REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE Dr. MOULAY TAHAR – SAIDA

FACULTE DE TECHNOLOGIE

DEPARTEMENT D'INFORMATIQUE

N° D'ORDRE : .....

THESE

Présentée by

GUENDOUZ Mohamed

Pour le degré de

DOCTORAT «L. M. D» en INFORMATIQUE

Spécialité: Informatique

Option: Web et Ingénierie des Connaissances

Big Data et Réseaux Sociaux

Défendu publiquement, en ...../...../2018

Devant le jury composé de:

HAMOU Reda Mohamed	M.C.A	Université de Saida	Président
BELALEM Ghalem	Professeur	Université d'Oran 1	Rapporteur
BARIGOU Fatiha	M.C.A	Université d'Oran 1	Rapporteur
TOUMOUH Adil	M.C.A	Université de Sidi Bel Abbès	Rapporteur
AMINE Abdelmalek	Professeur	Université de Saida	Directeur de thèse

Année Universitaire 2017-2018

Laboratoire GeCoDe, Université de Saida

Remerciements

Je ne saurais présenter cette thèse sans exprimer ma gratitude tout d'abord au seigneur mon Dieu qui n'a jamais cessé de me combler de ses grâces.

Ma gratitude va ensuite au professeur AMINE Abdelmalek qui m'encadre depuis mon PFE de licence et qui n'a pas cessé de m'apporter tout le support dont j'avais besoin durant ces années et ne cesse d'être disponible jusqu'aujourd'hui. Merci pour tout.

Je remercie l'ensemble des membres du jury, qui m'ont fait l'honneur de bien vouloir étudier avec attention mon travail : Monsieur Ghalem Belalem, Professeur à l'université d'Oran 1. Madame Fatiha Barigou, Maître de Conférences à l'université d'Oran 1. Monsieur Toumouh Adil, Maître de Conférences à l'université Djillali Liabès de Sidi Bel Abbès, pour avoir accepté d'être rapporteurs de cette thèse, et enfin Monsieur Hamou Mohamed Reda, Maître de Conférences à l'université Tahar Moulay de Saida, pour m'avoir fait l'honneur d'accepter de présider ce jury.

Résumé —

Le monde a connu durant les deux dernières décennies une évolution phénoménale de données dans tous les domaines. Aujourd'hui, les données existent en des très grands volumes et en différentes variétés, et sont générées avec une vitesse rapide. Ce phénomène connu sous le nom de Big Data, a attiré l'attention de presque tout le monde ces dernières années, d'un simple utilisateur aux grandes entreprises de technologie. Ces données viennent de plusieurs sources et sont produites par de multiples acteurs. L'e-commerce, les objets connectés, et Internet sont parmi les sources qui génèrent plus de données, mais dernièrement on remarque une augmentation rapide d'un genre de données un peu spécial, dite données sociales. Ce genre de données vient des réseaux sociaux en ligne et qui est généralement produit par des simples utilisateurs, il existe en grandes quantités et en différents formats, texte, image, et vidéo. L'analyse des données sociales peut révéler plusieurs informations et connaissances pertinentes qui peuvent aider les chercheurs a mieux comprendre les comportements d'utilisateurs des réseaux sociaux.

Dans cette thèse, nous nous intéressons principalement à l'analyse des réseaux sociaux en ligne (ARS) dans un environnement big data en utilisant de nouvelles approches et techniques afin d'améliorer les résultats des méthodes existantes. Nous avons abordé plusieurs problématiques liées aux réseaux sociaux telles que la détection des communautés et la recommandation. Nous avons proposé notamment une nouvelle approche de détection de communautés en utilisant un nouveau algorithme d'optimisation inspiré des jeux d'artifices, les résultats obtenues des évaluations montrent des bonnes performances de cette approche. Nous avons aussi proposé un système de recommandation des projets open source pour les développeurs sur le réseau social GitHub en utilisant une approche de filtrage collaborative. Ce travail est soutenu par un prototype de site web permettant la recommandation des nouveaux projets open source aux utilisateurs.

Mots clés : Big Data, Réseaux Sociaux en Ligne, Analyse des Réseaux Sociaux, Détection de Communautés, Systèmes de Recommandation.

Abstract —

Over the past two decades, the world has experienced phenomenal changes in data across all domains. Today, data exist in very large volumes and in different varieties, and it is generated in a fast speed. This phenomenon known as Big Data, has attracted the attention of almost everyone in recent years, from a simple user to large technology companies. These data come from many sources and are produced by multiple actors. E-commerce, Connected Objects, and Internet are among the sources that generate more data, but lately we notice a rapid increase in a kind of somewhat special data, known as social data. This kind of data comes from online social networks and is usually produced by simple users, it exists in large quantities and in different formats including text, image, and video. The analysis of social data can reveal several relevant information and knowledge that can help researchers better understand the behaviors of social networking users.

In this thesis, we focus on the analysis of online social networks (SNA) in a big data environment using new approaches and techniques to improve the results of existing methods. We discussed several issues related to social networks such as community detection and recommendation. We proposed a new approach to community detection using a new optimization algorithm inspired by fireworks explosion in the air, the results obtained from evaluations show good performances of this approach. We also proposed a system of recommending open source projects for developers on the GitHub social network using a collaborative filtering approach. This work is supported by a prototype web site allowing the recommendation of new open source projects to users.

Keywords : Big Data, Online Social Networks, Social Networks Analysis, Community Detection, Recommendation systems.

خلاصة

على مدى العقدين الماضيين، شهد العالم تغيرات هائلة في البيانات عبر جميع المجالات. واليوم، توجد البيانات بكميات كبيرة جدا وفي أصناف مختلفة، ويتم توليدها بسرعة عالية. هذه الظاهرة المعروفة باسم البيانات الكبيرة ،جذبت انتباه الجميع تقريبا في السنوات الأخيرة، من مستخدم بسيط الى شركات التكنولوجيا الكبيرة. وتأتي هذه البيانات من مصادر عديدة وتنتجها جهات فاعلة متعددة. تعتبر التجارة الإلكترونية، أنترنت الأشياء، والإنترنت هي من بين المصادر التي تولد المزيد من البيانات، ولكن لاحظنا مؤخرا زيادة سريعة في نوع من البيانات الخاصة إلى حد ما، والمعروفة باسم البيانات الاجتماعية. هذا النوع من البيانات يأتي من الشبكات الاجتماعية على الانترنت وعادة ما تنتج من قبل المستخدمين العاديين، فهو موجود بكميات كبيرة وبأشكال مختلفة تضمن النص والصورة والفيديو. تحليل البيانات الاجتماعية يمكن أن تكشف عن العديد من المعلومات والمعرفة ذات الصلة التي يمكن أن تساعد الباحثين على فهم أفضل لسلوك مستخدمي الشبكات الاجتماعية.

في رسالة الدكتوراه هذه، نركز على تحليل الشبكات الاجتماعية على شبكة الإنترنت في بيئة البيانات الكبيرة باستخدام نهج وتقنيات جديدة لتحسين نتائج الأساليب القائمة. ناقشنا العديد من القضايا المتعلقة بالشبكات الاجتماعية مثل الكشف عن المجتمعات وتحليل أنظمة التوصية. واقترحنا نهجا جديدا للكشف عن المجتمعات باستخدام خوارزمية التحسين الجديدة المستوحاة من الألعاب النارية، والنتائج التي تم الحصول عليها من التقييمات تظهر الأداء الجيد لهذا النهج. واقترحنا أيضا نظام توصية للمشاريع المفتوحة المصدر للمطورين على الشبكة الاجتماعية جتحب باستخدام نهج التصفية التعاوني. ويدعم هذا العمل موقع نموذجي على شبكة الإنترنت يسمح بالتوصية بمشاريع جديدة مفتوحة المصدر للمستخدمين.

الكلمات المفتاحية البيانات الكبيرة، الشبكات الاجتماعية، تحليل الشبكات الاجتماعية، كشف المجتمعات، أنظمة التوصية.

Table des matières

Table des figures

Liste des tableaux

Table des sigles et acronymes

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
ARS	Analyse des Réseaux Sociaux
B2B	Business to Business
B2C	Business to Consumer
BDAS	Berkeley Data Analytics Stack
CNM	Clauset-Newman-Moore algorithm
DFS	Distributed File System
DMFWA	Discrete Modified FireWorks Algorithm
EC2	Elastic Compute Cloud
ETL	Extract Transfom Load
FWA	FireWorks Algorithm
GA-Net	Genetic Algorithms - Networks
GFS	Google File System
GN	Generated Networks
GN	Girvan et Newman algorihtm
HDFS	Hadoop Distributed File System
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
J2EE	Java 2 Platform, Enterprise Edition
JMS	Java Messaging Service
JSON	JavaScript Object Notation
KDD	Knowledge Discovery in Databases
KNN	K-Nearest Neighbors algorithm
MAE	Mean Absolute Error
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
MQ	Message Queue
MSR2014	Mining Software Repositories Challenge 2014
NDFS	Nutch Distributed File System
NMI	Normalized Mutual Information

NoSQL	Not Only Structured Query Language
NYT	New York Times
OASIS	Organization for the Advancement of Structured Information Standards
PDF	Portable Document Format
RDD	Resilient Distributed Dataset
RPC	Remote Procedure Call
SGBDR	Système de Gestion de Bases de Données Relationnelles
SGBD	Système de gestion de base de données
SMS	Short Message Service
SNA	Social Network Analysis
SQL	Structured Query Language
SRS	Services de Réseaux Sociaux
TCP/IP	Transmission Control Protocol/Internet Protocol
URL	Uniform Resource Locator
YARN	Yet Another Resource Negotiator
YASNS	Yet Another Social Networking Service
kNN	k-Nearest Neighbor
NBC	Naive Bayes Classifier

Introduction générale

Nous vivons l'âge du Big Data, chaque jour des énormes quantités de données sont produites par différents acteurs de technologie tels que les entreprises, les gouvernements, et les utilisateurs des services Internet. Les données peuvent être générées à partir de plusieurs sources, parmi elles on trouve les systèmes d'e-commerce qui génèrent les données à partir des transactions commerciales faites dans les supermarchés ou les transactions commerciales faites en ligne par les utilisateurs des sites web marchands. Les transactions commerciales en ligne sont souvent stockées dans des bases de données, et peuvent atteindre des millions chaque jour. Une autre source qui génère également de grandes quantités de données est les objets connectés, ce sont généralement des périphériques électroniques implantés dans des endroits distribués géographiquement, les données générées par ces périphériques sont complexes et présentent parfois des bruits qui nécessitent un prétraitement spécifique avant leur exploitation. Les caméras de surveillance sont un exemple populaire de ces objets connectés. Ces deux sources s'ajoutant à d'autres produisent des données de tailles volumineuses et de différents formats. Cependant, la majorité des données qui existent aujourd'hui viennent d'une seule source, les réseaux sociaux en ligne. Ces derniers ont connu ces dernières années une grande incrémentation du nombre d'utilisateurs dépassant un milliard aujourd'hui. Cette augmentation est due au développement des technologies de web et des moyens d'accès à Internet et leur disponibilité tels que les ordinateurs portables et les smartphones, ce qui a rendu Internet plus accessible aux milliards de populations du monde entier, en conséquence les réseaux sociaux en ligne sont devenus très populaires.

Big Data est une discipline qui s'intéresse au traitement des gros volumes de données, elle s'introduit dans tous les niveaux, de la collecte et stockage de données à l'analyse et l'exploitation de ces données. A l'instant, il n'existe aucune définition standard du terme Big Data, et les définitions qui existent aujourd'hui présentent chacune des concepts différents du même phénomène. Cependant, on peut donner des caractéristiques à ces données pour les distinguer des autres, ces caractéristiques sont souvent connues sous le nom de 4 Vs. Le terme 4Vs désigne Volume, Vélocité, Variété, et Véracité. Ce sont quatre propriétés utilisées pour décrire le phénomène de Big Data comme suit : Volume pour décrire le volume très important des données. Vélocité, ou vitesse, fait référence à l'énorme rapidité avec laquelle les données sont générées et traitées. Variété, les données traitées se présentent en différent formats structurées, semi-structurées, et non-structurées. Véracité, qui correspond à la fiabilité des données et qui inclut des questions sur la qualité des données et leur authenticité.

Objectifs de la thèse

Dans un premier temps, nous avons étudié la problématique de détection de communautés dans les réseaux sociaux. Ce problème qui ne cesse d'attirer l'attention de plusieurs chercheurs de différents domaines et disciplines. En effet, le problème de détection de communautés est considéré comme un problème complexe et difficile, cela signifie que les algorithmes tradition-

nels ne peuvent pas le résoudre dans un temps raisonnable. En plus le problème lui-même présente quelques difficultés telles que le nombre de communautés qui peuvent avoir un réseau, et la taille de chaque communauté qui n'est pas la même pour toutes les communautés. La majorité des méthodes qui ont été proposées pour résoudre ce problème sont basées sur des algorithmes de la théorie des graphes qui demandent généralement des moyens de calculs performants afin de les exécuter. En outre, ces méthodes présentent quelques limitations pour les réseaux à grandes échelles, notamment le temps d'exécution qui augmente rapidement. Pour résoudre ces limitations, nous avons proposé dans cette thèse une nouvelle méthode de détection de communautés basée sur l'algorithme de feux d'artifice ou Fireworks Algorithm (FWA). L'algorithme de FWA est un algorithme d'optimisation bio-inspiré évolutif proposé en 2010 par [TZ10]. Cet algorithme est inspiré des explosions des feux d'artifice dans le ciel, il a été conçu pour résoudre les problèmes d'optimisation dont l'espace de recherche des solutions est continu. Nous avons développé une version discrète de cet algorithme et nous l'avons appliqué au problème de détection de communautés. Nous avons testé notre approche sur plusieurs benchmarks de réseaux sociaux réels et artificiels, les résultats obtenus ont montré une bonne performance de notre approche face à ce problème. Nous allons détailler cette approche dans le chapitre 3.

Dans un deuxième temps, nous nous sommes intéressés aux systèmes de recommandation au sein des réseaux sociaux, et particulièrement dans le réseau social professionnel destiné aux développeurs GitHub. GitHub est un site web d'hébergement de projets open source publics et privés qui offrent plusieurs fonctionnalités des réseaux sociaux. Le site GitHub héberge des millions de projets open source accessibles publiquement en utilisant son moteur de recherche, les recherches sont faites manuellement et GitHub n'offre aucune recommandation pour choisir un tel ou un tel projet. Afin de suggérer des projets intéressants aux utilisateurs de ce site web, nous avons proposé un système de recommandation des projets open source basé sur le filtrage collaboratif. Ce système est implémenté en utilisant la méthode de recommandation Top-N et la méthode Jaccard pour le calcul des similarités. Nous avons testé le système sur un dataset populaire qui contient des données collectées du GitHub, les résultats obtenus à partir de ces évaluations montrent les bonnes performances de notre système notamment en matière de précision et de rappel. Le chapitre 4 discute l'architecture et le fonctionnement de ce système en détails.

Plan de la thèse

Cette thèse est organisée en 4 chapitres. Les deux premiers chapitres sont consacrés aux traitements des Big Data et analyse des réseaux sociaux respectivement. Tandis que les deux derniers chapitres présentent les travaux faits dans le cadre de cette thèse.

Le chapitre 1présente un état de l'art sur les Big Data. Tout d'abord, quelques définitions populaires du phénomène Big Data citées par les grands acteurs du domaine de la technologie sont abordées avant de passer à une description des caractéristiques des big data connues sous le nom de "4Vs". Enfin, les différentes étapes du processus big data sont présentées et discutées telles que la génération, l'acquisition, le stockage, et l'analyse de données. Pour chacune des étapes précédentes, des définitions sont données, et des méthodes et outils sont présentés.

Le chapitre 2 est consacré à l'analyse des réseaux sociaux. En premier lieu, quelques définitions et une brève historique des réseaux sociaux sont mentionnées. Ensuite, les méthodes et les techniques utilisées dans le domaine de l'analyse de ces réseaux sociaux sont abordées, tout en soulignant quelques notions utilisées dans la théorie des graphes telles que les graphes et les matrices d'adjacence ainsi que d'autres méthodes et métriques telles que le calcul de centralité et la modularité.

Le chapitre 3 est consacré à la présentation de notre approche proposée pour la détection de communautés en utilisant l'algorithme des feux d'artifices (Fireworks Algorithm). Ce chapitre commence par définir la problématique étudiée et enchainant ensuite avec une discussion des méthodes et approches proposées dans la littérature pour résoudre cette problématique. Ensuite une description de l'approche utilisée ainsi les modifications faites sur l'algorithme des feux d'artifices pour l'adapter au problème sera entamée. Enfin, une étude expérimentale qui évalue et compare notre approche avec d'autres algorithmes de détection de communautés est présentée et discutée avant de conclure le chapitre.

Le chapitre 4 discute notre deuxième travail concernant le système de recommandation des projets open source pour les développeurs sur le réseau social GitHub. Nous avons discuté dans le début de ce chapitre les différentes méthodes et approches utilisées dans les systèmes de recommandation avant de présenter site GitHub et ses fonctionnalités. Ensuite, l'architecture de l'approche utilisée est présentée juste après. Enfin, le chapitre discute les résultats obtenus après l'évaluation de notre approche sur des données réelles collectées du site web GitHub.

Enfin, une conclusion générale est présentée et un ensemble de perspectives sont discutées dans la fin de cette thèse dans le chapitre 4.9.

Chapitre 1

Big Data : État de l'art

Sommaire

1.1 Introduction

Les deux dernières décennies ont connu une grande augmentation des données dans tous les domaines en général et dans les domaines clés tel que l'e-commerce et les réseaux sociaux en particulier. Cette explosion de données a conduit à l'apparition d'un nouveau terme : Big Data ou Grandes Données, ce terme qui est principalement utilisé pour décrire les énormes quantités de données qui existent aujourd'hui, ne cesse pas d'attirer l'attention des chercheurs scientifiques dans différent domaines, les industriels et les gouvernements, il constitue aussi une matière riche pour la presse surtout dans les dernières années.

Dans ce chapitre, nous introduisons les big data en générale, nous commencerons d'abord par quelques définitions de ce terme et nous enchaînerons avec la description des fonctionnalités du big data et la définition de ses caractéristiques connus par le terme : les 4 V. Nous détaillerons également les différentes tâches qui constituent le processus de big data y compris la génération et l'acquisition, le stockage et l'analyse des données.

1.2 Définitions et caractéristiques des Big Data

La notion de big data est un concept abstrait. Aucune définition précise ou standard ne peut être donnée à ce terme, il n'est pas propre à un seul domaine, en fait sa définition varie d'un domaine à un autre et aussi selon les gens qui s'y intéressent en tant que chercheur, industriel ou un simple utilisateur. Littéralement, le mot anglais big data signifie grosses données, mégadonnées ou données massives, il est utilisé pour décrire un très grand ensemble de données qui ne peut pas être géré et traité par les systèmes informatiques traditionnels dans un temps acceptable.

En raison de différents contextes scientifiques et techniques, les entreprises, les chercheurs et les personnes engagés dans l'analyse des données ont des définitions différentes de big data. Les définitions suivantes peuvent nous aider à mieux comprendre les idées économiques, scientifiques et technologiques profondes de big data.

Définition 1 (Gartner) : cette définition est parmi les définitions les plus citée, elle est incluse dans un rapport de recherche [Dou01] du META Group (devenu Gartner) sortie en 2001. Malgré que ce rapport est ancien par rapport à la tendance actuelle et qui ne fait aucune mention de l'expression "big data", il est considéré comme une définition clé. Dans ce rapport, Gartner ont proposé une définition tridimensionnelle qui englobe les "trois V" : volume, vélocité et variété. Ce rapport fait référence à la taille croissante des données, le taux croissant auquel elles sont produites et la variété de formats et de représentations employées. Ce modèle (les trois V) a été réutilisé après par Gartner en 2012 [BL12] et développé par IBM [IBM13] et d'autres pour inclure un quatrième V : Véracité. La véracité comprend des questions de qualité et de précision en ce qui concerne les données et les résultats d'analyse de ces données.

Définition 2 (Oracle) : au contraire de Gartner, Oracle n'utilise aucun des Vs pour définir les big data. Au lieu de cela, Oracle prétend que le big data est la dérivation de la valeur à partir de la prise de décision commerciale basée sur les bases de données relationnelles,

renfoncée par des nouvelles sources de données non-structurées [Ora12]. De telles nouvelles sources incluent les blogs, les réseaux sociaux, les réseaux de capteurs, les données multimédia et d'autres formes de données qui varient en taille, en structure, en format et en d'autres facteurs. Oracle affirme donc que le big data est le renforcement des opérations actuelles par l'ajout des données supplémentaires issus de sources inhabituelles.

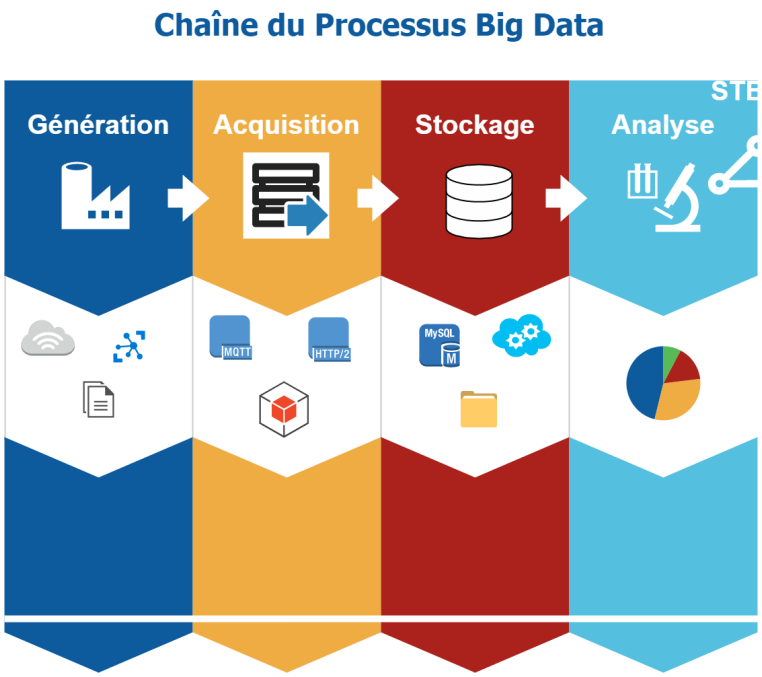
Définition 3 (Intel) : Intel est l'une des rares entreprises qui ont fourni des chiffres concrets dans leurs définitions du big data. Intel lie le phénomène de big data aux entreprises qui génèrent une moyenne de 300 téraoctets de données par semaine [Int12], en fait cette définition et l'une des résultats principales d'un sondage fait par l'entreprise à propos de big data. 200 managers IT de grandes entreprises ont participé dans ce sondage pour savoir comment ils s'adressent les problèmes d'analyse des big data dans leurs entreprises. Intel a trouvé aussi que les transactions commerciales gérés par les SGBD relationnelles représentent la majorité des données analysés, suivis par les documents, les E-mails, les données des capteurs, données d'imagerie, les blogs et les médias sociaux.

Définition 4 (Microsoft) : Microsoft définie le big data comme suit : "big data est le terme utilisé de plus en plus pour décrire le processus d'application de la puissance de calcul importante – les nouvelles méthodes d'apprentissage automatique et d'intelligence artificielle – à des ensembles d'informations très volumineux et souvent très complexes" [Mic13]. Cette définition indique clairement que le big data nécessite l'application d'une puissance de calcul très importante. En plus de ça, elle introduit de nouvelles technologies : l'apprentissage automatique et l'intelligence artificielle qui ont été négligés par les définitions précédentes. Ceci a introduit le concept de l'existence des technologies connexes qui sont des éléments essentielles d'une telle définition, en particulier, les bases de données NoSQL et la plateforme Apache Hadoop [Had].

Bien que les définitions précédentes présentent des idées différentes sur les big data, elles partagent toutes au moins une des affirmations suivantes : le volume, la complexité et les technologies. Volume : les données traitées sont souvent d'une taille très volumineuse. Complexité : les données ont des structures et des formats complexes qui nécessitent un prétraitement particulier pour chaque type. Technologies : l'utilisation des outils et des technologies spécifiques et évolués pour l'analyse de ces données.

Définition : Nous définissons les big data comme les ensembles volumineux des données structurées et non-structurées dont les techniques et les méthodes traditionnelles de stockage et d'analyse de données deviennent incapables et très imitées devant eux. Et qui demandent un traitement spécifique en utilisant des nouvelles méthodes suivant un processus standard appelé chaine de big data. La figure 1.1 montre les étapes de ce processus.

FIGURE 1.1: Chaine du processus big data



1.2.1 Caractéristiques

1. **Volume**
   La caractéristique principale des big data est le volume, les données big data ont toujours des volumes très importantes et qui ne cesse pas d'augmenter jour après jour. Aujourd'hui, on estime qu'on génère plus de 2 trillion de giga-octets de données chaque jour à partir des réseaux de télécommunication (téléphones portables), les interactions des utilisateurs sur les systèmes des réseaux sociaux tels que Facebook et Twitter, et bien d'autres sources de données. En 2010, Thomson Reuters a estimé dans son rapport annuel [Reu] qu'il y avait plus de 800 exaoctets de données dans le monde entier et qui sont toujours en augmentation.

   Dans la même année, EMC, leader de marché des dispositifs de stockage de données, a estimé que le nombre de données été proche de 900 exaoctets avec une augmentation possible de 50 % chaque année. Personne ne sait vraiment combien de nouvelles données sont générées par jour ou par an, mais une chose est vrais, c'est que la quantité d'informations recueillies aujourd'hui est énorme.

2. **Vélocité**
   La vélocité ou la vitesse, décrit la fréquence élevée dont les données sont générées et à quelle vitesse doivent-elles être traitées. Aujourd'hui, les données sont disponibles en temps réel, ce qui demande l'utilisation de nouvelles techniques pour analyser et traiter les données rapidement. Car, les méthodes traditionnelles de traitement des données prennent beaucoup de temps. En plus, certaines données doivent être traitées en temps réel, par exemple envoyer un SMS, publier un message sur son mur Facebook, et faire

un achat par la carte de crédit.

3. **Variété**
   Les données big data sont souvent présentes en différents genres et formats. La variété est une propriété très importante des données traitées, et qui décrit la diversité de forme de données qui existent aujourd'hui. Les systèmes traditionnels de traitement des données traitent souvent des données structurées (données suivant certaines règles), qui peuvent être facilement stockées dans des bases de données relationnelles. Mais, avec l'arrivée du big data, les données structurées sont complétées par des données non-structurées (des données qui ne suivent aucune règle) telles que les photos et les vidéos publiées par les personnes sur les réseaux sociaux. Ces nouveaux types de données nécessitent également des systèmes de gestion et de traitement de données spécifiques, car les systèmes traditionnels ne suffit pas et sont généralement très limités devant ces types de données.
4. **Véracité**
   La véracité fait référence à l'authenticité des données. Les données doivent être contrôlées avant qu'elles seront analysées surtout dans les applications sensibles telles que les applications médicales par exemple.

1.3 Génération de données

La génération de données est la première étape dans le processus du big data, elle constitue une partie indispensable dans ce processus, sans données on ne peut rien faire. Les données générées sont très variées (documents, images, signaux, ...) et très volumineuses, et sont généralement extraites ou générées à partir de plusieurs sources de données qui sont souvent distribuées géographiquement. Ces sources de données peuvent être des capteurs (appareils métrologiques, caméras de surveillance), des bases de données ou des réseaux sociaux en ligne, ...etc.

Malgré le grand nombre des sources qu'ils existent aujourd'hui et leurs diversité, les données d'entreprises restent la première source pour les big data, ceci inclut : les documents et les transactions commerciales.

1.3.1 Données d'entreprises

Dans un rapport publié en 2013 par IBM en collaboration avec Saïd Business School de l'université d'Oxford [IBM], les résultats du sondage ont montré que les données internes des entreprises représentent la majorité des données collectées et analysées par ces derniers. On y trouve principalement des transactions, des bases de données de journalisation et des emails, la plupart de ces données sont gérées par des SGBDR.

L'évolution de l'informatique en général et les moyens de transportation, stockage et de traitement de données numériques spécialement ont beaucoup contribué dans l'amélioration de la rentabilité des départements commerciaux des grandes et moyennes entreprises. Au cours des dernières décennies, le volume des données commerciales a connu une très grande augmentation et les derniers rapports estiment que ce volume va continuer de doubler tous

les trois ans, dans lequel, le chiffre d'affaires du commerce en ligne, B2B et B2C va atteindre 450 milliard de dollars par jour. Ce volume de données commerciales qui ne caisse jamais d'augmenter nécessite une analyse approfondie en temps réel afin de mieux exploiter son potentiel. Par exemple, Amazon, le leader des ventes en ligne traite environ 20 millions de transactions par jour.

1.3.2 Données IoT

IoT signifie Internet of Things ou Objets Connectés, c'est le terme utilisé pour décrire un réseau de capteurs distribués qui se communiquent à travers Internet d'où vient le mot Internet of things. Ils représentent une source très importante de données et on les trouve presque dans tous les domaines comme l'industrie, l'agriculture, les centres hospitaliers et le transport.

Un réseau IoT se compose de trois couches : la couche de détection, la couche réseau et la couche application. La couche de détection est responsable de l'acquisition des données, on y utilise les capteurs pour accomplir cette tâche, par exemple, dans un champ de culture on utilise des capteurs spécifiques pour collecter la température, l'humidité et d'autres indices. La deuxième couche, qui est le réseau est responsable de la transmission et du traitement de données collectées des capteurs, des architectures réseau spéciales peuvent être utilisées pour assurer la communication entre les capteurs ils-mêmes, cependant, le réseau Internet est utilisé pour faire la transmission à distance. Enfin, la couche application prend en charge les applications spécifiques des IoT.

Les données générées par les IoT ont des caractéristiques uniques par-rapport aux d'autres données, cela est dû à la complexité de l'architecture et la nature même des équipements utilisés. Voici quelques caractéristiques des réseaux IoT :

1. Gros volume de données : le grand nombre des équipements utilisés dans les réseaux IoT rend le volume des données générées très volumineux.
2. Hétérogénéité : les équipements utilisés sont très variés, ils peuvent générer des données numériques simples comme la température, ou des données multimédias complexes comme par exemple les vidéos.
3. Forte corrélation en temps et en géo-location : l'emplacement géographique du dispositif et le temps de la capture – le moment où le dispositif a capté la donnée – sont des propriétés très importantes dans l'analyse des données générées. Par exemple : dans un système de télésurveillance, deux extraits vidéos d'une même caméra à deux temps différents ne sont pas les même.
4. Données bruitées : les équipements peuvent produire une grande quantité de bruits lors de l'acquisition et la transmission de données dans les réseaux IoT. Cela, peut-être à cause des équipements, de l'état du réseau ou la météo dans certain cas. Ces données bruitées seront traitées avant d'être analysées pour enlever le bruit et seules les données pertinentes seront utilisées, on appelle généralement cette phase : la phase de nettoyage de données.

1.3.3 Données Web

Les données Web représentent toutes les données produites sur le web et qui sont générées par les utilisateurs de ce dernier. Parmi ces données on trouve : les recherches faites sur les moteurs de recherche telle que Google, Yahoo ou Microsoft Bing, les sujets partagés dans les blogs et les forums et les messages de discussion instantané échangés entre les utilisateurs. Les types et les formats de ces données peuvent-être variés entre les textes, les photos et les vidéos, mais généralement elles sont en format texte. Certes, que ces données ne sont pas d'une grande valeur pour les individus, mais, une fois collectées et analysées, elles peuvent fournir des informations et des connaissances précieuses.

1.3.4 Données biomédicales

Le grand développement des technologies de mesures biologiques au début du siècle 21 a fait rentrer le domaine de biomédecine dans l'ère de big data. Les appareils de mesure produisent des gros volumes de données en temps réel, ces données peuvent-être utilisées pour construire des modèles analytiques intelligents, efficaces et précis pour les applications de biomédecine.

L'achèvement du projet HGP (Human Genome Project) [NHG] et le développement continu de la technologie de séquençage conduisent également à des applications généralisées de Big Data sur le terrain. Les volumes de données générées par le séquençage génétique passent par une analyse spécialisée en fonction des différentes demandes d'applications, pour le combiner avec le diagnostic clinique des gènes et fournir des informations précieuses pour un diagnostic précoce et un traitement personnalisé de la maladie. Un séquençage du gène humain peut générer des données brutes de 100 à 600 Go. Dans la Banque Nationale Chinoise de Gènes à Shenzhen, il y'avait 1,3 millions d'échantillons, dont 1,15 million d'échantillons humains et 150 000 échantillons d'animaux, de plantes et de micro-organismes en 2014. À la fin de 2017, 20 millions d'échantillons biologiques traçables été stockés. Il est prévisible qu'avec le développement des technologies de la biomédecine, le séquençage génétique deviendra plus rapide et plus pratique, ce qui fera croître sans cesse les données massives de la biomédecine.

En outre, les données générées par les soins médicaux cliniques et la R&D médicale augmentent également rapidement. Par exemple, le Centre médical de l'Université de Pittsburgh (UPMC) a stocké 2 To de telles données. Explories, une société américaine, fournit des plates-formes pour la collection des données cliniques, des données d'exploitation et de maintenance et des données financières. À l'heure actuelle, environ 13 millions d'informations ont été regroupées, avec 44 articles de l'ordre de 60 To. Practice Fusion, une autre société américaine, gère les dossiers médicaux électroniques d'environ 200 000 patients.

En dehors de ces petites et moyennes entreprises, d'autres sociétés informatiques bien connues, telles que Google, Microsoft et IBM ont investi massivement dans la recherche et l'analyse computationnelle des méthodes liées aux big data biologiques à grande échelle. Selon IBM, la forte augmentation des images médicales et des dossiers médicaux électroniques, les professionnels de la santé peuvent utiliser les big data pour extraire des informations cliniques utiles à partir de données massives afin d'obtenir les antécédents médicaux et les effets du

traitement prévu, améliorant ainsi les soins aux patients et réduisant les coûts.

1.4 Acquisition de données

L'acquisition de données est la deuxième étape dans le processus de big data, cette tâche consiste en la collection, le filtrage et le nettoyage de données avant qu'elles seront stockées. Une fois les données brutes générées, elles doivent être rassemblées et envoyées à un système dédié où elles seront filtrées et en cas où elles contiennent des bruits, ce qui est habituel, elles seront nettoyées pour pouvoir éliminer la redondance et supprimer les données inutilisables, cela va nous permettre de réduire la taille des données et d'optimiser les moyens de stockage. Il existe des protocoles standards et des outils efficaces pour accomplir cette tâche et pour garantir que seules les données pertinentes seront stockées et traitées. Dans ce qui suit nous allons présenter quelques protocoles d'acquisition de données les plus populaires aujourd'hui.

1.4.1 Protocoles

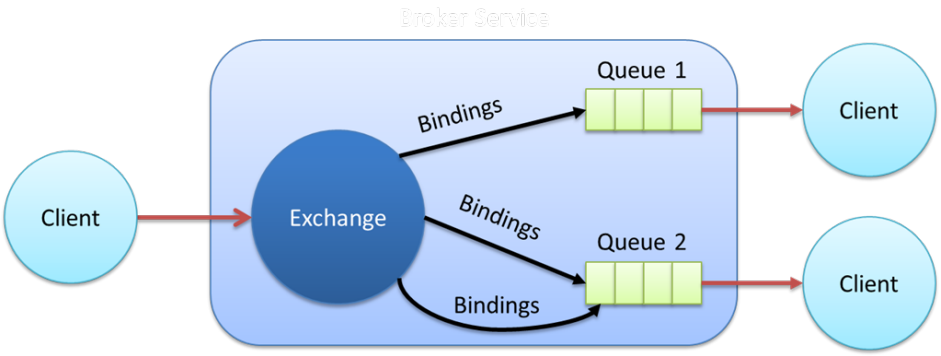
Plusieurs protocoles ont été conçus par des grandes entreprises comme des projets internes, et un petit nombre de ces protocoles ont été publiés publiquement. Dans la suite de cette section, nous présenterons les protocoles d'acquisition des données open source couramment utilisés.

1.4.1.1 AMQP

AMQP (l'acronyme en anglais de Advanced Message Queuing Protocol) [AMQ] est un protocole open source pour les systèmes de messagerie orientés message (middleware orienté message) crée par la banque JPMorgan Chase [Cha] pour gérer la communication entre ses différents partenaires. La spécification du protocole a été élaboré par un consortium international qui inclut plusieurs grandes entreprises tel que Microsoft, Cisco Systems, Red Hat, en aout 2011 le protocole est devenue un standard OASIS.

Le protocole AMQP permet aux applications de communiquer avec un serveur de messagerie orienté messages d'une façon asynchrone en utilisant un format de message standard, ce qui élimine les problèmes d'interopérabilité présents dans les solutions précédentes tel que IBM MQ [MQ]. Cela va permettre aux développeurs d'implémenter des applications clients et des serveurs sans se soucier du système d'exploitation ou du langage de programmation, par exemple, un serveur de messagerie écrit en langage C++ transférera des messages d'une application client écrit en langage Java vers un autre client écrit en langage Python.

FIGURE 1.2: Le modèle du protocole AMQP



La figure 1.2 représente le modèle de base du protocole AMQP. Dans ce modèle, le producteur et le consommateur sont considérés comme des applications client, le composant entres ces deux entités est un serveur qui gère l'échange des messages entres eux, il est appelé broker (message broker en anglais) ou agent de messages, mais l'appellation la plus utilisée et broker. Le rôle de broker est d'assurer l'envoi et la réception des messages entre les clients, faut noter qu'il peut y avoir plusieurs producteurs et consommateurs au même temps. Pour bien gérer les flux de messages reçus et les envoyer à leurs destinations, le broker AMQP utilise trois composants principaux, Exchange (échange), Queue (files d'attente) et Binding (règles de mapping). Une fois le broker reçoit un message du producteur, le composant Exchnage distribuera des copies de ce message à des queues en utilisant les règles de mapping, ensuite un consommateur peut chercher/tirer le message de la file d'attente.

1.4.1.2 JMS

JMS, acronyme de Java Messaging Service, est une API (Interface de Programmation Applicative) qui permet aux applications écrites en Java de dialoguer entre eux grâce à des brokers de messages. JMS a été intégré à la plate-forme J2EE à partir de la version 1.3. JMS définit plusieurs entités tel que :

Un fournisseur JMS : un outil qui implémente l'API JMS pour échanger les messages : ce sont les brokers de messages.

Un client JMS : une application/composant qui envoi et/ou reçoit les messages.

Un producteur/éditeur JMS : un client JMS qui crée et envois des messages.

Un consommateur/abonné JMS : un client JMS qui reçoit des messages.

Un message JMS : un objet contenant les données transférées entre les clients JMS.

Le protocole JMS supporte deux modèles de diffusions de messages présentés sous-dessous :

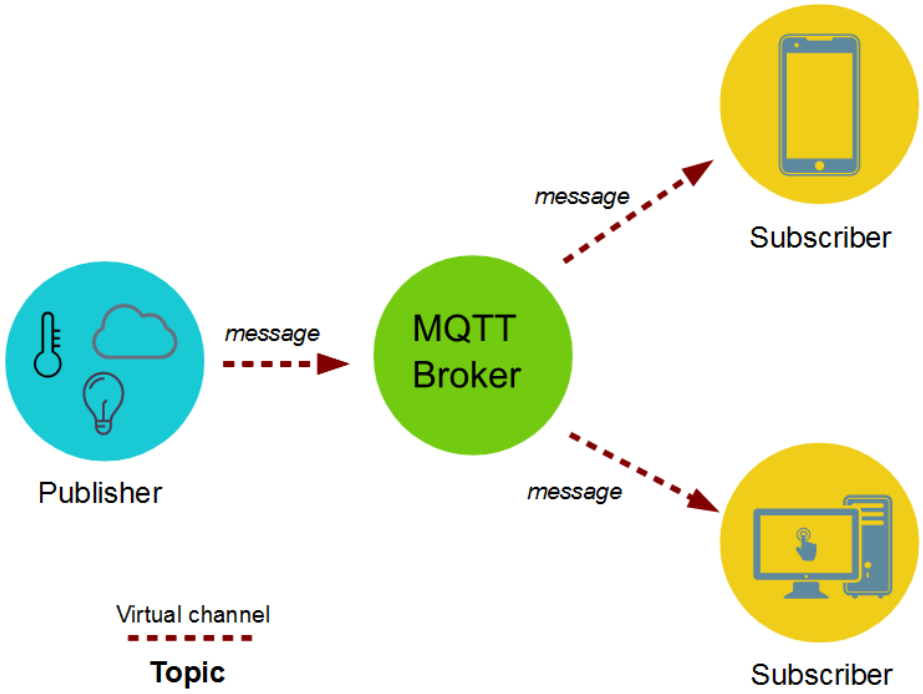
Point à point : dans un modèle de messagerie point à point, les messages sont acheminés vers un consommateur individuel en utilisant une file d'attente. Une fois le message est envoyé par le producteur, il sera stocké dans une file d'attente jusqu'à ce que le consommateur le lise, ensuite le message sera supprimé de la file.

Publication/Souscription : dans ce modèle, un message peut être lu par un ou plusieurs consommateurs en utilisant la notion de sujet (Topic). Chaque consommateur doit s'abonner à un sujet (souscription). Seuls les messages émis à partir de cet abonnement sont accessibles par le consommateur.

1.4.1.3 MQTT

MQTT [MQT], l'acronyme de MQ Telemetry Transport, est un protocole de messagerie extrêmement simple et léger basé sur le protocole TCP/IP et qui utilise le modèle publication/souscription. Il a été inventé par Dr Andy Stanford-Clark d'IBM, et Arlen Nipper de Arcom en 1999, puis rendu open source, il est aujourd'hui un standard OASIS. Le protocole MQTT est conçu principalement pour les appareils contraints et les réseaux à faible bande passante, il est bien adapté aux réseaux d'objets connectés où les appareils utilisés se disposent généralement de ressources limitées. La figure 1.3 montre l'architecture de base de ce protocole.

FIGURE 1.3: L'architecture de base du protocole MQTT



Le protocole MQTT utilise aussi un broker de messages pour gérer l'échange des messages entre les différents clients connectés au serveur, en plus les fonctionnalités standards d'un broker, le broker MQTT gère aussi l'authentification des clients en utilisant un mot d'utilisateur et un mot de passe mais ce n'est pas le cas toujours, certains broker publiques acceptent les clients anonymes. MQTT fonctionne en utilisant les principes de méthodes ou de verbes, les clients utilisent ces méthodes pour communiquer avec le broker de messages (le serveur). Pour qu'un client puisse envoyer/recevoir un message, il doit ouvrir une session

avec le broker et cela par la création d'une connexion TCP/IP vers le broker en utilisant la méthode **CONNECT**, une fois la session est approuvée par le serveur, le client peut envoyer des messages aux autres clients en utilisant la méthode **PUBLISH** ou recevoir des messages d'eux en utilisant la méthode **SUBSCRIBE**. Lorsqu'un client souhaite mettre fin à une session MQTT, il envoie un message **DISCONNECT** au broker. Faut noter que lors de l'envoi des messages, les clients MQTT peuvent choisir un niveau de qualité de service, le niveau de service le plus simple est le niveau 0 ou le service non confirmé (Unacknowledged Service). Dans ce niveau, le client envoie le message une seule fois au broker et ce dernier le transmit une seule fois aux autres clients abonnés. Aucun mécanisme ne garantit la réception du message et le broker ne l'enregistre pas non plus. Le tableau **??** présente quelques méthodes MQTT ainsi que leurs variantes.

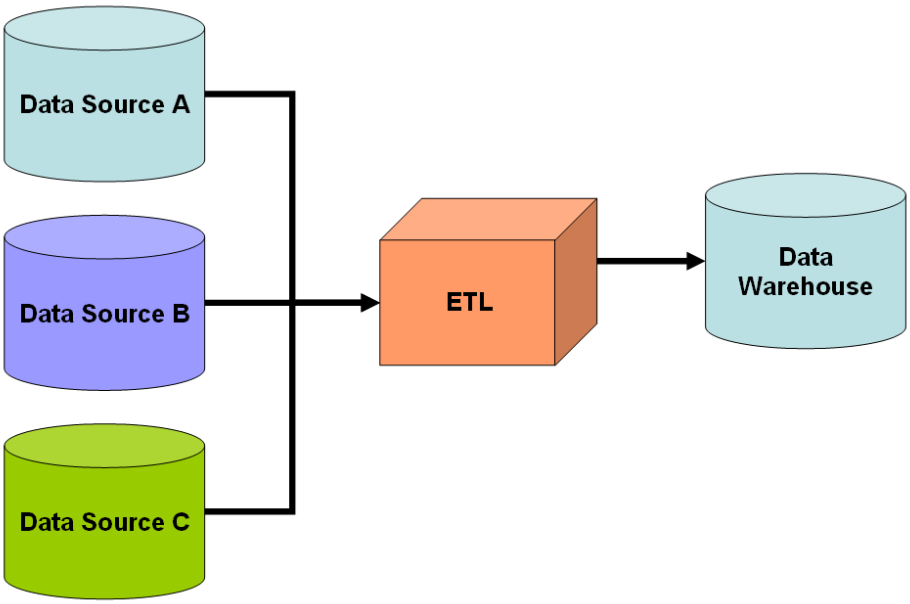
Table 1.1: Méthodes MQTT

Méthode	Description
CONNECT	Demander une connexion au serveur
CONNACK	Réponse envoyée par le serveur après demande de connexion
PUBLISH	Publier (envoyer) un message
PUBACK	Réponse envoyée après envoi d'un message PUBLISH
PUBREC	Message envoyé par le serveur informant le client que son message PUBLISH a été reçu
PUBREL	Message envoyé par le client pour informer le serveur de la sortie du message
PUBCOMP	Le serveur informe que l'envoi du message est finie
SUBSCRIBE	Demande d'abonnement d'un client au serveur
UNSEBSCRIBE	Un client demande d'annuler son ancien abonnement à un topic
DISCONNECT	Le client se déconnecte

1.4.2 Intégration de données

L'intégration de données est le processus de combinaison de multiples données provenant de différentes sources et de fournir aux utilisateurs une vue unifiée de ces données [Len02]. Ce processus comprend généralement trois tâches, la première tâche consiste à extraire les données de différentes sources (bases de données, documents, emails, ...), ensuite vient la deuxième tâche qui consiste à appliquer des transformations sur les données reçues. En fin, envoyer les données résultantes vers les systèmes ou les bases de données cibles. Ce processus est généralement connu sous le nom de ETL qui signifie en anglais, Extract Transfom Load. Une des approches ETL les plus populaire et couramment utilisée est l'approche d'entrepôt de données (data warehouse). La figure 1.4 montre un schéma simple d'un entrepôt de données utilisant le processus ETL.

FIGURE 1.4: Schéma d'un data warehouse utilisant le processus ETL



L'intégration de données semble une discipline facile, mais en réalité elle est compliquée. Il n'y a aucune approche universelle standard pour l'intégration de données, un grand nombre de solutions qui existent aujourd'hui ont été développées généralement pour résoudre des problèmes bien précis. Certaines approches d'intégration de données peuvent bien marcher que d'autres pour une entreprise selon ses besoins.

1.4.3 Nettoyage de données

Le nettoyage de données est la procédure de détection et de correction d'erreurs présentes sur les données, que ce soit sur des bases de données, documents, ou fichiers. Cette procédure comporte généralement cinq sous-procédures [MM00] classées dans l'ordre suivant : définition des types d'erreurs, chercher et identifier les erreurs, correction, documenter les exemples d'erreurs détectées, et enfin la modification des procédures de saisie de données pour réduire le nombre d'erreurs dans le futur.

Parmi les types d'erreurs, on trouve les erreurs lexicales, syntaxiques, et de formatage lié aux fautes de saisies, par exemple, à la place de mettre le caractère "F" dans la cellule de sexe, l'agent de saisie met "21" ce qui représente ici une erreur. On trouve aussi les erreurs sémantiques qui sont généralement liées à la violation de contraintes des systèmes, et les erreurs de couverture dues au manque de valeurs pour une donnée ou l'absence de la donnée elle-même. En corrigeant toutes ces erreurs, le nettoyage de données nous permet de garder que les données pertinentes pour les systèmes, il est très important pour conserver la cohérence de données. La procédure de nettoyage de données fait partie de nombreux systèmes de traitement de données dans plusieurs domaines notamment dans le commerce où une seule erreur peut mettre tout le système dans une situation dangereuse.

1.5 Stockage de données

Le stockage de données arrive juste après l'étape de génération et acquisition de données, il est dans la deuxième position de la chaine big data tel qu'il est montré dans la figure **??**. Le stockage de données consiste en le stockage et la gestion des données volumineuses tout en gardent la fiabilité et la disponibilité des données en satisfaisant les besoins des applications nécessitant l'accès à ces données.

Un système de stockage de donnée big data doit assurer le stockage de quantités illimitées de données, gérer un taux élevé d'opérations d'écriture/lecture de données (requêtes), supporter différents modèles de données ainsi que les données structurées et non-structurées à la fois. Évidemment, un tel système ne peut pas satisfaire tous ces besoins, mais au cours des dernières années, de nombreux nouveaux systèmes de stockage ont été développés pour satisfaire au moins une partie de ces besoins. Dans cette section, nous présentons les technologies de stockage de données les plus utilisées aujourd'hui pour faire face aux défis émergés par les données big data.

1.5.1 Technologies de stockage des données

L'explosion de données structurées et non-structurées dans différents domaines ces dernières années, surtout dans le web tel que les données des réseaux sociaux (textes, images, vidéos, ...), et l'incapabilité des systèmes de gestion de bases de données traditionnels devant ce phénomène ont conduit à l'apparition d'une nouvelle génération de systèmes de stockage de données avancés, capables de gérer d'immenses quantités de données d'une façon fiable, rapide, et sécurisée. Ces nouveaux mécanismes de stockage de données big data peuvent être classifiés en trois niveaux, systèmes de gestion de fichiers, bases de données, modèles de programmation pour l'interrogation des données.

Nous discuterons dans cette section les approches populaires de cette nouvelle génération de systèmes de stockage, tel que les systèmes de stockage distribués à base de fichiers, les bases de données NoSQL, et le stockage en nuage (cloud storage).

1.5.1.1 Systèmes de fichiers distribués

Un système de fichiers distribué (DFS en anglais) est un système de gestion de fichiers où les fichiers (données) sont stockés sur des serveurs (réseaux) et non pas sur une seule machine. Un DFS fournit aux utilisateurs les même fonctionnalités d'un système de gestion de fichiers classique, les fichiers sont accessibles et traités comme si'ls étaient stockés sur une seule machine. Mais en plus des fonctionnalités de base d'un système de gestion de fichiers classique, un DFS offre d'autres fonctionnalités comme : le contrôle d'accès, l'authentification des utilisateurs, la réplication de données stockés sur les serveurs, et bien plus. Un système de fichiers distribué devrait avoir les caractéristiques suivantes :

Transparence : les utilisateurs doivent accéder au système indépendamment de leur connexion, être en mesure d'effectuer les mêmes opérations sur les DFS et les systèmes de fichiers locaux et ne devraient pas se préoccuper des défauts dues à la nature distribuée du

système de fichiers grâce à des mécanismes de tolérance de panne. La transparence peut également être observée en termes de performance : les manipulations de données devraient être au moins aussi efficaces que sur les systèmes de fichiers classiques.

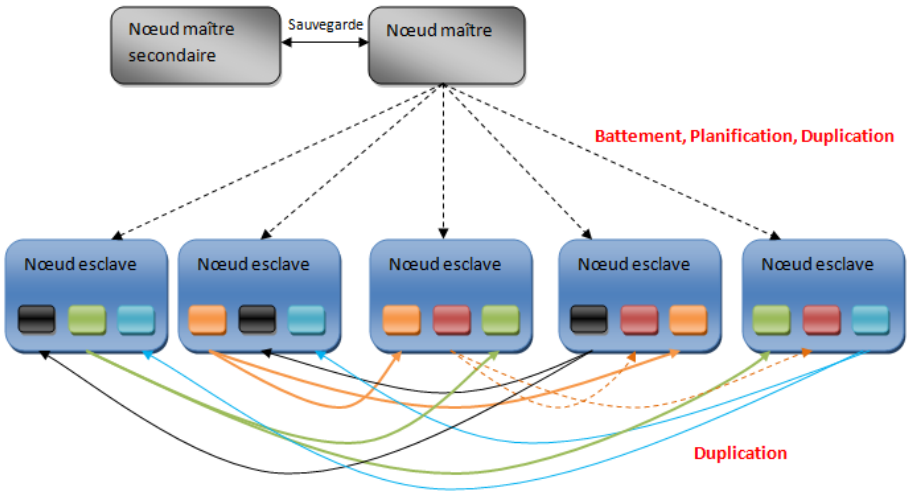
Tolérance aux pannes : un système à tolérance de panne ne doit pas être arrêté en cas de panne temporaire. Les défauts considérés sont : les pannes de réseau et de serveur qui rendent les données et les services indisponibles, l'intégrité et la cohérence des données lorsque plusieurs utilisateurs accèdent simultanément aux données.

Évolutivité : c'est la capacité de gérer efficacement un grand nombre de serveurs qui sont ajoutés dynamiquement et continuellement dans le système. Habituellement, il s'agit d'environ dizaines de milliers de nœuds (serveurs).

Il existe plusieurs systèmes de fichiers distribués, qui ont été proposés par des grandes entreprises et aussi par la communauté open source pour répondre à des besoins bien spécifiques, un des systèmes les plus populaires aujourd'hui est HDFS [Had]. HDFS ou Hadoop Distibuted File System est un système de fichiers distribué écrit en langage JAVA et développé par Apache Software Foundation [Apac], il a été conçu avec l'intention de supporter les serveurs avec des ressources limitées. Ce genre de serveurs tombent en panne fréquemment mettant les données en danger, le HDFS est donc conçu pour être extrêmement tolérant aux pannes.

Le HDFS est basé sur une architecture Maître/Esclave (master/slave), dans cette architecture, un seul serveur gère l'espace de noms du système de fichiers et l'accès aux fichiers par les clients, ce serveur appelé NameNode est considéré comme un serveur maître. En plus, il y'a un certain nombre de serveurs appelés DataNodes, ces serveurs sont considérés comme des esclaves et ils gèrent le stockage des fichiers. La figure 1.5 montre le schéma d'un cluster HDFS.

FIGURE 1.5: Schéma d'un cluster HDFS



NameNode : le serveur maître, souvent une machine Linux qui exécute le code du NameNode. Ce serveur gère l'espace de noms, les opérations d'ouverture, fermeture, et

renommage des fichiers et des dossiers. Il détermine également le mapping des blocs de fichiers dans les serveurs DataNodes.

DataNodes : les serveurs esclaves, sont responsables d'effectuer les opérations de lecture/écriture des fichiers selon les requêtes des clients. Les DataNodes effectuent également d'autres opérations telles que la création, la suppression, et la réplication de blocs de fichiers selon les instructions du Namenode.

Bloc : les fichiers stockés sur HDFS sont divisés en un ou plusieurs segments qui sont eux-mêmes distribués en copies sur les différents DataNodes. Ces segments sont appelés blocs, la taille par défaut d'un bloc est égale à 64 MO mais elle peut être augmentée en fonction des besoins.

1.5.1.2 Les bases de données NoSQL

Les technologies de systèmes de gestion de bases de données ont connu une évolution croissante pendant les 30 dernières années. Plusieurs systèmes ont été développés pour gérer les grands ensembles de données et pour supporter les différentes applications. Les bases de données relationnelles sont les plus connus et les plus utilisées. Depuis leurs création dans les années 70, ces systèmes de gestion de bases de données relationnelles (SGBDR) ont constitué un composant essentiel de plusieurs systèmes informatiques, ils ont été utilisé dans divers domaines tel que, les banques, le commerce électronique, ...etc. Mais à partir des années 2000, le monde a connu une explosion de données avec l'apparition des réseaux sociaux et les objets connectés. La grande quantité de données et la vitesse dont elles sont générées, en plus la variété de ces données qui sont généralement des données non-structurées ont rendu les SGBDR très limités devant ce nouveau phénomène. Pour faire face à ces problèmes, de nouveaux systèmes été développés par certains grandes entreprises notamment Google et Amazon faisant apparaitre une nouvelle génération de base de données, connu après sous le nom de NoSQL.

NoSQL est une nouvelle génération de bases de données conçus pour supporter les données volumineuses, qui sont différentes des bases de données relationnelles dans la façon elles gèrent le stockage et l'interrogation des données. Pour le moment, il n'existe aucune définition standard de NoSQL (ce sujet reste à débattre), mais on peut retirer quelques caractéristiques communs des NoSQL telles que :

Non-relationnel : les bases de données NoSQL n'utilisent pas le modèle relationnel pour gérer les relations entre les données, au lieu de cela, elles utilisent les structures embarquées pour stocker les données et les relations entre eux dans un même endroit.

Marche très bien sur les architecture distribués : on peut facilement faire marcher un système de base de données NoSQL sur un cluster.

Schéma libre : au contraire des SGBDR qui utilisent un schéma pour représenter les données et les relations, les bases de données NoSQL ne respectent pas un schéma précis.

Open source : les bases de données NoSQL sont généralement développées et distribuées publiquement en open source.

Les bases de données NoSQL peuvent être regroupées en quatre grandes catégories :

Clé/valeur
Les bases de données clé/valeur sont les plus simples des bases de données NoSQL. Elles utilisent le modèle clé/valeur pour stocker les données où chaque clé est unique et elles ne respectent aucun schéma. Les clients peuvent utiliser ces clés après pour récupérer des valeurs selon des critères. Les valeurs représentent les données et elles peuvent être complètement des données non-structurées ou des données structurées. Les bases de données NoSQL sont caractérisées par leur performance, leur grande capacité d'extensibilité, elles peuvent être facilement étendues, et par leur temps de réponse rapide aux requêtes. Plusieurs bases de données NoSQL de cette catégorie ont été développées durant les dernières années, les plus populaires parmi eux sont, Amazon DynamoDB [Ama], Redis [Red], Memcached [Mem], et Couchbase [Coua].

Orienté colonne
Une base de données orienté colonne sauvegarde les données en utilisant des colonnes plutôt que des lignes. Ce modèle est proche de tables de bases données relationnelles à l'exception qu'avec une base de données orienté colonne, le nombre de colonnes est dynamique, et il peut varier d'un enregistrement à un autre, évitant les colonnes de valeur NULL. Les bases de données orienté colonne ont la capacité d'ajouter des colonnes d'une façon dynamique, au contraire des bases de données relationnelles où le nombre de colonnes est fixé dès la création du schéma de la table. Les bases de données orienté colonne sont principalement inspirées par BigTable [Big] de Google, parmi eux on trouve, Apache HBase [Apaa] et Cassandra [Cas].

Orienté document
Les bases de données orienté document reposent sur le modèle clé/valeur, à la différence que, les valeurs sont des documents en format structuré ou semi-structuré. Tout comme les bases de données clé/valeur, les documents peuvent être interrogés à l'aide d'une clé unique. Cependant, il est possible d'accéder aux documents en interrogeant leur structure interne, par exemple en demandant tous les documents contenant un champ avec une valeur spécifiée. Cette catégorie de bases de données est utilisée généralement dans les systèmes de gestion de contenu. Les implémentations les plus populaires sont, MongoDB [Mon], CouchDB [Coub], et SimpleDB [Aws].

Orienté graphe
Les bases de données orienté graphe se basent sur la théorie des graphes. Les données sont stockées en utilisant les concepts de graphes comme les nœuds et les arêtes. Un nœud représente une entité (par exemple une personne), et une arête représente une relation entre deux entités (par exemple une amitié). En utilisant ce modèle on peut facilement représenter des concepts du monde réel tel que les réseaux sociaux. Neo4J [Neo] est la solution la plus populaire dans cette catégorie.

1.5.1.3 Les bases de données NewSQL

NewSQL est une nouvelle génération de bases de données relationnelles modernes, conçues pour fournir les mêmes performances des NoSQL, tout en gardant les propriétés transaction-

nelles solides des bases de données relationnelles. Les bases de données NewSQL utilisent le modèle relationnel pour représenter les données (tables), et le langage SQL pour l'interrogation. Mais elles sont distribuées et généralement implémentées entièrement en mémoire pour conserver la rapidité même sur les données volumineuses.

Les bases de données NewSQL présentent plusieurs avantages par rapport aux bases de données NoSQL, en particulier, l'utilisation du langage SQL comme langage principal d'interrogation, garantir les propriétés ACID des transactions, et la compatibilité avec les outils SQL standards. Cependant, leur approche de calcule en-mémoire (In-Memory) est confrontée à quelques limitations comme l'incapacité de traiter des données volumineuses de la taille de téraoctets, et la nécessité d'avoir un matériel spécifique avec une capacité de mémoire (RAM) très importante.

Tout comme les NoSQL, il existe plusieurs catégories de bases de données NewSQL telles que :

Nouvelle conception : cette catégorie de bases de données NewSQL sont écrites à partir de zéro pour supporter des architectures distribuées. Parmi les solutions NewSQL de cette catégorie on trouve, Google Spanner [] et VoltDB [].

Moteur de stockage MySQL : ce sont des moteurs MySQL qui implémentent les caractéristiques des bases de données NewSQL tout en gardant la même interface d'interrogation de MySQL. Les moteurs de stockage NewSQL pour MySQL les plus populaires sont, ScaleDB [], TokuDB [].

Middleware : les solutions de cette catégorie fournissent une couche "sharding", pour permettre la division automatiques de données sur plusieurs serveurs. Les middlewares les plus connus sont, ScaleBase [], ScaleArc [], et dbShards [].

1.5.1.4 Les plateformes d'interrogation des données big data

Les plateformes d'interrogation big data fournissent des solutions pour interroger des données stockées dans des architectures distribuées et gérées par des systèmes de fichiers distribués tel que, HDFS et GFS. Ces solutions ont pour objectif de permettre aux leurs clients d'interroger les données en utilisant des requêtes proches de celles du langage SQL, mais elles utilisent des approches différentes.

Apache Hive [] est parmi les solutions les plus populaires, ce système fait partie d'Hadoop offre la possibilité d'interroger des données structurées stockées en HDFS en utilisant un langage proche du SQL. Hive exécute les requêtes en les traduisant dans des tâches MapReduce [], en conséquence, les requêtes Hive ont une forte latence, même pour les petits ensembles de données. L'avantage principal de Hive est l'utilisation des requêtes proches de SQL, et la flexibilité d'évoluer les schémas facilement.

1.5.1.5 Stockage en nuage

Le stockage en nuage ou en cloud est une méthode simple et efficace pour sauvegarder, accéder, et partager les données sur Internet. Le stockage en nuage peut être utilisé par

les utilisateurs simples et les entreprises. Pour les utilisateurs, il les permet de sauvegarder et accéder à leurs données de n'importe quel endroit et sur n'importe quel périphérique connecté à Internet d'une manière fiable et sécurisée. Les utilisateurs peuvent utiliser ce mode de stockage pour partager des grands fichiers avec d'autres utilisateurs, ou pour faire des sauvegardes en ligne de leurs fichiers locaux. De même, le stockage en nuage offre aux entreprises des solutions de stockage en ligne distribuées et sécurisées avec des prix moins chers par rapport à une solution traditionnelle.

Techniquement, on peut distinguer deux types de solutions de stockage en ligne, un stockage en objet, et en bloc. Dans le stockage en objet, les différents fichiers téléchargés sur le serveur sont considérés comme des objets où chaque objet représente un seul fichier. En revanche, le stockage en bloc sauvegarde les données dans des volumes appelés blocs, chaque bloc est considéré comme un disque dure qui permet l'accès aléatoire aux données sauvegardées sur cet objet. Le stockage en bloc fonctionne bien pour les applications qui demandent un accès direct en lecture/écriture aux données.

1.6 Analyse de Big Data

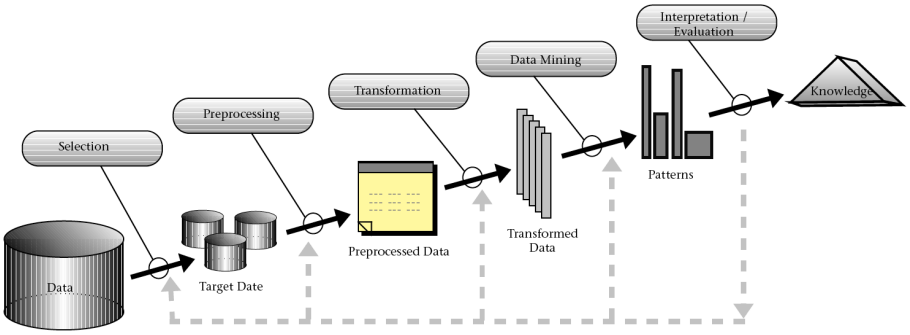
Les grandes quantités de données collectées présentes sous différents formats contiennent des quantités d'informations et de connaissances très importantes. Ces informations et connaissances, une fois identifiées et extraites peuvent nous aider à créer des systèmes d'aides à la décision fiables ainsi que nous aider dans la conception et le développement des applications spécifiques à un domaine. Par exemple, une collection de séquences vidéo d'un système de surveillance peut être utilisée pour améliorer un système de détection de personnes suspectes.

L'analyse de données est la dernière étape dans la chaine de big data, elle implique principalement les méthodes traditionnelles d'analyse de données, les architectures d'analyse de big data, et les outils logiciels. Dans cette section, nous présentons les méthodes d'analyse de données traditionnelles, ensuite, nous introduisons quelques plateformes d'analyse utilisées dans l'analyse des big data.

1.6.1 Fouille de données

La fouille de données (data mining en anglais) est un processus qui vise à extraire des connaissances cachées et inconnues jugées pertinentes à partir d'ensembles de données. Ce processus est aussi appelé Extraction des Connaissances à partir de Données, Knowledge Discovery in Databases (KDD) en anglais, il est illustré par la figure 1.6. Le processus KDD prend en entrée des données brutes et fournit en sortie des modèles trouvés dans ces données. Pour cela, KDD sélectionne un sous-ensemble de données à partir des données brutes désigné comme données cibles (target data). Ensuite, les données cibles seront prétraitées pour les préparer à être analysées par les algorithmes de fouille de données. Après, un algorithme de fouille de données sera appliqué sur l'ensemble de données cibles prétraitées pour extraire des modèles. Enfin, les modèles extraits seront évalués pour s'assurer de leur performance et ils seront aussi interprétés afin de nous fournir un aperçu.

FIGURE 1.6: Processus d'extraction de connaissances à partir de données (KDD)



Pour résumer, le processus KDD est constitué de sept étapes essentielles triées comme suit :

1. **Étude du domaine :** La toute première étape consiste à bien comprendre le domaine d'application. Il s'agit d'une étape préparatoire qui définit la scène pour comprendre ce qu'il faut faire avec la transformation, les algorithmes et la représentation.
2. **Sélection de données cibles :** La première étape dans le processus KDD consiste à sélectionner un sous-ensemble de données à partir des données originales qui sont souvent stockées dans des bases de données. Cette étape a pour but d'assurer la qualité des données cibles qui seront utilisées au long du processus.
3. **Prétraitement des données :** Cette étape consiste à prétraiter les données cibles sélectionnées dans l'étape précédente. Le prétraitement inclut le nettoyage pour supprimer les données bruitées, et la réparation des données incomplètes. En plus d'autres prétraitements spécifiques à chaque type de données.
4. **Transformation de données :** La troisième étape consiste à transformer les données de telle sorte qu'elles seront adaptées aux algorithmes de fouille de données. Par exemple, les données textuelles doivent être transformées en données numériques pour qu'elles soient exploitables par les algorithmes.
5. **Fouille de données :** Dans cette étape, un algorithme de fouille de données sera choisi pour être appliqué sur l'ensemble de données transformées. Le résultat de cette étape est un modèle d'apprentissage qui sera ensuite utilisé pour extraire des connaissances à partir des données.
6. **Évaluation du modèle :** Une fois le modèle généré, il sera évalué sur un ensemble de données de test pour valider ses performances.
7. **Représentation des connaissances :** La dernière étape dans le processus KDD consiste à représenter les connaissances extraites des données par le modèle d'apprentissage en utilisant des techniques spécifiques.

Dans le processus KDD, les étapes de sélection, prétraitement, et transformation de données sont toutes considérées comme des étapes de prétraitement. Ces étapes prennent en entrée des données et donnent en sortie des données aussi, la différence est que chaque étape va appliquer un certain nombre de traitements et modifications sur les données en entrée afin de les nettoyer ou les faire adaptées pour les étapes suivantes. Après ces étapes, vient l'étape la plus importante dans tout le processus KDD, elle s'agit de l'étape de fouille de données (data mining). Dans cette étape, un algorithme d'apprentissage automatique doit être choisi et fixé, ensuite appliqué sur les données prétraitées. Le type d'apprentissage qu'on souhaite appliqué doit être fixé dans la première étape du processus KDD, l'étape d'étude du domaine. Il existe plusieurs types d'apprentissage automatique tel que la classification supervisée et non-supervisée, et la régression. Par exemple, si nous somme devant un problème de détection comme la détection de spams ou la détection de plagiat, dans ce cas, la classification supervisée est le type d'apprentissage le plus adapté pour ces deux problèmes. Pour chaque type d'apprentissage il existe beaucoup d'algorithmes - dits algorithmes d'apprentissage automatique - qu'on peut les utiliser. Nous allons discuter dans la suite les types d'apprentissage et nous allons présenter quelques algorithmes populaires dans chaque type.

Dans l'étape qui suit l'étape de fouille de données on évalue le modèle généré par l'algorithme d'apprentissage par rapport aux objectifs définis dans la première étape, et cela en utilisant des techniques et des méthodes d'évaluation adaptées au type d'apprentissage. Par exemple, dans une classification supervisée on utilise généralement les métriques Rappel, Recall, et F-Measure pour évaluer le classificateur.

Finalement. La dernière étape du processus KDD consiste à utiliser concrètement le modèle d'apprentissage généré dans une application réel pour qu'on puisse extraire des nouvelles connaissances à partir de nouvelles données. Si on reprend l'exemple précédent de détection de spams, dans ce cas, une fois le modelé est validé, on peut l'intégrer directement dans une application ou un système de messagerie email. Ensuite, les emails seront filtrés en spam/ham en utilisant ce modèle.

1.6.1.1 Représentation des données

Dans le processus KDD (aussi dans la fouille de données), les données sont représentées dans un tableau où chaque ligne du tableau représente une instance, ou une observation, et chaque colonne représente une caractéristique, ou un attribut. Chaque case (i, j) dans le tableau représente la valeur de l'instance de la ligne i pour l'attribut de la colonne j. Tous les attributs ont un type, par exemple, on peut y'avoir un attribut "Nom" de type texte qui représente le nom d'un utilisateur, ou un attribut "Age" de type entier pour sauvegarder l'âge de l'utilisateur. Un ensemble d'instances représente un data set, ou un jeu de donnée.

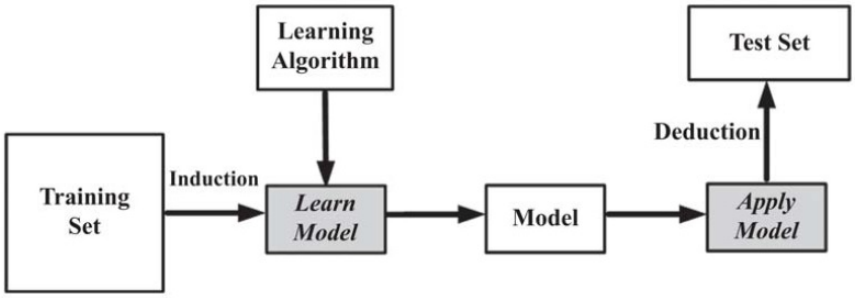
1.6.1.2 Apprentissage supervisé

La première catégorie des algorithmes d'apprentissage automatique est celle des algorithmes d'apprentissage supervisé. Dans cette catégorie d'algorithmes les valeurs de l'attribut qui représente la classe dans le dataset sont connues en avance et avant l'exécution de l'algorithme. Les données de ce type de dataset sont appelées données étiquetées (dataset étiqueté),

ou données d'entrainement. Les instances dans ces dataset sont en forme de paires (x, y) où x est un vecteur représentant les valeurs des attributs du dataset, et y est l'attribut de la classe, généralement un scalaire. Les algorithmes d'apprentissage supervisé visent à construire un modèle qui mappe l'entrée x à la sortie y. D'une manière grossière, la tâche est de trouver une fonction $m()$ de telle sorte que $m(x) = y$, c'est-à-dire pour chaque instance x fournie, la fonction doit trouver la valeur de y (la classe).

Les algorithmes d'apprentissage supervisé sont également fournies avec dataset non-étiqueté (données non-étiquetées), appelé aussi dataset de test, dans lequel les instances sont dans la forme de $(x, ?)$ et les valeurs y sont inconnues. Étant donné la fonction $m()$ générée par l'algorithme sur les données d'apprentissage et une instance x non-étiquetée (classe inconnue), on peut calculer $m(x)$, dont le résultat est la prédiction de l'étiquette pour l'instance x.

FIGURE 1.7: Diagramme d'apprentissage supervisé



Le processus d'apprentissage supervisé est représenté par le diagramme dans la figure 1.7. Il commence par un dataset d'apprentissage où les attributs et les classes sont connus. Un algorithme d'apprentissage supervisé est appliqué sur l'ensemble de ces données dans un processus appelé induction. Dans le processus d'induction, le modèle est généré. Le modèle mappe les valeurs des attributs aux valeurs de l'attribut de la classe. Ensuite, le modèle est utilisé sur un dataset de test dans lequel la valeur de l'attribut de classe est inconnue pour prédire les classe des instances de ce dataset, ce processus est appelé processus de déduction.

L'apprentissage supervisé peut être divisé en classification et régression. Lorsque l'attribut de la classe est discret, on l'appelle classification, et lorsque l'attribut de la classe est continu, c'est une régression tout simplement. Parmi les méthodes de classification les plus populaires on trouve les arbres de décision [Qui93], le naïve bayes [JL95], le k plus proches voisins (KNN) [AK91], Machine à vecteurs de support [Pla98 ; Kee+01 ; HT98], et les réseaux de neurones [FS98]. Et pour la régression on trouve aussi plusieurs méthodes telles que la régression linéaire [Wei05] et la régression logistique [CH92 ; LHF05 ; SFH05].

1.6.1.2.1 L'algorithme de Naïve Bayes

Parmi de nombreuses méthodes qui utilisent le théorème de Bayes, le classificateur bayésien naïf (NBC) est le plus simple. Étant donné deux variables aléatoires X et Y, le théorème de

Bayes stipule que :

$$P(X,Y) = \frac{P(X \mid Y)P(Y)}{P(X)} \tag{1.1}$$

Dans le NBC, Y représente la variable de classe et X représente les entités d'instance. Soit $X = (x_1; x_2; x_3; \ldots; x_m)$, où x_i représente la valeur de la caractéristique i. Supposons $y_1; y_2; y_3; \ldots; y_n$ représente les valeurs que l'attribut de classe Y peut prendre. Ensuite, la valeur d'attribut de classe de l'instance X peut être calculée en mesurant :

$$\arg\max_{y_i} P(y_i \mid X) \tag{1.2}$$

En basent sur le théorème de Bayes :

$$P(y_i, X) = \frac{P(X \mid y_i)P(y_i)}{P(X)} \tag{1.3}$$

Notez que $P(X)$ est constant et indépendant de y_i, donc nous pouvons ignorer le dénominateur de l'équation 1.3 en maximisant l'équation 1.2. Le NBC assume également une indépendance conditionnelle pour faciliter les calculs, c'est-à-dire, étant donné la valeur d'attribut de la classe, les autres attributs deviennent conditionnellement indépendants. Cette condition, bien qu'irréaliste, fonctionne bien dans la pratique et simplifie grandement le calcul.

1.6.1.2.2 L'algorithme des k plus proches voisins

Comme son nom l'indique, K-plus proches voisins ou kNN utilise les k instances les plus proches, appelées voisins, pour effectuer la classification. L'instance en cours de classement se voit attribuer la classe que la majorité de ses k voisins sont affectés. L'algorithme est décrit dans l'algorithme 1. Lorsque $k = 1$, la classe du voisin le plus proche est utilisée comme classe prédite pour l'instance à classer. Pour déterminer les voisins d'une instance, nous devons mesurer sa distance à toutes les autres instances en fonction d'une métrique de distance. Généralement, la distance Euclidienne est employée. Cependant, pour les espaces de plus grande dimension, la distance Euclidienne devient moins significative et d'autres mesures de

distance peuvent être utilisées.

Algorithme 1.1 : Algorithme de K-plus Proches Voisins kNN

Entrées : Instance i, Dataset d'attributs de valeur réelle, k (nombre de voisins), Mesure de distance d

Sorties : L'étiquette de classe pour l'instance i

1 Calculer les k voisins les plus proches de l'instance i en fonction de la mesure de distance d.
2 l = l'étiquette de classe majoritaire parmi les voisins de l'instance i. S'il y'a plus d'une étiquette majoritaire, on sélectionne une au hasard.
3 Classifier l'instance i en classe l.

1.6.1.3 Apprentissage non-supervisé

Contrairement aux algorithmes d'apprentissage supervisé qui nécessitent un dataset étiqueté pour faire l'apprentissage afin de construire un classificateur, les algorithmes d'apprentissage non-supervisé ne nécessitent pas forcément des données étiquetées. Ça signifie que le dataset ne contient pas nécessairement un attribut de classe. C'est la principale différence entre l'apprentissage supervisé et non supervisé. Le but principal des algorithmes d'apprentissage non-supervisé est de regrouper un dataset de telle sorte que les instances du même groupe sont plus similaires, cette tâche est appelée "clustering" ou "partitionnement de données".

Tout algorithme de clustering nécessite une mesure de distance. Les instances sont placées dans différents groupes en fonction de leur distance par rapport aux autres instances. La mesure de distance la plus populaire pour les valeurs continues est la distance Euclidienne :

$$d(X,Y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots (x_n - y_n)^2} = \sqrt{\sum_{i}^{n} (x_i - y_i)^2} \tag{1.4}$$

Où $X = (x_1; x_2; \ldots; x_n)$ et $Y = (y_1; y_2; \ldots; y_n)$ sont des vecteurs n-dimensionnelles des caractéristiques dans R^n. Une liste de quelques mesures de distance couramment utilisées est fournie dans le Tableau 1.2.

TABLE 1.2: Mesures de distance

Nom de la mesure	Formule
Minkowsky	$\sqrt[p]{\sum_i^n \lvert x_i - y_i \rvert^p}$
Manhattan ($p = 1$)	$\sum_i^n \lvert x_i - y_i \rvert$
Euclidienne ($p = 2$)	$\sqrt{\sum_i^n (x_i - y_i)^2}$
Chebychev ($p = \infty$)	$\max_b^a \sum_i^n \lvert x_i - y_i \rvert$
Hamming	$\sum_i^n \lvert x_i xor y_i \rvert$ Où $X = (x_1; x_2; \ldots; x_n)$ et $Y = (y_1; y_2; \ldots; y_n)$ sont des vecteurs binaires
Jaccard	$\frac{\lvert X \cap Y \rvert}{\lvert X \cup Y \rvert}$ Où X et Y sont des ensembles de valeurs discrètes

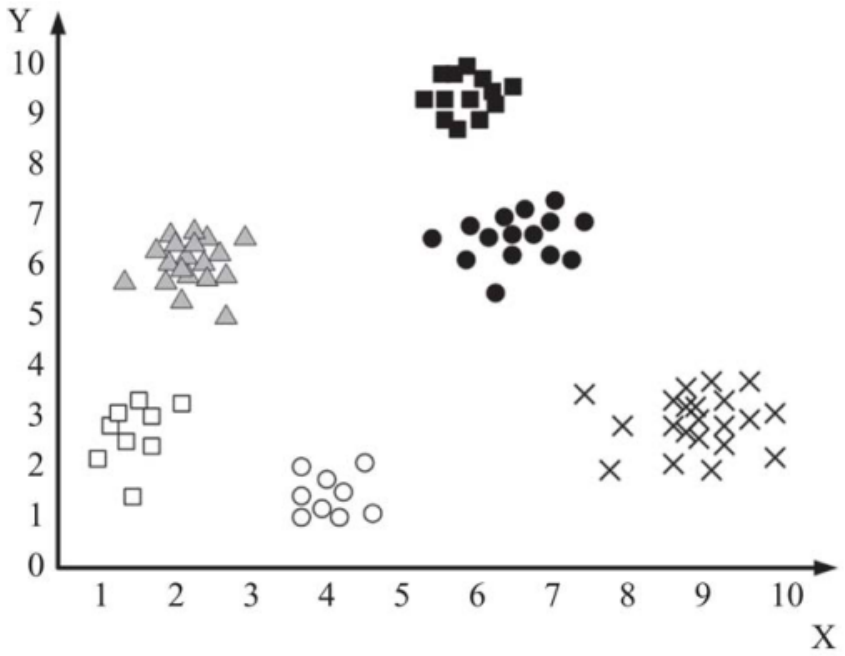
Une fois qu'une mesure de distance est sélectionnée, les instances sont groupées à l'aide de cette mesure. Les clusters sont généralement représentées par des notations compactes et abstraites. Les "centroïdes de cluster" sont un exemple courant de cette notation abstraite. Enfin, les clusters sont évalués. Il y'a encore un large débat sur la question de l'évaluation des clusters en raison du manque d'étiquettes de clusters dans l'apprentissage non-supervisé.

Il existe plusieurs types d'algorithmes de clustering. Dans cette section, nous discutons les algorithmes de clustering par partitionnement, en particulier l'algorithme de k-moyennes (k-means) qui est un des algorithmes de clustering les plus utilisés.

1.6.1.3.1 L'algorithme de K-Means

Les algorithmes de clustering par partitionnement divisent l'ensemble de données en un ensemble de clusters. En d'autres termes, chaque instance est affectée à un seul cluster et aucune instance ne reste non attribuée aux clusters. K-means [Mac+67] est un exemple bien connu de cette catégorie d'algorithmes. Le résultat de l'application de l'algorithme K-means ($k = 6$) sur un échantillon de données est montrée dans la Figure 1.8. Dans cette figure, l'ensemble de données comporte deux attributs et les instances peuvent être visualisées dans un espace à deux dimensions. Les instances sont représentées à l'aide de symboles représentant le cluster auquel elles appartiennent. Le pseudocode de l'algorithme K-means est fourni dans l'algorithme 2.

FIGURE 1.8: Résultat de l'application de K-Means sur un exemple de dataset.



L'algorithme commence par k centroïdes initiaux. En pratique, ces centroïdes sont des instances choisies aléatoirement de l'ensemble de données. Ces instances initiales forment l'ensemble initial de k clusters. Ensuite, nous assignons chaque instance à l'un de ces groupes en fonction de sa distance au centroïde de chaque groupe. Le calcul des distances entre les instances et les centroïdes dépend du choix de mesure de distance. La distance Euclidienne est la mesure de distance la plus utilisée.

Algorithme 1.2 : Algorithme de K-means

Entrées : Un dataset d'attributs de valeur réelle, k (nombre de clusters)
Sorties : Un regroupement de données en k clusters

1 Considérons k instances aléatoires dans l'espace de données comme les centroïdes des clusters initial.;
2 **tant que** *Les centroïdes n'ont pas convergé* **faire**
3 Affectez chaque instance au cluster ayant le centroïde le plus proche.;
4 Si toutes les instances ont été affectées, recalculez les centroïdes des clusters en utilisant les instances dans chaque cluster.
5 **fin**

Après avoir assigné toutes les instances à un cluster, les centroïdes sont recalculés en prenant la moyenne de toutes les instances à l'intérieur des clusters (d'où le nom k-moyennes). Cette procédure est répétée en utilisant les centroïdes nouvellement calculés. Notez que cette procédure est répétée jusqu'à la convergence. Le critère le plus commun pour déterminer la convergence est de vérifier si les centroïdes ne changent plus. Cela équivaut à des affectations

de regroupement des instances de données qui se stabilisent. En pratique, l'exécution de l'algorithme peut être arrêtée lorsque la distance Euclidienne entre les centroïdes en deux étapes consécutives est bornée par une petite positive ϵ. Comme alternative, les implémentations de k-means essaient de minimiser une fonction objective. Une fonction objective bien connue dans ces implémentations est l'erreur de distance au carré :

$$\sum_{i}^{k} \sum_{j=1}^{n(i)} \parallel x_j^i - c_i \parallel^2 \tag{1.5}$$

Où x_j^i est la $ji\grave{e}me$ instance du cluster i, $n(i)$ est le nombre d'instances dans le cluster i, et c_i est le centroïde du cluster i. Le processus s'arrête lorsque la différence entre les valeurs de la fonction objective de deux itérations consécutives de l'algorithme k-means est limitée par une petite valeur.

Notez que K-means est très sensible aux k centroïdes initiaux, et différents résultats de clustering peuvent être obtenus sur un seul ensemble de données en fonction des k centroïdes initiaux. Ce problème peut être mitigé en exécutant K-means plusieurs fois et en sélectionnant l'affectation de cluster qui est observée le plus souvent ou est plus souhaitable en fonction d'une fonction objective, telle que l'erreur carré. Puisque k-means suppose que les instances appartenant au même cluster sont celles qui ont trouvé le centroïde du cluster le plus proche que les autres centroïdes de l'ensemble de données, on peut supposer que toutes les instances d'un cluster tombent dans une hypersphère, avec le centroïde étant son centre. Le rayon de cette hypersphère est défini en fonction de l'instance la plus éloignée de ce cluster. Si les clusters qui doivent être extraites sont non sphériques, K-means a des problèmes pour les détecter. Ce problème peut être résolu par une étape de prétraitement dans laquelle une transformation est effectuée sur l'ensemble de données pour résoudre ce problème.

1.6.2 Plateformes d'analyse des big Data

1.6.2.1 Apache Hadoop

Hadoop a été créé par Doug Cutting, le créateur d'Apache Lucene [Apab], la bibliothèque de recherche de texte largement utilisée. Hadoop a ses origines dans Apache Nutch [Nut], un moteur de recherche web open source, lui-même une partie du projet Lucene.

Construire un moteur de recherche web à partir de zéro était un objectif ambitieux car non seulement le logiciel requis pour explorer et indexer les sites Web est complexe à écrire, mais il est également difficile de fonctionner sans une équipe dédiée aux opérations, car il y'a beaucoup de parties mobiles. C'est aussi chez Mike Cafarella et Doug Cutting, qui ont estimé qu'un système supportant un index d'un milliard de dollars coûterait environ un demi-million de dollars en matériel, avec un coût de fonctionnement mensuel de 30 000 $ .Néanmoins, ils croyaient que c'était un objectif valable, ouvrirait et démocratiserait finalement les algorithmes des moteurs de recherche.

Nutch a été lancé en 2002, un système de recherche qui a rapidement émergé. Cependant, ils ont réalisé que leur architecture ne serait pas à la hauteur des milliards de pages sur le Web. L'aide était à portée de main avec la publication en 2003 d'un article décrivant l'architecture du système de fichiers distribué de Google, appelé GFS, qui était utilisé dans la production de Google. GFS, ou quelque chose du genre, permettrait de résoudre leurs besoins de stockage de fichiers générés dans le cadre du processus d'exploration et d'indexation Web. En particulier, GFS libérerait du temps pour des tâches administratives telles que la gestion des nœuds de stockage. En 2004, ils se sont lancés dans l'écriture d'une implémentation open source, le NFS Distributed Filesystem (NDFS).

En 2004, Google a publié l'article qui a introduit MapReduce dans le monde. Au début de l'année 2005, les développeurs de Nutch avaient une implémentation MapReduce opérationnelle dans Nutch et au milieu de l'année, tous les principaux algorithmes de Nutch avaient été portés pour fonctionner en utilisant MapReduce et NDFS.

NDFS et l'implémentation de MapReduce dans Nutch étaient applicables au-delà du domaine de la recherche, et en février 2006, ils ont quitté Nutch pour former un sous-projet indépendant de Lucene appelé Hadoop. À peu près au même moment, Doug Cutting a rejoint Yahoo, qui a fourni une équipe dédiée et les ressources nécessaires pour transformer Hadoop en un système fonctionnant à l'échelle du Web. Cela été démontré en Février 2008 lorsque Yahoo a annoncé que son index de recherche de production était généré par un cluster Hadoop à 10 000 cœurs.

En janvier 2008, Hadoop a réalisé son propre projet de haut niveau chez Apache, confirmant son succès et sa communauté diversifiée et active. A cette époque, Hadoop était utilisé par de nombreuses autres sociétés que Yahoo, comme Last.fm, Facebook et le New York Times.

Dans un exploit bien publicité, le New York Times a utilisé le service du cloud computing EC2 d'Amazon pour parcourir quatre téraoctets (4 TO) d'archives numérisées à partir du papier, les convertissant en fichiers PDF pour le Web. Le traitement a duré moins de 24 heures en utilisant 100 machines, et le projet n'aurait probablement pas été lancé sans la combinaison du modèle de paiement à l'heure d'Amazon (qui a permis au NYT d'avoir accès à un grand nombre de machines pendant une courte période) et de la facilité d'utilisation de Hadoop.

En avril 2008, Hadoop a battu un record mondial pour devenir le système le plus rapide pour trier un téraoctet de données. En cours d'exécution sur un cluster de 910 nœuds, Hadoop a trié un téraoctet en 209 secondes, battant le vainqueur de l'année précédente de 297 secondes. En novembre de la même année, Google a signalé que sa mise en œuvre MapReduce a trié un téraoctet en 68 secondes.

Depuis lors, Hadoop a connu une adoption rapide et généralisée des entreprises. Le rôle de Hadoop en tant que plateforme de stockage et d'analyse à usage général pour le Big Data a été reconnu par l'industrie, et cela se reflète dans le nombre de produits qui utilisent ou intègrent Hadoop d'une manière ou d'une autre. Il existe des distributions Hadoop provenant

des grands fournisseurs d'entreprise établis, notamment EMC, IBM, Microsoft et Oracle, ainsi que des sociétés Hadoop spécialisées telles que Cloudera, Hortonworks et MapR.

Apache Hadoop et l'écosystème Hadoop

Bien que Hadoop est connu surtout pour MapReduce et son système de fichiers distribué (HDFS, renommé NDFS), le terme est également utilisé pour une famille de projets connexes qui relèvent de l'infrastructure de calcul distribué et de traitement de données à grande échelle.

Tous les principaux projets de l'écosystème Hadoop sont hébergés par Apache Software Foundation, qui fournit un support pour une communauté de projets de logiciels open source, y compris le serveur HTTP d'origine à partir duquel il tire son nom. À mesure que l'écosystème Hadoop se développe, d'autres projets apparaissent, pas nécessairement hébergés chez Apache, qui fournissent des services complémentaires à Hadoop ou s'appuient sur le noyau pour ajouter des abstractions de plus haut niveau.

Les projets Hadoop sont brièvement décrits ici :

Common : Un ensemble de composants et d'interfaces pour les systèmes de fichiers distribués et les E/S générales (sérialisation, Java RPC, structures de données persistantes).

Avro : un système de sérialisation pour un RPC multilingue efficace et un stockage de données persistant.

MapReduce : Un modèle de traitement de données distribuées et un environnement d'exécution qui s'exécute sur un grand nombre de machines .

HDFS : Un système de fichiers distribué qui fonctionne sur de grands nombre de machines.

Pig : Langage de flux de données et environnement d'exécution pour l'exploration de très grands ensembles de données. Pig fonctionne sur les clusters HDFS et MapReduce.

Hive : un entrepôt de données distribué. Hive gère les données stockées dans HDFS et fournit un langage d'interrogation basé sur SQL pour interroger les données.

HBase : Une base de données distribuée, orientée colonne. HBase utilise HDFS pour son stockage sous-jacent et prend en charge les calculs par lots utilisant MapReduce et les requêtes ponctuelles (lectures aléatoires).

ZooKeeper : un service de coordination distribué et hautement disponible. ZooKeeper fournit des primitives telles que des verrous distribués pouvant être utilisés pour créer des applications distribuées.

Sqoop : Un outil pour le transfert en masse efficace de données entre des entrepôts de données structurés (tels que des bases de données relationnelles) et HDFS.

Oozie : service d'exécution et de planification des workflows de travaux Hadoop (y compris les travaux Map-Reduce, Pig, Hive et Sqoop).

1.6.2.2 Apache Spark

Apache Spark est une plateforme de cluster computing conçue pour être rapide et polyvalente.

Sur le plan de la vitesse, Spark étend le modèle populaire MapReduce pour prendre en charge plus de types de calculs, y compris les requêtes interactives et le traitement de flux. La rapidité est importante dans le traitement de grands ensembles de données, car cela signifie la différence entre l'exploration interactive des données et l'attente de minutes ou d'heures. L'une des principales fonctionnalités de Spark pour la vitesse est la possibilité d'exécuter des calculs en mémoire, mais le système est également plus efficace que MapReduce pour les applications complexes exécutées sur disque.

D'un point de vue général, Spark est conçu pour couvrir un large éventail de charges de travail qui nécessitaient auparavant des systèmes distribués séparés, notamment des applications par lots, des algorithmes itératifs, des requêtes interactives et la diffusion en continu. En prenant en charge ces tâches de travail dans le même moteur, Spark facilite et permet de combiner différents types de traitement, ce qui est souvent nécessaire dans les pipelines d'analyse de données en production.

Spark est conçu pour être hautement accessible, offrant des API simples en Python, Java, Scala et SQL, ainsi que de riches bibliothèques intégrées. Il s'intègre également étroitement avec d'autres outils Big Data. En particulier, Spark peut s'exécuter dans des clusters Hadoop et accéder à n'importe quelle source de données Hadoop.

Une brève histoire de Spark

Spark est un projet open source qui a été construit et maintenu par une communauté de développeurs prospère et diversifiée. Spark a commencé en 2009 comme un projet de recherche dans l'UC Berkeley RAD Lab (devenu plus tard AMPLab.) Les chercheurs du laboratoire travaillaient auparavant sur Hadoop Map-Reduce et ont observé que MapReduce était inefficace pour les tâches informatiques itératives et interactives. Ainsi, dès le début, Spark a été conçu pour être rapide pour les requêtes interactives et les algorithmes itératifs, apportant des idées comme le support de stockage en mémoire et la récupération efficace des erreurs. Des articles de recherche ont été publiés sur Spark lors des conférences. Certains des premiers utilisateurs de Spark étaient d'autres groupes à l'UC Berkeley, y compris des chercheurs en apprentissage automatique tels que le projet Mobile Millennium, qui utilisait Spark pour surveiller et prédire la congestion du trafic dans la baie de San Francisco. En très peu de temps, plusieurs organisations externes ont commencé à utiliser Spark, et aujourd'hui, plus de 50 organisations l'utilise, et des dizaines d'entre eux parlent de leurs cas d'utilisation lors des événements communautaires Spark tels que Spark Meetups et Spark Summit. En plus d'UC Berkeley, les principaux contributeurs à Spark comprennent Databricks, Yahoo, et Intel.

En 2011, l'AMPLab a commencé à développer des composants de plus haut niveau sur Spark, tels que Shark (Hive on Spark) et Spark Streaming. Ces composants et d'autres sont

parfois désignés sous le nom de Berkeley Data Analytics Stack (BDAS).

Spark : une pile unifiée

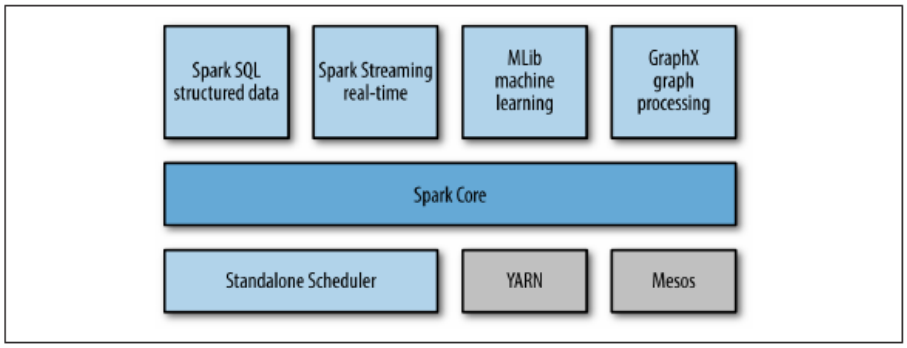
Le projet Spark contient plusieurs composants étroitement intégrés. À sa base, Spark est un «moteur de calcul» qui est responsable de la planification, de la distribution et de la surveillance des applications consistant en de nombreuses tâches de calcul sur de nombreux postes de travail ou un cluster de calcul. Parce que le moteur de base de Spark est à la fois rapide et polyvalent, il alimente plusieurs composants de plus haut niveau spécialisés pour diverses tâches, telles que le SQL ou l'apprentissage automatique. Ces composants sont conçus pour interopérer étroitement, nous permettant de les combiner comme des bibliothèques dans un projet logiciel.

Une philosophie d'intégration étroite présente plusieurs avantages. Premièrement, toutes les bibliothèques et les composants de plus haut niveau de la pile bénéficient d'améliorations dans les couches inférieures. Par exemple, lorsque le moteur principal de Spark ajoute une optimisation, les bibliothèques SQL et d'apprentissage automatique s'accélèrent automatiquement. Deuxièmement, les coûts associés à l'exécution de la pile sont minimisés, car au lieu d'exécuter 5 à 10 systèmes logiciels indépendants, une organisation doit en exécuter une seule. Ces coûts comprennent le déploiement, la maintenance, les tests, le support et autres. Cela signifie également qu'à chaque fois qu'un nouveau composant est ajouté à la pile Spark, chaque entreprise utilisant Spark pourra immédiatement essayer ce nouveau composant.

Enfin, l'un des plus grands avantages d'une intégration étroite est la possibilité de créer des applications qui combinent de manière transparente différents modèles de traitement. Par exemple, dans Spark, vous pouvez écrire une application qui utilise l'apprentissage automatique pour classer les données en temps réel lorsqu'elles sont ingérées à partir de sources de diffusion. Simultanément, les analystes peuvent interroger les données résultantes, également en temps réel, via SQL (par exemple, pour joindre les données avec des fichiers journaux non structurés). En outre, des ingénieurs de données et des spécialistes des données plus expérimentés peuvent accéder aux mêmes données via le Shell Python pour une analyse ad hoc. D'autres peuvent accéder aux données dans des applications batch autonomes. Pendant tout ce temps, l'équipe informatique doit maintenir un seul système.

Ici, nous présenterons brièvement chacun des composants de Spark, montré dans la Figure 1.9.

FIGURE 1.9: La pile Spark



Spark Core
Spark Core contient les fonctionnalités de base de Spark, notamment des composants pour la planification des tâches, la gestion de la mémoire, la récupération des erreurs, l'interaction avec les systèmes de stockage, etc. Spark Core héberge également l'API qui définit les ensembles de données réparties résilientes (RDD), qui sont la principale abstraction de programmation de Spark. Les RDD représentent une collection d'éléments répartis sur de nombreux nœuds de calcul pouvant être manipulés en parallèle. Spark Core fournit de nombreuses API pour construire et manipuler ces collections.

Spark SQL
Spark SQL est le package de Spark pour travailler avec des données structurées. Il permet d'interroger les données via SQL ainsi que la variante Apache Hive de SQL, appelée HQL (Hive Query Language), et prend en charge de nombreuses sources de données, notamment les tables Hive, Parquet et JSON. En plus de fournir une interface SQL à Spark, Spark SQL permet aux développeurs de mélanger les requêtes SQL avec les manipulations de données programmées supportées par les RDD en Python, Java et Scala, au sein d'une même application, combinant ainsi SQL et analytique complexe. Cette intégration étroite avec l'environnement informatique riche fourni par Spark rend Spark SQL différent de tout autre outil d'entrepôt de données open source. Spark SQL a été ajouté à Spark dans la version 1.0.

Shark était un ancien projet SQL-on-Spark de l'Université de Californie à Berkeley, qui a modifié Apache Hive pour qu'il fonctionne sur Spark. Il a maintenant été remplacé par Spark SQL pour fournir une meilleure intégration avec le moteur Spark et les API de langage.

Spark Streaming
Spark Streaming est un composant Spark qui permet le traitement des flux de données en direct. Les exemples de flux de données comprennent les fichiers journaux générés par les serveurs Web de production ou les files d'attente de messages contenant des mises à jour de statut publiées par les utilisateurs d'un service Web. Spark Streaming fournit une API pour manipuler les flux de données qui correspondent étroitement à l'API RDD du Spark Core, facilitant l'apprentissage du projet par les programmeurs et leur déplacement entre les applications manipulant les données stockées en mémoire, sur disque ou arrivant en temps réel. Sous son API, Spark Streaming a été conçu pour offrir le même degré de tolérance aux

pannes, de débit et d'évolutivité que Spark Core.

MLlib
Spark est livré avec une bibliothèque contenant la fonctionnalité d'apprentissage automatique (ML), appelée MLlib. MLlib fournit plusieurs types d'algorithmes d'apprentissage automatique, y compris la classification, la régression, la mise en grappe et le filtrage collaboratif, ainsi que des fonctionnalités de support telles que l'évaluation de modèle et l'importation de données. Il fournit également des primitives ML de niveau inférieur, notamment un algorithme générique d'optimisation de descente de gradient. Toutes ces méthodes sont conçues pour évoluer à travers un cluster.

GraphX
GraphX est une bibliothèque pour manipuler des graphes (par exemple, le graphe d'un réseau social) et effectuer des calculs en parallèle. Comme Spark Streaming et Spark SQL, GraphX étend l'API Spark RDD, ce qui nous permet de créer un graphe orienté avec des propriétés arbitraires attachées à chaque sommet et bord. GraphX fournit également divers opérateurs pour manipuler des graphes (par exemple, subgraph et mapVertices) et une bibliothèque d'algorithmes de graphes communs (par exemple, le PageRank et le comptage de triangles).

Gestionnaires de cluster
Sous le capot, Spark est conçu pour évoluer efficacement d'un à plusieurs milliers de nœuds de calcul. Pour atteindre cet objectif tout en maximisant la flexibilité, Spark peut exécuter plusieurs gestionnaires de cluster, y compris Hadoop YARN, Apache Mesos et un gestionnaire de cluster simple inclus dans Spark lui-même appelé le planificateur autonome. Si vous installez simplement Spark sur un ensemble de machines vide, le planificateur autonome fournit un moyen facile de démarrer ; Cependant, si vous disposez déjà d'un cluster Hadoop YARN ou Mesos, le support de Spark pour ces gestionnaires de cluster permet également à vos applications de les exécuter.

Qui utilise Spark, et pour quoi

Parce que Spark est un cadre général pour l'informatique en grappe, il est utilisé pour une gamme variée d'applications. On peut définir deux groupes d'utilisateurs : les scientifiques de données et les ingénieurs. Examinons de plus près chaque groupe et comment il utilise Spark. Sans surprise, les cas d'utilisation typiques diffèrent entre les deux, mais nous pouvons grossièrement les classer en deux catégories, la science des données et les applications de données. Bien sûr, ce sont des disciplines imprécises et des modèles d'utilisation, et beaucoup de gens ont des compétences des deux, jouant parfois le rôle de l'enquêteur de données, puis «changer de chapeau» et écrire une application de traitement de données durcie. Néanmoins, il peut être éclairant de considérer séparément les deux groupes et leurs cas d'utilisation respectifs.

Tâches de science des données
La science des données, discipline émergente depuis quelques années, est centrée sur l'analyse de données. Bien qu'il n'y ait pas de définition standard, un chercheur de données est pour nous une personne dont la tâche principale est d'analyser et de modéliser les données. Les

spécialistes des données peuvent avoir de l'expérience en SQL, en statistiques, en modélisation prédictive (apprentissage automatique) et en programmation, généralement en Python, Matlab ou R. Les scientifiques ont également l'expérience des techniques nécessaires pour transformer les données en formats pouvant être analysés. (Parfois appelé «conflit de données»).

Les scientifiques utilisent leurs compétences pour analyser des données dans le but de répondre à une question ou de découvrir des idées. Souvent, leur flux de travail implique une analyse ad hoc, ils utilisent donc des Shells interactifs (par opposition à la création d'applications complexes) qui leur permettent de voir les résultats des requêtes et des extraits de code en un minimum de temps. La vitesse de Spark et les API simples brillent dans ce but, et ses bibliothèques intégrées signifient que de nombreux algorithmes sont disponibles dès la sortie de la boîte.

Spark prend en charge les différentes tâches de la science des données avec un certain nombre de composants. Le Shell Spark facilite l'analyse interactive des données à l'aide de Python ou de Scala. Spark SQL dispose également d'un Shell SQL séparé qui peut être utilisé pour l'exploration de données à l'aide de SQL ou Spark SQL peut être utilisé dans le cadre d'un programme Spark standard ou dans le Shell Spark. L'apprentissage automatique et l'analyse des données sont pris en charge par les bibliothèques MLLib. En outre, il est possible d'appeler des programmes externes dans Matlab ou R. Spark permet aux spécialistes des données de résoudre des problèmes avec des tailles de données plus importantes qu'avec des outils comme R ou Pandas.

Parfois, après la phase d'exploration initiale, le travail d'un data scientist sera «produit» ou étendu, durci (c'est-à-dire toléré aux pannes) et réglé pour devenir une application de traitement des données de production, elle-même composante d'une application commerciale. Par exemple, l'enquête initiale d'un data scientist peut conduire à la création d'un système de recommandation de production intégré dans une application Web et utilisé pour générer des suggestions de produits aux utilisateurs. Souvent, c'est une personne ou une équipe différente qui dirige le processus de production du travail des informaticiens, et cette personne est souvent un ingénieur.

Applications de traitement de données
L'autre cas d'utilisation principal de Spark peut être décrit dans le contexte de la personne de l'ingénieur. Pour notre propos, nous considérons les ingénieurs comme une grande classe de développeurs de logiciels qui utilisent Spark pour construire des applications de traitement de données de production. Ces développeurs ont généralement une compréhension des principes de l'ingénierie logicielle, tels que l'encapsulation, la conception d'interface et la programmation orientée objet. Ils ont souvent un diplôme en informatique. Ils utilisent leurs compétences en ingénierie pour concevoir et construire des systèmes logiciels qui mettent en œuvre un cas d'utilisation métier.

Pour les ingénieurs, Spark fournit un moyen simple de paralléliser ces applications entre les clusters et cache la complexité de la programmation des systèmes distribués, de la communication réseau et de la tolérance aux pannes. Le système leur donne suffisamment de contrôle

pour surveiller, inspecter et régler les applications tout en leur permettant d'exécuter rapidement les tâches courantes. La nature modulaire de l'API (basée sur le passage de collections d'objets distribuées) facilite le travail dans des bibliothèques réutilisables et le test en local.

Les utilisateurs de Spark choisissent de l'utiliser pour leurs applications de traitement de données, car il offre une grande variété de fonctionnalités, est facile à apprendre et à utiliser, et est mature et fiable.

1.7 Conclusion

Nous avons présenté dans ce chapitre un état de l'art sur les big data. Nous avons commencé par requérir et présenter les définitions les plus populaires de ce terme, et nous avons aussi donné notre propre définition des big data. Ensuite, nous avons détaillé les quatre étapes du processus big data notamment la génération, l'acquisition, le stockage, et l'analyse de données. La première étape de ce processus est la génération de données, avant tout traitement on doit d'abord collecter les données à partir des sources telles que les objets connectés ou les réseaux sociaux. Dans la plupart des cas, ces données collectées nécessitent un prétraitement spécifique avant qu'elles soient stockées, des prétraitements tels que le nettoyage sont souvent appliqués. La deuxième étape consiste à stocker les données en utilisant sois des techniques traditionnelles comme les bases de données relationnelles sois des nouvelles techniques adaptées aux tailles et types des données big data. Les techniques les plus utilisées aujourd'hui sont les bases de données NoSQL et le stockage en nuage. La dernière étape du processus big data est la plus importante, elle s'agit de l'analyse de données. L'analyse des données consiste à appliquer les méthodes de fouille de données sur les données big data afin d'extraire des informations utiles et des connaissances cachées. Ces méthodes sont appliquées en utilisant des techniques spécifiques comme le MapReduce et avec des outils comme Hadoop et Apache Spark.

Chapitre 2

Analyse des réseaux sociaux

Sommaire

2.1 Introduction

Avec l'émergence des médias sociaux, le Web est devenue un espace vivant et dynamique dans lequel des milliards d'individus à travers le monde entier interagissent, partagent, publient et mènent de nombreuses activités quotidiennes. Les médias sociaux permettent aux gens d'être connectés les uns avec les autres n'importe où et à tout moment, ce qui nous permet d'observer le comportement humain d'une façon sans précédent. Les médias sociaux nous offrent des occasions d'or pour comprendre les individus à une grande échelle et d'extraire des modèles de comportement humain. En comprendraient mieux les individus, nous

pouvons concevoir des nouveaux systèmes informatiques évalués adaptés aux besoins des individus qui leur serviront mieux. Ce nouveau monde des médias sociaux n'a pas de frontières géographiques et ne cesse pas de créer des énormes quantités de données.

Malheureusement, les données sur les médias sociaux sont significativement différentes des données traditionnelles dont nous connaissons déjà les exploiter. Généralement, les données générées par les utilisateurs sont bruitées et non-structurées, avec des relations sociales complexes comme les liens d'amitiés. Ce nouveau type de données nécessite de nouvelles approches d'analyse de données qui peuvent combiner les théories sociales avec les méthodes de calcul statistique et les techniques de fouille de données. Cela a conduit à l'apparition d'un nouveau domaine interdisciplinaire : l'analyse des réseaux sociaux.

L'analyse des réseaux sociaux, en anglais : social networks analysis (SNA) est un domaine multidisciplinaire qui s'intéresse en l'étude approfondie des réseaux sociaux, et cela en fournissent des algorithmes et des méthodes pour caractériser la structure des réseaux sociaux, les positions stratégiques dans ces réseaux, les sous-réseaux spécifiques et les décompositions de personnes et d'activités. Ce domaine a suscité beaucoup d'attention ces dernières années à cause de l'éclatement des données sociales sur le Web qui a conduit à des réseaux sociaux très grands qui n'ont jamais existé.

Dans ce chapitre, nous dériverons d'abord les réseaux sociaux en ligne, présentant leurs caractéristiques et les raisons de la croissance de leur popularité. Ensuite, nous examinerons les méthodes et les modèles traditionnels qui sont utilisés pour construire et représenter les réseaux sociaux. Enfin, nous fournissons un aperçu des différentes métriques et algorithmes pour l'analyse des réseaux sociaux en ligne.

2.2 Les réseaux sociaux

Nous présenterons ici les réseaux sociaux en général et leurs caractéristiques avant de passer aux réseaux sociaux en ligne où nous les définiront et nous discuterons aussi leur historique. Ensuite, nous discuterons les différentes représentations des réseaux sociaux et nous donnerons quelques définitions liées à ces représentations.

Définition : Les réseaux sociaux

Il n'y a pas une mais plusieurs définitions de réseaux sociaux qui varient selon le domaine concerné, par exemple : la théorie des graphes définit un réseau social comme un graphe dont les nœuds représentent les acteurs sociaux ou les objets et les arêtes représentent les relations entre ces acteurs et objets. Cependant, un réseau social est plus qu'un simple graphe, un réseau contient d'autres informations supplémentaires sur les acteurs et les liens. Ce sont des caractéristiques d'acteurs comme par exemple : le prénom et le sexe d'une personne.

Une autre définition très utilisée issue de la sociologie qui définit un réseau social comme un ensemble d'acteurs qui sont liés par des relations sociales, ces acteurs peuvent être des individus, des organisations ou des groupes d'acteurs. Il existe un grand nombre de relations entre les acteurs d'un réseau social, on peut classifier ces relations en trois grandes catégories

comme suit :

1. Relations explicites entre humains
2. Interactions entre acteurs
3. Affiliations entre acteurs

Les relations explicites incluent toutes les relations que nous pouvons définir entre personnes dans la vie réel par exemple : parent, frère, cousin, amitié, simple connaissance, collègue, ...etc, entre personnes et organisations par exemple : membre, employé, ...etc, et à l'intérieur des organisations par exemple : propriétaire, gestionnaire, ...etc.

Les interactions représentent tous les échanges qui pourraient être observés entre des acteurs tels qu'une discussion, une collaboration, une réunion ou toute action impliquant au moins deux acteurs. Certaines interactions impliquent activement tous les acteurs concernés comme une discussion synchrone. D'autres sont initiées par certains acteurs et ciblent d'autres acteurs comme un seul message qui a un expéditeur et un destinataire.

Les affiliations correspondent à toute similitude entre les acteurs qui les relie comme par exemple : partager les mêmes attributs, les mêmes intérêts, les mêmes activités, les mêmes objets ou les mêmes organisations.

Définition : Les réseaux sociaux en ligne

On peut définir un réseau social en ligne comme un service web qui permet aux utilisateurs de créer ou de construire un profil publique ou semi-publique dans un environnement fermé, de créer des liens explicites avec d'autres utilisateurs (amitié, suivi, ...) ou d'autres éléments du système, de naviguer dans le réseau social en parcourant les liens et les profils d'autres utilisateurs.

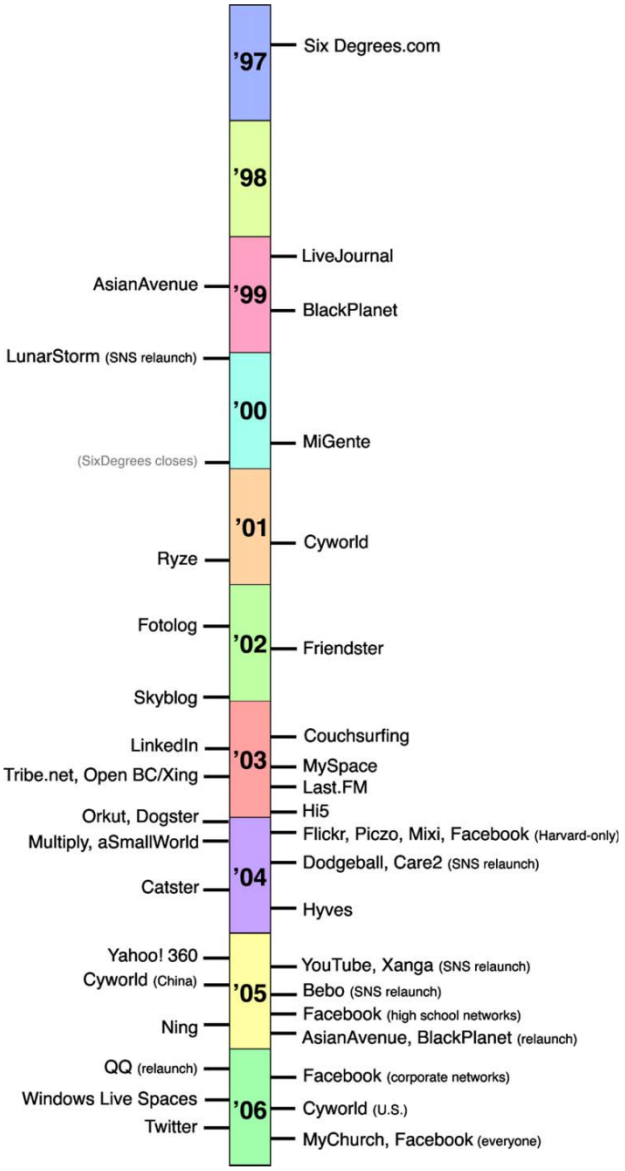
2.3 Historique des réseaux sociaux en ligne

Maintenant, nous donnons une brève histoire des réseaux sociaux en ligne. L'histoire a commencée en 1995 avec l'apparition de Classmates.com [Cla] qui est considéré comme le premier site web à fournir aux leurs utilisateurs la possibilité de se faire connecter entre eux. Classmates.com a été créé pour un objectif bien précis : aider les utilisateurs du site à se reconnecter avec leurs anciens camarades d'école, collège, lycée ou même avec des anciens collègues de travail. Ce site a permis aussi aux utilisateurs de consulter les réseaux d'autres anciens élèves d'école ou lycée. Malgré l'idée originale derrière le site web, Classmates.com avait quelques inconvénients, par exemple, le site n'a pas permis aux utilisateurs de faire des relations directes avec d'autres utilisateurs, au lieu de cela, il permettait aux utilisateurs de se lier entre eux uniquement via les écoles auxquelles ils avaient assisté.

En 1997, le site SixDegrees.com [Deg] a fait son apparition sur le web, ce site qui a permis à ses utilisateurs de créer des profils, se connecter directement à des personnes sur le réseau, et d'échanger des messages avec leurs contacts, a été considéré comme le premier vrai réseau social en ligne ayant toutes les caractéristiques d'un tel système et qui répond à la définition

donnée en haut. Bien que, le site web à réussi d'attirer des millions d'utilisateurs, il n'a pas réussi commercialement et suite à ça il été fermé en 2000 avec de grandes pertes. L'histoire des réseaux sociaux n'a pas arrêté ici, au contraire, plusieurs nouveaux réseaux sociaux en ligne ont vu le jour comme : LiveJournal [Liv], AsianAvenue, BlackPlanet [Bla], CyWorld [Cyw], et Ryze [Ryz]. Le site web Rize.com a été lancé en 2001 par Adrian Scott pour aider les gens à créer et enrichir leurs réseaux professionnels, le site offre des abonnements gratuits et payants aux utilisateurs souhaitent rejoindre le site web. Ce site est considéré aujourd'hui comme le premier d'une nouvelle génération de réseaux sociaux professionnels.

FIGURE 2.1: Chronologie des dates de création des sites de réseaux sociaux en ligne les plus populaires



A partir des années 2000, l'accès à Internet commence à être plus accessible et plus facile aux populations du monde entier, ce qui a rendu les sites des réseaux sociaux plus populaire et les gens plus connectés. Friendster [Fri] est un meilleur exemple des réseaux sociaux qui ont cartonné dans les débuts des années 2000, ce site web a été lancé en 2002 par un des membres de Ryze.com comme un complément au ce réseau social. En plus des fonctionnalités basiques d'un réseau social, Friendster a permis aux leurs utilisateurs de partager du contenu multimédia en ligne avec leurs contacts ainsi les utilisateurs peuvent partager des photos, vidéos, et commentaires. Toutes ces fonctionnalités ont rendu le site web très populaire en un peu de temps, en réalité le site a connu une croissance rapide dans le nombre des utilisateurs durant les premières années. Avec la croissance de sa popularité, Friendster commence à affronter des difficultés techniques et sociales, d'un part les serveurs et les bases de données ont été pas bien conçu pour gérer un grand flux de données sur le site web, et le site a commencé à tomber en panne presque régulièrement.

À partir de 2003, de nombreux nouveaux sites de réseaux sociaux ont été lancés, incitant l'analyste de logiciels sociaux Clay Shirky à utiliser le terme YASNS : "Yet Another Social Networking Service". La plupart ont pris la forme de sites centrés sur le profil afin de reproduire le succès de Friendster, ou pour cibler des démographies spécifiques. Alors que les SRS généraux sollicitent un large public, des sites professionnels tels que LinkedIn [Lin], Xing [XIN] (anciennement openBC) se concentrent sur les professionnels tels que les hommes d'affaires et les demandeurs d'emplois.

En outre, au fur et à mesure que les phénomènes liés aux réseaux sociaux et au contenu généré par les utilisateurs ont augmenté, les sites web de partage des médias ont commencé à mettre en œuvre les fonctionnalités SRS et à devenir eux-mêmes des SRS. Les exemples incluent Flickr (partage de photos), Last.FM (écoute de la musique) et YouTube (partage de vidéos).

Avec la pléthore de startups lancées dans la Silicon Valley, peu de gens ont prêté attention aux SRS qui ont gagné en popularité ailleurs, même ceux construits par de grandes sociétés. Par exemple, Orkut de Google qui n'a pas réussi à construire une base d'utilisateurs durable aux États-Unis, mais une "invasion brésilienne" a fait d'Orkut le SRS national du Brésil. Windows Live Spaces de Microsoft a également lancé à la réception américaine tiède mais il est devenu extrêmement populaire ailleurs.

Peu d'analystes ou de journalistes ont remarqué le lancement de MySpace [Mys] à Santa Monica, en Californie, à des centaines de kilomètres de la Silicon Valley. MySpace a été lancé en 2003 pour rivaliser avec des sites comme Friendster, Xanga et AsianAvenue, selon le cofondateur Tom Anderson, les fondateurs voulaient attirer des utilisateurs de Friendster éloignés. Après que des rumeurs aient émergé selon lesquelles Friendster adopterait un système basé sur les frais, les utilisateurs ont posté des messages encourageant les gens à rejoindre les autres SRS, y compris MySpace. Pour cette raison, MySpace a pu se développer rapidement en capitalisant sur l'aliénation de Friendster de ses premiers utilisateurs. Un groupe particulièrement important qui a encouragé les autres à changer de groupe était un groupe de rock indépendant qui ont été expulsés de Friendster pour ne pas avoir respecté les règles du profil. Bien que MySpace n'ait pas été lancé avec des groupes en tête, ils ont été bien accueillis. Des

groupes de rock indépendants de la région de Los Angeles ont commencé à créer des profils, et les promoteurs locaux ont utilisé MySpace pour annoncer des offres VIP pour des clubs populaires. Intrigué, MySpace a contacté des musiciens locaux pour voir comment ils pourraient les soutenir. Les groupes n'étaient pas la seule source de croissance de MySpace, mais la relation symbiotique entre les groupes et les fans ont aidé MySpace à s'étendre au-delà des anciens utilisateurs de Friendster. La dynamique des groupes et des fans était mutuellement bénéfique : les groupes voulaient être en mesure de contacter les fans, tandis que les fans voulaient attirer l'attention de leurs groupes préférés et utilisaient les connexions Friend pour signaler l'identité et l'affiliation.

De plus, MySpace s'est différencié en ajoutant régulièrement des fonctionnalités basées sur la demande des utilisateurs et en permettant aux utilisateurs de personnaliser leurs pages. Cette caractéristique est apparue parce que MySpace n'a pas empêché les utilisateurs d'ajouter du HTML dans les formulaires qui encadraient leurs profils, une culture de code de copier/coller a émergé sur le Web pour aider les utilisateurs à générer des arrière-plans et des mises en page MySpace uniques.

Les adolescents ont commencé à rejoindre MySpace en masse en 2004. Contrairement aux utilisateurs plus âgés, la plupart des adolescents n'étaient jamais sur Friendster, certains ont rejoint parce qu'ils voulaient se connecter avec leurs groupes préférés, d'autres ont été introduits sur le site par des membres plus âgés de la famille. Lorsque les adolescents ont commencé à s'inscrire, ils ont encouragé leurs amis à se joindre à eux. Plutôt que de rejeter les utilisateurs mineurs, MySpace a changé sa politique d'utilisateur pour autoriser les mineurs. Au fur et à mesure que le site grandissait, trois populations distinctes ont commencé à se former : les musiciens/artistes, les adolescents, et la foule sociale. Dans l'ensemble, les deux derniers groupes n'interagissaient pas les uns avec les autres, sauf à travers les bandes. En raison du manque de couverture médiatique en 2004, peu d'autres ont remarqué la popularité croissante du site.

Puis, en juillet 2005, News Corporation a acheté MySpace pour 580 millions de dollars, attirant ainsi l'attention des médias. Par la suite, les problèmes de sécurité en proie à MySpace. Le site a été impliqué dans une série d'interactions sexuelles entre adultes et mineurs, ce qui a déclenché une action en justice. Une panique morale concernant les prédateurs sexuels s'est rapidement répandue, bien que la recherche suggère que les préoccupations étaient exagérées.

Un phénomène mondial

Alors que MySpace a attiré la majorité de l'attention des médias aux États-Unis et à l'étranger, les réseaux sociaux proliféraient et gagnaient en popularité dans le monde entier. Friendster a gagné du terrain dans les îles du Pacifique, Orkut est devenu le premier réseau social au Brésil avant de croître rapidement en Inde, Mixi a été largement adopté au Japon, LunarStorm a décollé en Suède, Hi5 été adopté dans les petits pays d'Amérique latine, d'Amérique du Sud et d'Europe, Bebo est devenu très populaire au Royaume-Uni, en Nouvelle-Zélande et en Australie. De plus, des services communautaires et de communication auparavant populaires ont commencé à implémenter des fonctionnalités sociales. Le service de messagerie instantanée chinois QQ est instantanément devenu le plus grand réseau social au

monde en ajoutant des profils et en rendant visible des amis, tandis que Cyworld a accaparé le marché coréen en introduisant des pages d'accueil.

Les services de blogs avec des fonctionnalités sociales complètes sont également devenus populaires. Aux États-Unis, les outils de blog avec des fonctionnalités sociales, tels que Xanga, LiveJournal et Vox, ont attiré un large public. Skyrock règne en France et Windows Live Spaces domine de nombreux marchés à travers le monde, notamment au Mexique, en Italie et en Espagne. Bien que les réseaux sociaux comme QQ, Orkut et Live Spaces sont aussi grands, sinon plus, que MySpace, ils sont peu couverts dans les médias américains et anglophones, ce qui rend difficile le suivi de leurs trajectoires.

Expansion des communautés de niche

Parallèlement à ces services ouverts, d'autres services de réseaux sociaux ont été lancés pour soutenir la démographie de niche avant de s'étendre à un public plus large. Contrairement aux services de réseaux sociaux précédents, Facebook a été conçu pour soutenir uniquement les réseaux collégiaux distincts. Facebook a commencé au début de 2004 sous la forme d'un réseau social à Harvard. Pour se joindre, un utilisateur devait avoir une adresse e-mail harvard.edu. Lorsque Facebook a commencé à soutenir d'autres écoles, ces utilisateurs devaient également avoir des adresses électroniques d'universités associées à ces établissements, une exigence qui a maintenu le site relativement fermé et a contribué à ce que les utilisateurs perçoivent le site comme une communauté intime et privée.

À compter de septembre 2005, Facebook a été élargi pour inclure des étudiants du secondaire, des professionnels dans les réseaux d'entreprise et éventuellement tout le monde. Le passage à l'inscription ouverte ne signifiait pas que les nouveaux utilisateurs pouvaient facilement accéder aux utilisateurs dans des réseaux fermés, l'accès aux réseaux d'entreprise exigeait toujours l'adresse .com appropriée, tandis que l'accès aux réseaux du secondaire exigeait l'approbation de l'administrateur. Contrairement à d'autres services de réseaux sociaux, les utilisateurs de Facebook sont incapables de rendre leur profil complet public à tous les utilisateurs. Une autre caractéristique qui différencie Facebook est la capacité des développeurs externes à construire des applications qui permettent aux utilisateurs de personnaliser leurs profils et d'effectuer d'autres tâches, telles que la comparaison des préférences de film et des historiques de voyage.

Alors que la plupart des services de réseaux sociaux se concentrent sur une croissance large et exponentielle, d'autres cherchent explicitement des publics plus restreints. Certains, comme aSmallWorld et BeautifulPeople, restreignent intentionnellement l'accès pour paraître sélectif et élite. Les autres sites centrés sur l'activité tels que Couchsurfing, les sites axés sur l'identité comme BlackPlanet et les sites axés sur l'affiliation comme MyChurch sont limités par leur cible démographique et ont donc tendance à être plus petits. Enfin, quiconque souhaite créer un site de réseau social de niche peut le faire sur Ning, un service de plateforme et d'hébergement qui encourage les utilisateurs à créer leur propre service de réseaux sociaux.

Actuellement, il n'existe pas de données fiables sur le nombre de personnes qui utilisent des services de réseaux sociaux, bien que des études de marché indiquent que les réseaux

sociaux en ligne sont de plus en plus populaires dans le monde entier. Cette croissance a incité de nombreuses entreprises à investir du temps et de l'argent dans la création, l'achat, la promotion et la publicité des services de réseaux sociaux.

La montée des services de réseaux sociaux indique un changement dans l'organisation des communautés en ligne. Alors que les sites Web dédiés aux communautés d'intérêts existent et prospèrent, les réseaux sociaux sont principalement organisés autour de personnes et non d'intérêts. Les premières communautés publiques en ligne telles qu'Usenet et les forums de discussion publics étaient structurées par thèmes ou selon des hiérarchies thématiques, mais les réseaux sociaux sont structurés en réseaux personnels (ou égocentriques), l'individu étant au centre de sa propre communauté. Ceci reflète plus fidèlement les structures sociales non médiatisées. L'introduction des fonctionnalités de services de réseaux sociaux a introduit un nouveau cadre organisationnel pour les communautés en ligne, et avec lui, un nouveau contexte de recherche dynamique.

2.4 Représentation des réseaux sociaux

Dans le domaine de l'analyse des réseaux sociaux, les analystes et les chercheurs scientifiques utilisent généralement deux outils mathématiques pour représenter un réseau social, les graphes et les matrices. Nous allons discuter en détails ces deux représentations dans la suite de cette section, nous commencerons d'abord par les graphes parce qu'ils sont les plus utilisés et nous enchaînerons avec les matrices. Pour toutes les deux représentations, nous donnons quelques définitions et notations utilisées par chaque méthode.

2.4.1 Les graphes

Il existe plusieurs types de graphes tels que les diagrammes et les graphes statistiques, au même temps, le nom graphe est utilisé pour décrire plusieurs choses différentes. Mais, dans l'analyse des réseaux sociaux, un seul type de graphe est utilisé principalement pour représenter un réseau. C'est le type des graphes qui se compose de points et de lignes, les points représentent les acteurs et les lignes représentent les relations entre ces acteurs.

Les sociogrammes sont les premières représentations graphiques proposées, ils ont été développées par le médecin Jacob Levy Moreno dans les années 1930 [MJ+34]. Un sociogramme est un diagramme de nœuds et d'arêtes qui représentent les acteurs et les relations qu'ils existent dans un réseau social respectivement. Il existe plusieurs variantes de sociogramme mais elles partagent toutes le même format qui consiste à utiliser un cercle marqué pour décrire un acteur ou un individu de la population étudiée, et un morceau de ligne entre une paire d'individus pour mentionner qu'il existe une relation entre les deux.

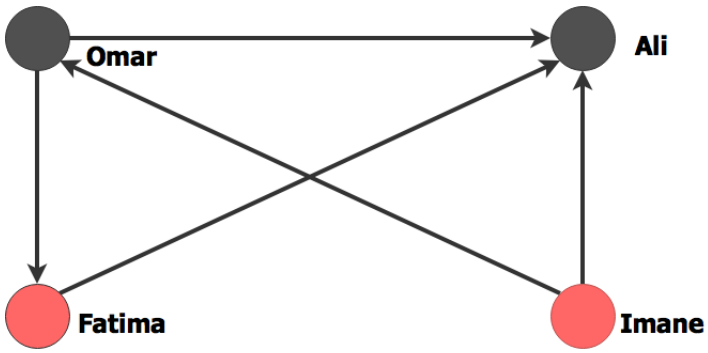
Supposons que nous avons un groupe de quatre personnes (Ali, Omar, Fatima, et Imane) et nous voulons représenter les relations qui peuvent avoir ces personnes entre eux dans un graphe sociogramme. Pour ce faire, nous commencerons d'abord par représenter chaque personne par un nœud avec une étiquette, dans ce cas, nous avons utilisé un cercle coloré avec le nom de la personne comme étiquette. La figure 2.2 montre le graphe de quatre nœud mais sans aucune relation.

FIGURE 2.2: Nœuds représentant les individus du groupe

Dans cet exemple, nous avons utilisé deux couleurs pour représenter le sexe de chaque personne, la couleur noire pour les hommes et la couleur rouge pour les femmes. Il faut savoir que les couleurs, les formes de dessin, ou les tailles des formes sont généralement utilisés pour représenter les attributs (les caractéristiques) des individus.

Nous voulons maintenant représenter les relations d'amitié qui existent au sein de ce groupe et nous nous intéressons uniquement à la relation "ami proche". Dans ce cas, chacune des quatre personnes peut choisir aucun, un, deux, ou tous les trois autres comme "ami proche". Dans notre exemple illustratif, Omar a choisi Ali et Fatima mais pas Imane, Fatima a choisi seulement Ali, Imane a choisi Ali et Omar, et Omar n'a choisi aucun autre. Nous représentons cette relation par une flèche qui commence du nœud (personne) sélecteur vers tous les autres nœuds choisis, comme le montre la figure 2.3.

FIGURE 2.3: Graphe des relations d'amitié

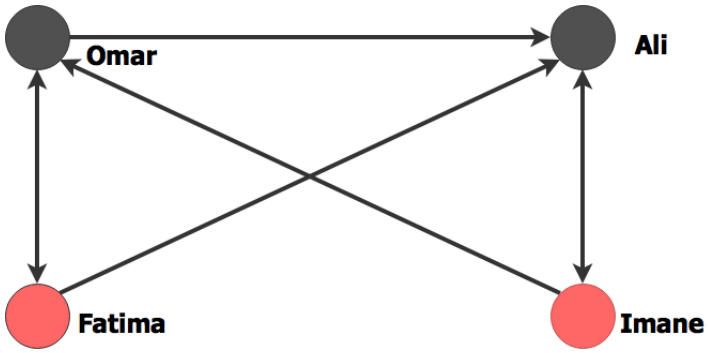


Supposons qu'il y'a aussi un autre genre de relation entre les quatre personnes, qui s'agit du mariage. Dans notre exemple, Ali est le mari d'Imane, et Omar est le mari de Fatima. On peut également représenter ce type de relation comme une flèche directe mais à double sens vu que la relation est réciproque comme dans la figure 2.4. On peut aussi utiliser un graphe simple pour représenter ce genre de relation en utilisant des lignes à la place des flèches orientées (flèche avec tête)

FIGURE 2.4: Graphe de la relation mariage

Les graphes nous permettent aussi de représenter différents types de relation dans un seul graphe, par exemple si on veut représenter la relation d'amitié et la relation de mariage entre les quatre personnes de notre exemple. Le résultat sera un graphe semblable au graphe dans la figure 2.5

FIGURE 2.5: Réseau social d'individus du groupe avec les deux types de relations : amitié et mariage



2.4.1.1 Définition

Mathématiquement, un graphe G est désigné par une paire $G(V, E)$ où V représente l'ensemble des nœuds (des fois appelés sommets ou cellules) et E représente l'ensemble des arêtes (liens ou arcs).

2.4.1.1.1 Nœuds

L'un des composants fondamentaux de n'importe quel graphe est les nœuds, ils représentent les individus d'un réseau social. Par exemple dans le cas d'un réseau social d'amis, les nœuds représentent les gens ou les personnes. L'appellation des nœuds varie selon le contexte, ils peuvent être appelés sommets, acteurs, ou même cellules. Par exemple, dans un réseau de télécommunication, les nœuds représentent les appareils, et les liens entre ces nœuds représentent les connexions entre les appareils. Dans un contexte social les nœuds sont appelés acteurs ou individu. La représentation mathématique d'un tel ensemble de nœuds est comme

suite :

$$V = \{v_1, v_2 \dots v_n\} \tag{2.1}$$

Où V désigne l'ensemble des nœuds et $v_i, 1 \leq i \leq n$ est un nœud unique. $|V| = n$ désigne la taille du graphe. Dans la figure 2.4 $n = 4$

2.4.1.1.2 Arêtes

Le deuxième composant d'un graphe est les arêtes qui relient les nœuds du graphe entre eux. L'ensemble des arêtes est généralement noté par E

$$E = \{e_1, v_e \dots e_n\} \tag{2.2}$$

Où $e_i, 1 \leq i \leq m$ est une arête, $m = |E|$ désigne la taille de l'ensemble des arêtes. Une arête peut être aussi représentée par la notation $e(v_1, v_2)$ ou (v_1, v_2) qui signifie qu'il y'a une arête entre les nœuds v_1 et v_2.

Une arête peut avoir des fois une direction ou une orientation, cela signifie qu'un nœud est connecté à un autre mais pas vice versa, dans ce cas l'arête est représentée par une flèche orientée et appelée arête directe. La figure 2.3 montre un graphe avec des arêtes directes, de même on peut appeler ce graphe un graphe direct. Dans les graphes directs, une arête $e(v_i, v_j)$ est représentée par une flèche qui commence du nœud v_i et finit au nœud v_j, une arête directe peut commencer et finir au même nœud, elle est appelée boucle ou auto-lien et elle est représentée par $e(v_i, v_i)$.

Dans le cas où l'arête n'a pas de direction (n'est pas orientée), elle est appelée indirecte, les nœuds sont liés dans les deux sens. Du coup, les arêtes $e(v_1, v_2)$ et $e(v_2, v_1)$ sont les mêmes. La figure 2.4 montre un graphe avec des arêtes indirectes, on remarque que $e(Ali, Fatima) = e(Fatima, Ali)$, dans ce cas le graphe est appelé un graphe indirect.

2.4.1.2 Types des graphes

Il existe généralement plusieurs types de graphes, le type peut varier selon le contexte ou selon la nature du réseau lui-même. Dans ce qui suit nous discuterons quelques types de graphes.

2.4.1.2.1 Les graphes directs, indirects, et mixés

Comme nous avons vu précédemment, les arêtes peuvent être directes ou indirectes selon la nature de la relation. Un graphe direct est un graphe qui ne contient que des arêtes directes, le contraire des graphes indirects qui ne contiennent que des arêtes indirectes. Un graphe mixé est un graphe qui peut avoir des arêtes directes et indirectes au même temps. Dans un graphe direct, on peut avoir deux arêtes entre deux nœuds i et j, une arête de i à j et une deuxième

de j à i. Tandis que dans les graphes indirects une seule arête peut être entre deux nœuds i et j.

Dans les réseaux sociaux en ligne, il y a plusieurs réseaux directs et indirects, par exemple, Facebook est un réseau indirect dans lequel si une personne A est ami de B alors B est aussi ami de A. Au contraire, Twitter est un réseau direct où A peut être suivi par B mais le contraire n'est pas vrai, B n'est pas forcément suivi par A.

2.4.1.2.2 Les graphes simples et les multigraphes

Dans les graphes simples, pour chaque nœud i et j une seule arête peut avoir lieu entre les deux nœuds. Cela est traduit par le fait qu'il existe un seul type de relation entre les acteurs de réseau, la figure 2.3 montre un exemple de graphe simple. Cependant, dans les multigraphes pour chaque nœud i et j il peut y avoir plusieurs arêtes entre les deux nœuds au même temps, la figure 2.5 montre un exemple d'un multigraphe avec deux types d'arêtes (deux relations, amitié et mariage).

2.4.1.2.3 Les graphes pondérés

Un graphe pondéré est un graphe où les arêtes sont associées avec des poids ou des pondérations. Par exemple, un graphe peut représenter une carte géographique où les nœuds représentent les villes et les arêtes sont les routes qui relient ces villes. Le poids associé à chaque arête représente la distance entre les villes. Un graphe pondéré peut être représenté par $G(E, V, W)$ où W (en anglais, weights) représente les poids associés à chaque arête, $|W| = |E|$.

Un exemple de réseaux pondérés est tout simplement un réseau Web. Un réseau Web représente la façon dont les sites Internet sont connectés où les nœuds représentent les pages web et les poids des arêtes représentent le nombre de liens entre ces pages web. Il existe un type particulier de graphes pondérés où les arêtes ont des poids binaires ($0/1 ou -/+$). Ce type d'arêtes s'appelle les arêtes signées (signed edges) et le type de ces graphes pondérés s'appelle les graphes signés (signed graphs). Les graphes signés sont généralement utilisés pour représenter les relations contradictoires qui peuvent exister entre les personnes, par exemple dans un réseau social d'amis les arêtes positives ($+$) représentent les relations d'amitié tandis que les arêtes négatives ($-$) représentent les relations d'hostilité entre les nœuds.

2.4.2 Les matrices d'adjacence

Les graphes sont un moyen très efficace pour représenter et visualiser les réseaux sociaux. Cependant, lorsqu'un réseau contient plusieurs acteurs et/ou différents types de relation, il devient visuellement très compliqué ce qui rend son analyse très difficile. Il est aussi possible d'utiliser les matrices pour représenter les graphes. L'utilisation des matrices permet également l'application des méthodes de calcul matricielle pour extraire des informations et des modèles à partir du réseau étudié.

Dans ce type de représentation matricielle, les lignes et les colonnes de la matrice représentent les nœuds et les valeurs qui contiennent les cellules représentent des informations sur les arêtes du graphe. Généralement, deux types de matrices sont utilisés, les matrices

d'adjacence et les matrices d'incidence.

Une matrice d'adjacence est une matrice carrée avec une dimension égale au nombre totale des nœuds dans le graphe, elle est bien adaptée au réseau avec un seul type d'acteurs. Cependant, il peut y avoir plusieurs matrices d'adjacence selon le type et les caractéristiques des relations dans le réseau. Les matrices binaires sont les plus communes, s'il existe une relation entre les deux nœuds v_i et v_j alors $A_{ij} = 1$, sinon s'il n'existe aucune relation alors $A_{ij} = 0$ dans le cas d'une arête non-pondérée. Dans le cas d'une arête pondérée on met le poids de l'arête comme valeur de la cellule. Ainsi une matrice d'adjacence d'un graphe non-pondéré $G(V, E)$ est notée par A ou :

$$A = \begin{cases} 1 & Si v_i est connecté au v_j \\ 0 & Sinon \end{cases} \tag{2.3}$$

Une matrice d'adjacence peut être aussi symétrique ou asymétrique selon le type de la relation, dans le cas d'une relation symétrique ou à double sens tel qu'une relation d'amitié ou fraternité on a $A_{ij} = A_{ji}$. Dans le cas d'une relation asymétrique ou à un seul sens tel qu'une relation de paternité on a $A_{ij} \neq A_{ji}$, dans ce cas, les lignes et les colonnes de la matrice représentent respectivement les nœuds de départ et les nœuds d'arrêt des arêtes dans le graphe.

2.5 Analyse des réseaux sociaux

L'analyse des réseaux sociaux est le processus d'extraction et d'étude des caractéristiques clés des réseaux sociaux afin de bien comprendre la structure et le fonctionnement de ces réseaux et de prédire et gérer leur évolution. Plusieurs disciplines peuvent être impliquées dans ce domaine, des disciplines liées à l'étude de l'humain tel que : la sociologie et la psychologie, les disciplines purement techniques tel que : la télécommunication. Généralement, les chercheurs scientifiques et les analystes des réseaux sociaux utilisent des méthodes et des algorithmes du domaine de la théorie des graphes pour conduire des analyses sur les réseaux sociaux. Dans cette section nous allons présenter quelque uns de ces méthodes et algorithmes et discuter aussi quelques applications de ce domaine notamment la détection des communautés.

2.5.1 Préliminaires

Dans tout ce qui suit, nous supposons que le graphe $G(E, V)$ représente un réseau social. Nous définissons le degré d'un nœud comme le nombre totale d'arêtes connectées à ce nœud, le degré d'un nœud v_i est noté par d_i. Dans le cas d'un graphe direct, chaque nœud v_i a un degré entrant (le nombre d'arêtes dirigées vers le nœud v_i) et un degré sortant (le nombre d'arêtes sortant de nœud v_i), les deux degrés sont notés par $d_i^{in}(d_i^-)$ et $d_i^{out}(d_i^+)$ respectivement.

2.5.2 Distribution de degrés

La distribution des degrés est une caractéristique importante des graphes, notamment les grands graphes (le cas des réseaux sociaux). Elle joue un rôle très important dans la description du réseau social étudié. La distribution des degrés est une fonction $P(k)$ qui décrit la fraction des nœuds ayant un degré égale à k, elle représente aussi la probabilité qu'un nœud choisi au hasard dans le réseau ait un degré égale à k. La distribution des degrés décrit comment les degrés sont distribués (répartis) entre les nœuds, et sa valeur est entre 0 et 1, $P(k) = 0$ signifie qu'il n'y a aucun nœud du graphe avec une degré égale à k. Dans un graphe indirect de n nœuds, $P(k)$ est définie comme suit :

$$P(k) = \frac{n_k}{n} \tag{2.4}$$

Où n_k est le nombre des nœuds ayant un degré k. Dans le cas d'un graphe indirect, nous avons deux degrés pour chaque nœud donc la distribution des degrés sera en deux dimensions et elle est notée par $P(k^{in}, k^{out})$, et qui signifie la fraction des nœuds ayant un degré entrant égale à k^{in} et un degré sortant égale à k^{out}.

2.5.3 Centralité

La centralité est une mesure utilisée dans la théorie des graphes pour identifier les nœuds importants dans un graphe. Dans un contexte social réel, la centralité définit les acteurs influenceurs dans un réseau. Plusieurs indicateurs peuvent être utilisés pour calculer la centralité dans un graphe, dans ce qui suit nous présenterons quelques types de centralité populaires.

2.5.3.1 Centralité de degré

Dans le monde réel, on considère souvent les personnes qui ont plusieurs relations avec les autres comme des personnes importantes. La centralité de degré [Fre78] transfert la même idée en une mesure. Cette mesure considère les nœuds avec plusieurs connexions comme des nœuds centraux. Dans un graphe indirect la centralité de degré C_d d'un nœud v_i est définie comme suit :

$$C_d(v_i) = d_i \tag{2.5}$$

Où d_i est le degré du nœud v_i. Dans un graphe direct, on peut utiliser soit le degré entrant, le degré sortant, ou la combinaison des deux pour calculer le degré de centralité :

$$C_d(v_i) = d_i^{in} \tag{2.6}$$

$$C_d(v_i) = d_i^{out} \tag{2.7}$$

$$C_d(v_i) = d_i^{in} + d_i^{out} \tag{2.8}$$

Si le degré entrant est utilisé, la popularité du nœud sera mesurée et dans ce cas nous parlons de prestige du nœud. Quand le degré sortant est utilisé, nous parlons de grégarisme de nœud ou sociabilité du nœud. Et si on combine les deux degrés, l'équation sera la même que l'équation.

2.5.3.2 Centralité des vecteurs propres

La centralité des vecteurs propres (eigenvector centrality) [Fre78] est une extension de la centralité des degrés. Au contraire de la centralité des degrés qui considère les nœuds avec plusieurs connexions comme nœuds importants, la centralité des vecteurs propres ne considère pas seulement les connexions directes des nœuds mais elle généralise la mesure par l'implication des nœuds voisins (ou les connexions entrantes dans le cas d'un graphe direct). Cela dit, un nœud qui reçoit plusieurs connexions n'est pas forcément un nœud important dans les yeux de la centralité des vecteurs propres, il se pourrait que les voisins de ce nœud ont des valeurs de centralité faibles ou nulles. De plus, un nœud avec une grande valeur de centralité des vecteurs propres n'est pas forcément un nœud fortement lié, dans ce cas le nœud peut avoir un nombre limité de voisins mais qui sont importants.

La centralité des vecteurs propres est une mesure pour les graphes directs et indirects, elle est notée par C_e et elle est définit comme suit :

$$c_e(v_i) = \frac{1}{\lambda} \sum_{j=1}^{n} A_{j,i} c_e(v_j) \tag{2.9}$$

Où A est la matrice d'adjacence du graphe et λ est un constant fixe.

2.5.3.3 Centralité de proximité

L'idée derrière la centralité de proximité [Fre78] est que les nœuds centraux sont les nœuds qui peuvent atteindre rapidement les autres nœuds du graphe. Pour calculer la centralité de proximité il faut d'abord calculer la moyenne des chemins les plus courts, supposons que $l_{i,j}$ est la longueur du chemin le plus court entre les nœuds i et j, qui signifie le nombre d'arêtes le long du chemin. La moyenne des longueurs des chemins les plus courts du nœud v_i est notée

par l_i et calculer comme suit :

$$l_i = \frac{1}{n} \sum_j l_{i,j} \tag{2.10}$$

2.5.3.4 Centralité d'intermédiarité

La centralité d'intermédiarité [Fre78] est une autre mesure de centralité globale proposée par Freeman. L'intuition de cette mesure est que, dans un graphe, un nœud est d'autant plus important qu'il est nécessaire de le traverser pour aller d'un nœud quelconque à un autre. Plus précisément, un sommet ayant une forte centralité d'intermédiarité est un sommet par lequel passe un grand nombre de chemins géodésiques (i.e. Chemins les plus courts) dans le graphe. Dans un réseau social, un acteur ayant une forte centralité d'intermédiarité est un sommet tel qu'un grand nombre d'interactions entre des sommets non adjacents dépend de lui. Dans un réseau de communication, la centralité d'intermédiarité d'un nœud peut être considérée comme la probabilité qu'une information transmise entre deux nœuds passe par ce nœud intermédiaire.

Soit $G = (V, E)$ un graphe (orienté ou non) d'ordre N La centralité d'intermédiarité d'un nœud $v_i \in V$ est définie par :

$$C^{int}(v_i) = \sum_{j=1}^{N} \sum_{k=1}^{N} \frac{g_{jk}(v_i)}{g_{jk}} \tag{2.11}$$

où $g_{jk}(v_i)$ est le nombre total de chemins géodésiques entre les nœuds v_j et v_k qui passent par le nœud v_i, et g_{jk} est le nombre total de chemins géodésiques entre les nœuds v_j et v_k.

2.5.4 Modularité

Beaucoup d'algorithmes peuvent identifier un sous-ensemble de partitions significatives, idéalement un ou juste quelques-uns, tandis que quelques autres, comme les techniques de clustering hiérarchique donnent un grand nombre de partitions qui ne signifie pas que toutes les partitions trouvées sont également bonnes.

Donc il est utile (parfois même nécessaire) d'avoir un critère quantitatif pour évaluer la qualité d'une partition d'un graphe. Une fonction de qualité est une fonction qui assigne un score à chaque partition d'un graphe. De cette façon on peut classer les partitions basées sur leur score donné par la fonction de qualité. Les partitions avec les bons scores sont bonnes, donc celle avec le plus grand score est par définition la meilleure. Néanmoins, Il faut garder à l'esprit que la réponse à la question de quand une division est meilleure qu'une autre dépend du concept spécifique de communauté et/ou la fonction de qualité adoptée.

Une fonction de qualité Q est additif s'il y a une fonction élémentaire q tel que pour une partition P d'un graphe :

$$Q(P) = \sum_{C \in P} q(C) \tag{2.12}$$

Où C est un cluster (communauté) de la partition P. Cette fonction stipule que la qualité d'une partition est donnée par la somme des qualités des groupes individuels. La plupart des fonctions de qualité utilisées dans la littérature sont additif, bien que ce ne soit pas une exigence nécessaire.

La fonction de qualité la plus populaire est la modularité, introduite par de Newman et Girvan [CNM04 ; New06]. C'est une mesure qui a pour but de caractériser l'efficacité d'une partition des sommets d'un graphe au regard de la densité des liens à l'intérieur des groupes et du nombre de liens entre ces groupes, ainsi que vis-à-vis du degré des sommets pris en compte.

La modularité compare la proportion des arêtes du réseau qui relient des sommets issus d'une même communauté à la proportion attendue dans un graphe équivalent dans le sens où les sommets ont la même distribution des degrés mais où les arêtes auraient été placées aléatoirement entre tous les sommets.

La modularité peut alors être écrite comme suit :

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(i, j) \tag{2.13}$$

Où A est la matrice d'adjacence, k_i et k_j sont respectivement les degrés des nœuds i et j, m est le nombre total de nœuds dans le réseau.δ est la fonction de Kronecker qui vaut 1 si les nœuds i et j sont dans le même groupe (communauté), sinon 0.

2.6 Conclusion

L'analyse des réseaux sociaux en ligne est un domaine multidisciplinaire qui combine plusieurs domaines de recherche tels que la sociologie, la théorie des graphes, et les statistiques. Ce domaine a pour but d'étudier les réseaux d'individus afin de mieux comprendre les comportements de ces individus au sein du réseau. Il existe plusieurs applications de l'analyse des réseaux sociaux en ligne, parmi les applications les plus populaires on trouve les systèmes de recommandation et la détection des communautés.

Dans ce chapitre, nous avons présenté en générale l'analyse des réseaux sociaux (L'ARS). Nous avons commencé par définir les réseaux sociaux réels et les réseaux sociaux en ligne.

Ensuite, nous avons présenté les moyens les plus utilisés pour représenter les réseaux sociaux : les graphes et les matrices d'adjacence. Après, nous avons donné quelques méthodes utilisées dans l'analyse des réseaux sociaux telles que les méthodes de calcul des centralités dans les graphes, et la modularité qui est une métrique proposée par Newman dans [New06] et utilisée pour évaluer la qualité d'une partition communautaire. Nous allons présenter dans le prochain chapitre le problème de détection de communautés dans les réseaux sociaux et nous allons aussi donner les méthodes et les techniques les plus utilisées pour détecter et identifier les communautés d'un réseau social telles que les méthodes de partitionnement des graphes, le clustering hiérarchique, et les méthodes basées sur l'optimisation de la modularité.

Chapitre 3

Détection de communautés dans les réseaux sociaux

Sommaire

3.1 Introduction

La détection de communautés est une filière du domaine de l'analyse des réseaux sociaux et de l'analyse des graphes. Cette filière constitue aujourd'hui un domaine de recherche très active ces dernières années, en raison du développement qui ont connu les réseaux sociaux en ligne. La détection de communautés n'est pas liée uniquement aux réseaux sociaux mais elle a d'autres applications dans d'autres différents domaines tel que : la biologie, les physiques, la télécommunication.

La détection de communautés a pour but de détecter et identifier les groupes d'individus qui ont une grande similarité entre eux (qui partagent les mêmes caractéristiques), elle est considérée comme un problème NP-difficile. Beaucoup d'algorithmes ont été proposés durant les dernières années pour surmonter ce problème, on peut classifier ces algorithmes en plusieurs familles selon l'approche utilisé dans l'algorithme (traditionnelle, heuristique, ...) ou selon le type de graphe (normal, direct, dynamique, ...). Dans la suite de ce manuscrit nous allons présenter les algorithmes et les méthodes les plus connus et les plus utilisés aujourd'hui dans la détection des communautés dans les réseaux sociaux.

3.2 Définition d'une communauté

Avant de discuter n'importe quel algorithme ou méthode, il est indispensable de définir d'abord c'est quoi une communauté. Il n'y pas une définition exacte ou standard de communauté. En réalité, la définition dépend généralement du domaine d'application, on peut trouver d'autre noms pour décrire le même concept tel que : groupe et cluster. Une définition de base a été donnée par Fortunato qui dit qu'il doit y avoir plus d'arêtes dans la communauté que d'arêtes qui relient les nœuds de la communauté au reste du graphe. Une autre définition donnée par Newman qui définit une communauté comme un groupe de nœuds avec des connexions denses à l'intérieur du groupe et des connexions creuses entre le groupe et les autres groupes du graphe.

Dans le contexte des réseaux sociaux, une communauté est définie comme un groupe d'individus qui partageant plus de similarités entre eux qu'avec les autres individus à l'extérieur du groupe, par exemple on peut regrouper les individus selon leur sexe, leur âge, ou leur niveau d'éducation.

3.3 Algorithmes de détection de communautés

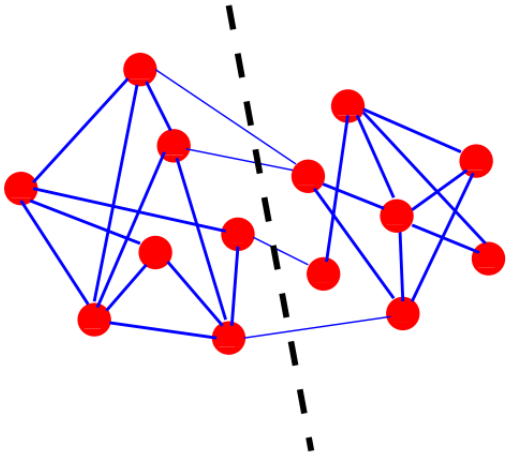
La détection de communautés vise à diviser un graphe en sous-graphes avec la remarque qu'un nœud peut être dans des différents sous-graphes. D'une façon formelle, pour un graphe $G(V, E)$, la tâche pour un algorithme de détection de communautés est de chercher l'ensemble des communautés $C_{i i=1}^{n}$.

Nous présenterons dans cette section quelques algorithmes de détection de communautés regroupés par l'approche qu'ils utilisent. Vu le grand nombre d'algorithmes qu'il existe aujourd'hui et afin de ne pas trop écrit, nous avons choisi de présenter uniquement les algorithmes connus dans le domaine sans les trop détaillés.

3.3.1 Approches basées sur le partitionnement des graphes

Le partitionnement des graphes et le processus de partitionner un graphe en un nombre g prédéfinie de groupes, de telle sorte que le nombre d'arêtes entre ces groupes soit minime. Le nombre d'arêtes qui sont entre les groupes de nœuds est appelé : le coût de coupe (cut size).

FIGURE 3.1: Exemple d'un graphe partitionné



L'algorithme de Kernighan-Lin est l'un des premiers algorithmes et qui est encore utilisé, cet algorithme vise à optimiser la fonction objective KL, c'est à dire minimiser le coût de coupe tout en gardant les tailles des groupes équilibrées. L'algorithme commence par diviser le graphe en deux partitions A et B de même taille (même nombre de nœuds dans chaque partition), ces deux partitions peuvent être générées d'une façon aléatoire ou en utilisant des informations sur le graphe. Ensuite, l'algorithme cherche un sous-ensemble de nœuds de chaque partition de telle sorte que leur échange conduit à une réduction du coût C_i (swapping cost), l'algorithme répète cette procédure jusqu'à ce que tous les nœuds soient visités. A la fin de chaque itération la paire (A', B') correspondant à la plus petite valeur de C_i sera choisie comme point de départ pour la prochaine série d'itérations. Il faut noter que les deux sous-ensembles doivent être de même taille, en plus le sous-ensemble peut être un nœud, c'est-à-dire on peut choisir un nœud de chaque partition à la place de choisir un sous-ensemble.

L'algorithme 3

Algorithme 3.1 : Algorithme de Kernighan-Lin

```
1 Diviserle graphe en 2 parties A, B de taille égale
2 répéter
3     Choisir a_i ∈ A, b_i ∈ B, de telle sorte que le cout soit minime et ni a_i, ni b_i été
        choisi avant
4     Échanger a_i et b_i
5     Soit C_i le cout des partitions après avoir échangé a_i et b_i
6 jusqu'à ce que tous les nœuds soient visités;
7 retourner (A', B') correspondant au plus petite valeur de C_i
```

L'algorithme de Kernighan-Lin marche uniquement avec deux partitions, une extension de l'algorithme a été proposé dans [KL70] pour rendre l'algorithme exploitable pour n'importe quel nombre de partitions. Cependant, la complexité temporelle et spatiale augmente rapidement avec le nombre de partitions.

Faut noter que dans le cas des algorithmes de partitionnement des graphes, on doit spécifier le nombre de groupes (communautés) auxquels on souhaite partitionner le graphe (le réseau). La taille de chaque communauté doit être aussi spécifiée sinon l'algorithme va essayer de générer des communautés avec des tailles équilibrées. Étant donné que le nombre de communautés n'est généralement pas connu à l'avance, les méthodes de partitionnement des graphes ne conviennent pas pour détecter les communautés dans de tels cas.

3.3.2 Approches basées sur le clustering par partitionnement

Le clustering par partitionnement est une famille populaire des méthodes d'identification de groupes d'objets similaires (appelés aussi clusters) dans un ensemble de données. Les algorithmes de cette famille considèrent les objets comme des points d'un espace métrique, dans le cas d'un graphe, chaque point de l'espace représente un nœud unique. En utilisant cette représentation, les algorithmes peuvent donc utiliser des mesures de distance définies sur cet espace pour calculer la similarité entre chaque paire de nœuds et ainsi regrouper les nœuds similaires dans des clusters.

Les algorithmes de clustering par partitionnement utilisent tous une fonction de qualité pour évaluer la qualité d'une partition, cette fonction se diffère d'un algorithme à un autre. La liste ci-dessous montre quelques exemples de ces fonctions :

L'un des algorithmes les plus populaires de clustering par partitionnement est l'algorithme des k-moyennes (k-means) que nous avons présenté dans la section 1.6.1.3.1. Cet algorithme permet de regrouper un ensemble de données en k cluster disjoint en utilisant une mesure de distance pour assigner les objets à leurs clusters. La fonction de qualité utilisée par l'algorithme pour mesurer la pertinence d'une partition est la somme des distances intra-cluster, et elle

est définie comme suit :

$$\sum_{k=1}^{K} \sum_{x_i \in S_k} d(x_j, c_k)^2 \tag{3.1}$$

Où K est le nombre total des clusters, S_k est l'ensemble des objets du cluster k, c_k est le centroïde du cluster k, x_i représente les données de l'objet i, et $d()$ est une mesure de distance. Une version de l'algorithme K-means adaptée aux graphes est donnée dans la figure 4

Algorithme 3.2 : Algorithme des K-mans adapté aux graphes

Entrées : Un graphe $G = (V, E)$ représenté par une matrice d'adjacence A, Un nombre K de communautés, Une mesure de distance d

Sorties : Un ensemble de communautés $S = \{S_1, S_2, \dots S_K\}$

Initialiser les centroïdes (centres des clusters) $\{C_1, C_2, \dots C_k\}$: Choisir aléatoirement K nœuds du graphe
répéter
  pour chaque *nœud* $v_i \in V$ **faire**
    Affecter le nœud x_i à l'une des communautés en fonction de la règle suivante :
    $argmin_{k,1 \leq k \leq K} d(x_i, C_k)$
    Calculer de nouveaux les centroïdes en utilisant l'équation suivante :
    $\forall k, 1 \leq k \leq K C_k = \frac{1}{|S_k|} \sum_{v_j \in S_k} a_j$
  fin
jusqu'à *condition*;

3.3.3 Approches basées sur le clustering hiérarchique

Dans la plupart des cas, le nombre de communautés ainsi que lueurs tailles sont inconnu. En générale on ne sait pas de grand choses sur la structure du réseau au les relations entre les individus. Dans de tels cas, on peut utiliser des algorithmes de clustering hiérarchique (regroupement hiérarchique) [].

Le clustering hiérarchique est une famille des algorithmes de regroupement qui visent à regrouper un graphe (un ensemble de données généralement) en un arbre hiérarchique de groupes appelé un dendrogramme. Le point commun entre tous les algorithmes de clustering hiérarchique est l'utilisation des mesures de similarité, une mesure de similarité est utilisée pour calculer la similarité entre chaque paire de nœuds du graphe même si ils ne sont pas voisin (liés). Cela dit les algorithmes de clustering hiérarchique visent à identifier les groupes de nœuds similaires et ils sont classifier en deux catégories : Algorithmes agglomératifs et algorithmes divisifs [].

Les algorithmes agglomératifs utilisent une approche ascendante (ou, en profondeur), c'est à dire l'algorithme considère en début tous les nœuds comme des groupes, ensuite il fusionne

les paires de nœuds similaires pour former des groupes jusqu'au arriver au sommet de la hiérarchie.

Au contraire des algorithmes agglomératifs, les algorithmes divisifs utilisent une approche descendante, le graphe complet est considéré comme une seule groupe en début, en suite l'algorithme divise le graphe en sous-groupes jusqu'au arriver au fond de la hiérarchie. L'algorithme le plus connus de cette catégorie des algorithmes est l'algorithme de Girvan et Newman [GN02]

3.3.3.1 Algorithme de Girvan et Newman

Les deux physiciens Michelle Girvan and Mark Newman ont joué un rôle très important dans le domaine de la détection des communautés dans les réseaux complexe. Leur premier algorithme [GN02] constitue une pièce très importante car il a marqué le début d'une nouvelle ère dans le domaine de la détection des communautés, il représente aujourd'hui un point de repère pour tous les chercheurs dans ce domaine.

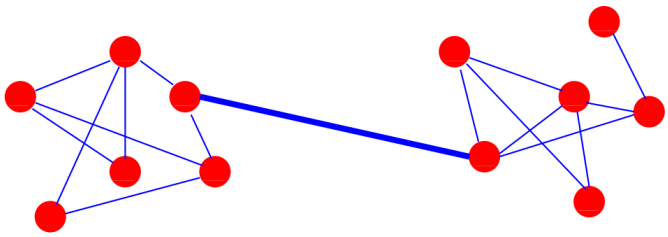
L'algorithme de Girvan et Newman est basé sur la centralité d'intermédiarité des arêtes (edge betweenness), l'idée est d'enlever les arêtes qui ont des valeurs de centralité d'intermédiarité grandes, ensuite recalculer la centralité d'intermédiarité pour toutes les arêtes qui restent après la suppression, cette procédure est répétée jusqu'à ce qu'il n'y ait pas d'arêtes. Le pseudo code 5 montre les étapes de l'algorithme.

Algorithme 3.3 : Algorithme de Girvan et Newman

1 **1** Calculer la centralité de toutes les arêtes du graphe.
2 **2** Supprimer l'arête avec la plus grande centralité.
3 **3** Recalculer les centralités des arêtes affectées par la suppression.
4 **4** Répéter les étapes 2 et 3 jusqu'à ce qu'il n'y ait pas d'arêtes affectées.

La centralité d'intermédiarité des arêtes est une extension de la centralité d'intermédiarité présentée dans la section 2.5.3.4. Girvan et Newman ont proposé cette extension pour les arêtes, qui est définie comme le nombre des plus courts chemins entre toutes les paires de nœuds qui passent par l'arête. L'algorithme donne des bonnes résultats mais le calcul des valeurs de centralités d'intermédiarité des arêtes exige une complexité temporelle élevée de l'ordre de $O(m^2, n)$ où m représente le nombre d'arêtes et n représente le nombre de nœuds. Il est donc utilisable uniquement pour les petits réseaux.

FIGURE 3.2: L'arête au milieu à une centralité d'intermédiarité beaucoup plus élevée que tous les autres arêtes, car tous les plus courts chemins reliant les nœuds des deux communautés traversent cette arête



Newman a proposé après un nouveau algorithme basé sur le premier mais qui utilise la mesure de modularité (Q) présentée dans la section **??** comme critère de sélection des partitions, l'idée est que les partitions qui ont une grande valeur de modularité représentent une meilleur structure communautaire. La mesure de modularité (Q) a été largement reprise et utilisée par beaucoup de chercheurs soit comme mesure d'évaluation pour évaluer les communautés, soit comme une fonction objective pour les algorithmes d'optimisation comme nous allons voir par la suite.

3.3.4 Approches basées sur l'optimisation de la modularité

3.3.4.1 Algorithme de Newman

Newman a proposé un algorithme glouton pour résoudre le problème de la détection des communautés en maximisant la valeur de modularité, il est le premier algorithme qui a utilisé la notion d'optimisation de modularité. C'est un algorithme de clustering hiérarchique agglomératif où les nœuds sont successivement groupés pour former des grandes communautés, de sorte que la modularité augmente après la fusion.

Comme chaque algorithme de clustering agglomératif, l'algorithme de Newman commence par n communauté où chaque nœud représente une communauté. Le rapport de changement dans la modularité lors de la fusion de deux communautés est représenté par δQ peut être calculé en temps constant ce qui rend l'algorithme plus rapide en exécution par rapport à l'algorithme de Girvan et Newman (GN). La complexité de l'algorithme est de l'ordre de $O(n^2)$ pour les graphes creux et $O((m+n) \times n)$ pour les autres graphes.

3.3.4.2 Algorithme de Clauset-Newman-Moore (CNM)

Tous les algorithmes présentés jusqu'à maintenant ont de limitations pour les grands graphes notamment dans le temps d'exécution. Pour éliminer cette limitation, Clauset [CNM04] a reproduit le même travail de Newman mais en exploitant certains raccourcis dans le problème d'optimisation et en utilisant une structure de données plus sophistiquée, Clauset a réussi de rendre l'algorithme plus rapide et avec des résultats comparables aux celles d'algorithme originale. La complexité de l'algorithme est de l'ordre de $O(md \log n)$ où d est la profondeur du dendrogramme décrivant les partitions successives trouvées lors de l'exécution

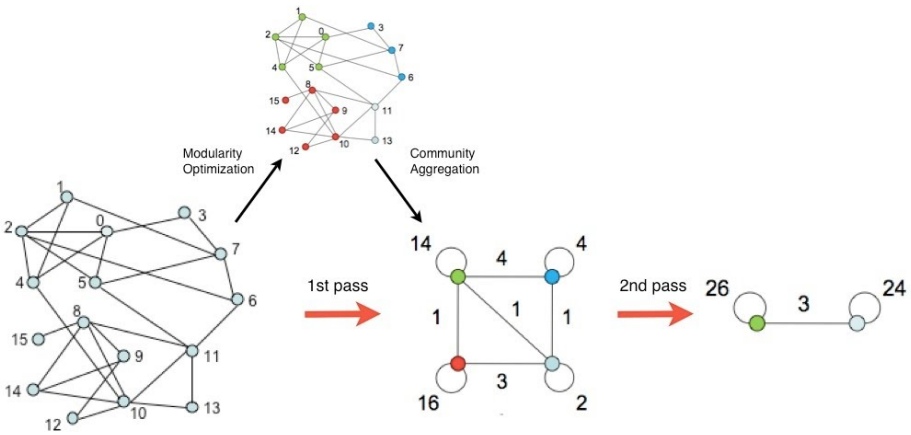
de l'algorithme. Dans le cas des graphes creux, la structure hiérarchique du réseau est généralement très forte et la valeur de d peut s'approcher facilement à $\log n$, dans ce cas la complexité de l'algorithme sera de l'ordre $O(m \log^2 n)$, ce qui permet d'exécuter l'algorithme pour des réseaux d'un million de nœuds ou plus dans un temps raisonnable. Pour évaluer les performances de l'algorithme, Clauset a testé l'algorithme sur un graphe de 400 000 nœuds et 2 million d'arêtes, ce graphe représente le réseau de recommandation de livres sur Amazon.com.

3.3.4.3 Méthode de Louvain

L'algorithme de Louvain a été proposé par Blondel et al. dans ce papier [Blo+08]. Cet algorithme utilise une approche itérative en deux phases répétitives. Dans la première phase, chaque nœud du réseau est assigné à une communauté différente, ainsi il y aura autant de communautés que de nœuds dans cette partition initiale. Ensuite, pour chaque nœud i, le gain de modularité de déplacer le nœud i de sa communauté à une des communautés de ces voisins est calculée. Si le gain est positive le nœud i sera placé dans la communauté pour laquelle ce gain est maximal, sinon le nœud restera dans sa communauté. Cette procédure est appliquée successivement et d'une façon répétitive pour tous les nœuds jusqu'à ce que aucune amélioration ne peut avoir lieu.

Dans la deuxième phase, un nouveau graphe (réseau) sera construit dont les nœuds sont les communautés trouvées dans la première phase et le poids d'arêtes entre chaque deux communautés (i.e. les nœuds du nouveau graphe) est calculé en utilisant la somme des poids des arêtes qui relient les nœuds des deux communautés. Une fois cette deuxième phase est terminée, on peut donc appliquer la première phase de l'algorithme sur ce nouveau réseau et ainsi de suite. La combinaison de ces deux phases est appelée passe, les passes seront donc répétées jusqu'à ce qu'il n'y ait plus de changements et qu'un maximum de modularité soit atteint.

FIGURE 3.3: L'algorithme de Louvain



3.4 Une approche bio-inspirée pour la détection de communautés basée sur l'algorithme de Fireworks

Au cours de la dernière décennie, plusieurs algorithmes ont été proposés pour résoudre le problème de la détection des communautés dans les réseaux complexes. Beaucoup d'entre eux sont basés sur les algorithmes d'intelligence collective et les algorithmes évolutifs. La plupart de ces algorithmes utilisent la densité de modularité comme fonction objective. Cependant, ces algorithmes tentent de trouver la meilleure solution sans tenir compte de la structure du réseau.

Dans ce qui suit, nous présenterons une nouvelle approche pour résoudre ce problème en utilisent une variante modifiée de l'algorithme de Fireworks (FWA). Une nouvelle stratégie d'initialisation et de nouvelles stratégies de mutation sont proposées, basées sur la stratégie de propagation d'étiquettes pour améliorer l'algorithme et pour accélérer sa convergence. L'algorithme proposé a été évalué sur des réseaux réels et synthétiques. Les résultats expérimentaux comparés à trois autres algorithmes connus montrent l'efficacité de l'utilisation de notre approche pour résoudre le problème de la détection des communautés dans des réseaux complexes en générale et dans les réseaux sociaux en particulier.

3.4.1 Problème de détection de communautés

Un réseau peut être représenté par un graphe $G(V, E)$ ou $V = \{v_1, v_2 \dots v_n\}$ représente les sommets ou les nœuds et $E \subseteq V \times V$ représente les bords ou les liens. La structure du réseau peut être représentée par une matrice d'adjacence A de $n \times n$, chaque élément A_{ij} peut être soit 1 ou 0, $A_{ij} = 1$ s'il existe un lien e_{ij} entre les nœuds v_i et v_j, sinon $A_{ij} = 0$.

Supposons que $S \subset G$ est un sou-graphe de G est i est un nœud qui appartient à S, $k_i^{in} = \sum_{i,j \in S} A_{ij}$ et $k_i^{out} = \sum_{i \in S, j \notin S} A_{ij}$, out sont respectivement le degré intérieur et le degré extérieur du nœud i Alors, la communautés représentée par le sous-graphe S a habituellement la propriété suivante :

$$\sum_{i \in S} k_i^{in} > \sum_{i \in S} k_i^{out} \tag{3.2}$$

Cette propriété signifie que la somme de tous les degrés dans la communauté est supérieure à la somme de tous les degrés vers le reste du réseau.

On peut formuler le problème de la détection des communautés comme un problème de maximisation de la modularité (Q). Cette métrique a été proposée par Newman pour évaluer la structure des communautés d'un réseau, Q est décrit mathématiquement par l'équation

suivante :

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(i,j) \tag{3.3}$$

Où k_i et k_j sont respectivement les degrés des nœuds i et j, m est le nombre total de nœuds dans le réseau. $\delta(i,j) = 1$ si les nœuds i et j sont dans le même groupe (communauté), sinon 0. Une grande valeur de Q signifie une meilleure structure communautaire. Sinon, la structure est plus ambiguë. Li [Li+08] a proposé une autre forme mathématique pour écrire la modularité, décrite comme suit :

$$Q = \sum_{i=1}^{N}\left[\frac{L(V_i,V_i)}{L(V,V)} - \left(\frac{L(V_i,V)}{L(V,V)}\right)^2\right] \tag{3.4}$$

La modularité a été utilisée dans de nombreuses méthodes de détection de communautés comme fonction objective à maximiser ou comme critère d'évaluation. Jusqu'à ce que, lorsque Fortunato et Barthelemy [FB07] prouvent que la modularité a une limitation de résolution. Pour surmonter cette limitation, Li et al. [Li+08] a proposé la densité de modularité (D) qui est basée sur le degré de modularité moyen, et elle est définie comme suit :

$$D = \sum_{i=1}^{N}\frac{L(V_i,V_i) - L(V_i,\overline{V})}{|V_i|} \tag{3.5}$$

Où $\frac{L(V_i,V_i)}{|V_i|}$ et $\frac{L(V_i,\overline{V})}{vertV_i|}$ représentent la moyenne de degré interne et externe de la *ieme* communauté, D vise à maximiser le degré interne et à minimiser le degré externe de la structure de communauté.

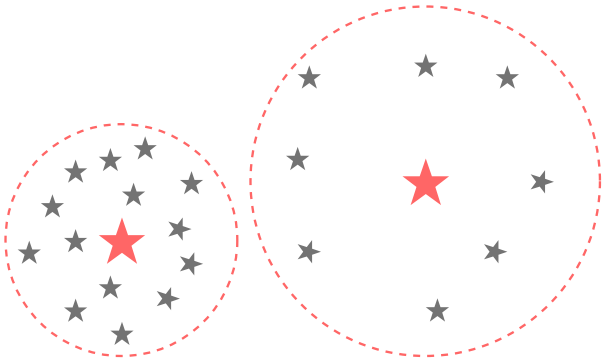
Dans notre approche, nous avons utilisé la densité de modularité (D) comme fonction objective dans l'algorithme FWA, en raison de son efficacité.

3.5 L'algorithme de Fireworks

L'algorithme des fireworks ou feux d'artifice en français (FWA) [TZ10] est un algorithme d'optimisation méta-heuristique issu de l'intelligence des essaims proposé par Tan en 2010, inspiré initialement par l'explosion des feux d'artifice dans le ciel, en particulier dans les festivals de printemps chinois où de nombreux feux d'artifice sont déclenchés pour générer des étincelles. L'idée de base derrière l'algorithme de fireworks est que la façon dont les feux d'artifices explosent dans le ciel est similaire à la façon de chercher la solution optimale dans les algorithmes de renseignement en essaim. Quand un feu d'artifice explose, un certain nombre

d'étincelles sont générées autour de lui. Le nombre et l'amplitude des étincelles générées varient d'un feu d'artifice à un autre, selon leur qualité. Les feux d'artifices de bonne qualité génèrent plus d'étincelles dans une petite amplitude, et les feux d'artifices avec une mauvaise qualité génèrent moins d'étincelles dans une grande amplitude. La figure 3.4 montre deux exemples d'explosion du feu d'artifice, une mauvaise et une bonne explosion.

FIGURE 3.4: Un exemple montrant deux explosions de jeux d'artifice



(a) Explosion Bonne (b) Explosion Mauvaise

À l'instar des autres algorithmes d'intelligence d'essaim, l'algorithme FWA comprend quatre parties : l'opérateur d'explosion, la règle de cartographie, l'opérateur de mutation et la stratégie de sélection. L'algorithme génère initialement une population de N feux d'artifice aléatoire, également appelée population d'individus. Ensuite, chaque feu d'artifice de la population explose et génère des étincelles autour de lui. Le nombre et l'amplitude de ces étincelles générées sont déterminés par l'opérateur d'explosion. Après cela, un opérateur de mutation gaussienne sera appliqué à chaque étincelle pour garder la diversité de la population. Enfin, l'algorithme conserve le meilleur individu dans la population et sélectionne le reste $(N-1)$ individus pour la prochaine génération en fonction de leurs distances. Ces étapes et ces stratégies sont décrites en détail comme suit :

3.5.1 Opérateur d'explosion

Lors de la première itération, le FWA génère N feux d'artifices aléatoires, chaque feu d'artifice génère des étincelles en utilisant des opérations d'explosion. L'opérateur d'explosion joue un rôle clé dans l'algorithme FWA. Il est responsable du calcul du nombre et de l'amplitude des étincelles générées. L'opérateur d'explosion se compose de deux paramètres ; Le premier est le nombre d'étincelles S_i et il est calculé en utilisant l'équation suivante :

$$S_i = m \times \frac{Y_{max} - f(x_i) + \varepsilon}{\sum_{i=1}^{N}(Y_{max} - f(x_i)) + \varepsilon} \tag{3.6}$$

Où S_i est le nombre d'étincelles pour chaque individu ou feu d'artifice, m est une constante représente le nombre total d'étincelles et Y_{max} est la valeur objective du pire individu parmi les

N individus de la population. La fonction $f(x_i)$ représente la fonction objective de l'individu x_i, tandis que le dernier paramètre : ε est utilisé pour empêcher le dénominateur de devenir nul.

Le deuxième paramètre est l'amplitude des étincelles A_i ; elle est calculée comme suit :

$$A_i = \widehat{A} \times \frac{f(x_i) - Y_{min} + \varepsilon}{\sum_{i=1}^{N}(f(x_i) - Y_{min}) + \varepsilon} \tag{3.7}$$

Où A_i désigne l'amplitude de chaque individu, $\widehat{A}$ est un nombre constant qui représente la somme de toutes les amplitudes, tandis que Y_{min} est la valeur objective du meilleur individu parmi les N individus. La signification de la fonction $f(xi)$ et du paramètre ε est la même que celle mentionnée dans la définition précédente.

L'amplitude d'explosion sera utilisée pour déterminer le déplacement de chaque étincelle générée autour du feu artificiel explosé. FWA génère différents mouvements aléatoires pour assurer la diversité de la population en fonction du nombre d'étincelles et de l'amplitude d'explosion de chaque feu d'artifice. L'opération de déplacement est définie comme suit :

$$\Delta x_i^k = x_i^k + U(-A_i, A_i) \tag{3.8}$$

Où $U(-A_i, A_i)$ désigne le nombre aléatoire uniforme dans les intervalles de l'amplitude A_i, et x_i^k représente la $k^{ième}$ position du $i^{ième}$ feu d'artifice.

3.5.2 Règle de mappage

La règle de mappage est utilisée pour maintenir les étincelles dans l'espace de recherche réalisable, si une étincelle se trouve dans la limite, sa position sera mappée à une nouvelle position dans la bordure. La règle de mappage est définie comme suit :

$$x_i = x_{min} + |x_i| \bmod (x_{max} - x_{min}) \tag{3.9}$$

Où x_i représente la $i^{ème}$ position de l'étincelle, tandis que x_{max} et x_{min} représentent la limite maximale et minimale de la position i.

3.5.3 L'opérateur de mutation gaussienne

L'opérateur de mutation gaussienne est utilisé pour maintenir la diversité de la population et pour faire rechercher dans tout l'espace de recherche global, étant donné une position d'un

individu désigné par x_i^k, l'opérateur gaussien est calculé comme suit :

$$x_i^k = x_i^k \times g \tag{3.10}$$

Où g est un nombre aléatoire dans la distribution gaussienne.

$$g = Gaussian(1, 1) \tag{3.11}$$

3.5.4 Stratégie de sélection

Après avoir appliqué l'opérateur d'explosion, l'opérateur de mutation et la règle de mappage, le FWA sélectionne les étincelles générées pour passer à la génération suivante. Dans FWA, la meilleure étincelle est toujours conservée et les autres$N-1$ étincelles sont sélectionnées à l'aide d'une stratégie à distance. Habituellement, la méthode euclidienne est utilisée pour calculer la distance entre deux étincelles ; Il est désigné par le terme d et défini comme suit :

$$d(x_i, x_j) = |f(x_i) - f(x_j)| \tag{3.12}$$

Où $f(x_i)$ et $f(x_j)$ sont des exercices individuels x_i et x_j respectivement.

Enfin, une méthode de roulette est utilisée pour calculer la possibilité de sélectionner un individu.

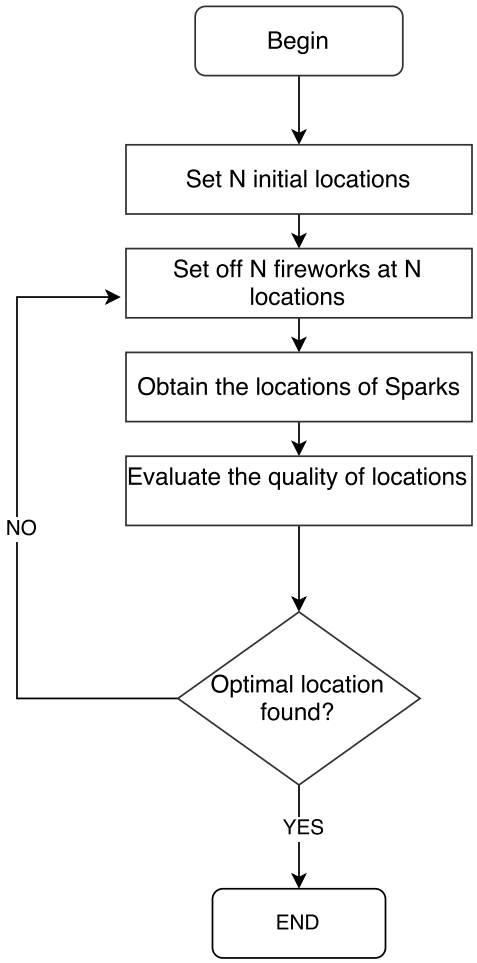
$$p(x_i) = \frac{R(x_i)}{\sum_{j \in k} R(x_j)} \tag{3.13}$$

Où $R(x_i) = \sum_{j=1}^{K} d(x_i, x_j)$ représente la somme des distances entre l'individu x_i et tous les autres individus, K est un ensemble contient toutes les étincelles générées ainsi que le N feux d'artifice. La figure 3.5 montre l'organigramme de l'algorithme du feu d'artifice.

3.6 Description de l'approche proposée

Étant donné que l'algorithme des fireworks est conçu pour traiter uniquement des problèmes d'optimisation continue, il ne peut pas être appliqué directement pour résoudre le problème de détection des communautés, ce qui constitue un problème d'optimisation discrète. Par conséquent, nous avons proposé un nouvel algorithme de fireworks discret, appelé

FIGURE 3.5: Diagramme de l'algorithme FWA



DMFWA, pour résoudre ce problème. Cette section est organisée comme suit : premièrement, le format de codage de la solution est donné. Après, la fonction objective de l'algorithme est présentée. Ensuite, notre stratégie d'initialisation proposée et l'opérateur de mutation sont décrits. Enfin, le cadre de l'algorithme proposé est élaboré.

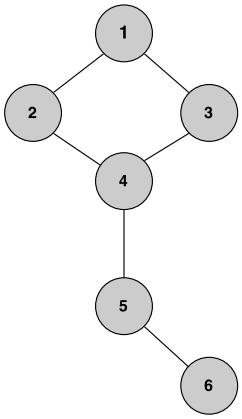
3.6.1 Représentation des individus

D'une manière générale, dans les algorithmes d'intelligence d'essaim, la représentation des individus, également appelée le codage, a un impact important sur la résolution du problème lui-même. Dans FWA, une bonne représentation des feux d'artifice peut réduire l'espace de recherche et accélère la convergence de l'algorithme. Dans cette approche, nous adoptons la représentation string-based.

Dans cette représentation, le nombre de positions est égal au nombre de nœuds dans le réseau. Un individu arbitraire X_i représente la structure communautaire du réseau où chaque position X_i^k représente l'étiquette $L(k)$ du nœud k. Par exemple, si $X_i^1 = X_i^3 = 1$, les nœuds 1 et 3 sont dans la même communauté marquée par 1, nous pouvons également

écrire, $L(1) = L(3) = 1$. Ce schéma de représentation est plus facile à coder et à décoder. En outre, le nombre de communautés sera déterminé automatiquement, et sera égal au nombre d'étiquettes uniques.

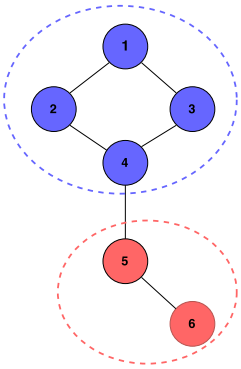
FIGURE 3.6: Encodage de la solution



(a) Graphe du réseau

Nodes	1	2	3	4	5	6
Firework	1	1	1	1	2	2

(b) Représentation d'un jeu d'artifice dans l'algorithme FWA



(c) Structure communautaire du réseau

Un exemple de codage de la solution et son réseau correspondant est illustré dans la figure 3.6. Comme l'indique la figure 3.6a, le réseau se compose de six nœuds numérotés de 1 à 6. Une solution optimale possible est donnée dans la figure 3.6b. Les nœuds 1 à 4 sont regroupés dans la première communauté marquée par 1, tandis que les nœuds5 et 6 sont dans la deuxième communauté marquée par 2. Cette solution est traduite par la structure de la communauté montrée dans la figure 3.6c.

3.6.2 Fonction objective

Plusieurs fonctions objectives ont été proposées pour résoudre des problèmes de détection communautaire. Dans cette approche, la densité de modularité (D) est utilisée comme fonction objective dans notre algorithme proposé. La densité de la modularité a été expliquée en détail dans la section **??**.

3.6.3 Initialisation de la première population

L'initialisation de la première population joue un rôle important ; Une meilleure initialisation peut générer de bonnes solutions et réduire le temps de recherche. Le FWA utilise une stratégie aléatoire pour initialiser la première population de feux d'artifice ; Il ne prend pas en compte toute connaissance préalable du problème spécifié. Dans cet article, nous présentons une nouvelle stratégie d'initialisation des feux d'artifice qui tient compte de la structure de réseau d'origine. Cette approche repose sur l'influence des voisins du nœud. La stratégie est la suivante :

1. Initialiser une population de N fireworks (N est égal à 50 feux d'artifice). 2. Pour chaque feu d'artifice X_i, $X_i^k = k$; Chaque nœud sera dans sa communauté. 3. Pour chaque feu d'artifice, sélectionnez un nœud aléatoire, obtenez le nœud k des voisins de j qui a le plus

haut degré. Affectez tous les voisins de k à la même communauté. Si le nœud j choisie au hasard n'a pas de voisins, l'algorithme sélectionne un autre.

3.6.4 L'opérateur de la mutation

La mutation est une opération importante dans l'algorithme des feux d'artifice. Il est responsable de la mise à jour des positions individuelles après chaque itération. L'opérateur de déplacement et la stratégie de mutation gaussienne est habituellement utilisée pour mettre à jour la position de l'individu dans l'algorithme original des feux d'artifice. Cependant, ces deux opérateurs ont été conçus initialement pour traiter des valeurs continues, et ils ne peuvent pas être utilisés dans notre algorithme proposé où les valeurs sont dans un espace de recherche discret.

Dans cette approche, nous proposons un nouvel opérateur de mutation basé sur la stratégie de propagation d'étiquettes introduite dans [Gon+14]. Cette stratégie utilise des étiquettes pour attribuer chaque nœud à une communauté et les nœuds ayant la même étiquette sont considérés comme étant dans la même communauté. L'étiquette du nœud de mise à jour de la stratégie de propagation de l'étiquette, en fonction de ses étiquettes de quartier. L'étiquette qui apparaît fréquemment dans son quartier est utilisée comme nouvelle étiquette. L'opérateur de mutation nouvellement défini est défini comme suit :

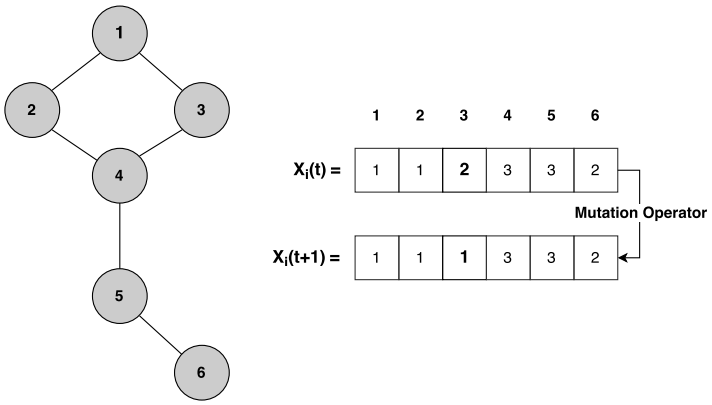
$$x_i^k = \begin{cases} x_i^k & if \quad random(0,1) \geq sigmoid(A_i) \\ Nbest_j & if \quad random(0,1) < sigmoid(A_i) \end{cases} \tag{3.14}$$

Où x_i^k est la $k^{éme}$ position actuelle du feu d'artifice i, qui représente l'étiquette actuelle du nœud k. $random(0,1)$ est un nombre généré au hasard entre 0 et 1. $Nbest_k$ est l'étiquette utilisée la plus fréquente par les voisins du nœud k. $sigmoid(A_i)$ est la fonction sigmoïde de l'amplitude A_i du feu artificiel i. L'amplitude A_i est calculée comme dans l'algorithme original des feux d'artifices en utilisant 3.7. La fonction sigmoïde est définie comme suit :

$$sigmoid(x) = \frac{1}{1 + \exp^{-x}} \tag{3.15}$$

Un exemple illustratif de cet opérateur de mutation nouvellement défini pour l'algorithme discret de feux d'artifice se trouve à la figure 3.7. Sur cette figure, $X_i(t)$ représente l'emplacement actuel du feu d'artifice X_i. L'étiquette du nœud 3 est 2, ce qui signifie qu'il se trouve dans la communauté marquée par 2 avec le nœud 6. Lors de la mise à jour de l'emplacement actuel du feu d'artifice vers le nouvel emplacement $X_i(t+1)$, l'étiquette du nœud 3 est changée en 1 parce que l'étiquette 1 est l'étiquette utilisée fréquemment dans ses nœuds de voisinage.

FIGURE 3.7: Description de la stratégie de mutation proposée



3.7 Expérimentation et discussion des résultats

Cette section présente et discute les résultats obtenus à partir de l'évaluation de notre méthode proposée sur les réseaux synthétiques et réels. Il montre également les résultats de la comparaison de notre méthode avec les trois autres méthodes : CNM, Infomap et GA-Net.

L'algorithme CNM est une méthode d'optimisation de la modularité gourmande rapide, proposée par Clauset et al dans [CNM04]. Cette méthode est en fait une mise en œuvre rapide de l'algorithme Girvan-Newman original. Cet algorithme est abrégé par l'acronyme CNM qui représente les premiers caractères du nom de l'auteur. Le deuxième algorithme est Infomap, qui a été proposé par [RB08]. Cet algorithme utilise une nouvelle théorie de l'information pour détecter les communautés dans un réseau complexe. Le dernier est l'algorithme GA-Net. Il a été proposé par Pizzuti [Piz08]. Cet algorithme utilise un algorithme génétique à objectif unique pour optimiser une fonction de condition physique, appelée score communautaire (CS), pour détecter les communautés à l'intérieur d'un réseau.

3.7.1 Paramètres expérimentaux

Nous avons choisi deux critères d'évaluation pour évaluer notre méthode proposée : la modularité (Q) et l'information Mutuelle Normale (NMI). L'information de mutation normalisée (NMI) mesure la similitude entre deux communautés, le véritable connu La communauté du réseau et celle détectée. Supposons que A et B sont deux partitions d'un réseau et C est la matrice de confusion, où C_{ij} est le nombre de nœuds qui se trouvent dans les deux communautés i et j des partitions A et B respectivement. Ensuite, $NMI(A, B)$ est calculé comme suit :

$$NMI(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij}\log(C_{ij}N/C_{i.}C_{.j})}{\sum_{i=1}^{C_A} C_{i.}\log(C_{i.}/N) + \sum_{j=1}^{C_B} C_{.j}\log(C_{.j}/N)} \tag{3.16}$$

Ou $C_A(C_B)$ est le nombre de communautés dans la partition $A(B)$, $C_{i.}(C_{.j})$ est la somme

des éléments de C dans la ligne i (colonne j) et N est le nombre de nœuds du réseau. Si $A = B$, alors $NMI(A, B) = 1$. Si A et B sont totalement différents, alors $NMI(A, B) = 0$.

3.7.2 Résultats d'évaluation sur des benchmarks

Nous avons d'abord évalué notre méthode proposée sur les réseaux de référence étendus GN pour voir son efficacité face à d'autres méthodes. Le GN a étendu les réseaux de référence proposés par Lancichinetti et al [LFR08] est une extension du réseau classique de référence GN proposé par Girvan et Newman dans [New06]. Il se compose de 128 nœuds divisés en quatre communautés de même taille que 32 nœuds. Le degré moyen de chaque nœud est 16. μ est un paramètre de mélange, utilisé pour contrôler le pourcentage de nœuds entre les nœuds de sa communauté et le total des nœuds du réseau. Lorsque $\mu < 0.5$, le réseau possède une structure communautaire forte. Au contraire, la structure de la communauté est ambiguë, et la détection de sa structure sera difficile.

Nous avons comparé notre méthode proposée (DMFWA) aux algorithmes Informap, CNM et GA-Net sur des réseaux étendus de 10 GN en rangeant la valeur du paramètre de mélange μ de 0,05 à 0,5. Les résultats de ces expériences sont donnés dans les figures 3.8 and 3.9.

La figure 3.8 montre les valeurs moyennes de NMI obtenues à partir d'algorithmes différents sur 30 courses, avec un paramètre de mélange μ allant de 0,05 à 0,5. On peut observer à partir de cette figure que lorsque $\mu \leq 0.2$, l'algorithme proposé et les trois autres algorithmes peuvent détecter avec succès la structure de la communauté exacte du réseau. Lorsque $\mu > 0.2$, la structure de la communauté du réseau devient de plus en plus ambiguë. Les algorithmes Infomap, CNM et GA-Net sont rapidement devenus infructueux et n'ont pas permis de détecter la structure de la communauté de réseau exacte. Infomap montre d'abord sa faiblesse et sa capacité de détection diminue de façon spectaculaire quand $\mu \geq 0.2$. La capacité de détection des algorithmes CNM et GA-Net diminue de $\mu \geq 0.25$ et 0.3 $\mu \geq 0.3$ respectivement. Au contraire, notre algorithme proposé montre sa prédominance sur les algorithmes comparés, et sa capacité de détection est stable.

Les valeurs de modularité moyenne (Q) obtenues à partir de différents algorithmes sont illustrées à la figure 3.9. À partir des valeurs rapportées dans cette figure, on peut observer que lorsque $\mu \leq 0.15$. Tous les algorithmes détectent avec grand succès la structure de la communauté de chaque réseau généré. Cependant, à partir de $\mu = 0.2$, la tâche de détection devient difficile et la capacité de détection de chaque algorithme diminue. L'algorithme CNM a donné le pire résultat de détection parmi les autres algorithmes, lorsque $\mu = 0.2$. La valeur de modularité obtenue à partir de cet algorithme est égale à 0,5, qui est plus petite que les autres valeurs obtenues, GA-Net (0.5201), Infomap (0.5598) et notre méthode (0.5982). Cette diminution des valeurs de modularité continue considérablement et atteint le maximum à $\mu = 0.35$. A cette valeur de μ , la structure communautaire du réseau généré est très ambiguë. Les algorithmes Infomap, CNM et GA-Net ne permettent pas de détecter les communautés. Parallèlement, notre algorithme proposé réussit à les détecter, avec une valeur de modularité égale à 0,3236, supérieure aux autres valeurs, y compris : Infomap (0.2063), GA-Net (0.1659) et CNM (0.1336). Ces valeurs de modularité rapportées dans la figure 3.9 montrent clairement l'avantage de notre algorithme proposé par rapport aux autres algorithmes comparés.

FIGURE 3.8: Les valeurs moyennes de la NMI obtenus sur 30 exécutions sur le benchmark de réseaux GN

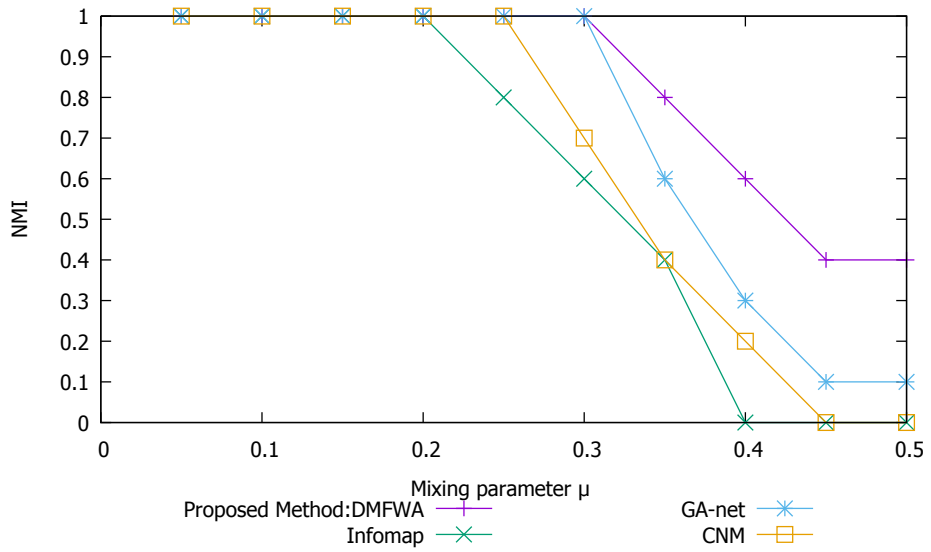
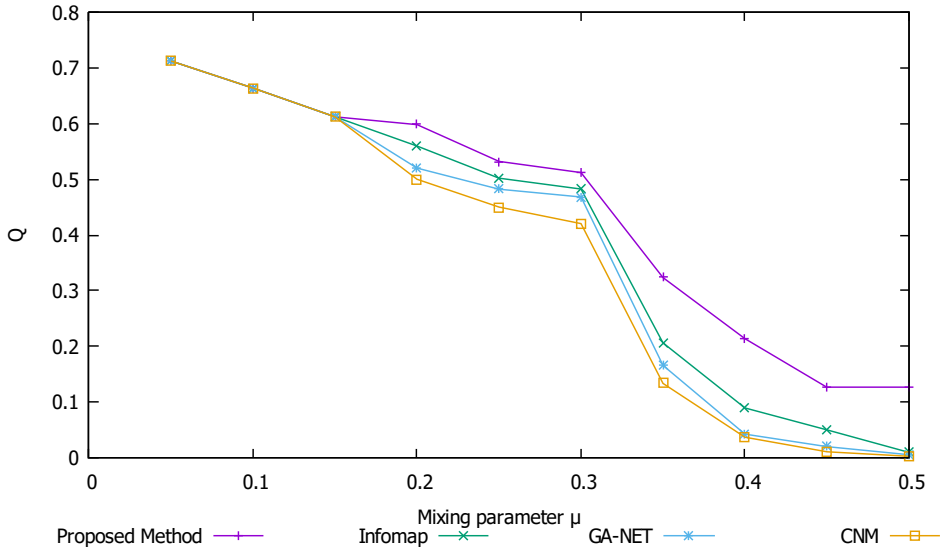


FIGURE 3.9: Les valeurs moyennes de Q obtenus sur 30 exécutions sur le benchmark de réseaux GN



Le tableau 3.1 montre les valeurs de modularité statistique obtenues sur 30 trajets individuels de chaque algorithme sur les réseaux de référence étendus GN avec des valeurs de μ différentes. Un test statistique non paramétrique utilisant le test de somme de rang de Wilcoxon [Wil45 ; Gar+09] a également été réalisé à un niveau de signification de 5% (0,05) sur les données de modularité (Q). Comme nul Hypothèse (H_0), nous avons supposé qu'il n'y a pas de différence significative entre les valeurs moyennes des deux ensembles d'observations (valeurs de modularité). Alors que l'hypothèse alternative (H_1) est qu'il existe une différence

significative dans les valeurs moyennes des deux ensembles. Lorsque la valeur p est inférieure à 0,05, l'hypothèse nulle sera rejetée. Ceci indique que notre méthode est statistiquement plus efficace que l'algorithme comparé.

Toutes les valeurs p enregistrées dans le tableau 3.1 sont inférieures à 0,05, ceci est contraire à l'hypothèse nulle. Cela signifie que l'hypothèse nulle sera rejetée, et les valeurs moyennes de modularité obtenues par notre méthode proposée sont statistiquement significatives. À partir de ces valeurs p, nous concluons que notre méthode proposée surpasse les trois autres algorithmes sur les réseaux de référence étendus GN. Sauf pour la première expérience où les valeurs p sont assez proches de 1, ce qui signifie qu'il n'y a pas de différence significative entre les valeurs de modularité obtenues par les quatre algorithmes. Les valeurs moyennes rapportées dans le même tableau prouvent cela.

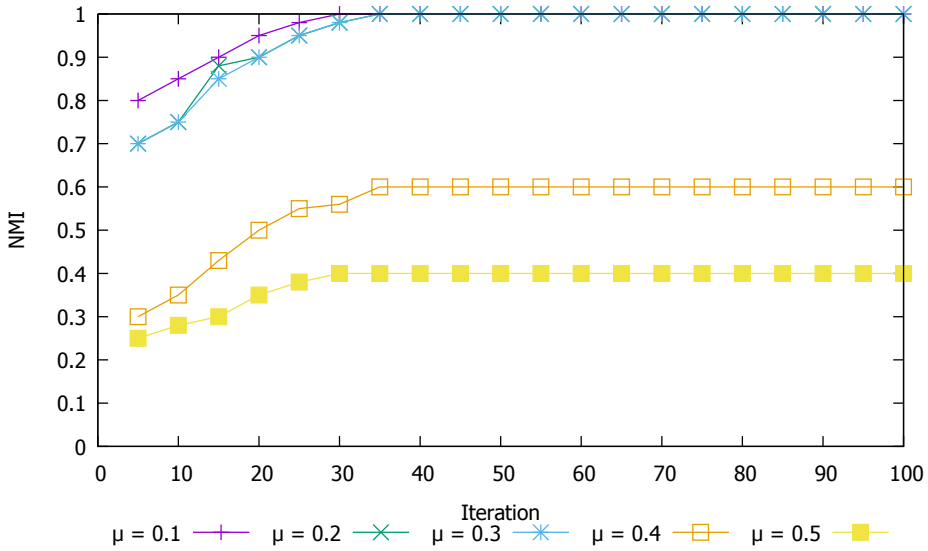
TABLE 3.1: Les résultats statistiques obtenus sur plus de 30 exécutions sur le benchmark de réseaux GN. La "valeur p" est produite par le test de somme de rang de Wilcoxon en comparant notre méthode proposée avec d'autres algorithmes

Réseau	Critère d'évaluation	Algorithme proposé	Infomap	GA-net	CNM
GN ($\mu = 0.1$)	Q_{avg}	**0.6638**	0.6638	0.6638	0.6638
	p-value	×	0.8493	0.8483	0.8653
GN ($\mu = 0.2$)	Q_{avg}	**0.5982**	0.5598	0.5201	0.5000
	p-value	×	0.0388	0.0254	0.0198
GN ($\mu = 0.3$)	Q_{avg}	**0.5121**	0.4825	0.4683	0.4205
	p-value	×	0.0238	0.0191	0.0138
GN ($\mu = 0.4$)	Q_{avg}	**0.2138**	0.0895	0.0425	0.0369
	p-value	×	0.0205	0.0175	0.0126
GN ($\mu = 0.5$)	Q_{avg}	**0.1265**	0.0098	0.0056	0.0028
	p-value	×	0.0385	0.0162	0.0119

Pour examiner la convergence de notre algorithme proposé, nous l'avons évalué sur les réseaux de référence étendus GN avec le paramètre de mélange $\mu \in [0.1, 0.2, 0.3, 0.4, 0.5]$ sur un certain nombre d'itérations allant de 10 à 100. Les résultats de cette expérience sont montrés à la figure 3.10. Comme on le voit sur cette figure, on peut constater que notre algorithme proposé converge rapidement vers une valeur stable de NMI, indépendamment de la valeur de μ, à partir de l'itération 30. Ceci est dû à l'impact de la stratégie d'initialisation Et l'opérateur de mutation proposé dans cet article qui rend l'algorithme plus rapide en convergence vers une solution optimale. Une conclusion qui peut être tirée des résultats présentés dans les figures 3.8, 3.9, et 3.10, c'est que notre algorithme proposé surpasse considérablement tous les autres algorithmes comparés et possède une grande capacité à détecter la structure de la communauté du réseau.

Nous avons présenté dans cette section les résultats expérimentaux obtenus à partir de notre méthode proposée et de trois autres algorithmes bien connus sur les réseaux de référence étendus GN. Cependant, cette référence synthétique présente certaines limites, telles que la taille de chaque communauté, ce qui est le même pour toutes les communautés. Ainsi, certaines

FIGURE 3.10: Convergence de l'algorithme proposé



fonctionnalités importantes des réseaux réels ne peuvent pas être reflétées. En raison de cela, nous avons effectué d'autres expériences sur des réseaux du monde réel. Les résultats de ces expériences seront présentés dans la section suivante.

3.7.3 Résultats d'évaluation sur des réseaux réels

Nous présentons maintenant les résultats de l'évaluation de notre algorithme proposé (DMFWA) sur quatre réseaux réels avec des structures communautaires connues, y compris le réseau de clubs de karaté de Zachary [Zac77], le réseau social Dolphins [Lus03], le livre de Krebs 'sur le réseau de politique américaine 1 et American College Football Network [GN02]. Ces réseaux sont décrits comme suit :

1. Zachary's Karate Club : C'est l'un des réseaux les plus populaires qui a été largement utilisé dans la littérature. Ce réseau représente un réseau social des membres d'un club de karaté étudié par [38]. Il se compose de 34 nœuds, chaque nœud représentant un membre de ce club. Cependant, au cours de cette étude, un conflit a eu lieu entre l'administrateur (nœud 34) et l'instructeur (nœud 1), ce qui a conduit à séparer les membres du club en deux groupes de clubs plus petits. Un groupe, formé autour de l'administrateur, compte 16 membres. Les 18 autres membres ont formé un groupe autour de l'instructeur.

2. Le réseau social des dauphins Ce réseau social représente les résultats de l'étude du comportement de 62 dauphins à nerf dans sept ans dans Doubtful Sound, Nouvelle-Zélande [23].

3. Le libellé de Krebs BooksKrebs 'sur le réseau de la politique contient 105 livres de politique américaine, recueillis par Krebs et vendus par Amazon.com.

4. American College Football American College Football Network [10], qui comprend 115

équipes de football américaines de 12 régions différentes des États-Unis, jouant un jeu de championnat entre eux pendant la saison de l'automne 2000. Chaque lien représente un jeu entre deux équipes, un total de 616 jeux a été joué.

Le Tableau 3.2 montre les caractéristiques de chaque réseau.

TABLE 3.2: Caractéristiques des réseaux réels utilisés

Réseau	Nœuds	Arêtes	Nombre de communautés réelles
Karate	34	78	2
Dolphin	62	159	2
Politics	105	613	3
Football	115	616	12

TABLE 3.3: Résultats d'expérimentation sur les réseaux réels et comparaison avec les algorithmes Infomap, CNM et GA-Net

Réseau	Critère d'évaluation	Algorithme proposé	Infomap	CNM	GA-net
Karate	NMI_{max}	**1**	0.7851	0.7691	0.6921
	NMI_{avg}	**1**	0.7672	0.7603	0.6805
	Q_{max}	**0.4198**	0.4091	0.4084	0.4103
	Q_{avg}	**0.4038**	0.3991	0.3989	0.4022
	p-value	×	0.0388	0.0358	0.0392
Dolphin	NMI_{max}	**0.9650**	0.7049	0.6980	0.8069
	NMI_{avg}	**0.9542**	0.705	0.6873	0.8005
	Q_{max}	**0.5093**	0.4312	0.4569	0.4743
	Q_{avg}	**0.5013**	0.4308	0.4498	0.4704
	p-value	×	0.0328	0.0348	0.0382
Politics	NMI_{max}	**0.7981**	0.5218	0.5301	0.6809
	NMI_{avg}	**0.7971**	0.5125	0.5298	0.6795
	Q_{max}	0.5271	0.4265	0.4298	**0.5816**
	Q_{avg}	0.5081	0.4201	0.4193	**0.5803**
	p-value	×	0.0326	0.0342	0.0674
Football	NMI_{max}	**0.9230**	0.6853	0.7695	0.9120
	NMI_{avg}	**0.9205**	0.6845	0.7687	0.9105
	Q_{max}	**0.6046**	0.3494	0.5386	0.6012
	Q_{avg}	**0.6041**	0.5485	0.5381	0.6007
	p-value	×	0.0219	0.0291	0.0315

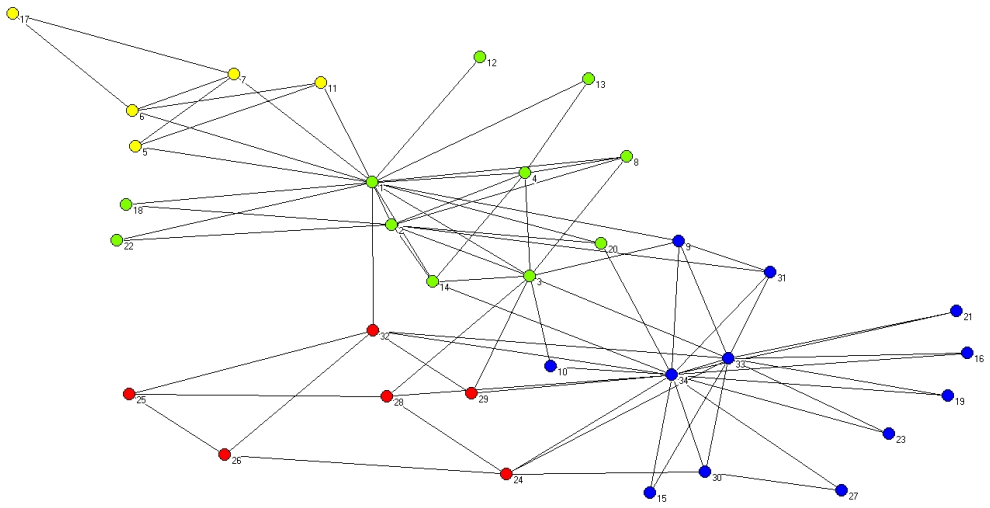
Les résultats de ces évaluations sont présentés dans le tableau 3.3 ainsi que les valeurs p obtenues par le test de base de Wilcoxon à un niveau de signification de 5% (0,05) sur les données de modularité (Q). Nous pouvons observer à partir de ce tableau que notre algorithme proposé fonctionne bien sur ces réseaux réels et surpasse les autres algorithmes comparés. Cependant, L'algorithme GA-Net montre une meilleure performance par rapport à notre algorithme proposé sur le réseau Krebs 'Books on US Politics, avec une valeur moyenne de

modularité égale à 0.5803 et 0.5081 pour notre algorithme. Cette petite différence de valeur est due au fait que l'algorithme CNM optimise une fonction de conditionnement physique efficace, appelée score communautaire, pour détecter les communautés à l'intérieur d'un réseau qui a bien fonctionné sur ce réseau.

Toujours du tableau 3.3, on peut constater qu'il existe une différence intéressante entre les valeurs de NMI obtenues à partir de notre algorithme proposé et les trois autres. Cette différence est clairement visible sur le réseau du club Karate, lorsque notre algorithme proposé détecte avec succès la bonne structure de la communauté. La valeur NMI obtenue sur ce réseau est égale à 1 pour notre algorithme, 0,7851 pour Infomap, 0,7691 pour CNM et 0,6921 pour l'algorithme GA-Net. D'après les valeurs NMI obtenues sur le réseau Krebs 'Books on US Politics, nous pouvons constater que notre algorithme donne la meilleure valeur NMI, égale à 0,7981. Ce qui signifie que notre algorithme proposé, réussit à détecter une structure communautaire suffisamment proche de celle correcte.

À partir des valeurs p du tableau 3.3, qui sont toutes inférieures à 0,05, nous concluons que notre méthode proposée est statistiquement plus efficace que les trois autres algorithmes sur tous les réseaux présentés, à l'exception du réseau Politics Books, où l'algorithme GA-Net donne des meilleurs résultats avec une valeur p value égale à 0,0674. La figure 3.11 montre la structure communautaire du réseau Zachary's Karate Club obtenu par notre algorithme. Comme le montre cette figure, le réseau est divisé en quatre communautés ayant une valeur de modularité Q égale à 0,4198. Les nœuds à couleur bleue représentent les membres qui ont rejoint le groupe de l'administrateur du club (nœud 34). Alors que les nœuds avec la couleur verte sont des membres qui ont rejoint le groupe de l'instructeur (nœud 1). Les nœuds aux couleurs rouge et jaune représentent les membres qui ont choisi de rester loin du conflit.

FIGURE 3.11: Structure communautaire du réseau Zachary's Karate Club obtenue par notre algorithme proposé



3.8 Conclusion

L'algorithme Fireworks (FWA) est un nouvel algorithme évolutif d'intelligence d'essaim qui a été largement utilisé dans de nombreux domaines comme un algorithme d'optimisation continue. Cependant, ses applications à des problèmes discrets manquent presque. Dans cet article, nous présentons une nouvelle variante modifiée discrète de l'algorithme de feux d'artifice pour résoudre le problème de la détection de la communauté dans les réseaux complexes en maximisant la valeur de la densité de la modularité. Nos principales contributions sont la conception d'une nouvelle stratégie d'initiation et de mutation, basée sur la méthode de propagation de l'étiquette.

Les résultats de l'évaluation de notre algorithme proposé sur le monde réel et les réseaux synthétiques par rapport à trois autres algorithmes montrent la supériorité de notre algorithme proposé par rapport aux autres. Au fur et à mesure que l'avenir fonctionne, l'algorithme proposé dans cet article utilise la densité de la modularité comme une fonction à objectif unique. Cependant, il peut être étendu à une technique d'optimisation multi-objective. Nous prévoyons également tester l'algorithme proposé sur les réseaux signés, car les expériences présentées dans cet article sont toutes effectuées sur des réseaux non signés.

Chapitre 4

Les Systèmes de recommandation

Sommaire

4.1 Introduction

L'importance croissante du Web comme support pour les transactions électroniques et commerciales a bien servi pour le développement de la technologie des systèmes de recommandation. Aujourd'hui, on trouve rarement un site web qui n'intègre pas la fonctionnalité de recommandation. Nous discuterons dans ce chapitre les méthodes et les algorithmes de création de systèmes de recommandation les plus populaires. Nous présentons aussi un nouveau système de recommandation des dépôts GitHub pertinents pour les développeurs. Ce système utilise une approche de filtrage collaboratif, dans notre approche on modélise les comportements des utilisateurs en tant que matrice utilisateur-élément afin qu'on puisse appliquer les différentes méthodes de recommandation, comme le calcul des similarités entre les utilisateurs (développeurs) et les éléments (projets). Nous avons évalué ce système de recommandation sur un ensemble de données réelles en utilisant des mesures d'évaluation connues, la conception et la mise en œuvre de ce système seront discutées en détail dans les sections suivantes.

4.2 Les systèmes de recommandation

Les personnes dans les médias sociaux prennent une variété de décisions sur une base quotidienne. Ces décisions concernent l'achat d'un produit, l'achat d'un service, l'ajout d'un ami et la location d'un film, entre autres. Une personne fait souvent face à de nombreuses options à choisir. Ces diverses options, la poursuite de l'optimalité, et la connaissance limitée que chaque individu a créé un désir d'aide extérieure. À certains moments, nous avons recours aux moteurs de recherche pour les recommandations. Cependant, les résultats dans les moteurs de recherche sont rarement adaptés à nos goûts particuliers et dépendent des requêtes, indépendamment des personnes qui les recherchent.

Des applications et des algorithmes sont développés pour aider les individus à décider facilement, rapidement et avec plus de précision. Ces algorithmes sont adaptés aux goûts des individus de telle sorte que des recommandations personnalisées sont disponibles pour eux. Ces algorithmes sont appelés algorithmes de recommandation ou systèmes de recommandation.

Les systèmes de recommandation sont couramment utilisés pour la recommandation de produits. Leur objectif est de recommander des produits qui seraient intéressants pour les individus. Formellement, un algorithme de recommandation prend un ensemble d'utilisateurs U et un ensemble d'éléments I et apprend une fonction f telle que :

$$f : U \times I \rightarrow \mathbb{R} \tag{4.1}$$

En d'autres termes, l'algorithme apprend une fonction qui attribue une valeur réelle à chaque paire d'éléments utilisateur (u, i), où cette valeur indique à quel point l'utilisateur u est intéressé par l'élément i. Cette valeur indique la note attribuée par l'utilisateur à l'élément i. L'algorithme de recommandation ne se limite pas à la recommandation d'élément et peut être généralisé pour recommander des personnes et du matériel, tels que des annonces ou du

contenu.

4.3 Types d'évaluations dans les systèmes de recommandation

La conception des algorithmes de recommandation est influencée par le système utilisé pour le suivi des notations. Les notations sont souvent spécifiées sur une échelle qui indique le niveau spécifié que d'appréciation ou d'antipathie de l'élément en question. Il est possible que les notations soient des valeurs continues, comme dans le cas du moteur de recommandation de blagues Jester [Gol+01], dans lequel les valeurs peuvent prendre n'importe quelle valeur entre -10 et 10. Ceci est cependant relativement rare. Habituellement, les évaluations sont basées sur des intervalles, où un ensemble discret de nombres ordonnés sont utilisés pour quantifier comme aimer ou ne pas aimer. De telles évaluations sont appelées notations basées sur les intervalles. Par exemple, une échelle de notation de 5 points pourrait être tirée de l'ensemble -2, -1, 0, 1, 2, dans laquelle une note de -2 indique une aversion extrême, et une note de 2 indique une forte affinité à l'article. D'autres systèmes peuvent tirer les notes de l'ensemble 1, 2, 3, 4, 5.

FIGURE 4.1: Exemple d'évaluations d'intervalles à 5 points

Le nombre d'évaluations possibles peut varier selon le système en question. L'utilisation de classements à 5, 7 et 10 points est particulièrement courante. Le système d'évaluation 5 étoiles, illustré dans la figure 4.1, est un exemple d'évaluation à intervalle. Au long de chacune des évaluations possibles, nous avons indiqué une interprétation sémantique du niveau d'intérêt de l'utilisateur. Cette interprétation peut varier légèrement d'un site web à l'autre, comme Amazon ou Netflix. Par exemple, Netflix utilise un système de notation 5 étoiles dans lequel le point 4 étoiles correspond à «vraiment aimé», et le point central 3 étoiles correspond à «aimé». Il y a donc trois notes favorables et deux notations défavorables dans Netflix, ce qui conduit à une échelle de notation déséquilibrée. Dans certains cas, il peut y avoir un nombre pair d'évaluations possibles et la note neutre peut être manquante. Cette approche est appelée un système de notation à choix forcé.

FIGURE 4.2: Exemple d'évaluations ordinales utilisées dans les évaluations des cours de l'Université de Stanford

On peut également utiliser des valeurs catégoriques ordonnées telles que : (Fortement en désaccord, En désaccord, Neutre, D'accord, Fortement d'accord) afin d'atteindre les mêmes objectifs. En général, ces notes sont appelées notes ordinales et le terme dérive du concept d'attributs ordinaux. Un exemple de notation ordinale, utilisé dans les formulaires d'évaluation de cours de l'Université de Stanford, est illustré dans la figure 4.2. Dans les évaluations binaires, l'utilisateur peut exprimer uniquement son admiration ou mépris pour l'élément et rien d'autre. Par exemple, les valeurs peuvent être 0, 1 ou des valeurs non spécifiées. Les valeurs non spécifiées doivent être prédites aux valeurs 0-1. Un cas particulier d'évaluation est celui des notations unaires, dans lesquelles il existe un mécanisme permettant à un utilisateur de spécifier un goût pour un élément mais aucun mécanisme pour spécifier une aversion. Les notations unaires sont particulièrement fréquentes, en particulier dans le cas d'ensembles de données à rétroaction implicite [HND15 ; HKV08 ; OK+98]. Dans ces cas, les préférences des clients sont dérivées de leurs activités plutôt que de leurs notations explicitement spécifiées. Par exemple, le comportement d'achat d'un client peut être converti en notes unaires. Lorsqu'un client achète un article, il peut être considéré comme une préférence pour l'article. Cependant, le fait de ne pas acheter un objet dans un large éventail de possibilités n'indique pas toujours une aversion. De même, de nombreux réseaux sociaux, tels que Facebook, utilisent des boutons de réactions, qui permettent d'exprimer le goût d'un article. Cependant, il n'existe aucun mécanisme pour spécifier l'antipathie pour un élément. Le paramètre de rétroaction implicite peut être considéré comme l'analogue d'achèvement de la matrice du problème d'apprentissage positif non marqué (PU) dans la classification des données [HND15].

4.4 Modèles de base des systèmes de recommandation

Les modèles de base des systèmes de recommandation fonctionnent avec deux types de données : (i) les interactions utilisateur-élément, telles que les notations ou les comportements d'achat ; et (ii) les informations des utilisateurs et des éléments tels que les profils textuels ou les mots clés pertinents. Les méthodes qui utilisent le premier sont appelées méthodes de filtrage collaboratif, tandis que les méthodes qui utilisent le dernier sont appelées méthodes de recommandation basées sur le contenu.

Notez que les systèmes basés sur le contenu utilisent également les matrices de notation dans la plupart des cas, bien que le modèle se concentre généralement sur les évaluations d'un seul utilisateur plutôt que sur celles de tous les utilisateurs. Dans les systèmes de recommandation basés sur la connaissance, les recommandations sont basées sur des exigences d'utilisateur explicitement spécifiées. Au lieu d'utiliser l'évaluation historique ou d'acheter des données, des bases de connaissances externes et des contraintes sont utilisées pour créer les recommandations. Certains systèmes de recommandation combinent ces différents aspects

pour créer des systèmes hybrides. Les systèmes hybrides peuvent combiner les forces de divers types de systèmes de recommandation pour créer des techniques plus performantes dans une grande variété de paramètres. Dans ce qui suit, nous discuterons brièvement ces modèles de base.

4.4.1 Modèles de filtrage collaboratif

Les modèles de filtrage collaboratif utilisent la puissance de collaboration des évaluations fournies par plusieurs utilisateurs pour faire des recommandations. Le principal défi dans la conception de méthodes de filtrage collaboratif est que les matrices de notation sous-jacentes sont rares. Prenons l'exemple d'une application de films dans laquelle les utilisateurs spécifient des classements indiquant qu'ils aiment ou n'aiment pas les films spécifiques. La plupart des utilisateurs n'auraient vu qu'une petite fraction du grand nombre de films disponibles. En conséquence, la plupart des notations ne sont pas spécifiées. Les notations spécifiées sont également appelées notations observées. Tout au long de cette thèse, les termes «spécifié» et «observé» seront utilisés de manière interchangeable. Les notations non spécifiées seront appelées "non observées" ou "manquantes".

L'idée de base des méthodes de filtrage collaboratif est que ces notations non spécifiées peuvent être prédites parce que les notations observées sont souvent fortement corrélées entre divers utilisateurs et éléments. Par exemple, considérons deux utilisateurs nommés Alice et Bob, qui ont des goûts très similaires. Si les notations, qui ont toutes deux été spécifiées, sont très similaires, leur similarité peut être identifiée par l'algorithme sous-jacent. Dans de tels cas, il est très probable que les notations dans lesquelles un seul d'entre eux a spécifié une valeur soient également susceptibles d'être similaires. Cette similarité peut être utilisée pour faire des inférences sur des valeurs incomplètement spécifiées. La plupart des modèles de filtrage collaboratif mettent l'accent sur l'utilisation de corrélations inter-éléments ou de corrélations inter-utilisateurs pour le processus de prédiction. Certains modèles utilisent les deux types de corrélations. En outre, certains modèles utilisent des techniques d'optimisation soigneusement conçues pour créer un modèle d'apprentissage à peu près de la même manière qu'un classificateur crée un modèle d'apprentissage à partir des données étiquetées. Ce modèle est ensuite utilisé pour prédire les valeurs manquantes dans la matrice, de la même manière qu'un classificateur prédit les étiquettes de test manquantes. Il existe deux types de méthodes couramment utilisées dans le filtrage collaboratif, appelées méthodes basées sur la mémoire et méthodes basées sur les modèles.

4.4.1.1 Méthodes de filtrage collaboratif basées sur la mémoire

Les méthodes basées sur la mémoire sont également appelées algorithmes de filtrage collaboratif basés sur le voisinage. Ces algorithmes ont été parmi les premiers algorithmes de filtrage collaboratif, dans lesquels les notes des combinaisons utilisateur-élément sont prédites en fonction de leur voisinage. Ces voisinages peuvent être définis de deux façons :

Filtrage collaboratif basé sur l'utilisateur : Dans ce cas, les évaluations fournies par les utilisateurs partageant les mêmes idées d'un utilisateur cible A sont utilisées pour formuler les recommandations pour A. Ainsi, l'idée de base est de déterminer les utilisateurs similaires

à l'utilisateur ciblé A et recommander des notes pour les notations non observées de A en calculant les moyennes pondérées des notes de ce groupe de pairs. Par conséquent, si Alice et Bob ont évalué les films d'une manière similaire dans le passé, alors on peut utiliser les notes observées d'Alice sur le film Terminator pour prédire la notes non-observée de Bob sur ce film. En général, les k utilisateurs les plus similaires à Bob peuvent être utilisés pour faire des prédictions d'évaluation pour lui. Les fonctions de similarité sont calculées entre les lignes de la matrice d'évaluations pour découvrir des utilisateurs similaires.

Filtrage collaboratif basé sur des éléments : Pour effectuer les prédictions de notation pour l'élément cible B par l'utilisateur A, la première étape consiste à déterminer un ensemble S d'éléments les plus similaires à l'élément cible B. Les classements dans l'ensemble d'éléments S spécifiés par A sont utilisés pour prédire si l'utilisateur A aimera l'élément B. Par conséquent, les notes de Bob sur des films de fiction scientifiques similaires comme Alien et Predator peuvent être utilisées pour prédire sa note pour le film Terminator. Les fonctions de similarité sont calculées entre les colonnes de la matrice d'évaluations pour découvrir les éléments similaires.

Les avantages des techniques basées sur la mémoire sont qu'elles sont simples à mettre en œuvre et que les recommandations qui en résultent sont souvent faciles à expliquer. D'un autre côté, les algorithmes basés sur la mémoire ne fonctionnent pas très bien avec les matrices de notation éparses. Par exemple, il peut être difficile de trouver des utilisateurs suffisamment similaires à Bob, qui ont évalué Gladiator. Dans de tels cas, il est difficile de prédire de manière robuste l'évaluation de Gladiator par Bob. En d'autres termes, de telles méthodes pourraient ne pas couvrir complètement les prédictions d'évaluation. Néanmoins, l'absence de couverture n'est souvent pas un problème, lorsque seuls les éléments de premier ordre sont requis.

4.4.1.2 Méthodes de filtrage collaboratif basées sur les modèles

Dans les méthodes basées sur des modèles, les méthodes d'apprentissage automatique et d'exploration de données sont utilisées dans le contexte des modèles prédictifs. Dans les cas où le modèle est paramétré, les paramètres de ce modèle sont appris dans le contexte d'un cadre d'optimisation. Des exemples de telles méthodes basées sur des modèles incluent des arbres de décision, des modèles basés sur des règles, des méthodes bayésiennes et des modèles de facteurs latents. Beaucoup de ces méthodes, telles que les modèles à facteurs latents, ont un niveau élevé de couverture, même pour les matrices de notation clairsemée.

Même si les algorithmes de filtrage collaboratif basés sur la mémoire sont appréciés pour leur simplicité, ils ont tendance à être de nature heuristique et ne fonctionnent pas bien dans tous les contextes. Cependant, la distinction entre les méthodes basées sur la mémoire et sur les modèles est quelque peu artificielle, car les méthodes basées sur la mémoire peuvent également être considérées comme des modèles basés sur la similarité, même s'ils sont heuristiques. Il est également montré que certaines variantes de méthodes basées sur le voisinage peuvent être formellement exprimées comme des modèles basés sur la régression. Les modèles de facteurs latents ont été popularisés dans les dernières années à la suite du concours du prix Netflix, bien que des algorithmes similaires aient été proposés beaucoup plus tôt dans le contexte d'ensembles de données incomplets (génériques) [AP01]. Récemment, il a été montré que

certaines combinaisons de méthodes basées sur la mémoire et sur les modèles fournissent des résultats très précis [Kor08].

4.4.2 Systèmes de recommandation basés sur le contenu

Dans les systèmes de recommandation basés sur le contenu, les attributs descriptifs des éléments sont utilisés pour faire des recommandations. Le terme "contenu" fait référence à ces descriptions. Dans ces méthodes, les évaluations et le comportement d'achat des utilisateurs sont combinés avec les informations de contenu disponibles sur les articles. Par exemple, considérons une situation dans laquelle John a fortement évalué le film Terminator, mais nous n'avons pas accès aux évaluations des autres utilisateurs. Par conséquent, les méthodes de filtrage collaboratif sont exclues. Cependant, la description de l'article de Terminator contient des mots-clés de genre similaires à ceux d'autres films de fiction scientifique, tels qu'Alien et Predator. Dans de tels cas, ces films peuvent être recommandés à John.

Dans les méthodes basées sur le contenu, les descriptions d'articles, qui sont étiquetées avec des évaluations, sont utilisées comme données d'apprentissage pour créer un modèle de classification ou de régression spécifique à l'utilisateur. Pour chaque utilisateur, les documents d'apprentissage correspondent aux descriptions des articles qu'il a achetés ou notés. La variable de classe correspond aux notations spécifiées ou au comportement d'achat. Ces documents d'apprentissage sont utilisés pour créer un modèle de classification ou de régression, qui est spécifique à l'utilisateur en question. Ce modèle spécifique à l'utilisateur est utilisé pour prédire si l'individu correspondant aimera un article pour lequel sa notation ou son comportement d'achat est inconnu.

Les méthodes basées sur le contenu ont certains avantages à formuler des recommandations pour de nouveaux éléments lorsque des données d'évaluation suffisantes ne sont pas disponibles pour cet élément. En effet, d'autres éléments ayant des attributs similaires peuvent avoir été évalués par l'utilisateur actif. Par conséquent, le modèle supervisé sera en mesure de tirer parti de ces notations en conjonction avec les attributs d'élément pour faire des recommandations même s'il n'y a pas d'historique des notes pour cet élément.

Les méthodes basées sur le contenu présentent également plusieurs inconvénients :

1. Dans de nombreux cas, les méthodes basées sur le contenu fournissent des recommandations évidentes en raison de l'utilisation de mots-clés ou de contenu. Par exemple, si un utilisateur n'a jamais consommé un élément avec un ensemble particulier de mots-clés, un tel élément n'a aucune chance d'être recommandé. En effet, le modèle construit est spécifique à l'utilisateur et les connaissances de la communauté provenant d'utilisateurs similaires ne sont pas exploitées. Ce phénomène tend à réduire la diversité des éléments recommandés, ce qui n'est pas souhaitable.
2. Même si les méthodes basées sur le contenu sont efficaces à fournir des recommandations pour de nouveaux éléments, elles ne sont pas efficaces pour fournir des recommandations aux nouveaux utilisateurs. En effet, le modèle d'apprentissage de l'utilisateur cible doit utiliser l'historique de ses évaluations. En fait, il est généralement important d'avoir un grand nombre de notations disponibles pour l'utilisateur cible afin de faire des prédic-

tions robustes sans sur-ajustement.

Par conséquent, les méthodes basées sur le contenu ont des différents échanges de systèmes de filtrage collaboratifs.

Bien que la description susmentionnée fournisse la vue conventionnelle basée sur l'apprentissage des méthodes basées sur le contenu, une vue plus large de ces méthodes est parfois utilisée. Par exemple, les utilisateurs peuvent spécifier des mots clés pertinents dans leurs propres profils. Ces profils peuvent être associés à des descriptions d'articles afin de faire des recommandations. Une telle approche n'utilise pas d'évaluations dans le processus de recommandation, et est donc utile dans les scénarios de démarrage à froid. Cependant, de telles méthodes sont souvent considérées comme une classe distincte de systèmes de recommandation, connus sous le nom de systèmes basés sur les connaissances, car les métriques de similarité sont souvent basées sur les connaissances actuelles. Les systèmes de recommandation basés sur les connaissances sont souvent considérés comme étroitement liés aux systèmes de recommandation basés sur le contenu, et on se demande parfois s'il existe une démarcation claire entre les deux classes de méthodes [Smy07].

4.4.3 Systèmes de recommandation basés sur les connaissances

Les systèmes de recommandation basés sur les connaissances sont particulièrement utiles dans le contexte d'articles qui ne sont pas achetés très souvent. Les exemples comprennent des biens immobiliers, des automobiles, des demandes touristiques, des services financiers ou des produits de luxe coûteux. Dans de tels cas, des notations suffisantes peuvent ne pas être disponibles pour le processus de recommandation. Comme les articles sont rarement achetés et avec différents types d'options détaillées, il est difficile d'obtenir un nombre suffisant d'évaluations pour une instanciation spécifique (c'est-à-dire une combinaison d'options) de l'article en question. Ce problème est également rencontré dans le contexte du problème de démarrage à froid, lorsque des évaluations suffisantes ne sont pas disponibles pour le processus de recommandation. De plus, la nature des préférences des consommateurs peut évoluer au fil du temps lorsqu'il s'agit de tels articles. Par exemple, le modèle d'une voiture peut évoluer de façon significative au cours de quelques années, ce qui fait que les préférences peuvent montrer une évolution correspondante. Dans d'autres cas, il peut être difficile de capter complètement l'intérêt des utilisateurs avec des données historiques telles que les notations. Un élément particulier peut avoir des attributs qui lui sont associés qui correspondent à ses diverses propriétés, et un utilisateur peut être intéressé uniquement par des éléments ayant des propriétés spécifiques. Par exemple, les voitures peuvent avoir plusieurs marques, modèles, couleurs, options de moteur et options intérieures, et les intérêts des utilisateurs peuvent être réglementés par une combinaison très spécifique de ces options. Ainsi, dans ces cas, le domaine de l'élément a tendance à être complexe en termes de propriétés variées, et il est difficile d'associer des classements suffisants au grand nombre de combinaisons disponibles.

De tels cas peuvent être traités avec des systèmes de recommandation basés sur la connaissance, dans lesquels les notations ne sont pas utilisées pour les recommandations. Le processus de recommandation est plutôt effectué sur la base de similarités entre les exigences du client et les descriptions d'éléments, ou l'utilisation de contraintes spécifiant les exigences de l'utili-

sateur. Le processus est facilité par l'utilisation de bases de connaissances, qui contiennent des données sur les règles et les fonctions de similarité à utiliser pendant le processus. En fait, les bases de connaissances sont si importantes pour le fonctionnement efficace de ces méthodes que l'approche tire son nom de ce fait. La spécification explicite des exigences entraîne un plus grand contrôle des utilisateurs sur le processus de recommandation. Dans les systèmes collaboratifs et basés sur le contenu, les recommandations sont entièrement déterminées par les actions / évaluations passées de l'utilisateur, les actions / évaluations de ses pairs ou une combinaison des deux. Les systèmes basés sur les connaissances sont uniques en ce sens qu'ils permettent aux utilisateurs de spécifier explicitement ce qu'ils veulent.

Les systèmes de recommandation basés sur les connaissances peuvent être classés sur la base du type d'interface (et des connaissances correspondantes) utilisé pour atteindre les objectifs susmentionnés :

4.4.4 Systèmes de recommandation basés sur les contraintes

Dans les systèmes basés sur les contraintes [Fel+11 ; FB08], les utilisateurs spécifient généralement des exigences ou des contraintes (par exemple, des limites inférieures ou supérieures) sur les attributs d'élément. Les règles spécifiques au domaine sont utilisées pour faire correspondre les exigences de l'utilisateur aux attributs de l'élément. Ces règles représentent les connaissances spécifiques au domaine utilisées par le système. De telles règles pourraient prendre la forme de contraintes spécifiques au domaine sur les attributs des articles (par exemple, "Les voitures avant l'année 1970 n'ont pas de régulateur de vitesse"). En outre, les systèmes basés sur les contraintes créent souvent des règles qui associent les attributs des utilisateurs aux attributs des éléments (par exemple, «les investisseurs plus âgés n'investissent pas dans des produits à très haut risque»). Dans de tels cas, les attributs de l'utilisateur peuvent également être spécifiés dans le processus de recherche. En fonction du nombre et du type de résultats renvoyés, l'utilisateur peut avoir l'opportunité de modifier ses exigences d'origine. Par exemple, ils peuvent relâcher certaines de leurs contraintes lorsque trop peu de résultats sont renvoyés, ou ils peuvent ajouter plus de contraintes. Ce processus de recherche est répété de manière interactive jusqu'à ce que l'utilisateur arrive aux résultats souhaités.

4.4.5 Systèmes de recommandation basés sur les cas

Dans les systèmes de recommandation basés sur des cas [Bri+05 ; Tre00 ; LR05 ; Smy07], des cas spécifiques sont spécifiés par l'utilisateur en tant que cibles ou points d'ancrage. Les métriques de similarité sont définies sur les attributs d'élément pour récupérer des éléments similaires dans ces cas. Les métriques de similarité sont souvent soigneusement définies d'une manière spécifique au domaine. Par conséquent, les métriques de similarité forment la connaissance de domaine qui est utilisée dans de tels systèmes. Les résultats retournés sont souvent utilisés comme de nouveaux cas cibles avec quelques modifications par l'utilisateur. Par exemple, lorsqu'un utilisateur voit un résultat renvoyé, ce qui est presque similaire à ce qu'il souhaite, il peut réémettre une requête avec cette cible, mais avec certains attributs modifiés au gré de l'utilisateur. Ce processus interactif est utilisé pour guider l'utilisateur vers les éléments d'intérêt.

Notez que dans les deux cas, le système offre à l'utilisateur la possibilité de modifier ses exigences spécifiées. Cependant, la manière dont cela est fait est différente dans les deux cas. Dans les systèmes basés sur des cas, des exemples (ou des cas) sont utilisés comme points d'ancrage pour guider la recherche en combinaison avec des mesures de similarité. Les interfaces de critique sont particulièrement populaires pour exprimer une rétroaction dans de tels systèmes, où les utilisateurs modifient itérativement un ou plusieurs attributs d'un élément préféré dans chaque itération. Dans les systèmes basés sur les contraintes, des règles (ou des contraintes) sont utilisées pour guider la recherche. La forme du guidage peut souvent prendre la forme de systèmes basés sur la recherche, où les utilisateurs spécifient leurs contraintes avec une interface basée sur la recherche.

Comment l'interactivité dans les systèmes de recommandation basés sur la connaissance est-elle atteinte ? Cette orientation se déroule selon une ou plusieurs des méthodes suivantes :

1. Systèmes conversationnels : dans ce cas, les préférences de l'utilisateur sont déterminées de manière itérative dans le contexte d'une boucle de rétroaction. La raison principale en est que le domaine de l'élément est complexe et que les préférences de l'utilisateur ne peuvent être déterminées que dans le contexte d'un système conversationnel itératif.

2. Systèmes basés sur la recherche : Dans les systèmes basés sur la recherche, les préférences des utilisateurs sont obtenues en utilisant une séquence prédéfinie de questions telles que : "Préférez-vous une maison dans une banlieue ou dans la ville ?" Dans certains cas, spécifiques les interfaces de recherche peuvent être configurées afin de permettre la spécification des contraintes de l'utilisateur.

3. Recommandation basée sur la navigation : Dans la recommandation basée sur la navigation, l'utilisateur spécifie un certain nombre de demandes de modification à l'élément actuellement recommandé. Grâce à un ensemble itératif de demandes de changement, il est possible d'arriver à un élément souhaitable. Voici un exemple d'une demande de modification spécifiée par l'utilisateur, lorsqu'une maison spécifique est recommandée : «Je voudrais une maison semblable à environ 5 milles à l'ouest de la maison actuellement recommandée.» De tels systèmes de recommandation sont également considérés comme systèmes critiques [MR11].

Il convient de noter que les systèmes fondés sur les connaissances et sur le contenu dépendent tous deux de façon significative des attributs des éléments. En raison de leur utilisation des attributs de contenu, les systèmes basés sur la connaissance héritent des mêmes inconvénients que les systèmes basés sur le contenu. Par exemple, tout comme les systèmes basés sur le contenu, les recommandations dans les systèmes basés sur la connaissance peuvent parfois être évidentes parce que l'utilisation des notations de la communauté (c'est-à-dire par les pairs) n'est pas exploitée. En fait, les systèmes fondés sur les connaissances sont parfois considérés comme les «cousins» des systèmes fondés sur le contenu [Smy07]. La principale différence est que les systèmes basés sur le contenu tirent des leçons du comportement passé des utilisateurs, alors que les systèmes de recommandation basés sur les connaissances recommandent en fonction de la spécification par l'utilisateur de leurs besoins et de leurs intérêts. Par conséquent, dans la littérature, les recommandations fondées sur les connaissances sont

considérées comme une catégorie distincte des recommandations fondées sur le contenu. Ces distinctions reposent à la fois sur les objectifs de ces systèmes et sur le type de données d'entrée utilisées.

4.4.6 Systèmes de recommandation basés sur les utilitaires

Dans les systèmes de recommandation basés sur les utilitaires, une fonction d'utilité est définie sur les caractéristiques du produit afin de calculer la probabilité qu'un utilisateur apprécie l'élément [GMM98]. Le défi central des méthodes basées sur les utilitaires est de définir une fonction d'utilité appropriée pour l'utilisateur à portée de main. Il convient de noter que tous les schémas de recommandation, qu'ils soient basés sur la collaboration, sur le contenu ou sur les connaissances, classent implicitement les éléments recommandés en fonction de leur valeur (ou utilité) perçue pour l'utilisateur cible. Dans les systèmes basés sur les utilitaires, cette valeur d'utilité est basée sur une fonction connue à priori. En ce sens, de telles fonctions peuvent être considérées comme une sorte de connaissance externe. Par conséquent, les systèmes basés sur les utilitaires peuvent être considérés comme un cas spécifique de systèmes de recommandation basés sur la connaissance.

4.4.7 Systèmes de recommandation démographique

Dans les systèmes de recommandation démographique, les informations démographiques sur l'utilisateur sont utilisées pour apprendre aux classificateurs qui peuvent cartographier des données démographiques spécifiques aux notations ou achats. Un ancien système de recommandation appelé Grundy [Ric79], recommandait des livres basés sur la bibliothèque de stéréotypes assemblés manuellement. Les caractéristiques de l'utilisateur ont été recueillies à l'aide d'un dialogue interactif. Le travail dans [Kru97] a observé que les groupes démographiques de la recherche marketing peuvent être utilisés pour recommander des articles. Un autre travail [Paz99] fait des recommandations de pages Web sur la base des caractéristiques démographiques des utilisateurs qui ont fortement évalué une page particulière. Dans de nombreux cas, les informations démographiques peuvent être combinées avec un contexte supplémentaire pour guider le processus de recommandation. Cette approche est liée à la méthodologie des systèmes de recommandation contextuels.

Des techniques plus récentes ont mis l'accent sur l'utilisation de classificateurs pour faire des recommandations. L'un des systèmes intéressants à cet égard était une technique qui extrayait les caractéristiques des pages d'accueil des utilisateurs afin de prédire leur probabilité d'aimer certains restaurants. Les classificateurs basés sur des règles [ASP98] sont souvent utilisés pour relier le profil démographique au comportement d'achat de manière interactive. Bien que l'approche dans [ASP98] n'ait pas été spécifiquement utilisée pour recommander des items spécifiques, elle peut facilement être jumelée à un système de recommandation. Bien que les systèmes de recommandation démographique ne fournissent généralement pas les meilleurs résultats, ils ajoutent de façon significative à la puissance des autres systèmes de recommandation en tant que composante des modèles hybrides ou d'ensemble. Les techniques démographiques sont parfois combinées avec des systèmes de recommandation basés sur la connaissance pour augmenter leur robustesse.

4.4.8 Systèmes de recommandation hybrides

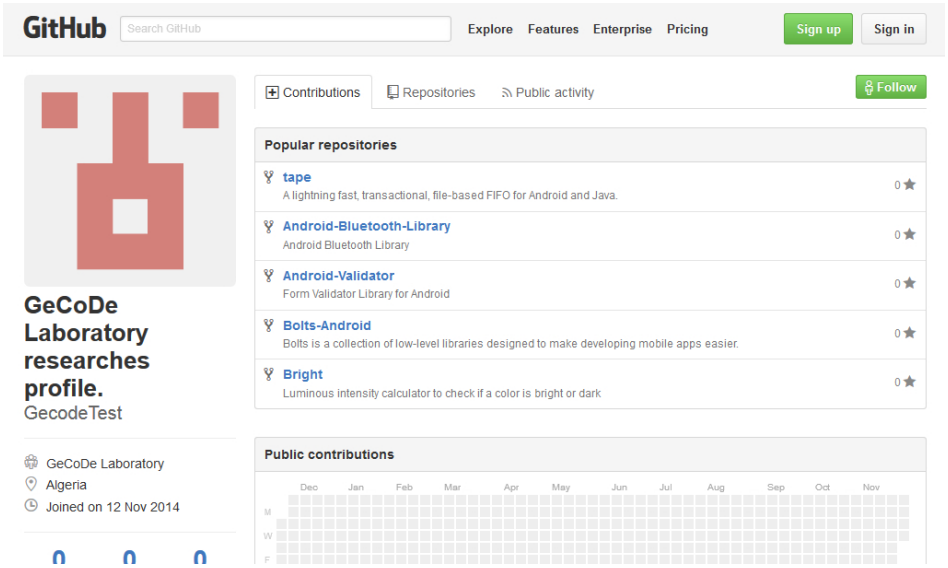
Les trois systèmes susmentionnés exploitent différentes sources d'entrée, et ils peuvent bien fonctionner dans différents scénarios. Par exemple, les systèmes de filtrage collaboratif s'appuient sur des notations communautaires, les méthodes basées sur le contenu reposent sur des descriptions textuelles et les propres notations de l'utilisateur cible, et les systèmes basés sur la connaissance reposent sur des interactions avec l'utilisateur dans le contexte des bases de connaissances. De même, les systèmes démographiques utilisent les profils démographiques des utilisateurs pour faire des recommandations. Il convient de noter que ces différents systèmes utilisent différents types d'entrée et présentent des forces et des faiblesses différentes.

Certains systèmes de recommandation, tels que les systèmes basés sur les connaissances, sont plus efficaces dans les environnements à démarrage à froid où une quantité importante de données n'est pas disponible. D'autres systèmes de recommandation, tels que les méthodes collaboratives, sont plus efficaces lorsque de nombreuses données sont disponibles. Dans de nombreux cas où une plus grande variété d'entrée est disponible, on a la possibilité d'utiliser différents types de systèmes de recommandation pour la même tâche. Dans de tels cas, il existe de nombreuses possibilités d'hybridation, où les divers aspects de différents types de systèmes sont combinés pour atteindre les meilleures recommandations. Les systèmes de recommandation hybride sont étroitement liés au domaine de l'analyse d'ensemble, dans lequel la puissance de multiples types d'algorithmes d'apprentissage automatique est combinée pour créer un modèle plus robuste. Les systèmes de recommandation basés sur un ensemble peuvent non seulement combiner la puissance de plusieurs sources de données, mais aussi améliorer l'efficacité d'une classe particulière de systèmes de recommandation (par exemple, les systèmes collaboratifs) en combinant plusieurs modèles du même type. . Ce scénario n'est pas très différent de celui de l'analyse d'ensemble dans le domaine de la classification des données.

4.5 Un nouveau système de recommandation des projets open source sur GitHub

GitHub [Gitb] est une plateforme de développement de logiciels très populaire et de codage social qui fournit un service d'hébergement de dépôts Git [Gita] sur le Web, permettant à n'importe quel développeur de participer à la documentation, au design, et notamment au codage des projets open source de manière libre. Pour participer à ces activités, un développeur doit tout d'abord créer un compte, lui permettant de partager ses propres projets, cloner des projets d'autres développeurs, ou tout simplement suivre les activités d'autres développeurs, la figure 4.3 montre un exemple de profil GitHub.

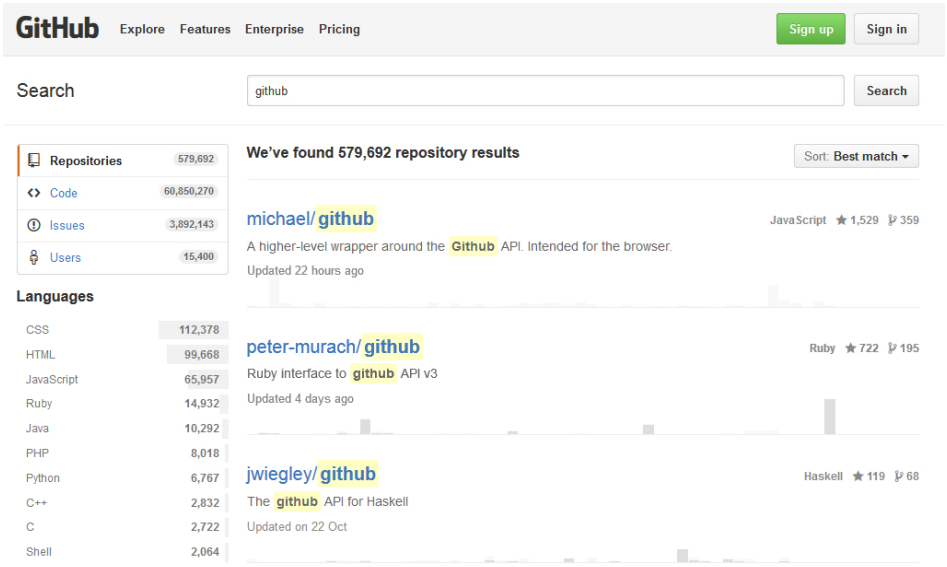
FIGURE 4.3: Exemple d'un profil sur le site GitHub



L'une des fonctionnalités les plus utiles implémentées sur GitHub est la fonctionnalité de copie de projet (fork en anglais), ce qui signifie faire une copie complète d'un projet. Le copiage d'un dépôt permet aux développeurs d'expérimenter librement sur projet sans affecter la copie originale, le copiage est considéré comme la première tâche à accomplir pour contribuer à un projet existant. Une autre fonctionnalité implémentée est la fonction étoile, lorsqu'un développeur donne une étoile à un projet, cela signifie qu'il s'intéresse à ce projet. Par exemple, un développeur intéressé par le développement de jeux mobiles peut donner des étoiles à des bibliothèques de jeux mobiles 2D telles que : AndEngine, LibGDX, cocos1d-x et autres.

Les développeurs cherchent toujours des bons projets open source pour créer des prototypes de produit ou pour améliorer leurs propres projets logiciels avec de nouvelles fonctionnalités, GitHub fournit à ses utilisateurs une fonctionnalité de recherche manuelle sans aucune recommandation automatique fournie, la Figure 4.4 montre un exemple d'une recherche faite sur le site. La recherche de dépôts de projets appropriés peut être une tâche difficile et peut prendre beaucoup de temps, elle peut également interpréter le processus de développement d'un projet, pour cette raison, l'existence d'un système de recommandation automatique pour les dépôts GitHub peut être très utile pour les développeurs afin réduire le temps de recherche et rendre les résultats de recherche plus pertinentes et mieux organisées. Les développeurs peuvent bénéficier différemment d'un tel système, selon leur type de profil et leurs compétences professionnelles, par exemple : un développeur professionnel cherche probablement de nouveaux défis de programmation ou même des opportunités de travail, alors qu'un débutant cherche généralement à améliorer ses compétences. Le problème qui se pose dans ces cas est la façon dont nous pouvons trouver un contenu pertinent sur GitHub et le recommander à un utilisateur.

FIGURE 4.4: Exemple d'une recherche faite sur le site GitHub



4.5.1 Analyse des données GitHub

Nous présentons dans ce qui suit certains travaux de recherche faits sur GitHub, y compris les travaux intéressés par l'analyse de ses données et les systèmes de recommandation proposés pour les différentes fonctionnalités de GitHub.

Dans [Zha+14], les auteurs ont étudié des différents comportements d'utilisateurs sur GitHub pour répondre à certaines questions. La question la plus importante était : "Quels types de comportement d'utilisateurs sont les plus appropriées pour recommander des projets open source pertinents ?". Après avoir effectué plusieurs expériences sur des données réelles recueillies sur GitHub, ils ont constaté que parmi tous les comportements d'utilisateur, le copiage est le plus adapté pour recommander des projets pertinents. Ce résultat est très utile pour nous, en fait, nous utilisons ces informations de copiage dans notre approche pour représenter la matrice utilisateur-objet . Un autre problème connu sur GitHub est l'attribution manuelle des examinateurs des demandes pulls (pull-requests), cette tâche peut prendre du temps, ce qui a conduit les auteurs dans [Yu+14] à proposer un système de recommandation des examinateurs pour les demandes pulls. Ils ont combiné la recherche d'information avec l'analyse des réseaux sociaux en profitant de la sémantique textuelle des demandes pulls, ils ont mis en place un système en ligne pour tester leur approche pour voir comment elle peut aider d'autres développeurs dans l'affectation des examinateurs de demandes pulls. Les résultats d'expérimentation de leur approche sur un ensemble de données réel utilisant les recommandations Top-N ont montré des bonnes performances.

D'autre part, plusieurs travaux ont été intéressés par l'analyse des données de GitHub, comme l'analyse de la structure du réseau des développeurs [Thu+13], et la visualisation des données GitHub [Hel+11].

4.5.2 Définition du problème et cas d'utilisation

Cette section définit formellement le problème abordé dans ce chapitre et montre un cas d'utilisation d'une telle solution avec un exemple illustratif.

On suppose que D est un ensemble de développeurs et R un ensemble de projets, l'objectif est de recommander des projets qui seraient intéressants pour les développeurs. Formellement, un algorithme de recommandation prend les ensembles D et R pour générer une fonction f telle que : $f : D \times R \to B$

En d'autres termes, la fonction attribue une valeur booléenne à chaque paire développeur-projet (d, r), où cette valeur indique si le développeur est intéressé par le projet r. Nous considérons $M = \{m_{ij}\}$ la matrice utilisateur-élément (user-item matrix) où $m_{ij} \in 0, 1$. $m_{ij} = 1$ signifie que le développeur i est déjà intéressé par le projet j tandis que $m_{ij} = 0$ signifie que le développeur n'est probablement pas intéressé par le projet ou ne le sait pas. La tâche est donc de trouver des projets qui peuvent être intéressants pour un développeur et qui ne sont pas déjà dans sa liste.

Nous donnons ici un exemple pour illustrer la problématique étudié dans ce chapitre. Considérons un développeur identifié par "gueno" qui s'intéresse au développement mobile, il s'intéresse déjà à ces projets : ("Libgdx", "ask", "bourbon", "fastadapter", "icepick"), nous considérons qu'il existe six développeurs qui se trouvent actuellement sur le site. Le tableau 4.1 montre un ensemble de données contenant les développeurs et leurs projets préférés, y compris le développeur "gueno". Le but est de recommander des nouveaux projets pour "gueno" en fonction de cet ensemble de données.

Table 4.1: Exemple d'un dataset

Développeur	Projets
gueno	Libgdx, ask, bourbon, fastadapter, icepick
mark	Libgdx, fastadapter, icepick, ExoMedia, FileDownloader
steve	Libgdx, fastadapter, FileDownloader, joda-time-android
ninja	bourbon, fastadapter
tim	ask, joda-time-android
hanna	icepick, FileDownloader, joda-time-android
gosling	icepick, bourbon, ExoMedia, FileDownloader

TABLE 4.2: La matrice utilisateur-élément extraite du dataset

Utilisateur/Élément	Libgdx	ask	bourbon	fastadapter	icepick	ExoMedia	FileDownloader	joda-timeandroid
gueno	1	1	1	1	1	0	0	0
mark	1	0	0	1	1	1	1	0
steve	1	0	0	1	0	0	1	1
ninja	0	0	1	1	0	0	0	0
tim	0	1	0	0	0	0	0	1
hanna	0	0	0	0	1	0	1	1
gosling	0	0	1	0	1	1	1	0

Nous proposons une technique de filtrage collaboratif pour résoudre ce problème. Tout d'abord, nous créons la matrice utilisateur-élément M_{ij} à partir d'informations rapportées dans le tableau 4.1, la matrice M_{ij} est montrée dans le tableau 4.2. Deuxièmement, nous calculons la similarité entre le développeur "gueno" et chaque développeur dans l'ensemble de données en utilisant la matrice du tableau 4.2, Après calcul, nous trient les développeurs en fonction de leurs similarités, le nouveau dataset de développeurs est le suivant :

Marc (3), steve (2), gosling (2), ninja (2), hanna (1), tim (1)

À partir de cette liste, nous pouvons dire que le développeur le plus similaire à "gueno" est "marc" avec 3 projets communs, puis vient les développeurs : "steve", "gosling" et "ninja" avec 2 projets communs. "Hanna" et "tim" ne sont pas similaires à "gueno" puisqu'ils ne partagent que 1 des 8 projets avec lui. Nous concluons que "marc", "steve" et "gosling" sont les développeurs les plus similaires à "gueno", la prochaine étape consiste à choisir des projets parmi les projets communs de ces trois développeurs et les recommandés à "gueno". Les projets communs entre "marc", "steve" et "gosling" sont : "FileDownloader" et "ExoMedia", de sorte que ces deux projets seront recommandés pour "geno" avec la liste triée suivante :

1. FileDownloader 2. ExoMedia

Notre but est d'automatiser les tâches ci-dessus en utilisant des techniques de filtrage collaboratif afin de calculer les similarités entre les développeurs et de trouver les projets pertinents pour eux.

4.6 Description de l'approche proposée

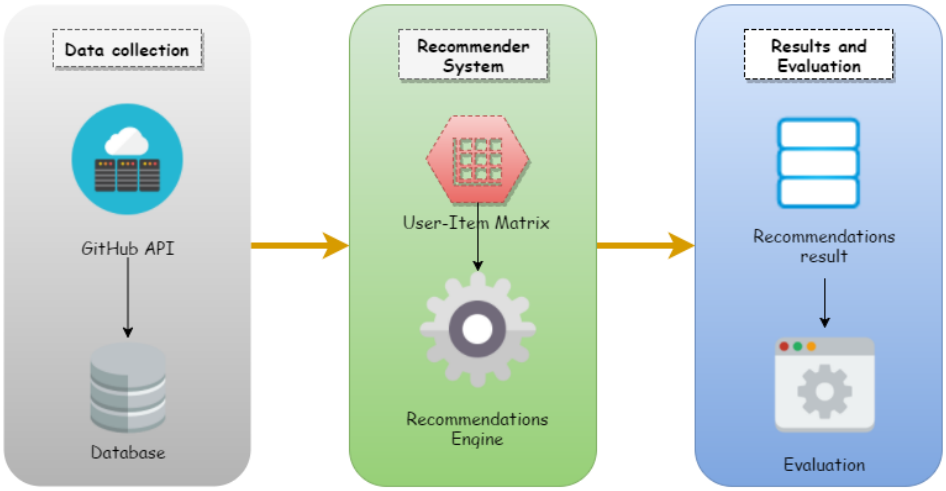
Cette section décrit en détail la conception et la mise en œuvre de notre système de recommandation. Elle présente également certaines méthodologies utilisées dans les systèmes de recommandation.

4.6.1 Architecture du système

Notre objectif est de créer un système capable de recommander automatiquement des projets GitHub pertinents pour les développeurs, la figure 4.5 illustre l'architecture de notre

système.

FIGURE 4.5: Architecture du système



Tout d'abord, les données GitHub sont traitées afin que nous puisse générer la matrice utilisateur-élément. Ces données sont enregistrées sous forme de JSON (JavaScript Object Notation) [JSO], nous commençons par extraire les projets copiés par chaque utilisateur dans la base de données. Le résultat de cette opération est une liste de paires de la forme clé-valeur, la clé représente l'utilisateur et la valeur représente la liste des projets copiés par ce même utilisateur. Deuxièmement, nous générons la matrice utilisateur-élément à partir de ces paires de données. Nous illustrons cette tâche en détail dans la section suivante. L'étape suivante consiste à faire des recommandations pour les développeurs en utilisant cette matrice utilisateur-élément générée. Un système de recommandation est un système qui peut prédire automatiquement les notes d'utilisateur pour un élément ou plusieurs éléments, ou dans l'autre cas, lui proposer une liste d'éléments triés. Ici, nous intéressons à ce deuxième cas particulier. Le système trouve une liste triés des projets pertinents pour un développeur. Enfin, la dernière étape consiste à évaluer notre système de recommandation, les paramètres d'évaluation seront discutés plus loin.

4.6.2 Modèle de matrice utilisateur-élément

Dans les techniques de filtrage collaboratif, une matrice utilisateur-élément est souvent donnée où chaque entrée est une valeur inconnue ou une note attribuée par un utilisateur à un élément. Dans cette approche, nous avons utilisé le comportement de copier un projet pour modéliser cette matrice, ce qui signifie que si un développeur a déjà copié déjà un projet, la valeur sera 1, sinon elle sera 0 (note inconnue). Le tableau 4.3 montre un exemple d'une matrice utilisateur-élément utilisée par notre système. Où $D_1 \dots D_4$ représentent les développeurs et $R_1 \dots R_3$ représentent les projets.

TABLE 4.3: Un exemple d'une matrice utilisateur-élément

Développeur/Projet	R_1	R_2	R_3
D_1	1	1	0
D_2	0	1	0
D_3	1	0	0
D_4	0	0	1

4.6.3 Méthode de recommandation

Dans cette approche, une technique de filtrage collaboratif basée "mémoire" est utilisée vu que notre objectif est de recommander une liste de projets à un développeur. Nous utilisons la méthode de recommandation Top-N. Dans les recommandations Top-N, la tâche consiste à recommander une liste d'éléments qui intéresseront l'utilisateur.

Afin de calculer ces recommandations, la méthode de recommandation Top-N analyse la matrice utilisateur-élément pour trouver des relations entre les utilisateurs ou les éléments. Il existe deux méthodes, la recommandation basée "utilisateur" et la recommandation basée "élément", dans notre approche document, la technique basée "élément" [BHK98] est utilisée.

4.6.3.1 Algorithme de recommandation Top-N basé "élément"

Dans cette section, nous décrivons comment fonctionne l'algorithme de recommandation Top-N basé "élément", au contraire de la technique basée "utilisateur", qui cherche d'abord des utilisateurs similaires à un utilisateur, puis combine leurs notes pour un élément, ce qui permet de prédire la note de l'utilisateur pour cette élément, la technique basés "élément" effectue un filtrage collaboratif directement en trouvant les éléments les plus similaires en calculant la similarité entre eux, les différentes étapes de cet algorithme sont détaillées comme suit :

1. Pour chaque projet, l'algorithme cherche les k projets les plus similaires pour celui-ci, ce nombre k est également appelé le nombre de voisins, les similarités sont calculées à l'aide d'une méthode de calcul de similarité.

2. L'algorithme supprime l'ensemble R des projets similaires extraits dans la première étape. R est l'ensemble des projets déjà copiés par l'utilisateur, cela implique l'élimination de tous les projets similaires déjà prélevés par l'utilisateur. Le résultat sera enregistré dans l'ensemble U.

3. Enfin, l'algorithme calcule la similarité entre chaque projet de l'ensemble U et l'ensemble R, c'est-à-dire calculer les similarités entre les projets copiés par utilisateur et la liste des k projets les plus similaires. Le résultat sera une liste de projets triés de manière décroissante selon leurs similarités.

4.6.3.2 Calcul de similarité

Il existe plusieurs méthodes pour calculer la similarité entre deux éléments. Cependant, beaucoup d'entre eux ne conviennent pas à notre cas (notes binaires, 1 ou 0), ils sont conçus pour des notes cinq étoiles où la note varie entre 0 et 5. Dans notre approche, nous utilisons méthode de calcul de similarité appropriée pour ce type de notes, C'est la méthode Jaccard [Lip99 ; LW71].

Le coefficient de similarité de Jaccard ou l'indice de Jaccard est une mesure pour comparer la similarité et la diversité entre les ensembles (éléments ou utilisateurs), il est défini comme la taille de l'intersection divisée par la taille de l'union des deux ensembles. Étant donné deux éléments i et j, le coefficient de similarité de Jaccard entre eux est désigné par $sim(i, j)$, donné par :

$$sim(i,j) = J = \frac{R_{11}}{R_{01} + R_{10} + R_{11}} \tag{4.2}$$

Où R_{11} représente le nombre de fois que les deux éléments i et j ont reçu une note de 1. R_{01} représente le nombre de fois où l'élément i a reçu une note de 0 et l'élément j a reçu une note de 1. R_{10} représente le nombre de fois où l'élément i a reçu une note de 1 et l'élément j a reçu 0.

Étant donné que notre approche utilise une notation 0/1 pour représenter les notes des utilisateurs pour les projets, le coefficient de similarité de Jaccard est la méthode de calcul de similarité la plus adaptée pour notre système, elle est également simple et facile à mettre en œuvre.

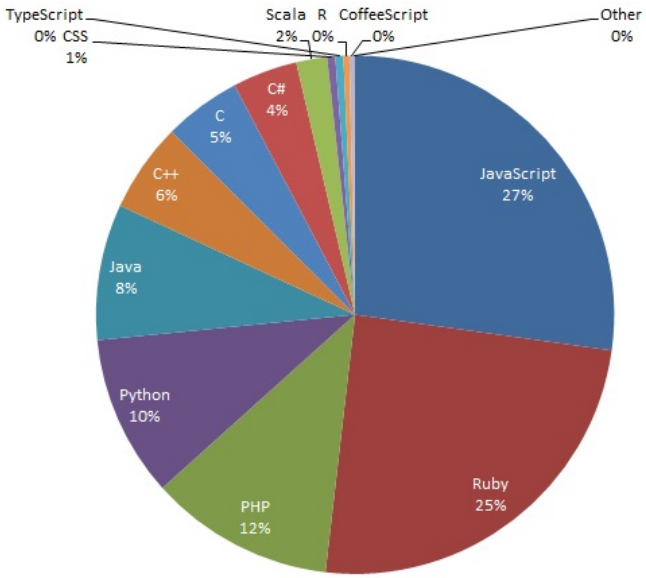
4.7 Dataset

Nous avons utilisé le dataset Mining Software Repositories Challenge 2014 (MSR2014) [Gou13 ; Gou+14], ce dataset a été créé spécialement pour ce défi, il inclut les données des 10 premiers projets logiciels favoris pour les principaux langages de programmation sur GitHub, ce qui donne 90 projets et leurs copies. Le tableau 4 résume quelques informations statistiques sur l'ensemble de données, la figure 4 montre la répartition des langages de programmation par rapport aux projets dans le dataset.

TABLE 4.4: Statistiques de dataset

Données	Nombre
Users	465,198
Repositories	1,204
Issues	126,308
Pull Requests	79,359
Commits	601,080

FIGURE 4.6: Distribution de langages de programmation par rapport aux projets



4.8 Expérimentations et discussion des résultats

Dans cette section, nous présentons les résultats expérimentaux des évaluations de notre système sur un ensemble de données réelles. Nous commencerons par donner les mesures d'évaluation utilisées, ensuite nous décrivons la configuration expérimentale. Enfin, nous discuterons les résultats obtenus à partir de ces expérimentations.

4.8.1 Mesures d'évaluation

4.8.1.1 Rappel, Précision et F-mesure

Ces trois paramètres sont largement utilisés dans la littérature pour évaluer les systèmes de recommandation en termes de pertinence. En fonction de la sélection des éléments pour

les recommandations et leur pertinence, nous pouvons avoir quatre types d'éléments décrits dans le tableau 4.5. A partir de ce tableau, nous pouvons définir des mesures qui utilisent les informations pertinentes fournies par les utilisateurs.

La précision est une mesure qui définit la fraction des éléments pertinents parmi les éléments recommandés :

$$P = \frac{N_{rs}}{N_s} \tag{4.3}$$

Rappel fournit la probabilité de choisir un élément pertinent pour la recommandation :

$$R = \frac{N_{rs}}{N_r} \tag{4.4}$$

Lorsque nous voulons mesurer ces deux évaluations, nous pouvons combiner à la fois la précision et le rappel en prenant leur moyenne harmonique dans la f-mesure :

$$F_1 = \frac{2 \times P \times R}{P + R} \tag{4.5}$$

TABLE 4.5: Partitionnement des éléments en fonction de leur sélection pour la recommandation et leur pertinence

	Sélectionné	Non-Sélectionné	Total
Pertinent	N_{rs}	N_{rn}	N_r
Non-Pertinent	N_{is}	N_{in}	N_i
Total	N_s	N_n	N

4.8.1.2 Erreur Absolue Moyenne (MAE)

L'erreur absolue moyenne (MAE) calcule la différence absolue moyenne entre les notes réelles et les notes prédites, la formule de calcul MAE est donnée par :

$$MAE = \frac{\sum_{ij} |\widehat{r}_{ij} - r_{ij}|}{n} \tag{4.6}$$

Où n est le nombre de notes prédites, $\widehat{r}_{ij}$ est la note prédite, et r_{ij} est la note réelle.

4.8.2 Configuration

Pour la mise en œuvre de notre système, on a utilisé Java sur une machine Linux (Ubuntu 14.04), avec les performances suivantes : Processeur Intel Core I3, RAM DDR3 de 8 Go. Pour l'évaluation, on a utilisé la méthode de validation croisée $k - fold$. Nous avons choisi k égale à 10.

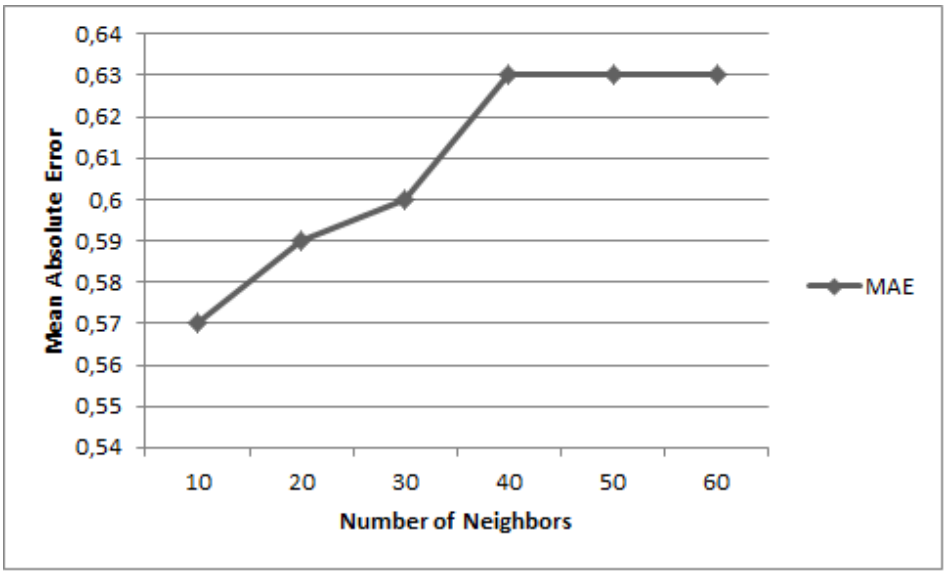
Dans la validation croisée en $K-fold$, l'ensemble de données est divisé de manière aléatoire en K (presque) pièces non superposées de taille égale, appelées plis. La performance totale du modèle est la moyenne des performances k.

4.8.3 Discussion des résultats

Dans cette section, les résultats de l'évaluation de notre système sont discutés et résumés dans différents formats, tableaux et images graphiques, pour montrer les résultats obtenus.

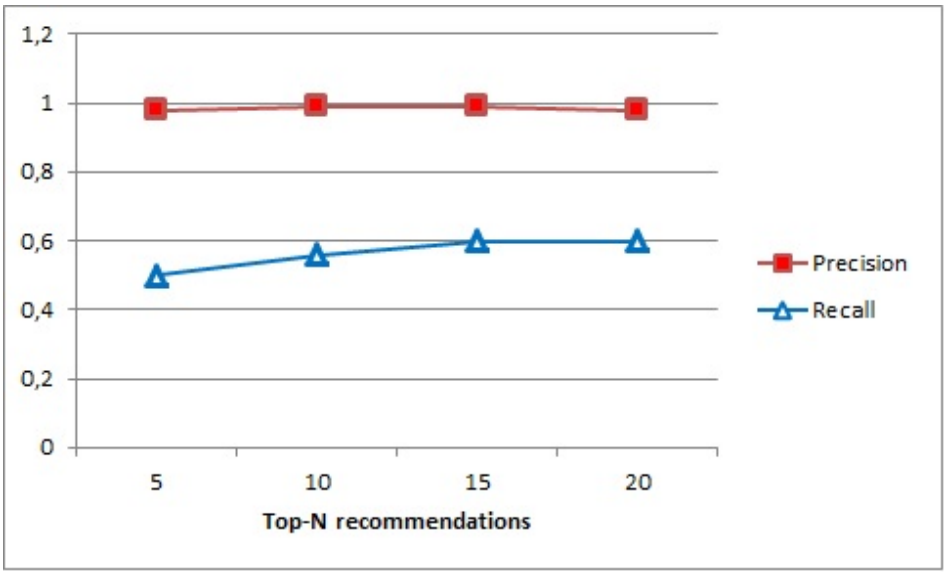
Dans la première expérimentation, notre objectif été de voir l'effet du nombre de voisins sur les résultats de la recommandation. Dans cette expérimentation, on a fixé le nombre de recommandations de Top N au nombre de total des éléments retournés, ce qui signifie que tous les projets recommandés seront considérés comme des résultats de recommandation avec la variation du nombre k de voisins. La figure 4.7 montre les résultats obtenus. Cette figure montre que le nombre de voisins peut affecter légèrement l'erreur absolue moyenne quand elle est inférieure à 40, sinon, lorsque le nombre de voisins est supérieur à 40, l'erreur moyenne absolue se stabilise à la valeur de 0.63. Cela signifie que notre système peut éviter de nombreuses erreurs tout en recommandant des projets très pertinents pour les développeurs.

FIGURE 4.7: L'efficacité du nombre de voisins sur l'erreur absolue moyenne



Une deuxième expérimentation a été effectuée, cette fois-ci pour voir l'effet du nombre de recommandations sur les valeurs de précision et de rappel. Les résultats de cette évaluation sont illustrés par la figure 4.8. Cette figure montre clairement la performance globale de l'approche utilisée. Nous observons que les valeurs de précision et de rappel sont stables pour toutes les valeurs de N, la précision atteint sa valeur la plus élevée à N égale à 15, ce qui signifie que tous les éléments renvoyés (projets recommandés) sont approximativement pertinents pour l'utilisateur. Toujours à partir de la figure 4.8, nous observons que le rappel est stable à la valeur de 0,6 à presque toutes les valeurs de N, cela signifie que la probabilité de sélectionner des projets pertinents pour les recommandations est plus élevée.

FIGURE 4.8: L'efficacité du nombre de recommandations sur les mesures de précision et de rappel



À partir de ces résultats, nous pouvons calculer les valeurs de F-Measure, nous l'avons trouvé stables à la valeur de 0,77 à toutes les valeurs de N, ce qui signifie que les résultats de la recommandation sont très précises. À partir de ces résultats d'évaluation, nous pouvons dire que notre système de recommandation a une bonne précision des recommandations. Cela le rend très utile pour les développeurs GitHub pour chercher des projets appropriés qui correspondent à leurs besoins. Toutefois, aucune conclusion finale ne sera donnée jusqu'à ce que ce système de recommandation soit mis en place concrètement, afin qu'on puisse vraiment voir comment il répondra aux besoins des développeurs.

Nous avons également développé un prototype de site Web dans lequel nous avons mis en œuvre notre solution proposée. La figure 4.9 montre une capture d'écran de ce prototype proposé. La liste à gauche fournit des liens vers les différentes fonctionnalités proposées par le système, comme le montre la figure, un développeur peut voir ses meilleurs projets, explorer d'autres projets ou recommander un projet à un ami. La liste à droite montre les projets recommandés pour le développeur par le système. Le système fournit pour chaque projet recommandé les informations suivantes :

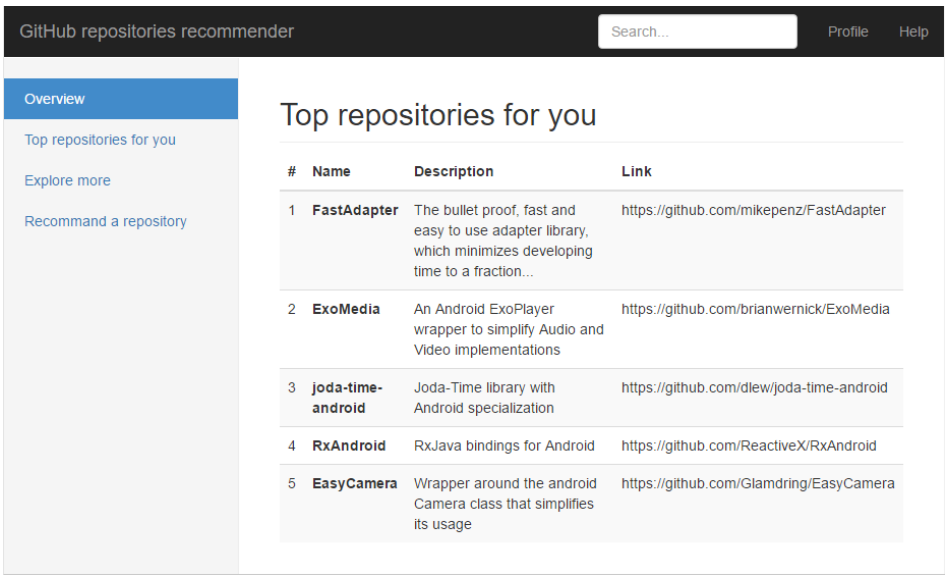
L'index : représente l'ordre du projet dans la liste.

Le nom : le nom du projet.

La description : la description du projet.

Le lien : le lien URL vers le projet sur GitHub.

FIGURE 4.9: Capture d'écran du prototype proposé



4.9 Conclusion

Nous avons présenté dans ce chapitre un nouveau système de recommandation pour les projets GitHub basés sur les techniques de filtrage collaboratif. Tout d'abord, certains travaux connexes sur ce domaine ont été discutés, puis l'architecture du système a été donnée, et enfin, un ensemble d'expérimentation ont été réalisé afin d'évaluer les résultats de notre système de recommandation.

Les résultats expérimentaux sur un ensemble de données réelles collectées à partir du site web GitHub, en utilisant des métriques bien connues et une méthode de validation croisée, montrent que notre système peut aider les développeurs à chercher des projets open source appropriés. La métrique d'erreur absolue moyenne de notre système a montré des résultats très encourageants. Les valeurs de précision et de rappel sont également précises.

Conclusion générale et perspectives

Les travaux présentés dans cette thèse sont des contributions aux domaines des big data et analyse de réseaux sociaux en ligne. Ces deux domaines fortement liés, contribuent les uns avec les autres pour résoudre des problèmes complexes réels. Nos objectifs dans cette thèse étaient de proposer des nouvelles méthodes et approches pour résoudre des problématiques liées à ces deux domaines afin d'améliorer les résultats des méthodes existantes et développer de nouvelles solutions plus adaptées. Parmi les problématiques que nous avons traitées, nous nous sommes beaucoup intéressés à la détection de communautés et les systèmes de recommandation, ces deux problématiques ont fait l'objet de deux chapitres dans ce manuscrit, les chapitres 3 et 4.

Les travaux réalisés dans cette thèse peuvent-être résumés comme suit :

Une approche bio-inspirée pour la détection de communautés : Nous avons traité la problématique de détection de communautés dans les réseaux sociaux, nous avons présenté différentes approches pour résoudre cette problématique. Dans un premier travail, nous avons utilisé l'algorithme d'optimisation bio-inspiré de feux d'artifice (Fireworks Algorithm), cet algorithme est conçu pour résoudre les problèmes d'optimisation dans les espaces de recherche continus, il est inspiré des explosions des feux d'artifice dans le ciel durant les fêtes du nouvel an chinois. Nous avons développé une variante discrète de cet algorithme afin de l'appliquer sur la problématique mentionnée. Nous avons aussi intégré une heuristique basée sur la stratégie de propagation d'étiquettes pour améliorer l'algorithme et pour accélérer sa convergence. Nous avons testé cette approche sur deux catégories de dataset, le premier dataset contient un ensemble de réseaux sociaux populaires, et le second représente un benchmark de réseaux sociaux génératifs appelé : GN Benchmark Networks. Les résultats de ces évaluations ont été comparés avec les résultats d'autres algorithmes populaires dans la littérature notamment : CNM, Infomap et GA-Net présentés dans le chapitre 3. Les résultats obtenus montrent l'efficacité de l'utilisation de notre approche pour résoudre le problème de la détection des communautés dans des réseaux complexes en général et dans les réseaux sociaux en particulier avec des valeurs de Q et NMI très satisfaisantes. Par exemple, dans le cas des réseaux benchmark GN ($\mu = 0.5$), notre algorithme a obtenu une valeur de $Q = 0.1265$ par rapport aux trois autres algorithmes (Informap : 0.0098, GA-Net : 0.0056, CNM : 0.0028), et pour le réseau réel Karate Club $Q = 0.4198$ par rapport aux autres (Informap : 0.4091, GA-Net : 0.4103, CNM : 0.4084)

Un système de recommandation de projets open source sur GitHub : Nous avons proposé un système de recommandation basé sur le filtrage collaboratif pour recommander des projets open source sur le site GitHub. Le site GitHub est considéré comme le premier réseau social pour les développeurs dans lequel ils peuvent créer, partager et contribuer aux projets open source, le site fournit une fonctionnalité de recherche intégrée pour effectuer des recherches dans sa large base de projets. Cette fonctionnalité de

recherche est basique et n'intègre aucune information sociale (le réseau de l'utilisateur lui-même), donc nous avons eu l'idée de développer un système de recommandation pour permettre aux développeurs de bien bénéficier de la large base de projets et d'améliorer aussi les résultats de recherche. Ce système est implémenté en utilisant la méthode de recommandation Top-N et la méthode Jaccard pour le calcul des similarités. Nous avons testé le système sur un dataset populaire qui contient des données collectées du GitHub appelé : Mining Software Repositories Challenge 2014 (MSR2014) [Gou13] en utilisant la méthode de validation croisée k-fold. Les résultats obtenus à partir de ces évaluations montrent des bonnes performances de notre système notamment en matière de précision et rappel avec une valeur de F-Measure égale à 0.77

Les travaux que nous avons fait, présentent des solutions efficaces aux problématiques abordées, cela est justifié par les bons résultats que nous avons obtenus lors de leurs expérimentations en utilisant des métriques prouvées et des benchmarks connus. Les expérimentations effectuées nous ont permis de confirmer les avantages et les performances des solutions proposées. Cependant, de nombreuses perspectives d'amélioration émergent de ces travaux.

Concernent la détection de communautés, nous envisageons d'appliquer notre approche sur d'autres types de réseaux complexes, particulièrement les réseaux signés et les réseaux dynamiques. Un réseau signé est un réseau complexe où chacun de ses liens est désigné comme étant positif ou négatif. Ce type de réseaux est utilisé pour modéliser de nombreux systèmes du monde réel, par exemple un réseau social dans lequel les liens positifs représentent des relations d'amitié tandis que les négatifs correspondent à des relations d'inimitié. La solution actuelle ne peut pas être appliquée directement sur ce type de réseaux sans faire de modifications au niveau de la structure du réseau ainsi qu'au niveau de la représentation des individus (codage/décodage). Le deuxième type de réseaux, appelé réseaux dynamiques, sont des réseaux qui changent leur structure dans le temps, cela dit, un réseau dynamique à l'instant i peut être totalement différent de lui-même à l'instant i1. Ce type de réseau nécessite un traitement spécifique et différent des autres réseaux. De ce fait, nous envisageons de développer une solution pour détecter et suivre l'évaluation des communautés dans les réseaux dynamiques.

En outre, il est nécessaire d'étudier la performance de la solution proposée sur les réseaux sociaux à grande échelle, nous n'avons pas pu faire des expérimentations sur ce genre de réseaux sociaux, car, cela nécessite l'accès à des datasets privés ainsi qu'à des moyens de calcul très performants.

En ce qui concerne le deuxième travail, nous envisageons d'étudier l'impact d'autres critères sur le résultat et la performance des recommandations. Notez que dans notre approche nous avons considéré le fait de copier un projet par un utilisateur comme une notation pour créer notre matrice utilisateur-élément. Ce travail ouvre également la voie vers d'autres nouveaux travaux liés à l'analyse de GitHub en général et à l'analyse et l'étude de son réseau social en particulier.

Annexe A

Liste des publications

A.1 Revues Scientifiques

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. Penguins Search Optimization Algorithm for Community Detection in Complex Networks. International Journal of Applied Metaheuristic Computing (IJAMC), 2018, vol. 9, no 1, p. 1-14 (IGI Global)

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. A discrete modified fireworks algorithm for community detection in complex networks. Applied Intelligence, 2016, p. 1-13 (Springer)

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. Recommending Relevant Open Source Projects on GitHub using a Collaborative-Filtering Technique. International Journal of Open Source Software and Processes (IJOSSP), 2015, vol. 6, no 1, p. 1-16 (IGI Global)

A.2 Chapitres de livre

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. Open source projects recommendation on GitHub. Optimizing Contemporary Application and Processes in Open Source Software. 2018. (IGI Global)

Mohamed Guendouz. A Discrete Black Hole Optimization Algorithm for Efficient Community Detection in Social Networks. Handbook of Research on Biomimicry in Information Retrieval and Knowledge Management. 2018. (IGI Global)

A.3 Conférences Nationales

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. Automatic Android Apps Categorization Using Machine Learning And Text Mining Techniques, 1st National Conference on Embedded and Distributed Systems (EDiS). Oran, 2015.

A.4 Conférences Internationales

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou. Machine Learning Based Classification of Android Apps through Text Features. EGC 2017 : 429-430, Grenoble, France, 23-27 January 2017.

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou, Automatic Android Apps Categorization Using Machine Learning And Text Mining Techniques, 2015, International Conference on Pattern Analysis and Intelligent Systems (PAIS), Tebessa, Algeria, 26-27 October 2015.

Mohamed Guendouz, Abdelmalek Amine, Reda Mohamed Hamou, Recommending relevant GitHub repositories : a collaborative-filtering approach, 2015, International Conference on Networking and Advanced Systems (ICNAS), Annaba, Algeria, 6-7 May 2015.

Bibliographie

[AK91] D. Aha et D. Kibler. « Instance-based learning algorithms ». In : *Machine Learning* 6 (1991), p. 37–66 (cf. p. 24).

[Ama] *Amazon DynamoDB – NoSQL Cloud Database Service*. `https://aws.amazon.com/dynamodb/`. (Accessed on 09/08/2017) (cf. p. 19).

[AMQ] AMQP. *AMQP*. `https://www.amqp.org/` (cf. p. 11).

[AP01] Charu C Aggarwal et Srinivasan Parthasarathy. « Mining massively incomplete data sets by conceptual reconstruction ». In : *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2001, p. 227–232 (cf. p. 85).

[Apaa] *Apache HBase – Apache HBase^{TM} Home*. `https://hbase.apache.org/`. (Accessed on 09/08/2017) (cf. p. 19).

[Apab] *Apache Lucene - Apache Lucene Core*. `https://lucene.apache.org/core/`. (Accessed on 02/07/2018) (cf. p. 29).

[Apac] *The Apache Software Foundation*. `https://www.apache.org/`. (Accessed on 09/08/2017) (cf. p. 17).

[ASP98] Charu C Aggarwal, Zheng Sun et S Yu Philip. « Online Generation of Profile Association Rules. » In : *KDD*. 1998, p. 129–133 (cf. p. 90).

[Aws] *AWS | Amazon SimpleDB – Simple Database Service*. `https://aws.amazon.com/simpledb/`. (Accessed on 09/08/2017) (cf. p. 19).

[BHK98] John S Breese, David Heckerman et Carl Kadie. « Empirical analysis of predictive algorithms for collaborative filtering ». In : *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 1998, p. 43–52 (cf. p. 97).

[Big] *Bigtable - Scalable NoSQL Database Service | Google Cloud Platform*. `https://cloud.google.com/bigtable/`. (Accessed on 09/08/2017) (cf. p. 19).

[BL12] Mark A Beyer et Douglas Laney. « The importance of 'big data' : a definition ». In : *Stamford, CT : Gartner* (2012), p. 2014–2018 (cf. p. 5).

[Bla] BlackPlanet. *BlackPlanet.com — Black Women, Men Meet to Chat, Discuss, Engage*. `http://www.blackplanet.com/` (cf. p. 41).

[Blo+08] Vincent D Blondel et al. « Fast unfolding of communities in large networks ». In : *Journal of statistical mechanics : theory and experiment* 2008.10 (2008), P10008 (cf. p. 63).

[Bri+05] Derek Bridge et al. « Case-based recommender systems ». In : *The Knowledge Engineering Review* 20.3 (2005), p. 315–320 (cf. p. 88).

[Cas] Apache Cassandra. *Apache Cassandra*. `http://cassandra.apache.org/` (cf. p. 19).

[CH92] S. le Cessie et J.C. van Houwelingen. « Ridge Estimators in Logistic Regression ». In : *Applied Statistics* 41.1 (1992), p. 191–201 (cf. p. 24).

[Cha] JPMorgan Chase. *JPMorgan Chase & Co.* `https://www.jpmorganchase.com/` (cf. p. 11).

[Cla] Classmates. *Classmates.com*. `http://www.classmates.com/` (cf. p. 40).

[CNM04] Aaron Clauset, Mark EJ Newman et Cristopher Moore. « Finding community structure in very large networks ». In : *Physical review E* 70.6 (2004), p. 066111 (cf. p. 54, 62, 72).

[Coua] *Couchbase*. `https://www.couchbase.com/`. (Accessed on 09/08/2017) (cf. p. 19).

[Coub] Apache CouchDB. *Apache CouchDB*. `http://couchdb.apache.org/` (cf. p. 19).

[Cyw] Cyworld. *Cyworld*. `http://www.cyworld.com/cymain/?f=cymain` (cf. p. 41).

[Deg] Six Degrees. *Six Degrees*. `http://sixdegrees.com/` (cf. p. 40).

[Dou01] Laney Douglas. « 3d data management : Controlling data volume, velocity and variety ». In : *Gartner. Retrieved* 6 (2001), p. 2001 (cf. p. 5).

[FB07] Santo Fortunato et Marc Barthelemy. « Resolution limit in community detection ». In : *Proceedings of the National Academy of Sciences* 104.1 (2007), p. 36–41 (cf. p. 65).

[FB08] Alexander Felfernig et Robin Burke. « Constraint-based recommender systems : technologies and research issues ». In : *Proceedings of the 10th international conference on Electronic commerce*. ACM. 2008, p. 3 (cf. p. 88).

[Fel+11] Alexander Felfernig et al. « Developing constraint-based recommenders ». In : *Recommender systems handbook*. Springer, 2011, p. 187–215 (cf. p. 88).

[Fre78] Linton C Freeman. « Centrality in social networks conceptual clarification ». In : *Social networks* 1.3 (1978), p. 215–239 (cf. p. 51–53).

[Fri] Friendster. *Friendster.com - Living the Game*. `http://www.friendster.com/` (cf. p. 42).

[FS98] Y. Freund et R. E. Schapire. « Large margin classification using the perceptron algorithm ». In : *11th Annual Conference on Computational Learning Theory*. New York, NY : ACM Press, 1998, p. 209–217 (cf. p. 24).

[Gar+09] Salvador García et al. « A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour : a case study on the CEC'2005 special session on real parameter optimization ». In : *Journal of Heuristics* 15.6 (2009), p. 617–644 (cf. p. 74).

[Gita] *Git*. `https://git-scm.com/`. (Accessed on 09/09/2017) (cf. p. 91).

[Gitb] *GitHub*. `https://github.com/`. (Accessed on 09/09/2017) (cf. p. 91).

[GMM98] Robert H Guttman, Alexandros G Moukas et Pattie Maes. « Agent-mediated electronic commerce : A survey ». In : *The Knowledge Engineering Review* 13.2 (1998), p. 147–159 (cf. p. 90).

[GN02] Michelle Girvan et Mark EJ Newman. « Community structure in social and biological networks ». In : *Proceedings of the national academy of sciences* 99.12 (2002), p. 7821–7826 (cf. p. 61, 76).

[Gol+01] Ken Goldberg et al. « Eigentaste : A constant time collaborative filtering algorithm ». In : *information retrieval* 4.2 (2001), p. 133–151 (cf. p. 82).

[Gon+14] Maoguo Gong et al. « Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition ». In : *Evolutionary Computation, IEEE Transactions on* 18.1 (2014), p. 82–97 (cf. p. 71).

[Gou+14] Georgios Gousios et al. « Lean GHTorrent : GitHub data on demand ». In : *Proceedings of the 11th working conference on mining software repositories*. ACM. 2014, p. 384–387 (cf. p. 98).

[Gou13] Georgios Gousios. « The GHTorent dataset and tool suite ». In : *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press. 2013, p. 233–236 (cf. p. 98, 105).

[Had] Apache Hadoop. *Apache Hadoop*. `http://hadoop.apache.org/` (cf. p. 6, 17).

[Hel+11] Brandon Heller et al. « Visualizing collaboration and influence in the open-source software community ». In : *Proceedings of the 8th working conference on mining software repositories*. ACM. 2011, p. 223–226 (cf. p. 93).

[HKV08] Yifan Hu, Yehuda Koren et Chris Volinsky. « Collaborative filtering for implicit feedback datasets ». In : *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee. 2008, p. 263–272 (cf. p. 83).

[HND15] Cho-Jui Hsieh, Nagarajan Natarajan et Inderjit Dhillon. « PU learning for matrix completion ». In : *International Conference on Machine Learning*. 2015, p. 2445–2453 (cf. p. 83).

[HT98] Trevor Hastie et Robert Tibshirani. « Classification by Pairwise Coupling ». In : *Advances in Neural Information Processing Systems*. Sous la dir. de Michael I. Jordan, Michael J. Kearns et Sara A. Solla. T. 10. MIT Press, 1998 (cf. p. 24).

[IBM] IBM. *Analytics : The real-world use of big data - How innovative enterprises extract value from uncertain data*. `https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=GBE03519USEN` (cf. p. 8).

[IBM13] IBM. *IBM What is big data ? - Bringing big data to the enterprise*. 2013. URL : `https://www-01.ibm.com/software/in/data/bigdata/` (cf. p. 5).

[Int12] Intel. *Peer Research Report : Big Data Analytics*. 2012. URL : `https://www.intel.com/content/dam/www/public/us/en/documents/reports/data-insights-peer-research-report.pdf` (cf. p. 6).

[JL95] George H. John et Pat Langley. « Estimating Continuous Distributions in Bayesian Classifiers ». In : *Eleventh Conference on Uncertainty in Artificial Intelligence*. San Mateo : Morgan Kaufmann, 1995, p. 338–345 (cf. p. 24).

[JSO] JSON. *JSON*. `http://www.json.org/` (cf. p. 96).

[Kee+01] S.S. Keerthi et al. « Improvements to Platt's SMO Algorithm for SVM Classifier Design ». In : *Neural Computation* 13.3 (2001), p. 637–649 (cf. p. 24).

[KL70] Brian W Kernighan et Shen Lin. « An efficient heuristic procedure for partitioning graphs ». In : *The Bell system technical journal* 49.2 (1970), p. 291–307 (cf. p. 59).

[Kor08] Yehuda Koren. « Factorization meets the neighborhood : a multifaceted collaborative filtering model ». In : *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2008, p. 426–434 (cf. p. 86).

[Kru97] Bruce Krulwich. « Lifestyle finder : Intelligent user profiling using large-scale demographic data ». In : *AI magazine* 18.2 (1997), p. 37 (cf. p. 90).

[Len02] Maurizio Lenzerini. « Data integration : A theoretical perspective ». In : *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM. 2002, p. 233–246 (cf. p. 14).

[LFR08] Andrea Lancichinetti, Santo Fortunato et Filippo Radicchi. « Benchmark graphs for testing community detection algorithms ». In : *Physical review E* 78.4 (2008), p. 046110 (cf. p. 73).

[LHF05] Niels Landwehr, Mark Hall et Eibe Frank. « Logistic Model Trees ». In : 95.1-2 (2005), p. 161–205 (cf. p. 24).

[Li+08] Zhenping Li et al. « Quantitative function for community detection ». In : *Physical review E* 77.3 (2008), p. 036109 (cf. p. 65).

[Lin] LinkedIn. *LinkedIn*. `https://www.linkedin.com/` (cf. p. 42).

[Lip99] Alan H Lipkus. « A proof of the triangle inequality for the Tanimoto distance ». In : *Journal of Mathematical Chemistry* 26.1 (1999), p. 263–265 (cf. p. 98).

[Liv] LiveJournal. *LiveJournal : Discover global communities of friends who share your unique passions and interests*. `https://www.livejournal.com/` (cf. p. 41).

[LR05] Fabiana Lorenzi et Francesco Ricci. « Case-based recommender systems : A unifying view ». In : *Intelligent Techniques for Web Personalization*. Springer, 2005, p. 89–113 (cf. p. 88).

[Lus03] David Lusseau. « The emergent properties of a dolphin social network ». In : *Proceedings of the Royal Society of London B : Biological Sciences* 270.Suppl 2 (2003), S186–S188 (cf. p. 76).

[LW71] Michael Levandowsky et David Winter. « Distance between sets ». In : *Nature* 234.5323 (1971), p. 34–35 (cf. p. 98).

[Mac+67] James MacQueen et al. « Some methods for classification and analysis of multivariate observations ». In : *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. T. 1. 14. Oakland, CA, USA. 1967, p. 281–297 (cf. p. 27).

[Mem] *memcached - a distributed memory object caching system*. `https://memcached.org/`. (Accessed on 09/08/2017) (cf. p. 19).

[Mic13] Microsoft. *The Big Bang : How the Big Data Explosion Is Changing the World*. 2013. url : https://news.microsoft.com/2013/02/11/the-big-bang-how-the-big-data-explosion-is-changing-the-world/ (cf. p. 6).

[MJ+34] Jacob Levy Moreno, Helen Hall Jennings et al. « Who shall survive ? » In : (1934) (cf. p. 45).

[MM00] Jonathan I Maletic et Andrian Marcus. « Data Cleansing : Beyond Integrity Analysis. » In : *Iq*. 2000, p. 200–209 (cf. p. 15).

[Mon] *MongoDB for GIANT Ideas | MongoDB*. https://www.mongodb.com/. (Accessed on 09/08/2017) (cf. p. 19).

[MQ] IBM MQ. *IBM MQ*. http://www-03.ibm.com/software/products/fr/ibm-mq (cf. p. 11).

[MQT] MQTT. *MQTT*. http://mqtt.org/ (cf. p. 13).

[MR11] Lorraine McGinty et James Reilly. « On the evolution of critiquing recommenders ». In : *Recommender Systems Handbook*. Springer, 2011, p. 419–453 (cf. p. 89).

[Mys] Myspace. *Myspace*. https://myspace.com/ (cf. p. 42).

[Neo] *Neo4j, the world's leading graph database - Neo4j Graph Database*. https://neo4j.com/. (Accessed on 09/08/2017) (cf. p. 19).

[New06] Mark EJ Newman. « Modularity and community structure in networks ». In : *Proceedings of the national academy of sciences* 103.23 (2006), p. 8577–8582 (cf. p. 54, 55, 73).

[NHG] NHGRI. *Human Genome Project - National Human Genome Research Institute (NHGRI)*. https://www.genome.gov/12011238/an-overview-of-the-human-genome-project/ (cf. p. 10).

[Nut] Apache Nutch. *Apache Nutch*. http://nutch.apache.org/ (cf. p. 29).

[OK+98] Douglas W Oard, Jinmook Kim et al. « Implicit feedback for recommender systems ». In : *Proceedings of the AAAI workshop on recommender systems*. T. 83. WoUongong. 1998 (cf. p. 83).

[Ora12] Oracle. « Oracle : Big data for the enterprise ». In : *Oracle White Paper* (2012) (cf. p. 6).

[Paz99] Michael J Pazzani. « A framework for collaborative, content-based and demographic filtering ». In : *Artificial intelligence review* 13.5-6 (1999), p. 393–408 (cf. p. 90).

[Piz08] Clara Pizzuti. « Ga-net : A genetic algorithm for community detection in social networks ». In : *Parallel Problem Solving from Nature–PPSN X*. Springer, 2008, p. 1081–1090 (cf. p. 72).

[Pla98] J. Platt. « Fast Training of Support Vector Machines using Sequential Minimal Optimization ». In : *Advances in Kernel Methods - Support Vector Learning*. Sous la dir. de B. Schoelkopf, C. Burges et A. Smola. MIT Press, 1998 (cf. p. 24).

[Qui93] Ross Quinlan. *C4.5 : Programs for Machine Learning*. San Mateo, CA : Morgan Kaufmann Publishers, 1993 (cf. p. 24).

[RB08] Martin Rosvall et Carl T Bergstrom. « Maps of random walks on complex networks reveal community structure ». In : *Proceedings of the National Academy of Sciences* 105.4 (2008), p. 1118–1123 (cf. p. 72).

[Red] *Redis*. `https://redis.io/`. (Accessed on 09/08/2017) (cf. p. 19).

[Reu] Thomson Reuters. *Thomson Reuters 2010 Annual Report*. `http://archive.annual-report.thomsonreuters.com/2010/` (cf. p. 7).

[Ric79] Elaine Rich. « User modeling via stereotypes ». In : *Cognitive science* 3.4 (1979), p. 329–354 (cf. p. 90).

[Ryz] Ryze. *Ryze - the original social network for business*. `https://www.ryze.com/` (cf. p. 41).

[SFH05] Marc Sumner, Eibe Frank et Mark Hall. « Speeding up Logistic Model Tree Induction ». In : *9th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Springer, 2005, p. 675–683 (cf. p. 24).

[Smy07] Barry Smyth. « Case-based recommendation ». In : *The adaptive web*. Springer, 2007, p. 342–376 (cf. p. 87–89).

[Thu+13] Ferdian Thung et al. « Network structure of social coding in github ». In : *Software maintenance and reengineering (csmr), 2013 17th european conference on*. IEEE. 2013, p. 323–326 (cf. p. 93).

[Tre00] Shari Trewin. « Knowledge-based recommender systems ». In : *Encyclopedia of library and information science* 69.Supplement 32 (2000), p. 180 (cf. p. 88).

[TZ10] Ying Tan et Yuanchun Zhu. « Fireworks algorithm for optimization ». In : *Advances in Swarm Intelligence*. Springer, 2010, p. 355–364 (cf. p. 2, 65).

[Wei05] Sanford Weisberg. *Applied linear regression*. T. 528. John Wiley & Sons, 2005 (cf. p. 24).

[Wil45] Frank Wilcoxon. « Individual comparisons by ranking methods ». In : *Biometrics bulletin* 1.6 (1945), p. 80–83 (cf. p. 74).

[XIN] XING. *XING – For a better working life*. `https://www.xing.com/` (cf. p. 42).

[Yu+14] Yue Yu et al. « Reviewer recommender of pull-requests in GitHub ». In : *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*. IEEE. 2014, p. 609–612 (cf. p. 93).

[Zac77] Wayne W Zachary. « An information flow model for conflict and fission in small groups ». In : *Journal of anthropological research* (1977), p. 452–473 (cf. p. 76).

[Zha+14] Lingxiao Zhang et al. « Recommending relevant projects via user behaviour : An exploratory study on github ». In : *Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies*. ACM. 2014, p. 25–30 (cf. p. 93).