons le positionnement de nos travaux par rapport à l'état de l'art. Pour les solutions s'appuyant sur la factorisation de Poisson, le travail fondateur de Blei [32] a été étendu dans de nombreuses directions pour tenir compte d'informations contextuelles autres que les notes des utilisateurs. En particulier, [17] a pris en compte les liens sociaux existant entre utilisateurs pour pondérer le score de prédiction. Nous avons adapté ce dernier

travail pour l'appliquer à la situation où les liens entre utilisateurs ne sont pas connus à l'avance mais inférés à partir de la similarité des déplacements des utilisateurs. L'axe sur la recommandation par factorisation de Poisson demeure très actif actuellement avec, par exemple, des propositions pour tenir compte de la description textuelle des éléments recommandés à l'aide de modèles d'apprentissage profond [60].

Pour les solutions de la factorisation de matrice applicables à un contexte dynamique, les fondements de la recommandation dite *online* ont été définis dans [73]. Puis des travaux tels que [49] ont proposé une méthode incrémentale pour factoriser une matrice, tandis que nous avons proposé une méthode pour intégrer des biais de notation dans un modèle dont la matrice est déjà factorisée, afin d'ajuster ces biais de manière incrémentale indépendamment de la factorisation.

Pour les solutions basées sur des parcours de graphe de similarité, le système Taagle [62] propose une méthode connexe à la nôtre mais pour le cas d'un moteur de recherche, *i.e.* lorsqu'il s'agit de retrouver les tags similaires à un mot clé, et non dans le cas d'un système de recommandation, *i.e.* lorsque la tâche consiste à suggérer des tags pour un couple (document, utilisateur). Notre approche se différencie également par la possibilité d'élaguer le parcours du graphe afin de réduire le temps de réponse sans dégrader la précision obtenue.

3.2 Recommandation continue à large échelle (thèse de Modou Gueye)

La recommandation exploite des données provenant des utilisateurs : on connaît un ensemble de faits - ou événements - décrivant quelle note un utilisateur a attribué à un élément. Une hypothèse fréquemment posée est que ces données sont statiques. Ces données d'usage constituent un jeu de données qui sert à entraîner le modèle de recommandation, *i.e.* optimiser les paramètres du modèle dans l'objectif de minimiser l'erreur de recommandation. Or dans de nombreux cas, les données d'usage ne sont pas statiques mais dynamiques : un utilisateur peut noter un élément à tout instant. Le jeux de données devient un flux d'événements générés en continu par les utilisateurs.

Tenir compte de cet aspect dynamique est difficile car cela fait émerger des contraintes contradictoires concernant l'entraînement d'un modèle de recommandation. D'une part l'entraînement doit tenir compte des données les plus récentes car cela tend à améliorer la

qualité de la recommandation. D'autre part, la méthode de calcul du modèle impose que les données d'entraînement soient entièrement connues au moment où le calcul démarre. Ainsi, le modèle est entraîné sans tenir compte des données qui arrivent pendant l'entraînement. Cela pose un problème d'autant plus important que la quantité de données non prises en compte est grande. Cela provoque une diminution significative de la qualité du modèle en comparaison avec une situation statique où toutes les données sont connues à l'avance.

La quantité de données non prises en compte dépend de deux paramètres : la durée de calcul du modèle et le débit du flux, mais ce dernier est généralement imposé par le contexte applicatif. Nous avons donc étudié des solutions en fonction de la durée de calcul, longue ou courte, du modèle. Ces deux approches sont détaillées ci-dessous.

3.2.1 Factorisation de matrice dans un contexte dynamique

Parmi les modèles de recommandation dont la complexité algorithmique élevée peut induire des longs temps de calcul, nous nous sommes concentrés sur la factorisation de matrice car son calcul est itératif avec une durée dépendant du nombre d'itérations, et car c'est un modèle reconnu pour la qualité des recommandations qu'il fournit. La question est : « Peut-on tenir compte des avis récemment arrivées mais sans recalculer le modèle ? » Nous avons apporté une réponse en cherchant à donner une signification concrète aux propriétés qui caractérisent un utilisateur. Or dans le modèle de factorisation de matrice un utilisateur est caractérisé par des poids sur des propriétés latentes qui sont par définition sans signification particulière. En conséquence, nous avons proposé d'étendre le modèle pour introduire des biais capturant la déviation du comportement d'un utilisateur. Ces biais sont initialisés lors du calcul du modèle. Puis ils sont continuellement mis à jour dès que des nouveaux avis arrivent. L'avantage est que contrairement au calcul du modèle, la mise à jour des biais est une opération peu complexe qui est suffisamment rapide pour être traitée entre l'arrivée de deux avis.

La mise à jour des biais n'apporte pas une qualité équivalente au recalcul du modèle, mais elle apporte cependant un gain de qualité mesurable. Nous avons montré de manière empirique que notre solution préserve plus longtemps la qualité de la recommandation d'un modèle. Une retombée intéressante est que cela permet de recalculer moins fréquemment un modèle et donc d'économiser des ressources de calcul. Nous avons publié ce travail dans EGC [47] et CNRIA [44], et une version étendue dans la revue AKDS [43].

3.2.2 Recommandation basée sur un parcours de graphe de similarité

La deuxième approche pour prendre en compte l'arrivée continue des avis des utilisateurs est d'opter pour une solution où le modèle a une faible complexité algorithmique et dont les paramètres sont suffisamment explicites pour pouvoir être ajustés partiellement en fonction des nouveaux avis qui arrivent. Le modèle que nous avons considéré est un graphe de similarité entre utilisateurs. Quand un nouvel avis arrive au sujet d'un certain élément et provenant d'un certain utilisateur, nous pouvons identifier les similarités à modifier. Ce sont celles entre l'utilisateur et ceux ayant au moins un avis sur un élément en commun avec lui.

En général ce type d'approche reposant sur un modèle simple donc rapide à calculer et à mettre à jour, présente l'inconvénient d'avoir un temps de prédiction plus long car il est nécessaire de parcourir le graphe de voisinage d'un utilisateur et la taille de ce voisinage peut être grande. Intuitivement, il y a un compromis à faire entre le temps d'entraînement (*i.e.* le calcul du modèle) et le temps de prédiction car ces deux durées varient en sens opposé l'une de l'autre. Nous avons choisi de positionner le curseur du côté des solutions ayant une faible durée d'entraînement, puis nous avons cherché à réduire la durée de prédiction.

Pour le cas de la recommandation de k tags, nous avons proposé un algorithme qui prend en compte la popularité des tags et les opinions du voisinage d'un utilisateur. Contrairement à l'approche existante qui considère un nombre fixe de plus proches voisins, nous avons proposé une méthode pour parcourir tous les voisins (et seulement eux) qui contribuent à la recommandation. Cela permet de calculer une recommandation à la demande avec un faible coût. Les résultats obtenus ont été publiés à SocialCom [42] et ACM RecSys [46]. Nous avons aussi proposé une méthode pour améliorer la précision d'une k-recommandation en réordonnant dynamiquement la liste des items candidats, travaux parus à ACM RecSys [45, 41].

3.3 Recommandation de points d'intérêts (thèse de Jean-Benoît Griesner)

3.3.1 Recommandation spatio-temporelle de points d'intérêts

Nous avons étudié la recommandation de points d'intérêts avec l'objectif de tenir compte de la date des événements émis par les utilisateurs. Cela permet de détecter des séquences

de points visités consécutivement par un utilisateur et d'obtenir la durée moyenne pour atteindre un point à partir d'une certaine zone géographique. Ce modèle simplifié reflète les moyens de déplacement existants ; nous l'avons intégré dans un modèle de recommandation géographique, ce qui a amélioré la qualité des prédictions. Ce travail est paru dans ACM RecSys [34] et dans la revue RNTI [33] en version étendue.

Cette approche est bien adaptée pour des recommandations à l'échelle d'une ville ou d'une région. En revanche elle devient inefficace à plus grande échelle (pays ou continent) car la représentation segmentée de l'espace géographique sous la forme d'une grille aboutit à un nombre de cellules trop important, sachant que la taille d'une cellule doit rester petite pour cerner le point d'intérêt à recommander.

3.3.2 Recommandation de points d'intérêts basée sur la factorisation de Poisson

Nous avons abordé le cas plus difficile d'une recommandation de points d'intérêts à l'échelle du globe. La quantité relative de points d'intérêts visité par utilisateur est très petite (*i.e.* faible densité), ce qui dégrade fortement la qualité des solutions existantes. Nous avons proposé une solution basée sur la factorisation de Poisson qui offre de bonnes performances en faible densité. Nous avons inféré un réseau d'utilisateurs à l'aide d'une fonction de similarité tenant compte de la distance et de la probabilité de transition entre deux points d'intérêts EGC [35].

3.3.3 Modèle de mobilité pour la recommandation de points d'intérêts

Nous avons étendu la solution proposée ci-dessus pour tenir compte des différents modèles de mobilité des utilisateurs. Nous avons proposé d'agréger les points visités en zones de visite et d'optimiser le niveau d'agrégation selon la mobilité des utilisateurs. Ce travail a été publié dans la conférence EGC [36] (où il a été sélectionné avec quatre autres travaux pour la catégorie des meilleurs articles de recherche appliquée) et aux journées BDA [67].

3.4 Enrichissement des données

Lors de nos travaux sur les modèles et algorithmes de recommandation, un effort important a été investi dans la préparation des données nécessaires à la validation expérimentale des solutions proposées. Deux aspects ont soulevé des problèmes de gestion de données : le

volume des données a conduit à concevoir des solutions plus performantes que celles dont nous disposions, et la présence de biais très prononcé reflétant la réalité des usages, *i.e.* la distribution du nombre d'utilisateurs associés à un item suit une loi de puissance. Par exemple quelques rares points d'intérêts sont associés à un nombre d'utilisateurs beaucoup plus grand que tous les autres points d'intérêt.

3.4.1 Enrichissement des données de déplacement avec des points d'intérêts

L'objectif suivi a été de concevoir une solution scalable pour nettoyer des données très fortement biaisées. Nous avons proposé une solution efficace pour enrichir des méga-données géo-localisées en les croisant avec une base décrivant des points d'intérêt et leurs catégories. Notre contribution principale est une nouvelle méthode pour calculer des jointures spatiales plus rapidement en optimisant la solution GeoSpark [85] existante afin de dédier plus de ressources aux points d'intérêt les plus populaires.

Nous avons appliqué notre méthode pour croiser deux grands jeux de données : d'une part YFCC [81] contenant plus de 100 millions de d'événements dont la couverture géographique est mondiale, et d'autre part, Geonames [1] contenant 10 millions de points d'intérêts annotés avec des catégories. Ce travail est paru dans la conférence INTERSOL 2020 [37].

[1] http ://www.geonames.org/export/

4 Optimisation de requête à large échelle pour le Web sémantique

Lors de la délégation d' O. Curé (UPEM) dans l'équipe en 2015, nous avons initié une activité de recherche à l'intersection entre nos expertises propres. Le raisonnement sur des données sémantiques pour O. Curé et, de mon côté, l'optimisation de requêtes parallèles et distribuées basée sur le coût. Ces travaux ont été financés par le projet CNRS MASTODONS/ARESOS (2014-2016).

Le format RDF (Resource Description Framework) est devenu un standard pour publier et intégrer des données et des connaissances à l'échelle du Web. RDF décrit de manière flexible et générale des entités et des liens entre entités pour former un *graphe de connaissances*. DBPedia[1] [53] est un des principaux graphes de connaissances, publié en libre accès au format RDF, contenant plusieurs milliards de liens provenant de Wikipedia et de nombreux autres jeux de données.

De plus, l'initiative Linked-Open-Data (LOD[2]) a permis d'inter-connecter une grande variété de graphes de connaissances pour former un méga-graphe appelé le *LOD Cloud*. La taille en continuelle croissance de cette méga-structure pose un problème récurrent de performance : interroger efficacement le *LOD Cloud* reste un défi à relever.

Le langage dédié à l'interrogation de graphes RDF est SPARQL. Nous avons étudié le traitement efficace de requêtes SPARQL sur des méga-graphes. L'objectif étant de proposer des solutions scalables (*i.e.* qui passent à l'échelle) qui s'appuient sur une plateforme de calcul parallèle telle que Apache Spark.

[1] `https://www.dbpedia.org/`
[2] `https://lod-cloud.net/`

4.1 Evaluation de requête et raisonnement : l'approche LiteMat

Nous avons défini une solution appelée *LiteMat* basée d'une part sur de la réplication et d'autre part sur de l'encodage sémantique.

Nous avons tout d'abord étudié les stratégies de réplication de très grands graphes RDF en fonction des requêtes fréquemment posées par des utilisateurs. J'ai supervisé T. Randriamalala (stage de Master 1) avec qui nous avons défini un nouvel algorithme parallèle de réplication de fragments de données qui minimise le taux de réplication global.

Le modèle RDF offre la possibilité d'enrichir les données avec des méta-données sémantiques rassemblées dans une *ontologie*. Une ontologie permet, par raisonnement logique, d'inférer d'autres connaissances. Nous avons abordé le problème de traiter efficacement des requêtes sémantiques qui nécessitent, pour être complètes, de raisonner sur la hiérarchie des concepts et propriétés définie dans l'ontologie. A la différence des solutions qui exploitent l'ontologie en amont [70, 69] afin de réutiliser des solutions de traitement de requêtes sans raisonnement, nous avons montré qu'un encodage hiérarchique des données RDF permet d'évaluer efficacement la requête et le raisonnement associé. Nous avons publié un article à la conférence IEEE BigData [22] et un poster à ISWC [20]. La prise en compte de l'héritage multiple dans l'ontologie est été publiée à ESWC [23]. Ces travaux ont également été présentés à la journée *Paris Summit on Big Data Management* organisée par Inria, l'Ecole Polytechnique et Télécom Paris.

4.2 Optimisation de requêtes SPARQL

Une requête SPARQL pouvant être traduite en un plan de jointures distribuées entre des ensembles de données RDF, nous avons défini un modèle de coût permettant de choisir parmi les deux opérateurs standards de jointure distribuée (jointure parallèle par hachage ou jointure par boucles imbriquées avec diffusion) celui qui minimisera les transferts de données, en tenant compte des différents modèles de stockage distribué implémentés dans la plateforme Apache Spark. Nos résultats expérimentaux montrent qu'une utilisation combinée des deux opérateurs de jointure permet d'accélérer significativement les requêtes. Ils ont fait l'objet de deux publications dans les ateliers internationaux SSWS@ISWC [21] et Grades@SIGMOD [64]. J'ai encadré deux étudiantes de Master 1 (Q. He et G. Kouloud) pour un projet logiciel visant à implémenter un optimiseur de requêtes SPARQL basé sur nos travaux. Le logiciel produit

est mis à disposition sur le site web de l'équipe[3]. Ce travail a été adapté pour des flux de données RDF et étendu à du raisonnement à base de règles logiques dont le formalisme s'inspire du langage *Answer Set Programming* ; les résultats ont été publié à la conférence Big Data [71, 72].

4.3 SemGraph

En 2018 et 2019, j'ai initié et animé le projet SemGraph [24] soutenu et financé par le laboratoire LIP6. Le budget annuel alloué à ce projet comprend 6 mois de stage de M2 et les missions qui en découlent. Ce projet vise à initier une nouvelle collaboration entre les équipes Bases de données et Complex Networks. Dans ce projet nous supposons qu'un grand graphe (tel que Wikidata) contient des informations sémantiques latentes dont les caractéristiques (motif de sous-graphe) pourraient être explicitées afin d'optimiser les requêtes posées de manière interactive sur un grand graphe.

L'objectif principal est d'indexer plus efficacement les données en bénéficiant de la connaissance acquise sur la structure du graphe sous-jacent. En rupture avec les approches récentes [51] communément employées qui indexent les structures simples du graphe telles que les étoiles ou les chemins de longueur 2, nous proposons d'indexer des structures topologiques plus complexes obtenues par une phase d'énumération de k-cliques. Le défi est de faire face au volume souvent très important des structures à indexer. Par exemple indexer les cliques n'est pas trivial car leur nombre est souvent beaucoup plus grand que le nombre d'arcs. En nous appuyant sur les travaux de dénombrement de structure semi-régulières [26], nous proposons une organisation hiérarchique des structures particulières. Nous étudions le gain en performance que cela apporte pour le traitement des requêtes des utilisateurs.

Nous avons étendu un algorithme efficace d'énumération de k-cliques (conçu en partie par Maximilien Danisch, membre de l'équipe Complex Nextworks) pour tenir compte des labels et de l'orientation des arcs. J'ai co-encadré l'étudiant S. Fernandez en stage de Master 1 (de juin à septembre 2018) qui a implanté une extension de l'algorithme et validé expérimentalement le bénéfice qualitatif de l'approche sur le graphe de connaissances YAGO3 [61] dans le cas de la fouille de triangles (une relation entre trois entités). En 2019, j'ai encadré deux étudiants de Master 1 (S. Sadaoui et S. Moulouel) pour un projet logiciel sur l'analyse topologique de données encyclopédiques. Cela a permis de quantifier plus précisément la pertinence de

[3] `http://www-bd.lip6.fr/wiki/en/site/recherche/logiciels/sparqlwithspark`

notre solution de fouille de triangles et d'identifier des classes de triangles pour lesquels la fouille réussit à inférer des nouvelles connaissances valides. Ces résultats font l'objet d'un article en préparation et ont impulsé le travail sur les requêtes en forme de triangle, présenté ci-dessous.

4.4 Triag : requêter les motifs triangulaires dans les grands graphes RDF

Lors de l'interrogation de graphes de connaissances, un motif intéressant et fréquemment interrogé est celui ayant une forme triangulaire. En effet un triangle peut représenter des interactions denses ou fortes entre trois entités. Par exemple, la recherche de collaborations étroites dans un réseau de personnes, s'exprime par une requête en forme de triangle reliant trois personnes A, B et C. En général, les arcs du graphe sont orientés et forment deux types de triangles : les triangles cycliques ou acycliques dont une illustration est présentée sur la figure 4.1 : supposons que les arcs représentent par exemple la relation « *donne à* », un triangle acyclique est tel que Bob donne à Alice et Diana, et Alice donne à Diana. Un triangle cyclique est tel que Diana donne à Hervé qui donne à Carole qui donne à Diana.

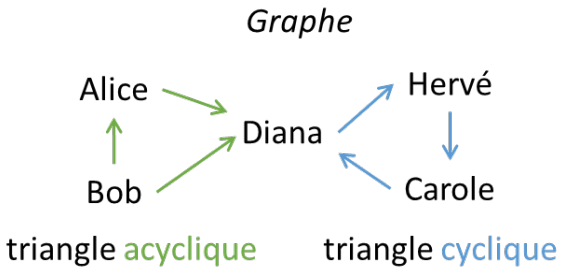

Figure 4.1 : Exemple de triangle cyclique et acyclique

Une requête définissant un motif en triangle peut être formulée dans le langage SPARQL et peut donc être traitée dans tout système supportant ce langage. Néanmoins, bien que la forme d'un motif triangulaire soit simple, son évaluation sur des très grands graphes de données réelles pose deux problèmes :

- un problème de performance dans un environnement parallèle et distribuée lié à la présence de données fortement biaisées,

- un problème de résultats incomplets car si on ne calcule pas les arcs inférés, il manque alors tous les motifs ayant au moins un arc inféré.

Nous détaillons ci-dessous les solutions que nous proposons pour ces deux problèmes.

4.4.1 Requête triangle en présence de données fortement biaisées

Afin de gérer des très grands graphes, on suppose un environnement parallèle et distribué (par exemple la plateforme Spark). Ce type d'environnement devient nécessaire quand le graphe est trop grand pour tenir sur une seule machine ou quand on veut disposer d'un nombre de processeurs (ou cœurs) supérieur à celui d'une seule machine, afin de paralléliser davantage les calculs. Dans ce contexte, les arcs sont distribués entre plusieurs machines. Le calcul d'une requête triangle consiste à traiter une première jointure pour obtenir les chemins de longueur 2 puis une deuxième jointure pour obtenir les triangles. La figure 4.2 détaille la première jointure qu'on appelle $J1_{acycl}$ (resp. $J1_{cycl}$) pour les requêtes en triangle acyclique et cyclique respectivement.

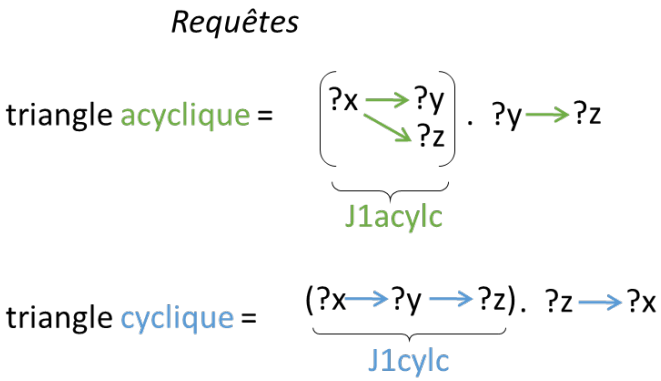

Figure 4.2 : Requêtes avec un motif en triangle cyclique ou acyclique

Dans l'environnement étudié, cette première jointure $J1$ produisant des chemins de longueur 2 est traitée de manière parallèle et distribuée. Or l'évaluation de cette jointure pose un problème de performance qui est dû principalement au biais sur les données. Un graphe de connaissances est biaisé lorsqu'il contient des nœuds dont le degré entrant et/ou sortant est très supérieur à celui des autres nœuds. Par exemple, dans un graphe décrivant les collaborations entre personnes, le nombre de collaborateurs est biaisé lorsque quelques personnes seulement (*e.g.*, Diana sur la figure 4.3) collaborent avec un très grand nombre de personnes

tandis que les autres personnes collaborent avec peu de personnes.

Lorsque les données sont fortement biaisées, on constate que le calcul de $J1_{acycl}$ et $J1_{cycl}$ n'est pas réparti uniformément sur toutes les machines : la majeure partie du temps écoulé consiste à calculer le résultat concernant le nœud dont le degré est beaucoup plus grand que celui des autres nœuds, ceci en n'utilisant qu'une seule machine parmi toutes les machines disponibles.

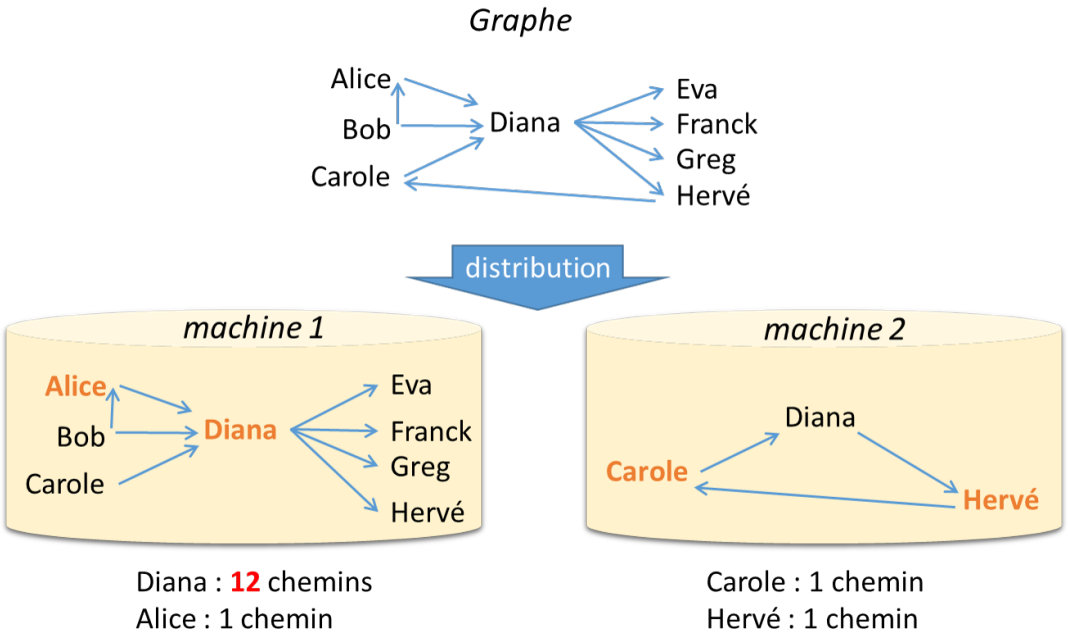

Figure 4.3 : Exemple de données biaisées

Par exemple, pour traiter $J1_{cycl}$, c'est à dire pour déterminer tous les chemins de longueur 2, les arcs sont distribués de telle sorte que pour chaque personne p, les arcs entrants et sortants de p sont rassemblés sur la même machine. La machine 1 reçoit les arcs entrant et sortant d'Alice et Diana, la machine 2 reçoit ceux de Carole et Hervé. Les autres personnes ne sont pas considérées car elles n'ont pas au moins un arc entrant et un arc sortant. Chaque machine peut ainsi calculer les chemins de longueur 2 en parallèle. Or la machine 1 produit 13 chemins ce qui est très supérieur aux 2 chemins produits par la machine 2. En conséquence, l'utilisation des deux machines n'est pas optimale.

En présence de très grands graphes, cela pose un problème de performance car des machines sont sous-utilisées et d'autres sont sur-utilisées : la plupart des machines passent leur temps à attendre que la machine traitant le nœud de plus fort degré (le nœud Diana

dans l'exemple) ait terminé son calcul. Face à cette situation, notre objectif est d'améliorer l'utilisation des ressources de calcul en évitant que le biais ne ralentisse le traitement des requêtes.

Nous avons proposé une méthode consistant à estimer, avant de traiter la jointure, l'impact d'un nœud ayant un biais élevé, sur le traitement des requêtes cycliques et acycliques. Le principe est de s'appuyer sur cette estimation pour « isoler » la partie biaisée du graphe afin de la traiter séparément du reste du graphe. Pour cela nous collectons le degré entrant et sortant de chaque nœud. Soit d_n^{in} (resp. d_n^{out}) le degré entrant (resp. sortant) du nœud n. Le nombre de chemins de longueur 2 (*i.e.* la requête $J1_{cycl}$ sur la figure 4.2) passant par n est :

$$C_n = d_n^{in} \times d_n^{out}$$

On note C_{total} le nombre total de chemins de longueur 2 dans le graphe G :

$$C_{total} = \sum_{n \in G} C_n$$

Soit p le nombre de processeurs (ou cœurs) dont on dispose pour calculer la jointure. Dans l'idéal, si la répartition du calcul était optimale alors chaque processeur produirait $\frac{C_{total}}{p}$ chemins. Cette valeur sert de point de repère pour séparer le graphe en deux parties :

- G_{biais} contient les nœuds ayant les plus fortes valeurs de C_n telles que $C_n > \frac{C_{total}}{p}$, ainsi que les arcs associés à ces nœuds. On remarque qu'en pratique pour la plupart des nœuds on a $C_n << \frac{C_{total}}{p}$ car le nombre de nœuds est très supérieur au nombre de processeurs p, *i.e.* un processeur traite un grand nombre de nœuds.
- $G_{principal}$ contient tous les autres nœuds et leurs arcs associés.

La requête $J1_{cycl}$ est tout d'abord évaluée sur $G_{principal}$ dont le biais est suffisamment faible pour ne pas ralentir le traitement.

Puis $J1_{cycl}$ est évaluée sur G_{biais} (la partie fortement biaisée du graphe) en utilisant un autre algorithme de jointure appelée *jointure par diffusion* et dont la performance n'est pas dégradée par la présence de biais. Les étapes de cet algorithme de jointure sont :

- Les arcs sortants de G_{biais} sont diffusés vers chaque machine et ils sont indexés selon leur nœud d'origine. Le surcoût induit par cette diffusion/indexation reste relativement faible car la taille de G_{biais} est petite.
- Les arcs entrants sont attribués aux machines selon leur nœud d'origine. Cela permet de répartir uniformément sur plusieurs machines les arcs entrants d'un nœud ayant

un degré élevé. Par exemple, les arcs $Alice \rightarrow Diana$, $Bob \rightarrow Diana$ et $Carole \rightarrow Diana$ sont attribués à trois machines différentes.

- Chaque arc entrant est associé avec un ou plusieurs arcs sortants pour former des chemins de longueur 2. Chaque machine produit approximativement la même quantité de chemins car le degré sortant moyen est identique sur chaque machine. Ainsi, la charge de calcul est équilibrée entre les machines.

Remarque : bien que la jointure par diffusion soit robuste au biais, il n'est pas optimal de l'utiliser sur le graphe G en entier car le surcoût de diffusion deviendrait trop élevé ; en conséquence, la durée totale de la jointure dépasserait la durée de la solution proposée. Cela justifie l'intérêt de traiter $J1_{cycl}$ avec deux algorithmes de jointure sur deux parties de G.

Pour calculer $J1_{acycl}$, nous adoptons une méthode similaire. Le nombre d'éléments du résultat pour un nœud n devient :

$$C'_n = d_n^{out} \times d_n^{out}$$

On remarque que $C'n$ dépend seulement du degré sortant car les deux arcs sont reliés par leur nœud d'origine, *i.e.* la variable x sur la figure 4.2. Le biais pour les requêtes acycliques est moins fort que pour les requêtes cycliques, car la valeur maximale de d_n^{out} est très inférieure à la valeur maximale de d_n^{in}. Néanmoins, la méthode proposée permet d'accélérer significativement la traitement de $J1_{acycl}$.

Finalement, les résultats intermédiaires $J1_{acycl}$ et $J1_{cycl}$ sont complétés comme indiqué sur la figure 4.2) pour obtenir le résultat final attendu, c'est-à-dire l'ensemble des triangles cycliques et acycliques du graphe. Cette dernière étape consiste en une jointure sur deux variables : (y, z) pour les triangles acycliques et (x, z) pour les triangles cycliques. Ainsi, pour chaque élément de $J1_{acycl}$ (ou $J1_{cycl}$) il existe au plus un seul triangle dans le résultat final car un seul arc peut relier les deux variables de jointure. Autrement dit, un graphe ne peut pas avoir plus d'un arc entre 2 nœuds. Cette propriété garantit l'absence de biais et permet de terminer les requêtes triangle sur le graphe entier, en suivant la méthode d'optimisation de requêtes que nous avons proposée dans un travail précédent [64]. Nous avons validé expérimentalement le bénéfice de l'approche proposée sur des grands graphes de connaissances, en particulier sur Yago3 [61] avec en utilisant un cluster 8 machines soit un total de 96 cœurs [65].

Conclusion. Ce travail a été publié dans la revue *Open Journal of Web Semantics* [65]. Il montre l'intérêt de s'affranchir d'hypothèses simplificatrices sur l'uniformité des données

afin de concevoir des solutions plus performantes car adaptées aux spécificités des données. La contribution principale est une méthode tenant compte du biais dans les données pour traiter une jointure plus efficacement. De plus, ce travail s'inscrit dans la démarche poursuivie actuellement pour améliorer les optimiseurs de requêtes dans les plateformes parallèles et distribuées d'analyse de données. Notamment, une solution connexe a été proposée récemment par Spark pour améliorer la performance des jointures en présence de biais [4]. Toutefois cette solution ne s'appuie pas sur une estimation a priori du biais mais détecte a posteriori l'occurrence d'une situation de sous-utilisation des ressources pendant une jointure. Cela déclenche l'arrêt du calcul courant qui est alors reformulé sur la partie non encore traitée des données. Cette approche complémentaire évite de pré-calculer des statistiques sur les données. Cependant la détection ajoute une certaine latence et une partie du traitement concernant les données biaisées est redondante car exécutée à la fois avant et après la détection de la situation de sous-utilisation.

4.4.2 Requête avec inférence de triangles

Un graphe de connaissances est un ensemble de triplets RDF tels qu'un sujet est relié à un objet par une propriété. Un triplet est noté (*sujet*, *propriété*, *objet*). Pour la propriété rdf:type, le sujet est une entité et l'objet est le type de cette entité. L'objectif de ce travail est de montrer que certaines requêtes en forme de triangle peuvent être exécutées plus efficacement que la méthode générale qui consiste à calculer deux jointures. Cela concerne les requêtes ayant le motif triangle décrit dans la figure 4.4. Il s'agit de trouver deux entités x et y ayant le même type T et reliées par une propriété p. L'idée est d'exploiter la connaissance du type des entités pour accélérer l'évaluation de cette requête.

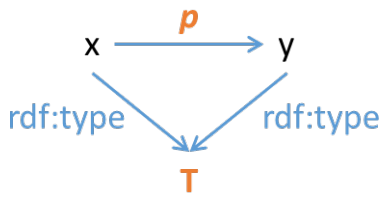

Figure 4.4 : Motif triangulaire des requêtes étudiées

Dans un graphe de connaissances, une partie des triplets décrit le schéma des données.

[4] `http://docs.databricks.com/delta/join-performance/skew-join.html`

Le schéma est défini à l'aide des propriétés spécifiques du standard RDFS (*e.g.*, *subClassOf*, *domain* ou *range*), et il contient les informations suivantes illustrées sur la figure 4.5 :

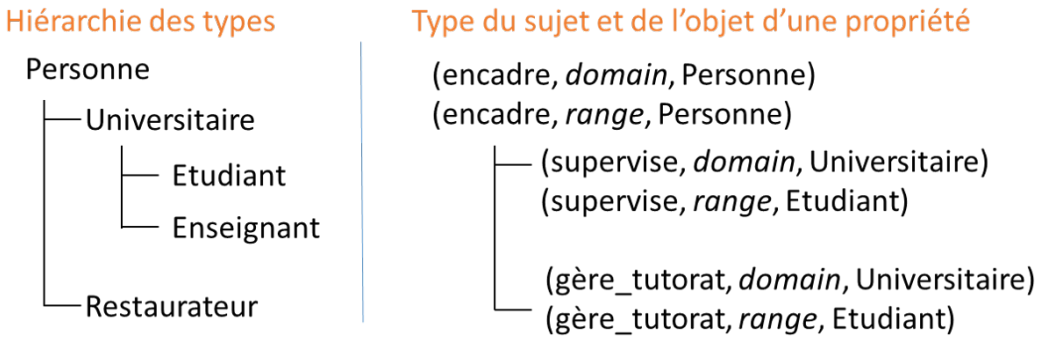

Figure 4.5 : Exemple de schéma

- la hiérarchie des types. Dans l'exemple suivant nous considérons une hiérarchie à trois niveaux : *Personne* a deux sous-types : *Restaurateur* et *Universitaire*, ce dernier a aussi deux sous-types : *Enseignant* et *Etudiant*.
- la hiérarchie des propriétés, par exemple *encadre* a pour sous-propriétés *supervise* et *gère tutorat*.
- pour toute propriété, le schéma précise le type des entités reliées par cette propriété. Par exemple, on sait que la propriété *encadre* relie toujours une personne avec une autre personne, la propriété *reside* relie toujours une personne avec un pays.

Le schéma des données permet de raisonner sur la base pour inférer des triplets de type, c'est à dire pour déduire du schéma les types des entités. Par exemple, sachant qu'un individu E est un enseignant, on peut inférer que c'est une personne bien que le graphe ne contienne pas explicitement le triplet (E, rdf:type, Personne).

Nous supposons que la méthode (appelée parfois saturation) consistant à compléter le graphe avec l'ensemble des triplets pouvant être inférés, n'est pas souhaitable car cela provoque une explosion de la taille du graphe lorsque le nombre de types par entité est grand et cela tend à dégrader les performances des requêtes. Par contre, le résultat de la requête de la figure 4.4 serait généralement incomplet s'il se limitait aux triplets existant dans la base car il manquerait les réponses pouvant contenir des triplets inférés. La figure 4.6 illustre ce problème avec un graphe contenant 6 triplets concernant 4 individus. L'information sur les types Personne et Universitaire n'est pas explicite dans le graphe mais peut être inférée à

partir du schéma.

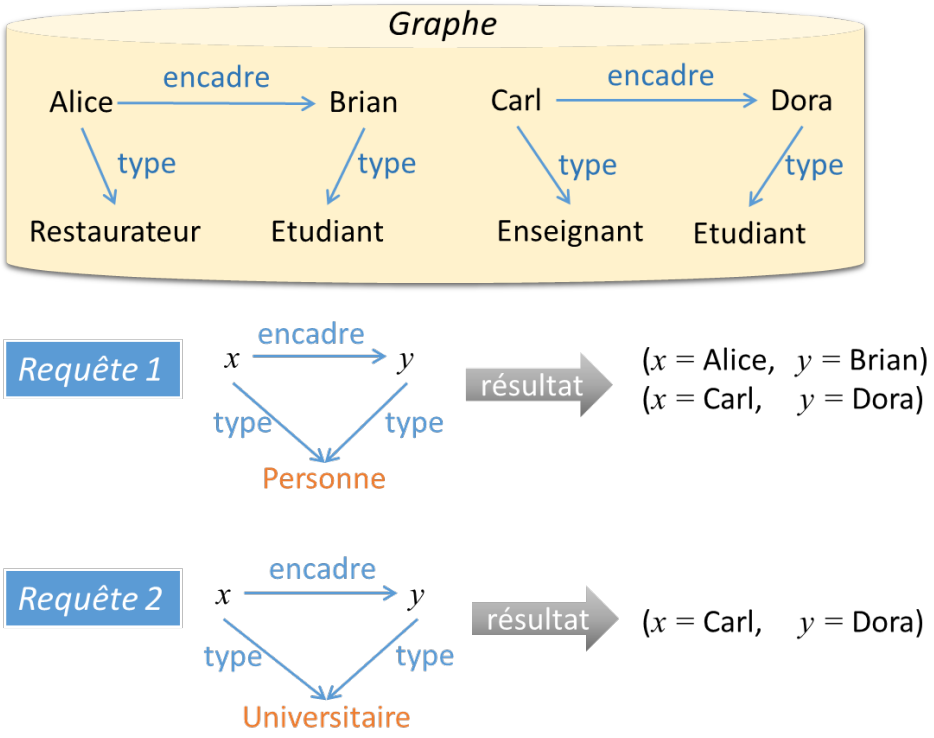

Figure 4.6 : Exemple de requêtes triangulaires nécessitant du raisonnement sur les types

La requête 1 recherche une personne qui encadre une autre personne. Le résultat contient tous les individus reliés par la propriété encadre : (Alice, encadre, Brian) et (Carl, encadre, Dora), chacun formant un triangle avec le concept *Personne*. En effet, on peut déduire du schéma que ces individus sont des personnes bien que le graphe ne contienne pas explicitement cette information. Ainsi, la requête 1 peut être simplifiée en un seul triplet : (x encadre y).

La requête 2 recherche un universitaire qui encadre un autre universitaire. Le résultat contient (Carl, encadre, Dora) car Enseignant et Etudiant sont des sous-concepts de *Universitaire*. Par contre, (Alice, encadre, Brian) n'est pas dans le résultat car le type d'Alice - Restaurateur - n'est pas un sous type d'*Universitaire*. L'évaluation de cette deuxième requête nécessite de déterminer si le type des entités reliées par la propriété encadre est plus spécifique ou égal à celui de la requête. Nous montrons que cette requête peut être exécutée efficacement en ajoutant un filtre sur le type des entités.

Pour résoudre ce problème, nous avons proposé une solution permettant de retrouver la

totalité des réponses des requêtes de la figure 4.4. Pour cela nous avons exploité la hiérarchie des types afin de déterminer le plus petit concept commun à deux concepts. Un encodage compact des types, prenant en compte leur hiérarchie, permet d'accélérer l'évaluation des tests d'inclusion de type. Une structure de données associe chaque propriété avec le plus petit type commun aux types du sujet et de l'objet, d'après les informations contenues dans le schéma. De plus, les éléments du résultats de la requête 4.4 tels que T est plus spécifique que le type commun déduit à partir du schéma, sont indexés pour être retrouvés efficacement. Ce travail a été validé expérimentalement et publié dans le workshop SBD@SIGMOD [66].

4.5 Bilan

Nous avons apporté plusieurs solutions pour l'optimisation de requêtes SPARQL dans les méga-graphes sémantiques. Nos travaux confirment le fait que l'optimisation de requêtes dans un environnement distribué reste un domaine vaste avec de nombreuses opportunités pour de prochaines contributions. Notamment, lorsque la sémantique des données et des requêtes peut être exploitée afin de réduire les calculs effectués. D'autres informations sémantiques plus riches pourraient bénéficier à l'évaluation des requêtes, par exemple savoir qu'une propriété est transitive (resp. symétrique) permettrait d'optimiser la représentation sous-jacente du graphe et de traiter beaucoup plus rapidement les requêtes interrogeant ces propriétés. Les résultats sur le traitement de requêtes en présence de données biaisées peuvent être appliqués de manière plus générale à d'autres type de graphes (réseaux sociaux, graphe d'usage, *etc*.).

De plus, ces travaux ont nécessité un effort de mise en œuvre conséquent pour déployer et maintenir une plateforme de calcul parallèle et distribué. Ce savoir-faire est un atout qui est mis à profit dans d'autres travaux de recherche entrepris dans l'équipe bases de données et lors de diverses collaborations extérieures.

5 Optimisation de workflow en science des données

5.1 Contexte

Depuis 2015, les plateformes de calcul parallèle ont gagné en généricité et en flexibilité. En science des données, elles sont maintenant largement adoptées pour concevoir et réaliser des chaînes de traitement de mégadonnées. On parle de workflow de *data science*. Toutefois, la taille des données manipulées et la complexité des traitements soulèvent des problèmes de performance : bien que les opérations élémentaires soient efficacement supportées par une plateforme, on constate qu'une chaîne de traitement composée de nombreuses opérations ne s'exécute pas toujours efficacement. Optimiser les performances d'un workflow pour la science des données est un défi difficile. En effet, le workflow peut contenir des opérations de fouille ou d'apprentissage dont les hyper-paramètres doivent être ajustés empiriquement. Ma démarche est de capitaliser sur l'expérience acquise dans l'optimisation de requêtes à large échelle afin de proposer des méthodes d'optimisation de chaînes de traitement en science des données. Je poursuis cette démarche en prenant en considération des méthodes de fouille de texte où le besoin d'analyser des mégadonnées de manière systématique est fort.

5.1.1 Epique : un workflow pour analyser des grands corpus scientifiques

J'ai participé au montage du projet ANR Epique[1] (2017-2021) coordonné par B. Amann responsable de l'équipe Bases de données.

Ce projet part du constat qu'il existe une demande croissante d'outils pratiques pour explorer l'évolution de la recherche scientifique publiée dans des archives bibliographiques telles que le Web of Science (WoS), arXiv, PubMed ou ISTEX. La mise en évidence de modèles

[1] http ://www-bd.lip6.fr/wiki/site/recherche/projets/epique/start

d'évolution significatifs à partir de ces archives documentaires a de nombreuses applications, par exemple en histoire des sciences ou dans la gouvernance de l'activité scientifique.

Le projet Epique vise à reconstruire l'évolution des sciences dans le temps, en utilisant des méthodes quantitatives pour analyser des grands corpus textuels de publications scientifiques. Il s'agit de produire une *cartographie* des sciences au fil du temps. Les domaines, représentés par des mots-clés, sont positionnés par période croissante : des plus anciennes en haut de la carte aux plus récentes en bas de la carte. Pour représenter le changement progressif du contenu d'un domaine, des liens connectent les domaines similaires appartenant à deux périodes consécutives. La structure de la carte est un graphe complexe : il contient des motifs d'évolution qu'il s'agit de mettre en évidence.

Une des nouveautés du projet est de rendre les cartes **interactives** afin d'augmenter les possibilités d'exploration et à terme découvrir des nouvelles formes d'évolution des sciences. Les paragraphes suivants détaillent mes principales contributions : la définition de méthodes d'extraction de domaine et d'alignement qui soient efficaces et rendent possible un usage interactif du workflow.

5.2 Extraction de domaines : calcul parallèle d'ensembles fréquents maximaux

J'ai pris la responsabilité du workpackage sur l'extraction de domaines ou *topics*. J'ai tout d'abord encadré le stagiaire de Master 2 Firas Atem (6 mois 03/2017-08/2017) sur la conception d'un algorithme parallèle efficace pour extraire les ensembles de mots-clé fréquents maximaux. Le code produit a été livré dans le projet Epique. Une première preuve de concept a été présentée à la conférence Bases de données avancées [16].

5.2.1 Contexte et Problématique

Ce travail concerne l'extraction de domaines ou *topics* dans des grands corpus de documents. On considère qu'un document est composé d'un ensemble de termes. Certains termes ont des affinités avec d'autres termes et forment des ensembles qui apparaissent dans au moins s documents ; ce sont des ensembles fréquents (EF) de termes avec un support de s.

Dans le but de caractériser les domaines étudiés dans un corpus de documents on veut

obtenir des ensembles fréquents qui soient suffisamment différents les uns des autres. Etant donné deux domaines, si tous les termes du premier domaine sont inclus dans le deuxième domaine, alors on considère que ces deux domaines n'en représentent qu'un seul : le premier est une description incomplète et on cherche à obtenir seulement le deuxième domaine plus complet. Autrement dit, on cherche les ensembles fréquents maximaux (EFM) définis ainsi : un ensemble fréquent est **maximal** s'il n'est pas contenu dans un autre ensemble fréquent.

De nombreux travaux en fouille de données ont étudié le calcul des EF. Les principaux algorithmes qui offrent de bonnes performances à large échelle sont résumés dans le livre [54]. Nous constatons que les solutions existantes pour calculer les EFM ne passent pas à l'échelle sur des grands corpus de plusieurs millions de documents. Notre objectif est de s'affranchir de cette limitation. Notons que des solution connexes ont été proposées pour calculer efficacement des ensembles fréquents qui satisfont d'autres critères de maximalité tels que l'entropie [74]. Ces solutions sont pertinentes pour une tâche de recherche d'information mais le sont moins pour décrire des domaines scientifiques. Une autre solution [7] s'intéresse à minimiser le coût monétaire du calcul des ensembles fréquents lorsque les ressources de calcul sont facturées à l'usage.

Nous proposons une solution efficace pour calculer les EFM sur des grands corpus. Les tableaux suivants résument les approches récentes sur lesquelles notre travail s'appuie.

- Pour le calcul des EF, *cf*. tableau 5.1, la solution FP-Growth [14] propose une structure de donnée compacte mais centralisée pour représenter les EF. La solution Parallel-FP [56], implantée dans la plateforme Spark, consiste à distribuer la structure de données de FP-Growth pour permettre le calcul les EF en parallèle.
- Pour le calcul des EFM, *cf*. tableau 5.2, la solution centralisée FP-Max [86] adapte la structure de FP-Growth pour représenter les EFM de manière compacte, et définit un algorithme pour les calculer. Mais l'algorithme FP-Max est séquentiel et utilise une structure globale accumulant tous les EFM déjà calculés. Cela permet de tester si un EF est inclus dans un EFM qui existe dans la structure globale. Cette approche globale ne peut pas être distribuée simplement.

Notre solution consiste à étendre la solution centralisée FP-Max afin de permettre le calcul parallèle et distribué des EFM. Les deux défis sont :

1. concevoir une structure de données distribuée pour représenter des EFM,

	FP-Growth	Parallel-FP
centralisée	✓	
distribuée	✗	✓

Table 5.1 : Méthode centralisée ou distribuée pour calculer les EF

	FP-Max	Solution proposée
centralisée	✓	
distribuée	✗	✓

Table 5.2 : Méthode centralisée ou distribuée pour calculer les EFM

2. adapter l'algorithme FP-Max pour isoler les opérations qui peuvent être traitées en parallèle.

5.2.2 Résumé des contributions

Les contributions améliorent l'efficacité du calcul des EFM. Un des principes suivi est de concevoir une structure de données **compacte** pour représenter les EFM, afin de réduire le surcoût de transfert de données entre les machines impliquées dans le calcul. Les contributions de ce travail sont les suivantes :

- Une structure de données distribuée et compacte pour représenter les EFM.
- Un algorithme parallèle et distribué pour extraire des EFM de manière indépendante, puis les fusionner afin de garantir que chaque ensemble est bien maximal vis-à-vis de tous les autres ensembles.
- Une validation de la scalabilité de l'approche sur des très grand corpus de documents.

Nous détaillons tout d'abord la structure de données distribuée pour représenter les EFM, puis nous présentons l'algorithme parallèle qui les calcule. Finalement nous présentons les expériences menées pour valider les performances de la solution.

5.2.3 Représentation distribuée des EFM

Nous précisons les notations utilisées pour définir les structures compactes qui représentent les EF et les EFM. Nous nous appuyons sur une représentation en arbre qui évite d'énumérer explicitement tous les EFM recherchés. Intuitivement, les EFM correspondent aux chemins de l'arbre, or le nombre de nœuds d'un arbre est en général très inférieur à la somme du nombre de nœuds de tous les chemins.

Notations

On note $(t_1, ..., t_n)$ les termes du corpus. La fréquence du terme t_i est notée f_i ; c'est le nombre de documents contenant t_i. Les termes sont totalement ordonnés par fréquence décroissante et, pour les ex aequo, par ordre alphabétique. L'ordre d'un terme est noté $ordre(t)$ et on a $t_1 > t_2$ ssi $ordre(t1) > ordre(t2)$.

EF_i représente tous les ensembles de termes qui contiennent t_i (dans les documents du corpus), et tels que le terme t_i est le plus petit terme de chaque ensemble. Notons que dans la figure 5.1 le terme t_i n'est pas représenté dans EF_i car chaque branche se termine implicitement par t_i.

EF_i est structuré de manière compacte par un arbre dans lequel tout préfixe commun à deux EF (ou plus) est factorisé. Pour connaître la fréquence d'un EF, chaque nœud de l'arbre contient le nombre de chemins passant par ce nœud et allant à la racine. La structure d'arbre est avantageuse car elle permet de fusionner plusieurs EF_i en un seul arbre dont la taille est inférieure à la somme des arbres à réunir ; le gain de taille étant d'autant plus grand que les EF partagent des termes communs.

EFM_i représente tous les EFM contenus dans EF_i : c'est un arbre tel que chaque branche est un EF *localement* maximal par rapport aux EF de EF_i.

Exemple : On considère le corpus suivant :
doc_1 = (santé soin vaccin),
doc_2 = (santé soin virus),
doc_3 = (crise hôpital soin virus),
doc_4 = (hôpital santé soin),
doc_5 = (santé virus).

Les termes sont triés par fréquence décroissante puis par ordre alphabétique. On a l'ordre : (soin, santé, virus, hôpital, crise, épidémie, vaccin).

Chaque document est trié selon cet ordre :
doc_1 = (soin santé vaccin),
doc_2 = (soin santé virus),
doc_3 = (soin virus hôpital crise),
doc_4 = (soin santé hôpital),
doc_5 = (santé virus).

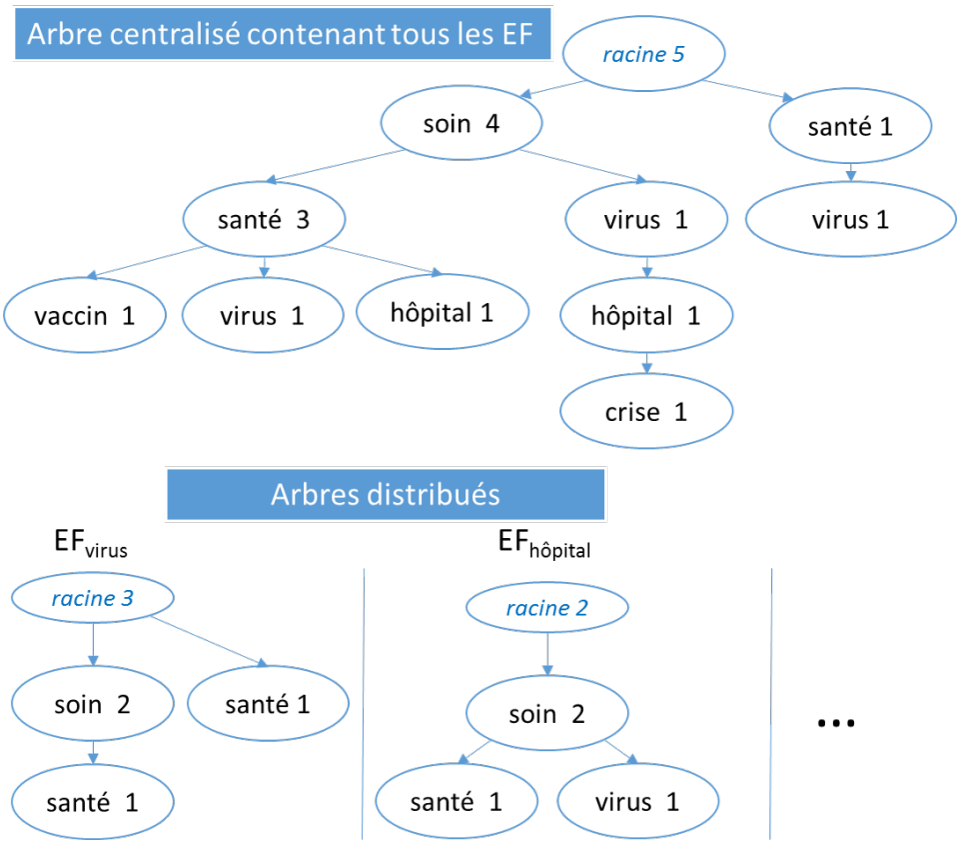

Figure 5.1 : Structure centralisée ou distribuée pour représenter les EF

Etant donné un support s, les EF_i sont construits pour chaque terme i et ne contiennent que les EF dont la fréquence est supérieure ou égale à s. La figure 5.1 montre l'arbre représentant tous les EF de cet exemple pour $s = 1$. C'est une structure centralisée. Au contraire, les arbres EF_{virus}, $EF_{hopital}$, etc. (un EF_i par terme) peuvent être distribués et manipulés indépendamment. Notons que dans l'arbre EF_{virus}, le nœud *soin* a une fréquence égale à 2 car il y a 2 chemins (virus,santé,soin,racine) et (virus,soin,racine) dans l'arbre centralisé.

Puis on détermine un arbre EFM_i correspondant à chaque EF_i. En général, l'arbre EFM_i a moins de branches que son EF_i correspondant car les branches non maximales, *i.e.* celles incluses dans d'autres branches, sont supprimées. Finalement, les EFM_i sont fusionnées pour obtenir un arbre contenant seulement les EF globalement maximaux, *cf.* figure 5.2

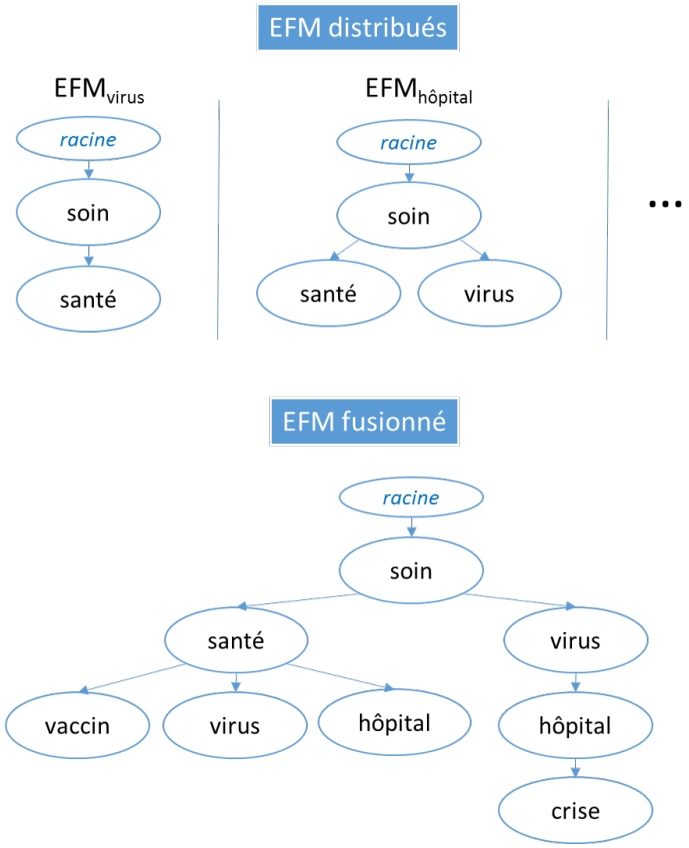

Figure 5.2 : EFM : maxima locaux distribués / maxima globaux après fusion

Calcul parallèle des EFM

La solution consiste à calculer les EF_i, puis les EFM_i puis à les fusionner. Chacune de ces étapes est divisée en plusieurs tâches indépendantes qui peuvent s'exécuter en parallèle. Pour cela, les documents sont distribués aléatoirement entre les machines. Puis chaque machine est désignée responsable d'une partie distincte des termes.

Calcul des EF_i. Cette étape est issue de la solution Parallel-FP et permet de construire les EF_i indépendamment les uns des autres. Chaque machine calcule *partiellement* la fréquence

des termes parmi les documents qu'elle contient. Les résultats partiels sont transmis aux machines responsables et additionnés pour obtenir les fréquences *complètes* des termes. Ce résultat est diffusé à toutes les machines pour qu'elles disposent de l'ordre total des termes.

Le calcul des EF_i est lui aussi parallélisé en fonction du terme i. Chaque machine trie chacun de ses documents puis génère les ensembles fréquents e_i tels que leurs termes sont triés. Les e_i sont transmis à la machine responsable de i pour qu'elle les rassemble dans l'arbre EF_i.

Calcul des EFM_i locaux. Cette étape se déroule localement sur chaque machine. Chaque EF_i est parcouru pour obtenir l'arbre EFM_i correspondant. Lors du parcours d'un EF_i, chaque EF est comparé avec les EFM déjà insérés dans EFM_i, afin de garantir que seuls des EF maximaux sont ajoutés dans EFM_i.

Calcul du résultat final La structure en arbre permet de fusionner efficacement deux EFM_i : chaque branche du premier est insérée dans le deuxième en prenant soin de ne pas introduire de branches redondantes. En particulier, nous avons défini un algorithme qui exploite l'ordre global des termes pour tester efficacement qu'un EF est maximal, *i.e.* qu'il n'existe pas de chemin dans l'arbre égal à cet EF. Les fusions successives aboutissent à un unique EFM_I tel que I est l'ensemble de tous les termes. Cet arbre global représente les EFM recherchés.

5.2.4 Validation expérimentale de l'approche

Nous avons implanté l'approche en langage scala sur la plateforme Apache Spark, et comparé les performances obtenues par rapport à l'état de l'art.

Méthode de référence

La méthode qui nous sert de référence exploite de la manière la plus directe possible les fonctions de calcul parallèle et distribué fournies par Apache Spark ; nous l'appelons $EFM_{Reference}$. Elle utilise Parallel-FP pour générer tous les EF. L'ensemble des EF obtenus est nommé $\mathcal{F}$ et f est un élément de $\mathcal{F}$. L'ensemble $\mathcal{M}$ des EFM est obtenu en sélectionnant les f qui ne sont pas inclus dans un autre EF. On a :

$$\mathcal{M} = \{f | f, f' \in \mathcal{F} \wedge \neg(\exists f'(f \subset f'))\}$$

$EFM_{Reference}$ est exécutée en deux étapes sur un cluster de n machines ou nœuds de calcul appelés $\{N_1, \cdots, N_n\}$. Nous détaillons les étapes en précisant leur coût. Dans le but de mesurer le bénéfice relatif de notre solution par rapport à $EFM_{Reference}$, nous ne tenons pas compte du coût de l'étape 1 car elle est commune aux deux approches.

1. Calculer $\mathcal{F} =$ Parallel-FP$(Dataset)$. La fonction Parallel-FP est invoquée pour calculer $\mathcal{F}$. A la fin du calcul, chaque machine N_i dispose d'une partie distincte $\mathcal{F}_i$ de $\mathcal{F}$.
2. Calculer $\mathcal{M} = \mathcal{F} \setminus \sigma_{f \subset f'}(\mathcal{F} \times \mathcal{F}')$. L'ensemble $\mathcal{F}'$ (=$\mathcal{F}$) est diffusé sur les n machines. Puis chaque machine n_i compare ses EF $f \in \mathcal{F}_i$ avec tous ceux de $\mathcal{F}'$ pour sélectionner les EF non maximaux. Ceux-ci sont retranchés de $\mathcal{F}_i$ pour obtenir $\mathcal{M}_i$. Au final chaque machine dispose d'une partie distincte $\mathcal{M}_i$ de $\mathcal{M}$.

Au total, le traitement de $EFM_Reference$ nécessite de transférer une quantité de données valant $n \times taille(\mathcal{F})$. Le nombre de tests d'inclusion est de l'ordre de $|\mathcal{F}|^2$.

Durée de calcul des EFM

Nous avons expérimenté notre solution sur le corpus Medline contenant 23 millions d'articles scientifiques publiés dans le domaine médical entre 1946 et 2016. Le nombre annuel de documents augmente avec les années pour dépasser 800 000 documents par an de 2012 à 2016. Le vocabulaire contient 27 300 termes et il y a en moyenne 10 termes par documents.

L'objectif de cette expérience est de mesurer la performance relative de la solution proposée pour différentes situations dont la complexité est de plus en plus grande. Or la complexité dépend directement du nombre d'EFM à calculer. Pour augmenter le nombre d'EFM dans le résultat du calcul, il suffit de diminuer la valeur s du support donnée initialement. En effet, soit D le nombre de documents manipulés, on sait par définition du support que chaque EFM doit être dans au moins $s \times D$ documents. Donc lorsque le support diminue en passant de s_1 à s_2 tel que $s_1 > s_2$, les nouveaux EFM présents dans un nombre de documents compris entre $D.s_2$ et $D.s_1$ sont ajoutés dans le résultat. En faisant varier le support nous mesurons une série de calculs d'EFM qui ont une complexité croissante : plus le support diminue, plus le nombre d'EFM à calculer augmente et plus le calcul est long. Le tableau 5.3 résume les mesures effectuées, avec le support s' étant la valeur normalisée du support s telle que $s' = s/nombre\ de\ documents$.

Lorsque $s' > 0,4\%$, la durée de notre solution (50 sec) est sensiblement identique à celle de

support s' (%)	$\|EFM\|$	Solution de référence (secondes)	solution proposée (secondes)	Gain
0,9	449	26	25	1,04
0,4	983	48	46	1,06
0,2	1858	118	50	**2,3**
0,1	4991	390	75	**5,2**
0,05	13528	3000	117	**25**

Table 5.3 : Comparaison de la durée de calcul des EFM : résumé des mesures

$EFM_{Reference}$. Puis l'écart se creuse rapidement : pour $s' = 0.1\%$ notre solution (75 sec) devient 5 fois plus rapide que $EFM_{Reference}$ (6,5 minutes). L'écart continue de se creuser rapidement et rend $EFM_{Reference}$ inapplicable à des cas où le nombre d'EFM produit dépasse le millier, tandis que la durée de notre solution reste relativement faible jusqu'à 50 000 EFM produits. Plus de détails sur les expériences sont présentés dans [48].

Interprétation des résultats expérimentaux

L'écart de performance entre notre solution et $EFM_{Reference}$ est dû principalement à la différence de quantité de données transférées entre les deux approches. Pour $EFM_{Reference}$ le surcoût de transfert est très élevé. Tous les EF sont transférés sur chaque machine avant de tester la condition d'inclusion (*cf.* le début de l'étape 2 présentée ci-dessus).

En comparaison, notre solution ne transfère jamais aucun EF ; ils sont générés localement sur chaque machine et servent à calculer les EFM_i locaux. De plus le nombre d'EFM_i produits localement s'avère être très inférieur au nombre d'EF dans F_i bien que EFM_i contienne des EF qui ne soient pas globalement maximaux. Autrement dit la plupart des EF non maximaux sont filtrés localement et n'induisent aucun surcoût de transfert. De plus, les EFM_i sont représentés de manière compacte par un arbre. La fusion des arbres, 2 à 2, ne provoque en pratique que très peu de transfert.

De plus, les deux approches ont un coût de calcul (*i.e.* coût CPU) différent qui explique en partie l'écart de performance observé. Cela concerne en particulier le test d'inclusion entre deux EF. Dans $EFM_{Reference}$ chaque EF f est comparé avec tous les autres tant qu'un autre EF f' contenant f n'a pas été trouvé. Au contraire dans notre solution, l'insertion d'un EF f dans

l'arbre EFM_i nécessite moins de comparaisons car seuls les chemins contenant un terme de f sont comparés.

5.3 Diversité des domaines et modèle d'évolution (thèse de Ke LI)

Entre 2018 et 2021, j'ai co-encadré la thèse du doctorant Ke LI intitulée *Exploring Topic Evolution in Large Scientific Archives with Pivot Graphs* Nous avons tout d'abord proposé une mesure de la qualité des domaines extraits, basée sur la diversité entre les domaines. Nous avons défini une méthode pour déterminer automatiquement le nombre optimal de domaines par période. Puis nous avons formalisé le concept d'évolution : nous avons défini un modèle de graphe pivot décrivant l'évolution passée et future d'un domaine, appelé aussi *topic*, et des métriques qui caractérisent la structure et le contenu d'une évolution. Nous avons défini un langage d'interrogation, basé sur ces métriques, permettant de décrire et filtrer des modèles d'évolution de topics significatifs. Ce langage, accompagné de méthodes de visualisation du résultat des requêtes, rend possible l'exploration de grands graphes d'évolution de sujets.

Ce travail est publié dans le workshop BigVis@EDBT [59]. Nous avons engagé un effort d'implantation important. Le logiciel résultant passe à l'échelle : il traite efficacement des corpus de très grande taille tels que la totalité du dépôt ArXiv sur 20 ans et les corpus des éditeurs Wiley et Elsevier - obtenus via le service ISTEX du CNRS - couvrant un demi siècle de publications tous domaines confondus. Le prototype est publié à la conférence internationale EDBT 2020 sous la forme d'un démonstrateur [58]. Une version plus complète de ce travail est publiée dans la revue Big Data Research [57].

5.4 Calcul de similarité pour un usage interactif

5.4.1 Résumé des contributions

Le workflow contient une étape d'appariement pour relier les domaines similaires situés dans des périodes de temps différentes. Cela permet à l'utilisateur d'explorer les domaines appariés entre eux sur plusieurs périodes consécutives, dans le but de mettre en évidence des formes d'évolution intéressantes. Pour interpréter plus facilement ces formes d'évolution, l'utilisateur souhaite contrôler le niveau de détail d'un domaine, *i.e.* le nombre de termes

décrivant un domaine, appelé *longueur de description*. Ainsi l'utilisateur invoque plusieurs fois le workflow de manière interactive pour différentes longueurs de description. Or le fait de changer de longueur de description provoque un changement des similarités entre les domaines. L'étape d'appariement est donc re-exécutée pour chaque longueur de description, ce qui induit un sur-coût important.

Dans ce contexte, nous avons proposé une méthode pour optimiser l'étape d'appariement. Nous avons proposé une représentation compacte d'un domaine pour un ensemble de longueurs de descriptions. Puis nous avons défini une méthode parallèle efficace pour factoriser le calcul des similarités entre les domaines ainsi représentés, tout en minimisant les transferts de résultats intermédiaires entre les différents nœuds de calcul. Nous avons optimisé la solution pour le cas où seules les similarités entre périodes temporelles consécutives sont demandées. Nous avons comparé le bénéfice des solutions proposées par rapport à la solution fournie nativement par la plateforme Apache Spark. Les résultats ont été publiés à la conférence IEEE BigData [68].

5.4.2 Problème de l'alignement basé sur la similarité

Pour ce travail, nous considérons un corpus de documents scientifiques qui a été pré-traité pour extraire des domaines par tranches de temps. C'est à dire que nous disposons des domaines scientifiques représentant les travaux de recherche publiés pendant une certaine période, ceci pour plusieurs périodes consécutives. Par exemple la figure 5.3 illustre deux périodes : les domaines de la première période p_1 sont les points bleus, et ceux de la période p_2 suivante sont les points verts. Pour mesurer la proximité entre deux domaines nous nous basons sur leur similarité. Un domaine est caractérisé par un ensemble de termes et chaque terme a un poids. Nous pouvons donc calculer la similarité entre deux domaines en les considérant comme des vecteurs et en appliquant la fonction cosinus. C'est le produit scalaire des deux domaines divisé par le produit de leur norme :

$$sim(d_i, d_j) = \frac{d_i.d_j}{||d_i||.||d_j||} \tag{5.1}$$

Le but est de construire un graphe d'évolution des domaines. Pour cela, il s'agit de relier un domaine d_2 de la période p_2 avec un domaine d_1 de la période p_1 précédente si les deux domaines sont proches. Un lien d'alignement, *cf.* l'arc orange sur la Figure 5.3, est défini si et

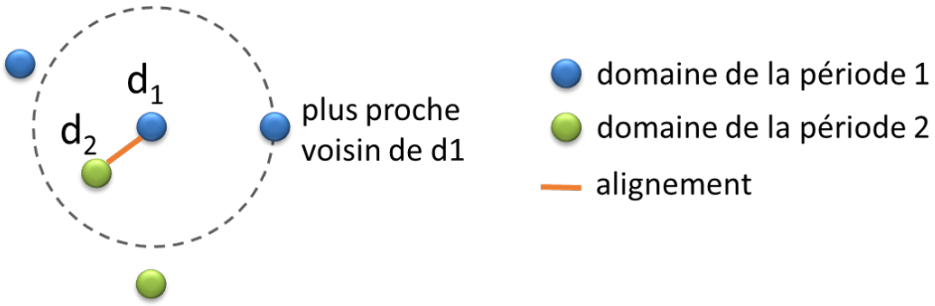

Figure 5.3 : Alignement de domaines basé sur la similarité

seulement si les deux conditions d'alignement suivantes sont satisfaites :

- Premièrement, un domaine n'est aligné qu'avec un domaine « relativement » similaire. Plus formellement, d_1 est relié à d_2 seulement si d_2 est plus proche de d_1 que ne l'est le plus proche voisin de d_1 parmi ceux de la période p_1. Autrement dit, d_2 est contenu dans le cercle en pointillés sur la Figure 5.3 dont le centre est d_1 et le rayon est la similarité entre d_1 et son plus proche voisin.
- Deuxièmement, d_1 est le meilleur candidat parmi les domaines de la période p_1 pour être relié à d_2. Autrement dit, il n'existe pas d'autre candidats « bleus » qui soit plus proche de d_2 que ne l'est d_1 ;

Le nombre de similarités à calculer est très grand et l'objectif de ce travail est de les calculer efficacement en prenant en compte une exigence importante : le nombre K de termes caractérisant un domaine, est choisi par l'utilisateur et **varie** au fil du scénario d'exploration. Nous appelons k la *longueur de description* d'un domaine. Le problème s'énonce alors comme suit : étant donné un ensemble de K valeurs de k, déterminer les K graphes d'alignement des domaines pour toutes les valeurs de k. La question qui en découle est : comment calculer efficacement, pour chaque longueur de description, les similarités entre tous les domaines de toutes les périodes consécutives ?

5.4.3 Méthode de référence pour l'alignement

Dans l'objectif de concevoir une solution efficace, nous considérons Apache Spark une plateforme de calcul sur un cluster de machines. Dans cet environnement, les opérations élémentaires sont traitées efficacement de manière parallèle et distribuée. Afin de disposer d'une solution de référence, nous avons d'abord proposé de nous appuyer directement sur

la fonction existant dans Spark qui calcule la similarité cosinus entre toutes les paires de vecteurs d'un ensemble de vecteurs donné.

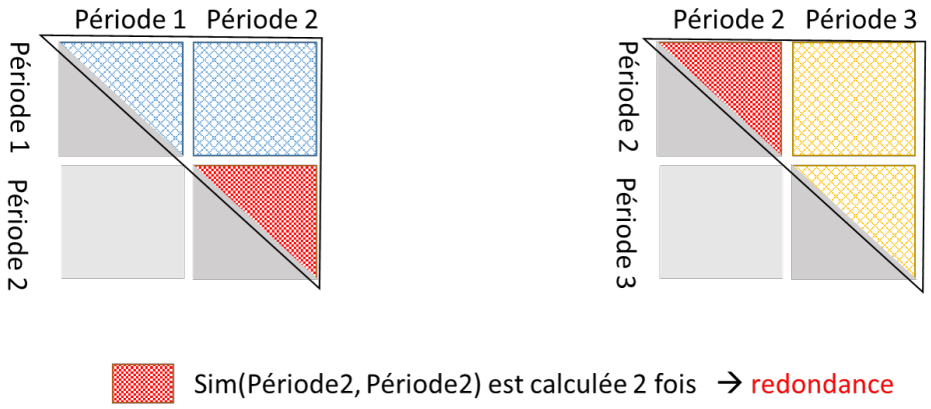

Figure 5.4 : Alignement séquentiel et distribué provoquant des calculs redondants

Pour cela, nous avons découpé les données en plages de deux périodes consécutives. Puis, comme indiqué sur la Figure 5.4, nous invoquons la méthode calculant la similarité pour chaque plage. Le calcul pour une plage est traité de manière parallèle et distribué et le résultat contient la similarité entre toutes les paires de domaines de la plage. Notons que la fonction cosinus étant commutative ($sim(d_1, d_2) = sim(d_2, d_1)$), le calcul se limite aux similarités contenues dans le triangle supérieur de la matrice des paires de domaines, *i.e.* rien n'est calculé pour les parties grisées de la Figure 5.4.

Cette solution a l'avantage d'être simple à mettre en œuvre mais présente un inconvénient important en termes de calculs redondants : les similarités internes à une période sont calculées deux fois, *cf.* les parties rouges sur la Figure 5.4, pour toutes les périodes excepté la première et la dernière période. Ce surcoût peut s'avérer rédhibitoire. Le deuxième inconvénient est que le calcul des similarités se fait indépendamment et séquentiellement pour les K longueurs de description, sans aucune réutilisation possible des calculs déjà effectués. En effet, la fonction calculant la similarité s'applique sur un ensemble de domaines dont la longueur de description est fixée. Cela aboutit à calculer K similarités entre chaque paire de domaines pour laquelle on veut connaître la similarité. La solution présentée ci-dessous vise à surmonter ces deux inconvénients.

5.4.4 Solution : alignement parallèle efficace

L'objectif de la solution est double : éviter de calculer plusieurs fois une similarité entre deux domaines de la même période, et factoriser la calcul des similarités entre deux domaines pour un ensemble de K longueurs de description. Cela nécessite de concevoir un nouvel algorithme pour calculer la similarité cosinus qui tienne compte de K. Pour éviter les calculs redondants, l'algorithme contient deux étapes illustrées sur la Figure 5.5.

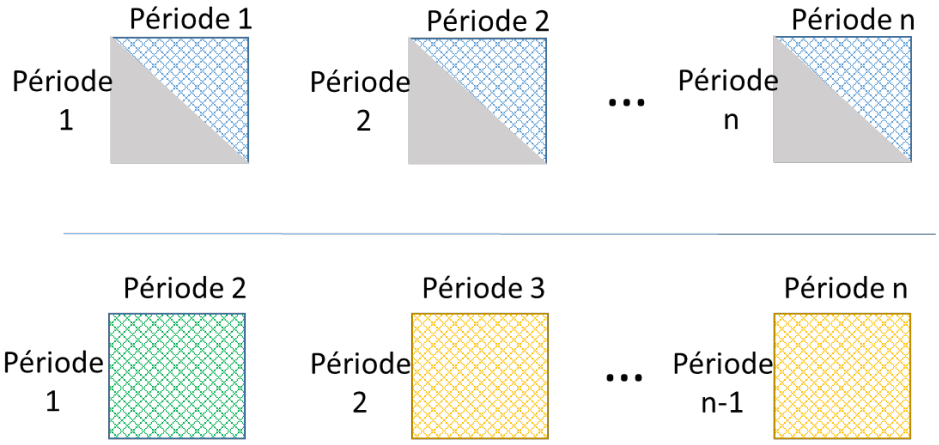

Figure 5.5 : Alignement parallèle sans redondance

La première étape calcule les similarités internes dans chaque période p_i. Les périodes sont distribuées sur les machines de telle sorte que chaque machine reçoit tous les domaines d'au moins une période. Par exemple, la partie supérieure de la Figure 5.5 montre que la machine 1 prend en charge la période p_1 et ainsi de suite pour les autres machines. Les similarités internes à une période peuvent donc être calculées localement avec l'avantage de ne pas générer de transferts de données entre les machines pendant le calcul comme cela se produit avec la méthode 5.4.3. La deuxième étape calcule les similarités et évalue la condition d'alignement entre deux périodes consécutives. Pour cela les domaines de la période p_i, complétés avec l'indice du plus proche voisin de chaque domaine, sont envoyés sur la machine contenant déjà ceux de la période t_{i+1}. Par exemple, les domaines de la période p_1 sont envoyés sur la machine de couleur verte pour traiter les similarités entre p_1 et p_2 et ainsi de suite. Puis l'alignement est évalué localement sans aucun transfert supplémentaire. Au total la quantité de données transférée pour les deux étapes est, au plus, égale au double de la taille des domaines, ceci quel que soit K. En comparaison avec la méthode 5.4.3, la quantité transférée est divisée par K.

Nous détaillons maintenant la méthode permettant de calculer de manière groupée les K similarités entre deux domaines. La similarité se basant sur le produit scalaire et la norme de deux vecteurs, il s'agit tout d'abord de calculer K produits scalaires en calculant **une seule fois** les termes communs entre deux domaines.

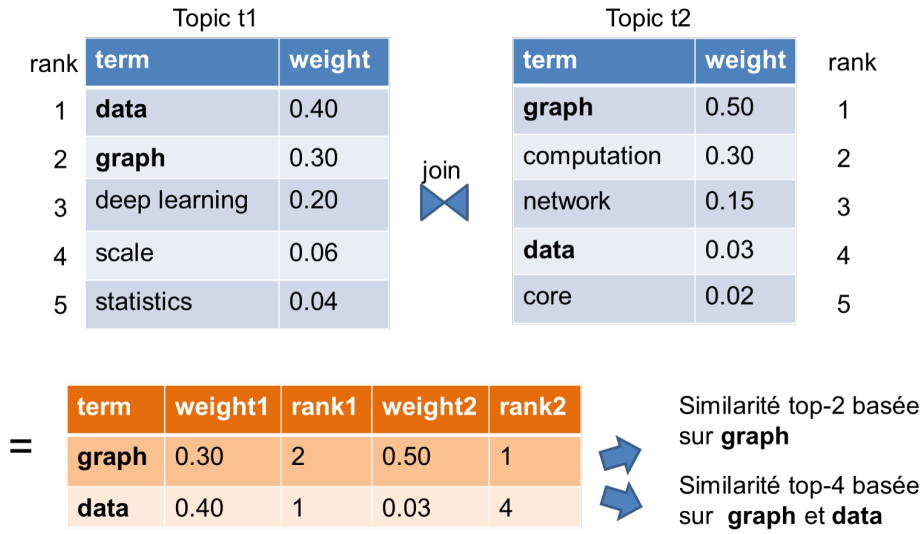

Figure 5.6 : Calcul groupé des K produits scalaires

Pour cela et comme le montre la figure 5.6, les termes d'un domaine sont classés par poids décroissant. Le rang obtenu correspond à la longueur de description k, *cf.* les deux tableaux bleus. Une équi-jointure sur les termes entre deux ensembles de domaines est calculée. Son résultat, illustré par le tableau orange de la figure 5.6, permet de connaître directement quels sont les termes communs entre deux domaines pour une valeur de rang k donnée.

Par exemple, pour $k = 2$, il suffit de sélectionner les lignes telles que $rank_1 \leq 2 \wedge rank_2 \leq 2$ pour déterminer que le terme *graph* est commun. De la même manière, pour $k = 4$, le terme commun à ajouter est *data*.

Finalement, les K normes de chaque domaine sont calculées incrémentalement pour toutes les longueurs de description $(l_1, l_2, ...)$ croissantes successives. Soit $\delta_i^{l_j}$ les termes du domaine $d_i^{l_j}$ qui ne sont pas dans $d_i^{l_{j-1}}$. Le principe est de réutiliser le résultat de $||d_i^{l_{j-1}}||$ pour calculer $||d_i^{l_j}||$ en appliquant : $||d_i^{l_j}||^2 = ||d_i^{l_{j-1}}||^2 + ||\delta_i^{l_j}||^2$.

Pour conclure, les expérimentations, détaillées dans [68], montrent le gain en performance significatif de l'approche (allant jusqu'à un facteur 4).

6 Bilan et Perspectives

6.1 Bilan des travaux récents

Depuis trois ans, nous avons axé nos récents travaux sur des problématiques liées à la science des données, avec l'objectif de capitaliser l'expérience acquise lors des années antérieures dans l'optimisation de requêtes à large échelle. La science des données met en avant des situations où la complexité des requêtes d'analyse de données s'est accrue ainsi que le volume des données manipulées et leur diversité. Les exigences concernant ces trois dimensions - requête, volume et diversité - se déclinent différemment selon le domaine applicatif. Plus précisément, nous avons collaboré avec des spécialistes de deux domaines afin d'identifier les caractéristiques spécifiques des données et des requêtes :

- Dans le Web sémantique, les données forment un graphe de connaissances de très grande taille et tel que le nombre de connexions (*i.e.* le degré) des nœuds est très irrégulier. Les requêtes consistent à trouver des motifs significatifs tels que des triangles. De plus, des informations sémantiques telles que la hiérarchie des types, complètent les données et doivent être prises en compte pour évaluer les requêtes en raisonnant sur le graphe.
- Dans le Web des sciences, les données sont peu structurées et forment des grands corpus de publications scientifiques. Des traitements préliminaires complexes (inférence statistique, fouille de texte) sont nécessaires pour obtenir des données sur lesquelles les utilisateurs peuvent exprimer des requêtes concernant l'évolution des sciences.

Dans ces deux domaines, nous avons abordé des problèmes de gestion de méga-données pour lesquels l'exigence d'efficacité est prépondérante. Les solutions proposées ont été conçues pour passer à l'échelle en s'appuyant sur un environnement de calcul parallèle et distribué. Nos résultats récents sont publiés dans [65, 66] pour les travaux appliqués au Web Sémantique et dans [68, 59, 57] et à travers la démonstration [58] pour ceux appliqués au Web des sciences.

6.2 Encadrement doctoral

Mon encadrement doctoral totalise **6 thèses soutenues plus une en cours d'encadrement**.

Liste des thèses soutenues co-encadrées :

1. Idrissa Sarr (2007-2010) Routing database transactions at large scale. Dirigée par A. Doucet (LIP6)
2. Modou Gueye (2012-2014) Large scale recommender systems. Co-encadrée par T. Abdessalem (Télécom Paristech)
3. Ndiouma Bame (2012-2015) Distributed query processing for biodiversity databases. Dirigée par B. Amann (LIP6)
4. Ibrahima Gueye (2013-2016). Traitement des transactions à large échelle avec ajustement dynamique de ressources à la demande. Co-encadrée par I. Sarr (UCAD) et S. Ndiaye (UCAD)
5. Jean-Benoît Griesner (2015-2018). Systèmes de recommandation de POI à large échelle. Co-encadrée par T. Abdessalem (Télécom Paristech).
6. Ke LI (2018-2021) Exploring Topic Evolution in Large Scientific Archives with Pivot Graphs. Co-encadrée par B. Amann (LIP6)

Tous ont obtenu un emploi directement après leur thèse. I. Sarr, M. Gueye et N. Bame sont enseignant-chercheurs titulaire à l'UCAD. I. Gueye est enseignant-chercheur titulaire à l'École Polytechnique de Thiès. J.-B. Griesner est ingénieur au sein de la branche recherche et développement de Qwant. K. Li est data scientist dans la startup G42 à Abu Dhabi.

La thèse en cours de co-encadrement est :

1. Hamed Rahimi (2021-) Semantization of large-scale scientific corpora. Co-encadrée par B. Amann (LIP6)

Voir le CV détaillé pour les autres encadrements d'étudiants en Master.

6.3 Publications

Parmi l'ensemble de nos publications, nous avons choisi d'en retenir certaines, plus significatives, en précisant quelques éléments d'intérêt :

1. L'article [29] publié dans la revue internationale Information Systems en 2007, intitulée *The leganet system : Freshness-aware transaction routing in a database cluster*. Il démontre la faisabilité du traitement de transactions en contrôlant le niveau de fraîcheur. Ce type de d'approche s'est généralisé par la suite dans de nombreux systèmes dits NewSQL.

2. L'article [10] est issu de l'encadrement de la thèse de Ndiouma Bame, et publié dans la revue internationale (pays francophones) ARIMA en 2015 et intitulé *Algorithmes de traitement de requêtes de biodiversité dans un environnement distribué*. Cet article a permis aux équipe du GBIF France d'améliorer leur service d'accès aux données de biodiversité hébergées au MNHN.

3. L'article [36] est issu de l'encadrement de la thèse de Jean-Benoit Griesner, et publié à la conférence nationale EGC en 2018, intitulé *ALGeoSPF : un modèle de factorisation basé sur du clustering géographique pour la recommandation de POI*. Cet article contribue au passage à l'échelle de solutions de recommandations. Une partie des contributions est actuellement en cours de transfert chez Qwant pour améliorer leur moteur de recherche géographique (Qwant map).

4. L'article [22] publié dans la conférence internationale IEEE BigData en 2015 est intitulé *LiteMat : A scalable, cost-efficient inference encoding scheme for large RDF graphs*. Il montre une collaboration réussie à l'intersection entre les domaines du web sémantique et des bases de données. Il a impulsé la production d'un logiciel open source pour requêter efficacement des très grandes bases de connaissances en s'appuyant sur la plateforme de calcul parallèle Apache Spark. Par la suite, l'article [65] publié dans la revue Open Journal of Web Semantic en 2020 a montré l'intérêt de prendre en compte les spécificités des données liées à un certain contexte (ici, le web sémantique) pour optimiser les requêtes.

5. L'article [57] publié dans la revue Big Data Research en 2021 synthétise les principaux résultats de la thèse de Ke LI. L'interdisciplinarité a été sans cesse prise en compte tout au long de la thèse avec des retombées importantes dans deux disciplines : (i) des résultats théoriques en science des données sur des modèles de données et leur manipulation, et

(ii) la production d'un logiciel pouvant servir directement aux chercheurs en histoire des sciences pour valider leurs hypothèses. Le co-encadrement a été très complémentaire et chacun a su trouver son rôle et le mettre au profit de la thèse.

Voir le CV détaillé pour le récapitulatif complet de nos publications classées par type (revue, conférence, chapitre de livre, ...)

6.4 Perspectives

Durant les prochaines années, nous prévoyons de renforcer nos travaux principalement autour de l'axe de la science des données. Nous avions commencé à aborder cet axe durant le projet interdisciplinaire ANR EPIQUE (2018 - 2021) qui se termine actuellement. Notre partenariat avec l'Institut des Systèmes Complexes - Paris Ile-de-France (ISC-PIF) a mis en évidence de nouveaux défis scientifiques parmi lesquels nous retenons en priorité :

- le passage à l'échelle des algorithmes de construction de phylomémies définis par l'ISC-PIF.
- la sémantisation de corpus scientifiques à large échelle

Notre expérience de recherche en gestion et analyse de données à large échelle (Big Data, web), en interrogation des données et en optimisation de requêtes sémantiques, sera mise à profit pour relever ces défis situés à l'intersection entre la science des données et le Web sémantique. Par ailleurs, notre collaboration avec le Sénégal se poursuit sous le forme du projet Senagro détaillé ci-après.

6.4.1 Passage à l'échelle des algorithmes de construction de phylomémies

Cette perspective est prévue à court terme. Il s'agit d'une suite directe à partir des résultats produits dans le projet EPIQUE. Le workflow d'analyse de l'évolution des sciences a été défini en commun par les partenaires du projet. Toutefois, il se décline en deux variantes complémentaires proposées chacune par un partenaire distinct. Notamment chaque workflow a sa propre représentation des topics :

- Dans le workflow réalisé au LIP6, les topics sont des vecteurs de termes obtenus directement, à partir d'un ensemble de documents, par une méthode d'inférence statistique

appelée LDA.

- Dans le workflow réalisé à l'ISC-PIF, les topics sont des clusters (sous graphes denses) du graphe de co-occurrence de termes, lui-même obtenu à partir d'un ensemble de documents. Puis les clusters sont reliés entre eux pour former des *phylomémies* [18].

Ces deux représentations des topics intéressent l'utilisateur final car elles permettent d'explorer différentes facettes de l'évolution des sciences. Néanmoins, un problème de performance a été identifié dans le workflow réalisé à l'ISC-PIF : le calcul des clusters et des phylomémies repose sur des algorithmes dont la complexité est grande et qui ne passent pas à l'échelle : cela empêche d'appliquer ce workflow sur des corpus multidisciplinaires, *i.e.* toutes sciences exactes et appliquées, de grande taille tels que le Web of Science [1]. Nous prévoyons de proposer une solution générique et transparente pour les utilisateurs, capable de distribuer la génération des clusters et des phylomémies sur une plateforme de calcul parallèle.

6.4.2 Sémantisation de corpus scientifiques à large échelle

Ce projet de recherche doctoral a été sélectionné pour bénéficier d'un financement, pour la période 2021-2024, alloué par l'institut SCAI (Sorbonne Center for Artificial Intelligence) auquel le laboratoire LIP6 est associé. Il se concrétisera par le co-encadrement, avec Bernd Amann, de la thèse de Hamed Rahimi qui débute en septembre 2021.

Les graphes de connaissances ont atteint un niveau de fiabilité très élevé et constituent une richesse immense pour mieux comprendre les données textuelles produites en masse chaque jour. Ils décrivent des connaissances en s'appuyant sur le standard RDF, et peuvent être interrogés à l'aide du langage de requêtes SPARQL. Un des plus grands graphes de connaissances est Wikidata [2] qui apporte des informations encyclopédiques de culture générale sur de nombreux concepts et entités. Toutefois, structurer des connaissances en RDF pour les intégrer dans un graphe de connaissances demande un effort important et on constate que de nombreuses connaissances sont actuellement produites sous la forme de documents textuels et ne se fondent pas sur le formalisme des graphes de connaissances. C'est le cas de la plupart des connaissances produites par les acteurs de la recherche scientifique au fil des années. Nous disposons ainsi de très grands corpus contenant des articles scientifiques publiés

[1] `https://www.webofscience.com`
[2] `https://www.wikidata.org`

sur plusieurs décennies. Afin de faciliter l'exploration et l'analyse globale des connaissances représentées dans ces corpus, les approches existantes consistent à décrire (indexer) les documents par des ensembles de termes qui ne captent pas toutes les nuances conceptuelles nécessaires pour construire des cartes décrivant les interactions et évolutions scientifiques. De ce fait, l'analyse plus sémantique des connaissances dans des corpus scientifiques se heurte à un manque de représentation sémantique des concepts scientifiques.

L'objectif général de ce projet est de proposer des nouveaux outils pour enrichir les grands corpus scientifiques avec des graphes de connaissances pour permettre une analyse plus fine des domaines scientifiques appellés *topics*, et de leur évolution. L'approche proposée consiste à combiner des méthodes de fouille de texte avec les technologies (RDF, SPARQL) et les ressources du web sémantique (Wikidata, DBPedia, Yago). Cela pose les deux défis suivants.

Enrichissement sémantique des topics. La représentation des topics définis par inférence statistique, (*cf*. le workflow EPIQUE) s'avère limitée. Il s'agit d'enrichir les topics en intégrant des données du web sémantique. En particulier, les topics peuvent être représentés de manière plus riche qu'un ensemble de termes pondérés, en ajoutant le type (ou le concept) associé au terme. Par exemple, certains termes définissent le sujet du topic, d'autres termes définissent la méthode utilisée pour traiter le sujet, les acteurs impliqués dans l'étude scientifique et leur rôle. Ces annotations de type peuvent ensuite être prises en compte pour calculer la similarité entre les topics et caractériser plus précisément la nature de l'évolution entre des domaines.

Recherche interactive de motifs d'évolution dans un grand graphe pondéré. Ce défi explore le traitement de requêtes complexes dans les graphes obtenus lors de la résolution du premier défi. Une requête cherche un sous-graphe partant d'un certain topic et contenant tous les topics connexes plus récents et tels que tous les arcs du sous-graphe en question ont une similarité supérieure à un seuil. Or la valeur du seuil déterminant le sous-graphe, n'est pas précisée dans la requête. La requête définit seulement une condition agrégative sur le sous-graphe, par exemple le degré d'évolution moyen du sous-graphe doit être supérieur à une borne. Le problème est donc de déterminer pour chaque topic candidat, la valeur du seuil de similarité telle que la condition agrégative soit satisfaite.

Cette requête n'est pas exprimable avec les langages de requêtes existants tels que SPARQL. Une méthode simple pour l'évaluer consiste à partir d'un topic quelconque, de parcourir progressivement les topics connexes jusqu'à obtenir un sous-graphe qui satisfait la condition

agrégative. Toutefois cette méthode ne passe pas à l'échelle : son coût devient prohibitif pour des grands graphes. D'autre part, il n'est pas envisageable de pré-calculer à l'avance tous les motifs puis de les indexer, car le délai induit empêcherait l'utilisateur d'interroger immédiatement les données qu'il fournit.

Approche scientifique envisagée. Dans un premier temps une approche algébrique sera privilégiée. On étudiera la définition et l'implantation de nouveaux opérateurs capturant la logique d'une jointure sémantique entre un graphe de connaissances et un ensemble de domaines décrits par des termes pondérés. L'accent sera également mis sur les requêtes qui effectuent des parcours transitifs dans les graphes. Des solutions seront étudiées pour optimiser l'analyse interactive, par exemple en anticipant les parcours demandés par les utilisateurs. Une représentation succincte du graphe sera étudiée afin d'améliorer les performances des requêtes. Une approche expérimentale sera préconisée. Les algorithmes proposés seront implantés sur la plateforme Apache Spark en veillant à ce que leur exécution soit effectivement parallèle et distribuée et en améliorant, le cas échéant, le passage à l'échelle des traitements sous-jacents. Concernant la validation qualitative des résultats, cette thèse bénéficiera de l'expertise de philosophes des sciences de Institut d'histoire et de philosophie des sciences et des techniques. Finalement, nous étudierons dans quelle mesure les algorithmes d'évaluation de requêtes proposés sont suffisamment généraux pour dépasser le cas de l'analyse de l'évolution des sciences.

6.4.3 Projet CNRS : Senagro Data Scikit

Pour la période 2021-2022, je suis porteur du projet SenAgro qui a été sélectionné en juin 2021 pour être financé par le CNRS dans le cadre de l'appel « Dispositif de Soutien aux Collaborations avec l'Afrique subsaharienne ». Le responsable du projet, côté Sud, est Idrissa Sarr enseignant-chercheur à l'Université Cheikh Anta Diop (UCAD) de Dakar, avec qui nous entretenons des collaborations scientifiques (co-encadrement de thèses et de stages, missions) depuis une quinzaine d'années. Le consortium inclut également l'Ecole Polytechnique de Thiès qui apporte une expertise complémentaire sur les flux de données. Ce projet aborde le défi sociétal d'améliorer la productivité de l'agriculture sénégalaise dont les pratiques manquent encore de modernité : l'absence de solution pour lutter contre les pathologies tend à réduire fortement les récoltes, ce qui compromet l'auto-suffisance alimentaire du Sénégal.

L'objectif de ce projet est de renforcer les compétences en data science des acteurs académiques impliqués dans la gestion des données de l'agriculture intelligente. Des missions bilatérales permettront la production d'un guide méthodologique et de divers contenus scientifiques qui seront exploités et vulgarisés à travers une école d'été. Les principales composantes du projet sont :

- l'élaboration d'un référentiel de compétences pour la gestion, l'exploitation et la valorisation des données ;
- l'édition d'un guide méthodologique sous forme de bonnes pratiques et recommandations pour orienter la mise en place d'une infrastructure de gestion, d'exploitation et de valorisation de données agricoles ;
- la conception et le développement de contenus scientifiques et pédagogiques à diffuser aux moyens de pages wiki, plateforme e-learning, publications scientifiques et une école d'été.

Le projet ciblera en priorité les thématiques suivantes car elles couvrent des notions essentielles à la conception d'une solution répondant aux besoins spécifiques d'une problématique de l'agriculture intelligente :

- La collecte de données via un réseau de capteurs,
- Les systèmes de stockage à large échelle,
- La manipulation des données et l'analyse des flux de données collectées,
- La visualisation des données,
- L'enrichissement sémantique des données.

Le projet Senagro Data Scikit constitue un premier jalon pour la mise en place d'un observatoire sur les données à usage agricole. De plus, le projet servira de levier pour développer de nouveaux axes de recherche à l'intersection entre la science des données et la biologie végétale en mobilisant des chercheurs des deux disciplines et de trois universités au Nord et au Sud.

6.4.4 Perspectives à plus long terme

A plus long terme, nos travaux ont l'ambition d'apporter des éléments de réponse concrets au défi plus général qui a émergé récemment et qui concerne les enjeux sociétaux de l'intelligence artificielle (IA) :

1) Mieux contrôler et comprendre les résultats des tâches d'apprentissage automatique et leur impact dans diverses chaînes de traitement complexes.

2) L'axe *Artifical Intelligence for Database systems* (AI4DB) : les méthodes prédictives issues de l'apprentissage automatique en IA tendent à être appliquées pour résoudre des problèmes de gestion de données *cf*. le tutoriel AI4DB à SIGMOD 2021 [55], la présentation Database Systems 2.0 de J. Gehrke aux workshops PhD en 2019 [30] et AIDM en 2021, et celle de D. Kossmann en 2021 [52]. Il s'agit de remplacer un modèle descriptif basé sur les paramètres caractérisant le système de gestion de données, par un modèle prédictif dont les paramètres latents sont appris sur un large échantillon de données et de réponses permettant d'entraîner le modèle. Ce changement de paradigme est l'objet d'une attention de plus en plus soutenue dans la communauté bases de données. Des résultats démontrent le bénéfice de ce nouvel angle d'approche pour résoudre plus efficacement des problèmes de gestion de données, notamment pour indexer des données et optimiser des requêtes. En particulier, [28, 83] propose un modèle d'apprentissage par renforcement pour déterminer l'ordre des jointures d'une requête. [28] propose d'apprendre la distribution du domaine des valeurs indexées pour construire un index plus compact donc plus performant. [8] propose un modèle basée sur un réseau de neurones pour apprendre le coût des opérateurs relationnels pour les requêtes réparties.

Nos travaux en science des données nous ont permis de nous familiariser avec les modèles prédictifs (*e.g.*, le modèle LDA utilisé dans le workflow Epique). Nous pensons que cette approche, consistant à prédire au lieu de décrire, a un fort potentiel pour apporter des éléments de solutions aux problèmes de gestion efficaces de méga-données sur lesquels nous avons acquis une grande expertise. Parmi les nombreuses pistes pouvant être explorées, nous focaliserons nos travaux sur le choix de modèles prédictifs pour améliorer la représentation et la manipulation des grands graphes. Nous explorerons les compromis entre les modèles prédictifs et descriptifs afin de proposer un cadre pour unifier ces deux modèles sans les opposer.

Une première étape pour amorcer ces travaux est le démarrage d'une collaboration avec la

startup Zeenea [3], sous la forme de co-encadrements de stages prévus en 2022. Le contexte est un ensemble de sources de données hétérogènes partagées à travers un *data lake*. Les sources ne possèdent pas ou peu d'information décrivant la sémantique des données. Il s'agit de retrouver une donnée pouvant être pertinente vis-à-vis d'un certain objectif métier en question. Or une récente méthode d'intégration de données relationnelles, reposant sur un modèle d'apprentissage profond, permet de définir et associer des concepts communs entre plusieurs fragments de données [15]. Un des stages explorera la possibilité d'adapter ce travail au contexte de Zeenea.

[3] `https://zeenea.com`

Bibliographie

[1] Daniel J. Abadi. *Spanner vs. Calvin : Distributed Consistency at Scale*. 2017. URL : https://fauna.com/blog/distributed-consistency-at-scale-spanner-vs-calvin.

[2] Daniel J. Abadi et Jose M. Faleiro. « An overview of deterministic database systems ». In : *Commun. ACM* 61.9 (2018), p. 78-88. DOI : 10.1145/3181853.

[3] Daniel Abadi et al. « The Beckman report on database research ». In : *Commun. ACM* 59.2 (2016), p. 92-99. DOI : 10.1145/2845915.

[4] Daniel Abadi et al. « The Seattle Report on Database Research ». In : *SIGMOD Rec.* 48.4 (2019), p. 44-53. DOI : 10.1145/3385658.3385668.

[5] Deepthi Devaki Akkoorath et Annette Bieniusa. *Antidote : the highly-available geo-replicated database with strongest guarantees*. 2016. URL : https://pages.lip6.fr/syncfree/attachments/article/59/antidote-white-paper.pdf.

[6] Deepthi Devaki Akkoorath et al. « Cure : Strong Semantics Meets High Availability and Low Latency ». In : *IEEE International Conference on Distributed Computing Systems (ICDCS)*. 2016, p. 405-414. DOI : 10.1109/ICDCS.2016.98.

[7] Sabeur Aridhi, Laurent d'Orazio, Mondher Maddouri et Engelbert Mephu Nguifo. « Cost Models for Distributed Pattern Mining in the Cloud ». In : *2015 IEEE TrustCom/BigDataSE/ISPA, Helsinki, Finland, August 20-22, 2015, Volume 2*. IEEE, 2015, p. 112-119. DOI : 10.1109/Trustcom.2015.569.

[8] Kassem Awada et al. « Cost Estimation Across Heterogeneous SQL-Based Big Data Infrastructures in Teradata IntelliSphere ». In : *Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020*. OpenProceedings.org, 2020, p. 534-545. DOI : 10.5441/002/edbt.2020.64.

[9] David F. Bacon et al. « Spanner : Becoming a SQL System ». In : *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. ACM, 2017, p. 331-343. DOI : 10.1145/3035918.3056103.

[10] Ndiouma Bame, Hubert Naacke, Idrissa Sarr et Samba Ndiaye. « Algorithmes de traitement de requêtes de biodiversité dans un environnement distribué ». In : *Revue Africaine de la Recherche en Informatique et Mathématiques Appliquées* 18 (2014), p. 1-18.

[11] Ndiouma Bame, Hubert Naacke, Idrissa Sarr et Samba Ndiaye. « Architecture répartie à large échelle pour le traitement parallèle de requêtes de biodiversité ». In : *11th African Conference on Research in Computer Science and Applied Mathematics (CARI)*. Algiers, Algeria, oct. 2012, p. 143-150.

[12] Ndiouma Bame, Hubert Naacke, Idrissa Sarr et Samba Ndiaye. « Optimisation de requêtes dynamiques pour l'analyse de la biodiversité ». In : *Revue Africaine de la Recherche en Informatique et Mathématiques Appliquées* 21 (2015), p. 21-47.

[13] Ndiouma Bame, Hubert Naacke, Idrissa Sarr et Samba Ndiaye. « Traitement décentralisé de requêtes de biodiversité ». In : *Colloque National sur la Recherche en Informatique et ses Applications, CNRIA 2013*. Ziguinchor, Senegal, mar. 2013, p. 8.

[14] C. Borgelt. « An implementation of the FP-growth algorithm ». In : *Workshop on Open Source Data Mining : Frequent Pattern Mining Implementations (OSDM)*. 2005.

[15] Riccardo Cappuzzo, Paolo Papotti et Saravanan Thirumuruganathan. « Creating Embeddings of Heterogeneous Relational Datasets for Data Integration Tasks ». In : *International Conference on Management of Data, SIGMOD*. ACM, 2020, p. 1335-1349. doi : 10.1145/3318464.3389742.

[16] Samuel Castillo, Hubert Naacke, Bernd Amann et David Chavalarias. *Exploring the evolution of science through interactive phylomemetic topic maps*. 32ème Conférence sur la Gestion de Données - BDA2016. Prototype Demonstration. Nov. 2016.

[17] Allison J.B. Chaney, David M. Blei et Tina Eliassi-Rad. « A Probabilistic Model for Using Social Networks in Personalized Item Recommendation ». In : *12th ACM Conference on Recommender Systems (RecSys)*. 2015, p. 43-50. isbn : 978-1-4503-3692-5. doi : 10.1145/2792838.2800193.

[18] David Chavalarias, Quentin Lobbe et Alexandre Delanoë. « Draw me Science - multi-level and multi-scale reconstruction of knowledge dynamics with phylomemies ». working paper (hal-03180347). Mar. 2021.

[19] Christine Collet et al. « De la gestion de bases de données à la gestion de grands espaces de données ». working paper or preprint. 2012.

[20] Olivier Curé, Hubert Naacke, Mohamed-Amine Baazizi et Bernd Amann. *HAQWA : a Hash-based and Query Workload Aware Distributed RDF Store*. The 14th International Semantic Web Conference, ISWC 2015. Poster. Oct. 2015.

[21] Olivier Curé, Hubert Naacke, Mohamed-Amine Baazizi et Bernd Amann. « On the Evaluation of RDF Distribution Algorithms Implemented over Apache Spark ». In : *The 11th International Workshop on Scalable Semantic Web Knowledge Base Systems*. Bethlehem, Pennsylvania, United States, oct. 2015, p. 16-31.

[22] Olivier Curé, Hubert Naacke, Tendry Randriamalala et Bernd Amann. « LiteMat : A scalable, cost-efficient inference encoding scheme for large RDF graphs ». In : *2015 IEEE International Conference on Big Data (Big Data)*. Santa Clara, CA, United States : IEEE, oct. 2015, p. 1823-1830. doi : 10.1109/BigData.2015.7363955.

[23] Olivier Curé, Weiqin Xu, Hubert Naacke et Philippe Calvez. « LiteMat, an Encoding Scheme with RDFS++ and Multiple Inheritance Support ». In : *The Semantic Web : ESWC 2019 Satellite Events - ESWC 2019 Satellite Events, Portorož, Slovenia, June 2-6, 2019, Revised Selected Papers*. Springer, 2019, p. 269-284.

[24] Maximillien Danish, Mohamed-Amine Baazizi et Hubert Naacke. *Analyse topologique et requêtes interactives dans des grands graphes sémantiques (project proposal)*. 2017. url : http://www-bd.lip6.fr/wiki/_media/site/recherche/projets/projetlip6-2018-short.pdf.

[25] Sudipto Das, Divyakant Agrawal et Amr El Abbadi. « ElasTraS : An elastic, scalable, and self-managing transactional database for the cloud ». In : *ACM Trans. Database Syst.* 38.1 (2013), 5 :1-5 :45. doi : 10.1145/2445583.2445588.

[26] Wenfei Fan, Chunming Hu, Xueli Liu et Ping Lu. « Discovering Graph Functional Dependencies ». In : *Intl Conf on Management of Data, ACM SIGMOD*. 2018, p. 427-439. doi : 10.1145/3183713.3196916.

[27] Juliana Freire, Philippe Bonnet et Dennis Shasha. « Computational Reproducibility : State-of-the-Art, Challenges, and Database Research Opportunities ». In : *International Conference on Management of Data (SIGMOD)*. New York, NY, USA : ACM, 2012, p. 593-596. isbn : 9781450312479. doi : 10.1145/2213836.2213908.

[28] Alex Galakatos et al. « FITing-Tree : A Data-aware Index Structure ». In : *International Conference on Management of Data (SIGMOD)*. ACM, 2019, p. 1189-1206. doi : 10.1145/3299869.3319860.

[29] Stéphane Gançarski, Hubert Naacke, Esther Pacitti et Patrick Valduriez. « The leganet system : Freshness-aware transaction routing in a database cluster ». In : *Information Systems* 32.2 (2007), p. 320-343.

[30] Johannes Gehrke. « Database Systems 2.0 ». In : *VLDB PhD Workshop, co-located with the 45th International Conference on Very Large Databases (VLDB)*. T. 2399. CEUR Workshop Proceedings. 2019.

[31] Seth Gilbert et Nancy A. Lynch. « Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services ». In : *SIGACT News* 33.2 (2002), p. 51-59. doi : 10.1145/564585.564601.

[32] Prem Gopalan, Jake M. Hofman et David M. Blei. « Scalable Recommendation with Hierarchical Poisson Factorization ». In : *Thirty-First Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2015, p. 326-335.

[33] Jean-Benoît Griesner, Talel Abdessalem et Hubert Naacke. « Intégration des Influences Géographique et Temporelle pour la Recommandation de Points d'Intérêt ». In : *Extraction et Gestion des Connaissances*. T. RNTI-E-30. RNTI. Reims, France, jan. 2016, p. 153-158.

[34] Jean-Benoît Griesner, Talel Abdessalem et Hubert Naacke. « POI Recommendation : Towards Fused Matrix Factorization with Geographical and Temporal Influences ». In : *9th ACM Conference on Recommender Systems, RecSys*. Vienne, Austria : ACM, sept. 2015, p. 301-304. doi : 10.1145/2792838.2799679.

[35] Jean-Benoît Griesner, Talel Abdessalem et Hubert Naacke. « Un Modèle de Factorisation de Poisson pour la Recommandation de Points d'Intérêt ». In : *17ème Journées Francophones Extraction et Gestion des Connaissances (EGC 2017)*. Grenoble, France, jan. 2017, p. 411-416.

[36] Jean-Benoît Griesner, Talel Abdessalem, Hubert Naacke et Pierre Dosne. « AL-GeoSPF : Un modèle de factorisation basé sur du clustering géographique pour la recommandation de POI ». In : *Extraction et Gestion de Connaissances (EGC'2018)*. Saint Denis, France, jan. 2018, p. 12.

[37] Ibrahima Gueye, Hubert Naacke et Stéphane Gançarski. « Enriching Geolocalized Dataset with POIs Descriptions at Large Scale ». In : *Innovations and Interdisciplinary Solutions for Underserved Areas - 4th EAI International Conference, InterSol 2020, Nairobi, Kenya, March 8-9, 2020*. T. 321. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer, 2020, p. 264-273.

[38] Ibrahima Gueye, Hubert Naacke et Idrissa Sarr. « Supporting Fluctuating Transactional Workload ». In : *26th International Conference on Database and Expert Systems Applications (DEXA)*. T. 9262. Lecture Notes in Computer Science. Valencia, Spain : Springer, sept. 2015, p. 295-303. doi : 10.1007/978-3-319-22852-5.

[39] Ibrahima Gueye, Idrissa Sarr et Hubert Naacke. « Exploiting the social structure of online media to face transient heavy workload ». In : *The Sixth International Conference on Advances in Databases, Knowledge, and Data Applications, DBKDA 2014*. Chamonix, France, avr. 2014, p. 51-58.

[40] Ibrahima Gueye, Idrissa Sarr et Hubert Naacke. « Gestion d'un workload transitoire via les graphes sociaux ». In : *12th African Conference on Research in Computer Science and Applied Mathematics (CARI)*. Saint-Louis, Senegal, oct. 2014, p. 201-212.

[41] Modou Gueye, Talel Abdessalem et Hubert Naacke. « A parameter-free algorithm for an optimized tag recommendation list size ». In : *RecSys '14*. RecSys '14 Proceedings of the 8th ACM Conference on Recommender systems. ACM. Foster City, United States : ACM, oct. 2014, p. 233-240. doi : 10.1145/2645710.2645727.

[42] Modou Gueye, Talel Abdessalem et Hubert Naacke. « A Social and Popularity-based Tag Recommender ». In : *SocialCom 2014 - The Seventh IEEE International Conference on Social Computing and Networking*. Sidney, Australia : IEEE, déc. 2014, p. 318-325. doi : 10.1109/BDCloud.2014.44.

[43] Modou Gueye, Talel Abdessalem et Hubert Naacke. « Dynamic Recommender System : Using Cluster-Based Biases to Improve the Accuracy of the Predictions ». In : *Advances in Knowledge Discovery and Management Volume 5*. T. 615. Studies in Computational Intelligence. Springer, 2015, p. 79-104. doi : 10.1007/978-3-319-23751-0.

[44] Modou Gueye, Talel Abdessalem et Hubert Naacke. « Factorisation multi-biais pour de meilleures recommandations ». In : *5ème Colloque National sur la recherche en informatique et ses applications (CNRIA)*. 8 pages. Ziguinchor, Senegal, avr. 2013.

[45] Modou Gueye, Talel Abdessalem et Hubert Naacke. « FoldCons : A Simple Way To Improve Tag Recommendation ». In : *5th ACM RecSys Workshop on Recommender Systems & the Social Web*. T. 1066. CEUR Workshop Proceedings. 4 pages. Hong Kong, Hong Kong SAR China : CEUR, oct. 2013, Session : Tags.

[46] Modou Gueye, Talel Abdessalem et Hubert Naacke. « STRec : An Improved Graph-based Tag Recommender ». In : *5th ACM RecSys Workshop on Recommender Systems & the*

Social Web. T. 1066. CEUR Workshop Proceedings. 8 pages. Hong Kong, Hong Kong SAR China : CEUR, oct. 2013, Session : Tags.

[47] Modou Gueye, Talel Abdessalem et Hubert Naacke. « Technique de factorisation multi-biais pour des recommandations dynamiques ». In : *EGC 2013 - 13e Conférence Francophone sur l'Extraction et la Gestion des Connaissances*. T. RNTI-E-24. Revue des Nouvelles Technologies de l'Information. 12 pages. Toulouse, France, jan. 2013, p. 365-376.

[48] Firas Hatem. « Large scale analysis of science evolution : topic extraction from scientific production ». rapport de stage de Master. 2017.

[49] Xiangnan He, Hanwang Zhang, Min-Yen Kan et Tat-Seng Chua. « Fast Matrix Factorization for Online Recommendation with Implicit Feedback ». In : *ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 2016, p. 549-558. doi : 10.1145/2911451.2911489.

[50] Ricardo Jiménez-Peris et al. « CumuloNimbo : A Cloud Scalable Multi-tier SQL Database ». In : *IEEE Data Eng. Bull.* 38.1 (2015), p. 73-83.

[51] Anurag Khandelwal et al. « ZipG : A Memory-efficient Graph Store for Interactive Queries ». In : *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. 2017, p. 1149-1164. doi : 10.1145/3035918.3064012.

[52] Donald Kossmann. « Democratizing AI : For Everybody and Everything, (Keynote at DIIP Meeting) ». keybnote. Fév. 2021.

[53] Jens Lehmann et al. « DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia ». In : *Semantic Web* 6.2 (2015), p. 167-195. doi : 10.3233/SW-140134.

[54] Jure Leskovec, Anand Rajaraman et Jeffrey D. Ullman. *Mining of Massive Datasets, 2nd Ed*. Cambridge University Press, 2014. isbn : 978-1107077232.

[55] Guoliang Li, Xuanhe Zhou et Lei Cao. « AI Meets Database : AI4DB and DB4AI ». In : *SIGMOD '21 : International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*. ACM, 2021, p. 2859-2866. doi : 10.1145/3448016.3457542.

[56] Haoyuan Li et al. « Pfp : parallel fp-growth for query recommendation ». In : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*. ACM, 2008, p. 107-114. doi : 10.1145/1454008.1454027.

[57] Ke Li, Hubert Naacke et Bernd Amann. « An Analytic Graph Data Model and Query Language for Exploring the Evolution of Science ». In : *Big Data Research* 26 (2021), p. 18. issn : 2214-5796. doi : https://doi.org/10.1016/j.bdr.2021.100247.

[58] Ke Li, Hubert Naacke et Bernd Amann. « EPIQUE : Extracting Meaningful Science Evolution Patterns from Large Document Archives ». In : *International Conference on Extending Database Technology (EDBT)*. Copenhagen, Denmark, mar. 2020.

[59] Ke Li, Hubert Naacke et Bernd Amann. « Exploring the Evolution of Science with Pivot Topic Graphs ». In : *International Workshop on Big Data Visual Exploration and Analytics BigVis*. EDBT 2020. Copenhague, Denmark, mar. 2020.

[60] Ngo Van Linh, Anh Nguyen Duc, Thai Binh Nguyen et Khoat Than. « Neural Poisson Factorization ». In : *IEEE Access* 8 (2020), p. 106395-106407. doi : 10.1109/ACCESS.2020.2994239.

[61] Farzaneh Mahdisoltani, Joanna Biega et Fabian M. Suchanek. « YAGO3 : A Knowledge Base from Multilingual Wikipedias ». In : *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. 2015.

[62] Silviu Maniu et Bogdan Cautis. « Taagle : efficient, personalized search in collaborative tagging networks ». In : *ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, 2012, p. 661-664. doi : 10.1145/2213836.2213926.

[63] Laure Millet et al. « Facing peak loads in a P2P transaction system ». In : *The First Workshop on P2P and Dependability (P2PDEP'12)*. P2P-Dep '12. Sibiu, Romania : ACM, mai 2012, p. 1-7. doi : 10.1145/2212346.2212347.

[64] Hubert Naacke, Bernd Amann et Olivier Curé. « SPARQL Graph Pattern Processing with Apache Spark ». In : *GRADES (Graph Data-management Experiences & Systems), Workshop, SIGMOD 2017*. Chicago, United States, mai 2017, p. 1-7.

[65] Hubert Naacke et Olivier Curé. « On Distributed SPARQL Query Processing Using Triangles of RDF Triples ». In : *Open J. Semantic Web* 7.1 (2020), p. 17-32.

[66] Hubert Naacke et Olivier Curé. « Triag, a framework based on triangles of RDF triples ». In : *Proceedings of The International Workshop on Semantic Big Data, SBD@SIGMOD 2020, Portland, Oregon, USA, June 19, 2020*. ACM, 2020, p. 31-36. doi : 10.1145/3391274.3393634.

[67] Hubert Naacke, Jean-Benoît Griesner, Talel Abdessalem et Pierre Dosne. « AL-GeoSPF : A Hierarchical Geographical Factorization Model for POI Recommendation ». In : *Journées Bases de Données Avancées (BDA)*. Nancy, France, nov. 2017, p. 11.

[68] Hubert Naacke, Ke Li, Bernd Amann et Olivier Curé. « Efficient similarity-based alignment of temporally-situated graph nodes with Apache Spark ». In : *High Performance Big Graph Data Management, Analysis, and Mining*. IEEE Int'l Conference on Big Data. Los Angeles, United States, déc. 2019.

[69] Yavor Nenov et al. « RDFox : A Highly-Scalable RDF Store ». In : *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II*. T. 9367. Lecture Notes in Computer Science. Springer, 2015, p. 3-20. doi : 10.1007/978-3-319-25010-6_1.

[70] Thomas Neumann et Gerhard Weikum. « The RDF-3X engine for scalable management of RDF data ». In : *VLDB J.* 19.1 (2010), p. 91-113. doi : 10.1007/s00778-009-0165-y.

[71] Xiangnan Ren, Olivier Curé, Hubert Naacke et Guohui Xiao. « BigSR : real-time expressive RDF stream reasoning on modern Big Data platforms ». In : *IEEE International Conference on Big Data*. Seatle, WA, United States, 2018, p. 811-820.

[72] Xiangnan Ren et al. « Strider R : Massive and Distributed RDF Graph Stream Reasoning ». In : *IEEE International Conference on Big Data*. Boston, United States, déc. 2017, p. 10.

[73] Steffen Rendle et Lars Schmidt-Thieme. « Online-updating regularized kernel matrix factorization models for large-scale recommender systems ». In : *ACM Conference on Recommender Systems (RecSys)*. ACM, 2008, p. 251-258. doi : 10.1145/1454008.1454047.

[74] Saber Salah, Reza Akbarinia et Florent Masseglia. « A highly scalable parallel algorithm for maximally informative k-itemset mining ». In : *Knowl. Inf. Syst.* 50.1 (2017), p. 1-26. doi : 10.1007/s10115-016-0931-2.

[75] Idrissa Sarr, Hubert Naacke et Stéphane Gançarski. « DTR : Distributed Transaction Routing in a Large Scale Network ». In : *VECPAR International Workshop on High-Performance Data Management in Grid Environments (HPDGrid)*. T. 5336. Lecture Notes in Computer Science. Toulouse, France : Springer, juin 2008, p. 521-531. doi : 10.1007/978-3-540-92859-1_46.

[76] Idrissa Sarr, Hubert Naacke et Stéphane Gançarski. « Failure-Tolerant Transaction Routing at Large Scale ». In : *International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA)*. Menuires, France : IEEE, avr. 2010, p. 165-172. doi : 10.1109/DBKDA.2010.9.

[77] Idrissa Sarr, Hubert Naacke et Stéphane Gançarski. « Routage Décentralisé de Transactions avec Gestion des Pannes dans un Réseau à Large Echelle ». In : *Journées de Bases de Données Avancées (BDA)*. Guilherand Granges, France, oct. 2008, p. 1-20.

[78] Idrissa Sarr, Hubert Naacke et Stéphane Gançarski. « Routage décentralisé de transactions avec gestion des pannes dans un réseau à large échelle ». In : *Revue des Sciences et Technologies de l'Information - Série ISI : Ingénierie des Systèmes d'Information* 15.1 (2010), p. 87-111. doi : 10.3166/isi.15.1.87-111.

[79] Idrissa Sarr, Hubert Naacke et Stéphane Gançarski. « TransPeer : Adaptive Distributed Transaction Monitoring for Web2.0 applications ». In : *ACM Symposium on Applied Computing : Track on Dependable and Adaptive Distributed Systems (SAC DADS)*. Sierre, Switzerland : ACM, mar. 2010, p. 423-430. doi : 10.1145/1774088.1774179.

[80] Michael Stonebraker. « Technical perspective - One size fits all : an idea whose time has come and gone ». In : *Commun. ACM* 51.12 (2008), p. 76. doi : 10.1145/1409360.1409379.

[81] Bart Thomee et al. « YFCC100M : the new data in multimedia research ». In : *Commun. ACM* 59.2 (2016), p. 64-73. doi : 10.1145/2812802.

[82] Alexander Thomson et al. « Fast Distributed Transactions and Strongly Consistent Replication for OLTP Database Systems ». In : *ACM Trans. Database Syst.* 39.2 (2014), 11 :1-11 :39. doi : 10.1145/2556685.

[83] Immanuel Trummer et al. « SkinnerDB : Regret-Bounded Query Evaluation via Reinforcement Learning ». In : *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. ACM, 2019, p. 1153-1170. doi : 10.1145/3299869.3300088.

[84] Zhou Wei, Guillaume Pierre et Chi-Hung Chi. « CloudTPS : Scalable Transactions for Web Applications in the Cloud ». In : *IEEE Trans. Serv. Comput.* 5.4 (2012), p. 525-539. doi : 10.1109/TSC.2011.18.

[85] Jia Yu, Jinxuan Wu et Mohamed Sarwat. « GeoSpark : a cluster computing framework for processing large-scale spatial data ». In : *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, Bellevue, WA, USA, November 3-6, 2015*. ACM, 2015, 70 :1-70 :4. doi : 10.1145/2820783.2820860.

[86] B. Ziani et Y. Ouinten. « Mining maximal frequent itemsets : A java implementation of FPMAX algorithm ». In : *International Conference on Innovations in Information Technology (IIT)*. 2009, p. 330-334. doi : 10.1109/IIT.2009.5413790.