ROYAUME DU MAROC
Université Mohammed V - RABAT -
جامعة محمد الخامس – الرباط –
Faculté des sciences
كلية العلوم
CENTRE D'ETUDES DOCTORALES - SCIENCES ET TECHNOLOGIES

N° d'ordre : 3411

THESE

En vue de l'obtention du : ***DOCTORAT***

Structure de Recherche : Intelligent Processing Systems & Security (IPSS)
Discipline : Informatique
Spécialité : Intelligence Artificielle et Big Data

Présentée et soutenue le : 30/12/2020 par :

Kaoutar EL HANDRI

New Top-k algorithms for Multicriteria Recommendation Systems based on Artificial intelligence and Big Data

JURY

Ismail KASSOU	PES, Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes (ENSIAS), Université Mohammed V de Rabat	Président
Abdellah IDRISSI	PH, Faculté des Sciences de Rabat, Université Mohammed V de Rabat	Directeur de thèse
Faouzia BENABBOU	PES, Faculté des Sciences de Ben M'sik, Université Hassan II de Casablanca	Rapporteuse/Examinatrice
Fatima-Zahra BELOUADHA	PES, Ecole Mohammadia d'Ingénieurs (EMI), Université Mohammed V de Rabat	Rapporteuse/Examinatrice
Nadia NEDJAH	PES, Faculty of Engineering, State University of Rio de Janeiro, Brazil	Rapporteuse/ Examinatrice
Iadjel BELLATRECHE	PES, Ecole Nationale Supérieur de Mécanique et d'Aérotechnique (ENSMA), Poitiers, France	Examinateur
Abderrahmane EZZAHOUT	PA, Faculté des Sciences, Université Mohammed V de Rabat	Invité

Année Universitaire: 2020/2021

Dedication

It is a pleasure for me to express my profound thanks to my dear mother and father for their unlimited moral support for their presence and encouragement throughout my studies. I also thank my husband, Abdelghani, without whom this work would not have been possible. I want to thank my family members equally for their endless encouragement, namely My brothers Youness, Yassin, and my lovely sister Iman. I would like to thank my best friends and colleagues from Mohammed V or other international universities for their encouragement and love. Eventually, this work is also dedicated to Sara, Youssef, my children, and my little princess Aicha. And those who have left us and are dear to us.

ACKNOWLEDGEMENTS

This thesis was carried out in the computer science department of the faculty of sciences in the Intelligent Processing Systems Security (IPSS) team.

First of all, I would like to thank my thesis supervisor, Professor Mr. **Abdellah IDRISSI** (PH) at the Faculty of Sciences, Mohammed V University in Rabat directing and supervising this work with great scientific rigor. The quality of his advice, the support, and the confidence he gave me allowed me to develop my ideas and give me another vision of research. These years spent in this team with him have given me an incredibly enriching experience, both in study and teaching and supervision.
I want to thank Professor Mrs. **Fouzia OMARY** PES at the Faculty of Sciences of Rabat and the director of IPSS laboratory for welcoming me to her laboratory and for her continuous support and encouragement during my doctoral research and studies.

I want to thank Professor Mr **Ismail KASSOU** (PES) at l'École Nationale Supérieure d'Informatique et d'Analyse des Systèmes (ENSIAS) in Mohammed the V university. He agreed to participate in my thesis committee as president of the jury and devoted precious time.

Furthermore, I would like to thank Professor Mrs. **Faouzia BENABBOU** (PES) at the Ben M'sik Faculty of Sciences, Hassan II University of Casablanca to have accepted the reporter's task and have devoted precious time to examining this manuscript.

Besides, I would like to thank Professor Mrs. **Fatima-Zahra BELOUADHA** (PES) at Ecole Mohammadia d'Ingénieurs (EMI), University Mohammed the V of Rabat, for taking this work and consecrated precious time to the reporter task and the examination task.

A special thanks to Professor Mrs. **Nadia NEDJAH** (PES) at Faculty of Engineering, State University of Rio de Janeiro, in Brazil, for giving me the honor of being a reporter and examiner for this thesis and have dedicated valuable time to review this.

Once again, I would like to thank the Professor Mr. **Ladjel BELETRECH** (PES) at the Ecole Nationale Supérieure de Mécanique et d'Aérotechnique (ENSMA), in Poitiers, France, for agreeing to examine my thesis and devoting precious time to the revision of this manuscript.

I would also thank Professor Mr. **Abderrahmane EZZAHOUT** (PA) at faculty of science Mohammed the V University of Rabat, for accepting to participate in my thesis committee of the jury.

Abstract

In the era of Big Data, Parallel Top_k Query Processing in Information Retrieval (IR) has received growing attention for both the industry and academia. This query handling allows users to retrieve only one of the most moving data objects in a minimum set of choices. Nevertheless, most existing studies in this field focus on a single processor in a usually centralized environment, which limits the scalability of the system. In addition, this problem becomes more than less performed when using Top_k in cases of multiple dimensions and Big Data analytic's. In this thesis, we propose a novel approach for sequential and parallel algorithms using a Distributed Recommender System (RS) based on Big Data platform. Our approach is combining Multi-Criteria Decision Aiding support (MCDA) and Dominating Query-Processing method, including sophisticated Artificial Intelligence techniques. In addition, we applied the Resilient Distributed Datasets (RDD) paradigm in Cloud Computing (CC) context. The platforms above present a promoting environment for Big Data processing. Extensive experimental results in terms of accuracy, performance, and scalability show, thanks to a new Multi-criteria collaborative filtering system, that our proposed hybrid recommendation systems outperform other approaches existing in the literature.

Keywords: Artificial Intelligence, Top_k query optimization, Skyline, Artificial Intelligence, Multi-Criteria Decision Aid (MCDA), Collaborative filtering (CF), Recommender System (RS), Parallel processing, Big Data.

RÉSUMÉ

À l'Ère du Big data, le traitement parallèle des requêtes Top_k dans la Recherche d'informations (RI) a reçu une attention croissante de la part de l'industrie et du monde universitaire. Cette gestion des requêtes permet aux utilisateurs de ne récupérer qu'un seul des objets de données les plus utiles dans un ensemble minimal de choix. Cependant, la plupart des études existantes dans ce domaine se concentrent sur un seul processeur dans un environnement généralement centralisé, ce qui limite l'évolutivité du système. Ce problème est encore moins performant lors de l'utilisation de Top_k dans les cas de dimensions multiples et d'analyses de données étendues. Dans cet article, nous fournissons un nouvel algorithme parallèle dans un système de recommandation (SR) entièrement distribué basé sur la plate-forme Apache Spark. L'idée principale de cette approche est d'implémenter l'Aide à la décision multicritère (ADMC) présentée par une approche d'optimisation dans les requêtes tout en utilisant la factorisation matricielle et la décomposition vectorielle unique comme une technique sophistiquée d'apprentissage machine. De même, l'application du paradigme des ensembles de données distribués résilient (RDD : resilientdistributeddatasets) qui se base sur l'ensemble de données distribuées résilientes dans le Cloud Computing (CC) et qui présente un environnement favorable à la gestion des grandes données. De nombreux résultats expérimentaux en termes de précision, de performances et d'évolutivité montrent que notre approche est réellement prometteuse par rapport à celles trouvées dans la littérature.

Mots clés: Intelligence artificielle, Top_k query optimization, Aide à la Decision Multi critères (ADMC), Collaborative filtering (CF), System de Recommandation, Traitement parallèle, Big Data.

Résumé détaillé

L'importance croissante du Web en tant que support pour le commerce électronique, les activités commerciales et les sites d'aide à la décision a permis l'évolution de la technologie des systèmes de recommandation.Um système de recommandations comme son nome le dit, est um moteur que permet de deviner les intérêts de consommateur pour leur proposer des produits et de service afin d'augmenter les recettes. Considérons par exemple le scénario Netflix d'un fournisseur de services de contenu dans lequel les utilisateurs peuvent rapidement fournir un retour d'information d'un simple clic. À l'aide des techniques de l'intelligence artificielle, développer les systèmes de recommandation (SR) intelligents, tout en intégrant l'aide à la décision, vise à personnaliser le contenu pour chaque utilisateur et à réduire la charge cognitive de l'information de l'utilisateur.

En outre, l'objectif de l'exploitation de Big data est non seulement pour créer la technologie du futur, mais aussi pour générer une analyse en temps réel des informations d'aujourd'hui. L'exemple des sociétés de réseaux sociaux comme Facebook et Twitter utilise la puissance des Big data que les utilisateurs créent chaque minute. En particulier, les données sont toujours en mouvement. Nous croyons que les Big Data finiront par s'imposer dans le monde de l'aide à la décision. Dans ce contexte, l'utilisation des requêtes de classement et les algorithmes Top_k est une exigence cruciale surtout quand il s'agit des environnements interactifs, tels que la recherche des services cloud, l'exploitation des métamoteurs de recherche sur le Web, les réseaux sociaux, le multimédia et dans tous les types de systèmes de recommandation. Cependant, dans ce genre de système, les différentes méthodes utilisées à savoir la méthode basée sur le Filtrage Collaboratif (FC), la méthode basée sur le contenu, ou la méthode hybride se concentrent généralement sur l'usage d'un seul processeur dans un environnement centralisé. Ce qui limite son efficacité et son évolutivité. Le volume de données devient de plus en plus important et est dans l'ensemble stocké et distribué sur plusieurs serveurs. Le vrai problème alors c'est de supporter les requêtes de classement dans les environnements distribués. Surtout que le volume de données et leur vitesse exigent des solutions parallèles efficaces qui surmontent les ressources restreintes d'un aspect centralisé. La présente thèse concerne la proposition d'un nouvel algorithme Top_k basé sur un système de recommandation multicritère dans le contexte du Big data. Il se rapporte plus particulièrement au domaine du traitement et de l'optimisation de requêtes Top_k dans des systèmes multi-objectifs. En effet, le traitement de ces requêtes consiste à trouver les k-objets qui ont les meilleurs scores globaux. Une requête dans une base de données combine différents attributs notés par une fonction score appelée aussi une fonction d'agrégation. La notation globale de chaque objet sera calculée par cette fonction, et nous pouvons retourner les meilleurs objets Top_k.

En outre, différentes techniques de Big Data sont utilisées pour résoudre ce problème. La transition vers les processeurs multi cœurs, qui est due aux limitations des processeurs mono cœurs, nécessite des calculs parallèles pour améliorer les performances des logiciels. C'est l'objectif

principal de la parallélisation de l'algorithme au sein d'un système de recommandation distribué et c'est ce que nous proposons dans cette thèse. Cela étant, notre étude s'est basée aussi sur les méthodes d'aide à la Décision multicritère (MCDA pour Multi Criteria Decision Aiding) qui sera impliquée dans la modélisation des préférences d'utilisateur dans les algorithmes Top_k. En effet, le problème de trouver la solution optimale de Pareto qui est formé de plusieurs solution non-dominés, et la méthode itérative d'optimisation numérique à critère unique pour les problèmes de recommandation multicritères sont des problèmes passionnants et difficiles. La MCDA permet d'aider à organiser et à synthétiser l'information afin que les décideurs puissent prendre de meilleures décisions. Il existe de nombreuses méthodes d'analyse multicritères pour générer des ensembles de solutions à partir de programmes multi objectifs. Ces méthodes peuvent être regroupées en deux approches. La première est basée sur la théorie de l'utilité en utilisant des critères d'agrégation à priori en un seul test, et la seconde approche est basée sur des méthodes de surclassement telles que ELECTRE I, III, IV, IS, PROMÉTHÉE I et PROMÉTHÉE II. L'usage de ces méthodes est fortement relié au domaine de FC. Ces derniers sont utilisés avec succès dans de nombreuses applications.
Dans cette thèse, nous étudions d'abord les techniques et algorithmes proposés dans la littérature, conçus pour le traitement multicritères de Top_k dans un contexte spécifique de type et de coût d'accès aux scores. Ensuite, nous nous distinguons de l'état de l'art qui traite du raffinement du résultat de Skyline et qui se concentre sur l'utilisation des méthodes MCDA, en proposant un algorithme "Top_{kws}", basé sur une méthode adaptée de l'analyse multicritères afin d'intégrer les exigences de l'utilisateur final dans la sélection. La première contribution porte sur le raffinement de Skyline dans le domaine de la QoS du Cloud Service qui a été proposé dans ce que nous avons appelé : « enhanced Cloud Service Research and Selection System (e-CSRSS) ». Ensuite, dans un cadre générique du système de recommandation capable d'exprimer les différents scénarios possibles, nous proposons un système générique de recherche et de sélection "GRSS" qui maintient l'ensemble actuel des k meilleurs objet dans leur ensemble, contrairement aux stratégies habituelles de profondeur qui se concentrent sur la melleure candidate. La plupart des études visant à affiner le résultat de Skyline constatent que l'interrogation d'une donnée multidimensionnelle peut aboutir soit à un très grand nombre de réponses, soit à un nombre insuffisant de réponses. Dans les deux cas, cela peut brouiller le choix de l'utilisateur final. Par conséquent, la compréhension des caractéristiques de la méthode de la somme pondérée en anglais: the weighted sum method (WSM) a de profondes implications. Cependant, malgré les nombreuses applications publiées pour cette méthode, il y a une discussion peu approfondie sur la signification conceptuelle des poids et des techniques qui peuvent améliorer l'efficacité de la méthode WSM. À notre connaissance les travaux cités dans l'état de l'art sur l'usage de MCDA dans le raffinement de Skyline et Top_k n'ont pas utilisé une adaptation de la méthode MCDA appliquée à l'optimisation de Top_k ou pour affiner le résultat du Skyline. Cette approche de base peut être considérée comme très puissante, car dans notre cas nous combinons les avantages de la requête Top_k et de la méthode (MCDA), à la requête Skyline. De plus, la méthode de la somme pondérée adaptée nous a permis de modéliser le problème dans un contexte multi objectif. Par ailleurs, presque tous les principaux algorithmes de Top_k comme ceux de Fagin souffrent de comportements statiques, et donc ne s'adaptent pas aux coûts d'accès à l'exécution. Dans certains cas, leurs heuristiques d'adaptation à l'exécution supposent spécifiquement des fonctions de notation moyenne pondérée. Par contre, pour avoir une bonne optimisation dans les problèmes multicritères, une des techniques MCDA puissantes et qui n'est pas compliqué en termes de calcul comme la méthode de la somme pondérée adaptée en anglais appelée bi-objective Weighted Sum Method (BWSM) peut être utile pour gérer la qualité et la scalabilité de

la sélection Top_k. Cette approche donne de meilleurs résultats en termes de temps de réponse et de qualité de résultats trouvés par rapport aux autres algorithmes. Ces résultats sont justifiés par l'application de différentes métriques et mesure de machine learning et de recommendation. En outre, les grandes infrastructures de données et les systèmes de recommandation distribués nécessitent l'utilisation d'un algorithme parallèle. Il y a cependant un intérêt croissant pour la parallélisation des requêtes Top_k. De plus, chaque algorithme a sa préférence par rapport aux autres algorithmes, des accès moins triés facilitent la gestion du traitement des requêtes Top_k. Les solutions proposées sont des parallélisations des algorithmes déjà existants, et toutes les approches partagent l'idée de partitionner l'ensemble des données en sous-ensembles, de traiter les sous-ensembles localement en parallèle, et finalement de fusionner les résultats. Par ailleurs, à l'ère des mégadonnées, l'importance du traitement Top_k est primordiale. Il nécessite une plate-forme évolutive, car il est pratiquement impossible pour les utilisateurs d'inspecter un grand nombre de résultats de requêtes non classifiées, simplement en raison de leur volume extrême. Pourtant, malgré l'importance du problème, il y a un manque d'algorithmes entièrement parallèles qui fonctionnent efficacement à grande échelle et qui révèlent les résultats corrects et exacts (c.-à-d. cas de Map-Reduce). Nous présentons un nouvel algorithme parallèle "$SPTop_{kws}$" dans un environnement distribué, basé sur les stratégies de FC et adaptable à plusieurs types de configuration de variétés et de coûts d'accès aux scores. Ce nouvel algorithme est une version parallèle du "Top_kws dans un contexte générique et distribué. Par conséquent, pour stocker de grands volumes de données, les systèmes de fichiers distribués sont une solution possible. Dans le contexte de l'analyse de données, un modèle distribué est avantageux à la fois en raison de la capacité de pousser le calcul vers différents nœuds d'un cluster et de l'évolutivité qu'il offre. Cependant, Apache Spark, à l'heure où nous écrivons ces lignes est considéré comme le plus grand projet open source au monde. En plus, en tant que nouvelle plate-forme d'analyse de données open source, il offre une gamme de fonctionnalités beaucoup plus complète que MapReduce de Hadoop puisqu'il résout les algorithmes itératifs en utilisant la mémoire interne. Il est considéré comme exécutant des programmes beaucoup plus rapidement que son homologue MapReduce principalement parce qu'il utilise les RDDs (Resilient Distributed Datasets) comme bloc de programmation qui sont conçues aussi pour gérer la défaillance de n'importe quel nœud de travail de cluster. Il en résulte une garantie de perte de données nulle. D'après l'état de l'art, Les techniques de CF basées sur la Méthode Singular Value Decomposition (SVD) sont au centre de nombreux algorithmes Top_k. Par contre, l'usage de SVD ne peut être efficace que si on peut agréer le problème commun dans le moteur de recommandation appelé "Cold Start Problem" ou bien le démarrage à froid. Ce problème signifie que nous ne pouvons pas faire de recommandations pour de nouveaux choix ou de services (par exemple les nouveaux services cloud, les nouveaux films, les nouveaux médicaments) pour les utilisateurs. Par conséquent, une approche réussie consiste à combiner la technique de Funk SVD avec d'autres méthodes telles que l'usage d'un algorithme basé sur le classement comme le traitement des requêtes Top_k, le traitement des requêtes à k dominances, et les méthodes basées sur le contenu. L'état de l'art des solutions proposées va dans le même sens que notre démarche d'optimisation où nous utilisons le modèle Funk SVD avec le traitement des deux types de requêtes qui sont Skyline et Top_k, en utilisant cette fois des MCDA adaptés au lieu des méthodes standard pour remédier à ce problème. Néanmoins, le calcul de SVD en général et de Funk SVD en particulier d'une grande matrice creuse est souvent effectué avec un algorithme itératif basé sur la multiplication de cette matrice creuse. Dans cette thèse nous avons exploité la bibliothèque MLib de Spark pour utiliser la SVD et profiter du parallélisme des techniques d'apprentissage machine intelligent, et d'analyse de données de manière parallèle et en temps réel. Finalement, pour chaque contribution de cette

thèse, des expériences intensives ont été menées en utilisant un ensemble de données réelles et synthétiques. Ainsi on a utilisé des métriques pour mesurer la qualité des algorithmes Top_k (en terme de prédiction et de la recommandation) dans des multiples configurations matérielles et logicielles de façon centralisée et distribuée. Mots-clés : Intelligence artificielle, Optimisation d'algorithme Top_k, Skyline, MCDA, Filtrage collaboratif (FC), Système de recommandation, Traitement parallèle, Big data.

CONTENTS

List of Tables

Table **Page**

List of Figures

Figure **Page**

CHAPTER 1

GENERAL INTRODUCTION

1.1 Context and problematic issues

At the era of Big Data, Information Retrieval (IR) technologies have gained outstanding prevalence in the last two decades. The development of techniques for processing ranking requests is a research focus very involved in the field of information research. Several applications require processing multi-criteria ranking queries, such as meta-search engines on the Web, search in social networks, exploring multimedia document databases. Opposed to the traditional Boolean queries, in which filtering is based on predicates that return true or false, ranking query use similarity predicates that return a relevance score. These queries designate an aggregation function that combines the individual scores produced by similarity predicates, allowing to calculate an overall score for each object. The k objects with the best overall scores are returned to the final result. Moreover, The study of the problem of query optimization is regularly linked to recommendation systems. However, the Web's growing importance as a support for e-commerce, business activities, and decision support sites has allowed the evolution of recommendation system technology. A key driver in this area is the Web's convenience, enabling users to express their opinions on what they like or dislike. Consider, for example, a content service provider's Netflix [6] scenario in which users can quickly provide feedback with a single click. Using these artificial intelligence techniques, developing intelligent recommendation systems (RS), while integrating decision support, aims to personalize content for each user and reduce the cognitive load of the user's information. Inspiration from this type of recommendation system led us to develop a multi-criteria recommendation system (RS) while incorporating decision support, which aims to personalize content for each user and reduce the user's information's cognitive load.

This research aims to determine how Spark can be used to implement and improve the well-known collaborative neighborhood-based filtering algorithm's scalability. This research is based on ideas presented in the implementations of other machine learning algorithms on Spark, including SVDFunk and Gradient Descent[6]. It also builds on several ideas implemented as part of Map-Reduce for collaborative element filtering presented by Schelter et al. [7], such as selective sampling of experienced users and the use of a scattering variable to store the element similarity matrix. In addition, this approach uses a hybrid MCDA-based recommendation that will be examined later in his thesis.

1.2 Background of Research and Basic Concepts

Before discussing Artificial Intelligence tools for RSs, let us articulate some statistical information in the decision-making within Big Data's circumstances as it is a part of our study. However, according to Splunk[8]: A novel published research shows that fewer than one in 100 European leaders systematically base their decisions on data. This survey, conducted in France, the UK, and Germany, shows that organizations are under increasing pressure with the digital transformation. More than half (53%) of European decision-makers make at least one strategic business decision a day, and an overwhelming majority (90%) make such decisions on the same day. For 66%, this is done even in less than two hours. While decisions are being made in an increasingly shorter period, the frequency of findings is also increasing: respondents report earning an average of seven strategic choices each week. This issue has implications for the way decision-makers make their choices. When asked about the criteria that guide their decisions, respondents primarily refer to their coworkers (52%), almost equal to their clients' feedback (51%).
Meanwhile, this European statistical example wasn't a unique example of the frequency of using Big Data. Still, today the decisions are consistently based on data worldwide, and with the digital transformation, it's that start to be an essential need. Adding that, the cases of large amounts of data and a high number of evaluation criteria for socio-economic objects are also quite frequent. The assessment may involve potentially millions of data entries and millions of different standards [9]. Using large amounts of data and with multiple criteria, which describe a process or an object, requires proper presentation of data in an understandable form of decision-makers to form and understand available methods of processing such data.
However, a majority agrees that there are areas where decisions would benefit from being evidence-based [9]. Overall, others see client growth as an opportunity to make better use of data. The two main obstacles are the need to act quickly and the obligation of asking for more data. To deal with the given limitations, improve the use of MCDA tools in the context of Big Data infrastructure is very required for rapidly extracting essential information using recommendation systems. Therefore what are these MCDA tools? How can specific methods be used in the context of new recommender systems generations? In the following chapter, we will overview from

state-of-the-art about the most related MCDA approaches in our studies.

1.2.1 Big Data management and its essential relationship with our study

The concept of Big Data is now commonly used, often too generically, to define an idea, a methodology, a solution, or a fascinating discovery that could revolutionize our vision of the world. This confusion arises mainly from the fact that this term is relatively broad, ultimately quite recent, and difficult to be defined because it is at the border of several fields. We quickly understand the words "data," but finally, why do we associate it with the adjective "Big"? What does the combination of the two words mean really in the present world, and what can we expect in the future?

Big Data took on its current meaning in the 1990s: a technological challenge to analyze large datasets, initially scientific but increasingly collected daily by various technical means. Besides, Big Data research is undoubtedly one of the most critical research areas in the last decade, as is its impact on industries. The research on the application of data analysis, such as in data mining, statistics, and information systems, has been ongoing for several decades. In the last ten years, industries and academia have begun to focus more attention on the development of studies and applications of Big Data. As can be seen, over the last decade, the volume of data has increased enormously with the development of technologies for data generation, retrieval, addressing, storage, and output, especially devices connected to the internet of things (IoT) and the new database technologies, e.g., NoSQL. IoT products, like smartphones, tablet PCs, and handheld devices, are being adopted worldwide by consumers as well as by all types of industries [10]. The International Data Corporation (IDC) predicts that: By 2025 [11], there will be 175 zettabytes of data, more than 41 zettabytes in 2019, and only two zettabytes in 2010. While 79.4 zettabytes of this are expected to be created by IoT devices, 87 zettabytes will reside in public cloud environments [10].

The trend of increasing data volume has revealed a large number of potential opportunities. In the early 2010s, the term "Big Data" appeared on the horizon. Big Data's characteristics were studied and explained, leading to the development of the research, analysis, and application path. Therefore, Big Data refers to the generation of massive data and the development of technologies capable of processing them to extract correlations or meaning [12]. Worldwide Big Data market revenues for software and services are projected to increase from $42B in 2018 to $103B in 2027, attaining a Compound Annual Growth Rate (CAGR) 10.48% according to Wikibon. Consequently, Forrester predicts that the global Big Data software market will be worth $ 31B in 2020, growing 14% from the previous year. The entire global software market is forecast to be worth $628B in revenue, with $302B from applications [13]. According to an Accenture study, 79% of enterprise executives agree that companies that do not embrace Big Data will lose their competitive position and face extinction. Even more, 83% have pursued Big Data projects to seize a competitive edge. 59% of executives say Big Data at their company would be improved through the use of AI,

according to PwC. However, These statics tell us that everybody lives under the effect of Big Data, and everyone has their vision towards this phenomenon. And consequently, everyone has his exploitation and predilection towards Big Data. However, the increase in the number of disciplines affected by this sector and the difference between these objective visions has raised many nuances to come out with a more precise and more generalized definition of the Big Data concept. The effervescence (buzz) around this phenomenon generates some confusion definition, if only because some critics find in it an argument of weariness. Although the word Big Data sometimes sounds like used to excess, we should not forget that the phenomenon of being is genuine. For defining Big Data, some researchers are based on the V notions to give a precious definition of this concept. Therefore, it is quite commonly accepted that Big Data is defined by the 3V, 4V, even the 8, or 10V. It is the most schematic and synthetic way possible to explain what is in this notion.
In what follows, we present the Big Data definition using 10 Vs:

1. **Volume**: Data volume refers to the large dataset generated by science and education, enterprise, and social interaction records. The amount of data plays a vital role in storage and processing. With the advent of Big Data technology from 2014 to 2016, statistics estimated 90% of data collected since the beginning of humankind had been generated during these two years [14]. The notion of volume can be expressed in numbers; Today, we talk about storing and processing exabytes (10^{18}) or even zettabytes (10^{21}), whereas barely ten years ago, we were talking about megabytes (10^{6}), stored on diskettes. Adding that this is because Big Data far exceeds the size of traditional operational databases or data warehouses[15]. Traditional databases are typically gigabytes or even terabytes in size. Large volumes of data are large enough that new measurement units are required, such as petabytes and exabytes [16], [14].

2. **Variety**: More than just volume, this data is also more diverse than ever before. This phenomenon is linked to the diversification of the Internet and digital technology uses[17]. There is an enormous variety of data sources, formats, and domains. We need to process not only structured data but also semi-structured and mostly unstructured data. The structured database includes a unified data format and can be easily managed using database tools. But unstructured and semi-structured data is more challenging to analyze and make decisions. Traditional data analysis systems have used the Relational Database Management System (RDBMS). Traditional RDBMSs have required expensive hardware and only apply to structured data [15]. As a result, semi-structured and unstructured data requires a more advanced processing infrastructure. Traditional management tools are unable to handle the high volume and heterogeneity of large data sets. However, it also adds to the complexity and is probably one of the barriers to the effective use of a large data set and decision making. That's what makes the mix complicated. That's why we can no longer call it a relational database. It is a great challenge to establish or build a system

that can directly integrate such a mixture of data in common Big data architecture[18]. Besides, on the web, people use various software and internet browsers, and they send data to the cloud differently [19]. Large data platforms would need to process data from various sources, such as e-mail, mobile devices, distributed sensors, web pages in different forms of text, images, audio, video, drones, and multimedia data. The diverse nature of the data can strengthen the system in some ways. It is also essential to keep in mind that most data comes directly from a real human interface, and errors are inevitable. We find that the variety of data directly affects its integrity. In other words, the more varied the data, the more errors it contains.

3. **velocity**: This refers to the speed about which new data is produced and progress. Consider just how quickly messages on social networks go viral in seconds, how quickly fraudulent banking transactions are detected in minutes, or how long it takes for software to analyze social networks and capture the behaviors that trigger a purchase, must take milliseconds! Big Data now allows us to analyze data while it is being generated, without having to research it in databases. Data can be input in two ways: batch data, where the data set is imported at one time, and transmitted is imported and processed while generating [15, 20]. Stream processing is the solution for choosing Big Data's analysis platform, as the real-time process is generally time-sensitive and requires faster, almost instantaneous analysis results. For example, Hadoop (High-availability distributed object-oriented platform) (Hadoop It is also the name of the yellow elephant cuddly toy of Cutting's son). [3] [21] is highly efficient for processing archive data in a batch mode, whereas Apache Spark [22] is more efficient for temporal and interactive job analysis[17, 23]. These technologies will be discussed furthermore in Chapter4.2.7.

4. **Variability**: In the context of big data refers to the number of inconsistencies in the data. These need to be detected by anomaly and outlier detection methods so that meaningful analyses can be performed. Also, it refers to the wide variety of data dimensions resulting from the many disparate data types and sources. And finally, the range of Big data can also relate to some inconsistent speed at which essential data are loaded into the database[15].

5. **Veracity**: It refers to the source or reliability of the data source, its context, and its importance for the resulting analysis [24, 25]. This is one of the impoverished characteristics of large data sets. As any of the discussed properties increase, veracity (confidence in the data) decreases. This is similar, but not identical, to the validity or volatility discussed following.

6. **Validity**: When the status of the data changes from exploratory to actionable, the data must be valid. A data set may not have veracity problems but may also not be valid if it is

[2]https://mbaron.developpez.com/tutoriels/bigdata/hadoop/introduction-hdfs-map-reduce/

not understood correctly. Like truthfulness, validity refers to the accuracy and correctness of the data for its intended use. According to Forbes, it is estimated that scientists spend 60% of their time cleaning data before they can analyze it[15, 26]. This property of Big Data is essential for finding hidden relationships between elements within vast sources of Big Data generation. Validity extends beyond the domain of the application and the application itself. The validity of Big Data generation sources and the resulting analysis must be precisely right if the study result is to be used in the decision-making process.

7. **Vulnerability**: One of the first things that come to mind when we raise Vulnerability is the issue of privacy. How do users of web and online systems and customer protect their data? Once this is done, a user will be more interested in hearing about the others Vs. - value. "We can support people find out how to extract the most benefit from their data, for example, by finding a cheaper product, that all depends on how we are currently adapted to think about data. Therefore, it is what the Vulnerability has just been imposed as a tenth V of Big data features that has a similar importance to the other nine Vs.

    - **Security** In their work [27], authors Moreno et al. assert that the main security weaknesses linked to Big Data stem from the lack of authentication mechanisms, such as user ID and password, and the absence of secure means of access to databases in the cloud. Big Data's highly dynamic distributed computing environment, involving vast amounts of elastic data, makes data security a major challenge. Traditional approaches to security, such as firewalls and Intrusion Detection Systems (IDS), are not designed to handle this massive amount of data for protecting the distributed file systems. Big Data platforms have tried to mitigate these risks by using security mechanisms to manage a group of data without security. Moreover, Kerberos is challenging to install and configure in the cluster. As a result, other security practices have emerged, such as implementing security controls closer to the data and applying cryptographic models in all data, both static and in transit. Other mechanisms are also trying to remediate this risk, such as the Blockchain [28, 29], quantum security [30], a cryptographically secured distributed database technology decentralized storage and transmission of information. That provides access to a single, unchangeable data source.
    - **Privacy** This concept is associated with the idea of what is ethically acceptable and what is not. In parallel with the necessity for a more explicit and thorough set of rules on this subject, the adoption of a code of conduct by businesses, governments, and public entities is also essential [29]. Such a code of conduct should identify the purpose of data collection, its coverage, and its collection limits.
    - **Standardization** Another vulnerability is the lack of widely accepted standards in the area of Big Data. Criteria are used to enable, promote, measure, and control technology in a wide range of communities [29, 31]. Today, there are a significant

number of communication protocols in various fields of large-scale data. The lack of standardization hinders innovation and creates barriers to user adoption of extensive data, as it becomes necessary to be familiar with many different protocols.

8. **Volatility**:We have chosen to talk about the aforementioned V because we see as many other researchers that this V depends on the three first Vs. So, we cannot understand the volatility without understanding the volume, variety, and velocity of Big Data. Indeed, Big Data's volatility refers to the lifetime, or more precisely, to the validity period of data and the duration for which they must be stored [15].
   For some sources, the data should always be there, but it may not be for others. It is necessary, then, to understand the requirements, availability, and lifetime of the data. The speed of creating new data must be taken into account: for example, Facebook reports 600 terabytes of incoming data per day. Google alone processes an average of more than "40,000 search queries per second," which amounts to approximately 3.5 billion searches per day [13]. Knowing what data is available and how long it can help us define the requirements and policies for maintaining voluminous data. Big Data helps us as a practitioner recognize better ways to develop and provide our products and services. This management is made possible only when big data is integrated into organizations and enterprises' business processes.

9. **Visualization**: Today, valuable data is becoming a business imperative because it enables organizations to achieve several objectives [32], such as i) Applying analysis beyond traditional use cases to support decisions in real-time, anywhere, anytime ii) Applying data to decision making by allowing different people to explore and analyze information and provide information to others—iii) Optimizing all types of decisions made by experts or automated systems [33]. However, visualization tools face technical challenges due to technical limitations in memory and poor scalability, functionality, and response time. Traditional graphs cannot be relied upon when trying to plot a billion data points. Therefore, different ways of representing data are required, such as data clustering, tree-maps, parallel coordinates, circular network diagrams, and cone trees. It is practically complicated to develop meaningful visualization considering the remaining variety of high-speed data sets and the complicated relationship between them. Consequently, visually representing the data has made Big Data much more adaptable and valuable[34]. Figure 1.1 is an example that shows how vital a presentation of this massive data of a LinkedIn personal network in an astonishing visual with links from the center can be for decision making or for analyzing a given situation [35].
   In the context of the figure 1.1, khan et al. [15] consider that complexity of data management is very difficult for giant data sets, especially when they come from different sources because linking, matching, and transformation are significant activities. The above complexity deals with the degree of correlation and inter-dependencies in large data structures.

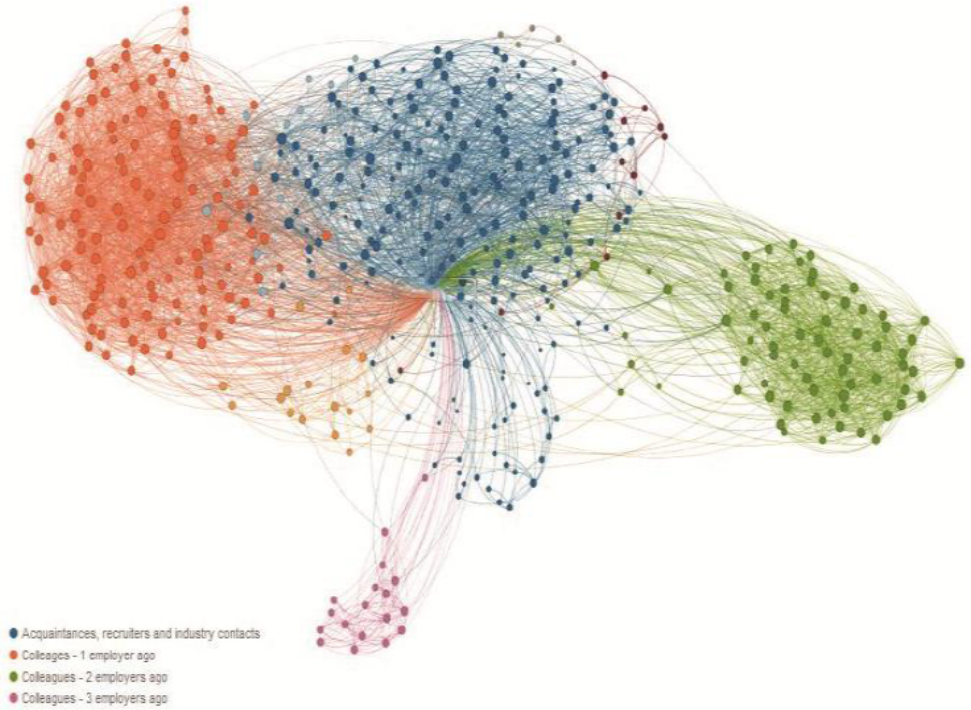


Figure 1.1: Massive data of a LinkedIn personal network in an astonishing visual with links from the center [35]

Small changes can have a considerable effect on a system's behavior or might not apply any changes at all. The authors define this concept as one other V of Big Data characteristics, which is Viscosity. Still, they also talk about 10 V because they did not consider the V of Vulnerability, which was explained above.

10. **Value**: It refers to the usefulness of data in decision making and is one of the most critical factors in Big Data as it directly impacts corporate profits. Besides, it noted that organizations agree that business information and analysis differentiates them in the industry as the best and worst performers [17]. Thus, the value lies in the careful study of accurate data. Since the 1990s and early 2000s, social scientists in various fields have increasingly relied on digitized empirical research methods. Surveys can be conducted via the Internet using research tools that collect or process data in digital form—adding that they have begun to use digital material that has not been specially created for research purposes. The introduction of big data technology and the exploitation of the Internet of Things and the 5G networks and the extent of the Internet in no small part of a given population has created vast amounts of digital data of unknown size and structure[15, 17]. While telephone companies and retailers generally do not share their customers' data with the academic community. Researchers have focused their efforts on the massive amounts of publicly available data on the Internet that often provide insights into previously

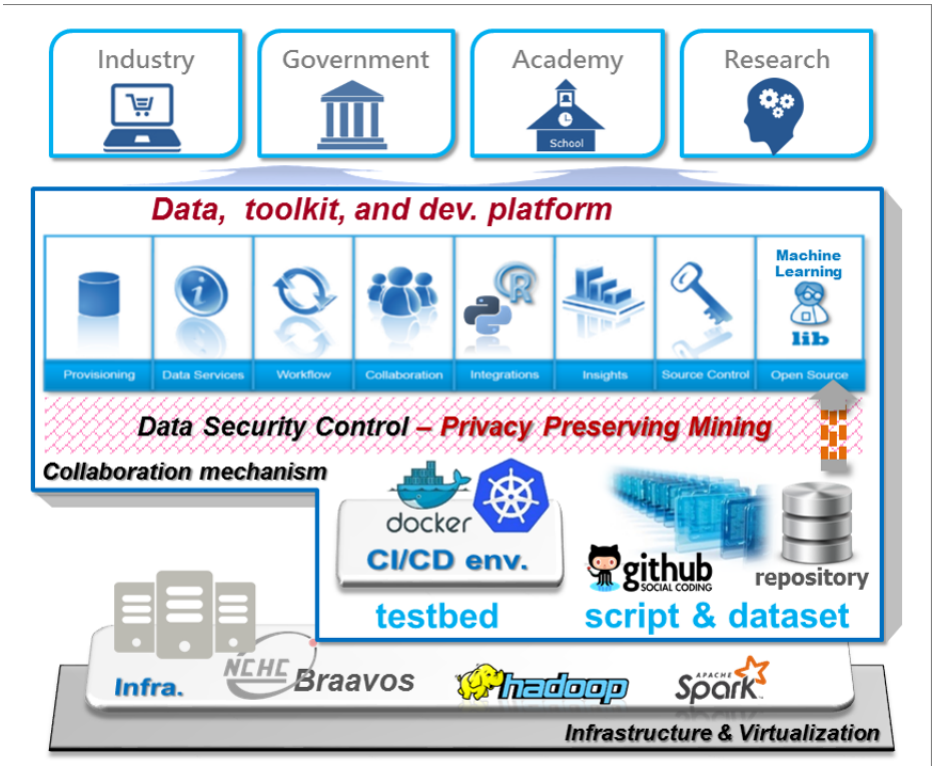


Figure 1.2: Big Data platform Figure reproduced from [5].

inaccessible topics. Subsequently, the methodological literature began to discuss research practices, opportunities, and drawbacks of online research. According to this vision, two aspects deserve further discussion: The current debate around the concept of "Big Data" and the question of finding "meaning" in digital media data [36]. We char the ides of the authors in [17] that consider this V as the most related V with others. That is because we are always interested in extracting maximum value from any extensive data set that we have to work with. In this case, we need to look for the real amount of that data. In short, we need to be careful about the investment in data storage. Storage can be cost-effective and relatively cheaper at the time of purchase. Still, such under-investment can be detrimental to valuable data, for example, storing clinical trial data for a drug or a virus vaccination [37] on an inexpensive and unreliable medium may save money today but may put the data at risk tomorrow. How we develop policies and structures that will eventually balance the reward and risk of data. At the same time, these policies and structures, if not carefully drafted and implemented, can prevent companies from extracting the real value from the data [38].

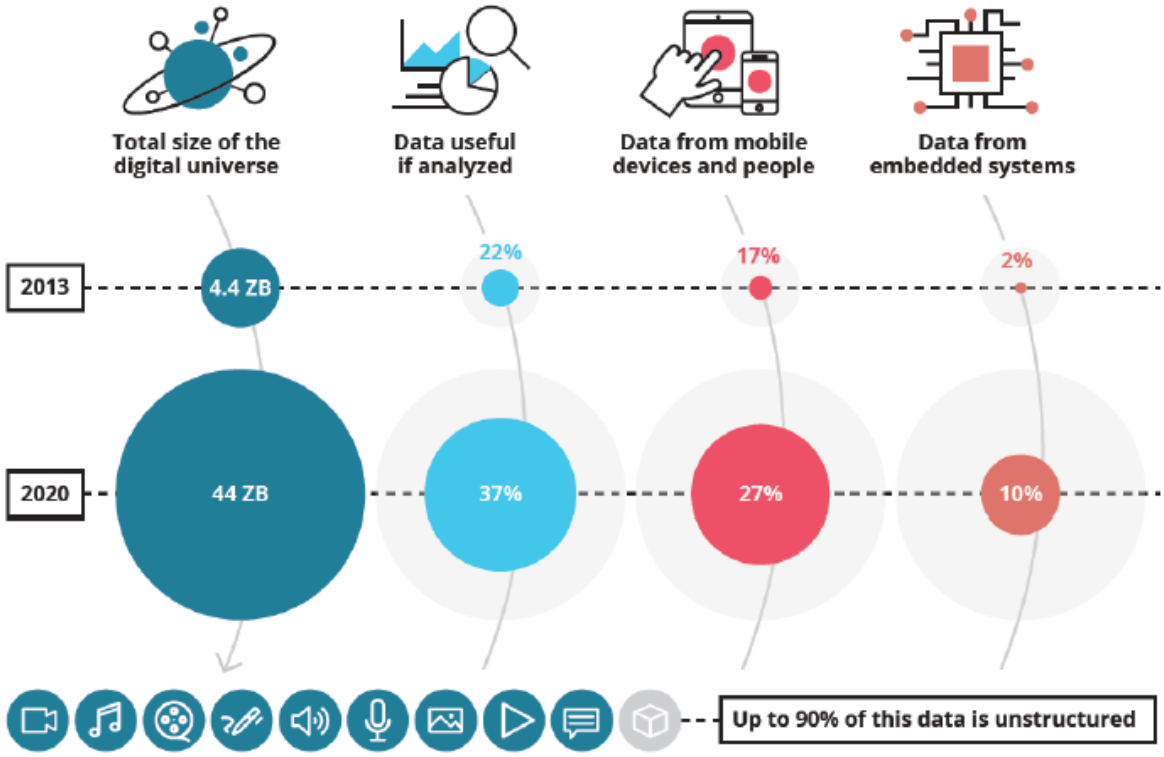


Figure 1.3: The expanding digital universe 2013-2020 [5].

1.2.2 Parallelism and Map-Reduce paradigm in Big Data architecture

A large number of Big Data architecture is starting to be used and especially the famous Hadoop Platform. Currently, Hadoop is the leading platform for Big Data 1.2. It is used for storing and processing huge volumes of data. Hadoop is an open-source software framework for storing data and running applications on clusters of standard machines. This framework allows the processing of massive Data on clusters ranging from one to numerous devices. Its insistence manages the distribution of data in the cluster machines. The Hadoop framework consists of many Open Source components [39], all connected to a set of core modules designed to capture, process, manage, and analyze large volumes of Data. These core technologies are : Hadoop Distributed File System (HDFS). This file system supports a conventional hierarchical directory but distributes files to a set of storage nodes on a Hadoop cluster. It is a programming model, and an execution framework for parallel processing of applications in batch mode [40], [41]. It is designed for managing a large number of potentially multiple files in a highly distributed environment (up to thousands of servers). It breaks down the processing of an operation (called a "job" at Hadoop) toward several steps, two of which are elementary, to optimize parallel data processing. These operations are respectively the mapping and the reducing while the mapping performs a specific operation on each element of the input list: from a file in the form <key, value>, it then generates an output

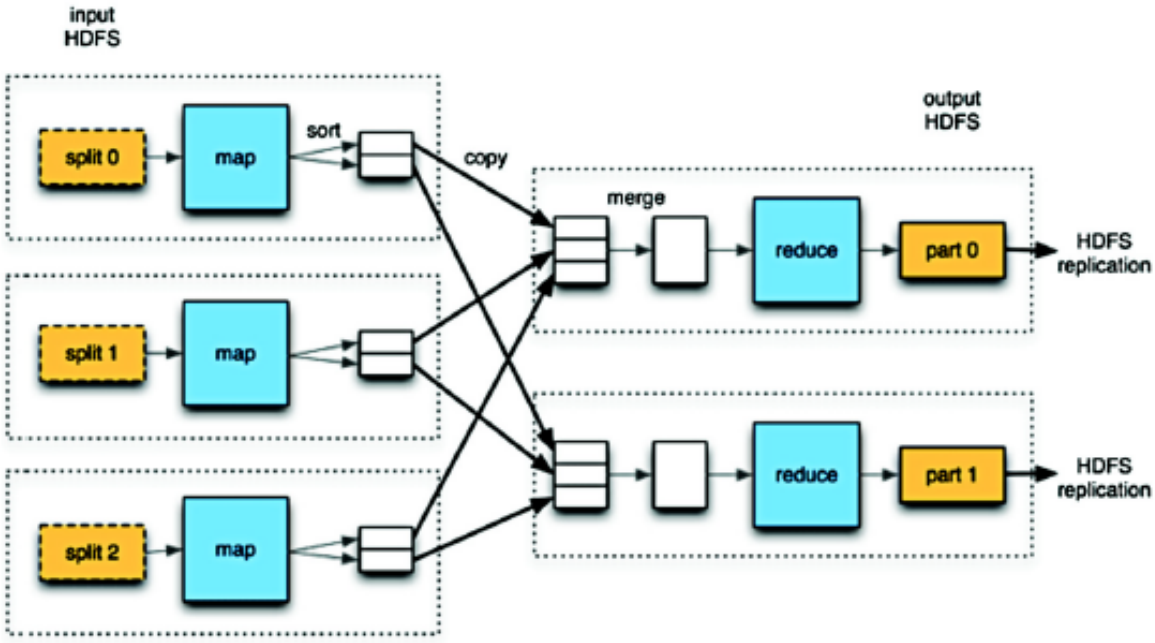


Figure 1.4: Map-Reduce work in Hadoop[5].

list in the same way as shown in figure 1.4. Another operation between Mapping and Reducing called Shuffling rearranges the elements of the list to prepare the Reducing. The Reducing is then processed, giving the final output. Thanks to YARN, Hadoop has seen several improvements; For example, YARN (Yet Another Resource Negotiator) takes care of scheduling tasks (jobs) and allocates resources to the cluster to run applications, and arbitrate when there is a conflict of resources. It also monitors the execution of jobs. Hadoop also has expressed a significant change, notably Hadoop 2.0. However, Hadoop 3.0 includes new features such as erasure coding to manage fault tolerance. Hadoop 3.x also reduces storage costs from 200% up to 50%. It has also implemented a new command-line tool known as Disk balancer. Thus, Hadoop 3.x has improved overall performance table 2.3 show the added value of Apache Hadoop3 over Apache Hadoop2.

1.2.3 Artificial Intelligence tools for Recommender Systems

In a competitive market, it is difficult for companies to offer products and services that respond directly to a particular customer's requirements. Customized online services help solve a significant problem of information overload by making it easier for customers to make decisions and improve user experience. The recommendation systems that are based on these personalized online services were created some 20 years ago. They were developed using techniques and theories from other artificial intelligence (AI) fields for user profiling and preference discovery. recently, there has been a considerable increase in application era numbers based on artificial

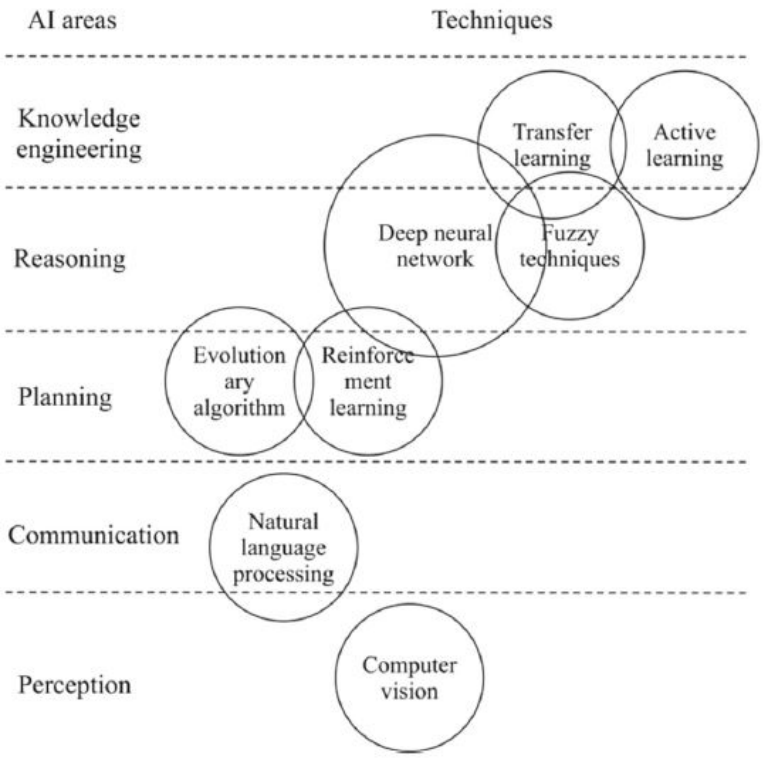


Figure 1.5: AI areas and techniques [42].

intelligence. Among these successes are Deepmind services AlphaGo, the AI-based program that won the famous award game of "Go" against a professional human player, and self-driving car, as well as others in the fields of computer vision and speech recognition. These constant advances in AI, Data analysis and large data sets represent an excellent opportunity to recommend systems to gain acceptance for AI's impressive achievements. Artificial intelligence is the set of methods, concept and techniques used to create machines capable of simulating intelligence. Current achievements of artificial intelligence can be used in diagnostic aid, decision support, solving complex problems such as resource allocation problems, assistance by machines in dangerous or high-precision tasks, task automation. Developing AI techniques is to achieve automated intelligent behavior that covers mainly six areas: knowledge engineering, reasoning, planning, communication, perception, and movement[43] etc. Definitely, knowledge engineering applies to techniques that are used for knowledge representation and modeling to enable machines to understand and process knowledge; Techniques for reasoning are developed for problem-solving and logical deduction; Planning is to help devices to set and achieve a goal; Communication aims to understand natural language and communicate with human; Perception plays the role of analyzing and processing inputs such as images or speech; and finally, motion is about movement and manipulation. Except for the motion, techniques in the first five areas can be applied to enhance and boost the growth of recommender systems due to the enormous information processing

requirements. as presented in Figure1.5.
The author zahang et al in [44] presented eight main models and methodologies as shown in Figure 1. namely : Deep neural networks, transfer learning, active learning and fuzzy techniques are representatives of knowledge and reasoning and are interconnected with each other. Evolutionary algorithms and AI domains and techniques enhancing learning are related to reasoning and planning, while natural language processing is the main technique for communication and perception, and computer vision is for image perception. Of the eight methods, natural language processing and computer vision are two areas of IA techniques application in recommendation systems.

1.3 Motivation

The motivation for this thesis's work is related to the fact that decision systems in general and recommendation systems in particular need to be improved and adapted in the context of comprehensive data technology. Nevertheless, Big Data has become part of our lives, and general data hides in them the solutions to many problems in the academic or industrial world. Big Data provides the raw ingredients to build sophisticated technologies such as supercomputers. Permit us citer the example which was given by the American industrial group Honeywell, which supplies equipment to companies announced that they have finally managed to bring the world's most powerful quantum computer to market in mid-2020 [45]," according to a release. "Quantum computing will enable us to address complex scientific and business challenges, leading to dramatic improvements in computing power, operating costs, and speed," said Darius Adamczyk [1]. Harnessing Big Data in creating tomorrow's technology and generating a real-time analysis of today's information, the example of social networking companies such as Facebook and Twitter use the power of the Big Data that users create every minute. In particular, Data is always on the move. We support the fact that Big Data will eventually make its mark in the world of decision support. Therefore, the application of the Big Data concept and technologies on decision making and Information Retrieval (IR) have made exceptional use over the last two decades with the explosion of Big Data users. This practice was incorporated particularly in the World Wide Web. That makes the Recommendation Systems (RSs) based on IR more studied and designed to maximize the degree of satisfaction of certain objective conditions. User satisfaction is one of the greatest. The optimization of Recommendation algorithms and query processing like Top_k to generate the best choices for users to meet their needs better and the development of IR have revolved around the definition of models and algorithms that best achieve this objective. The recommendation algorithms above have been particularly influential, drawing on the MCDA domain of multi-criteria decision support and applying the collaborative filtering widely used in next-generation RSs to meet user needs. These systems are referred to in our context as multi-

[1] https://www.honeywell.com/en-us/newsroom/pressreleases/2020/03/honeywell-achieves-breakthrough-that-will-enable-the-worlds-most-powerful-quantum-computer

criteria recommendation systems (MCRS). Also, the impact of the Big Data management on the Top_k algorithms runtime and quality of results given by these algorithms motivated our research to extract, filter, analyze and evaluate the results found in Big Data platforms applying an intelligent collaborative filtering system. Besides, inspired by the principal relationship existing between some paradigms like Top_k and Skyline, we combine them in this work to have more enhanced decision-making tools while incorporating users' choice by the MCDA methods. -we detailed the relationship mentioned above later in this document- Then, this motivation led us to perform this combination in a Big Data context while using the well-known technologies in this field to construct a complete RS.
However, Big Data technology such as Map-Reduce and Spark Frameworks has been widely adopted because of its qualities, and several applications have been designed and implemented. Nevertheless, several issues and challenges still exist regarding scalability and organizational performance. Top_k algorithms are very involved in terms of runtime; Map-Reduce is a solution to parallelize these algorithms while optimizing execution time and ensuring the excellent quality of results requested by the user. Consequently, This thesis deals with the performance challenges using The Top_k in sequential version and a parallel model such as Map-Reduce and RDD technologies.

- How to benefit from the combination of the Skyline and the Top_k
- How to combine the Top_k algorithm with sophisticated machine learning techniques and use Map-Reduce and Spark to parallel the recommendation system?
- How to deal with the problem of the sparse matrix generated by the user while using collaborative filtering approaches?
- How to adapt these approaches for dealing with a challenging problem like Cloud Computing, sport management and other different application domains?

The following subsection hyaline the principal contributions to answering these questions.

1.4 Thesis Contribution

The major contributions of this thesis are:

1. A new method was proposed based on a combined approach of the Top_k and Skyline using an adapted MCDA approach to return the most relevant answer to the user. The enhanced Cloud Service Research and Selection System (e-CSRSS) system was applied to the Cloud Computing QoS dataset. This approach was used while managing the size of Skyline points in the high dimensional and the bridge between Skyline and Top_k query; in other words, the principal objective was to use the advantage of Top_k query processing

based on MCDA and apply it on the Skyline query. This processing allows the user of the Skyline query to determine the number of Top_k most relevant answers to be returned. As mentioned earlier, the behavior makes the Skyline query alone a weak concept and produces many incomparable result objects with increasing numbers of dimensions. The system was applied in the real-life scenario when the user can specify the output size. study between the Top_k weighted sum (Top_{kws}) algorithm and Fagin Algorithm (FA) using the correlation metrics evaluation. An extensive enhancement of the e-CSRSS to a Generic and complete system called Generic Research and Selection System (GRSS) shows an equilibrium tool to handle more general scenarios of users' choice. The above-mentioned approach is tested on a real-life dataset of the sports management domain and a synthetic dataset of the Cloud Computing QoS. The extensive comparison with other algorithms like TA and NRA shows its performance.

2. Modeling of the Top_{kws} algorithm based on the Map-Reduce paradigm. A parallel concept of the adapted MCDA method presented by the Bi-objective Weighted Sum (BWS) method is used in the score function of the algorithm

3. An optimization approach is proposed to improve the recommender system's scalability, which includes a dimension reduction approach. Sparsity is addressed by applying an associative retrieval framework and related distribution algorithms to explore transitive associations between users through their past requirements and feedback. The reduction is based on the Funk SVD model [46] in the training process and an adapted bi-objective weighted sum method in the prediction process.

4. A novel parallel algorithm $SPTop_{k_{WS}}$ is proposed using RDD to improve the $Top_{k_{WS}}$ algorithm performance, while combining Top_k-parallel recommendation processing and MCDA optimization.

5. Improvement of the Generic Research and Selection System presented in [1] is proposed by combining the Skyline and Top_k dominating query processing and using a hybrid recommendation based on MCDA and CF approaches. And applications in other domains like Health care and corona-virus investigations.

1.5 International Publications

- International Journals

    1. [1] **K. EL HANDRI** and A. IDRISSI, Comparative study of Top_k based on fagins algorithm using correlation metrics in cloud computing qos, International Journal of Internet Technology and Secured Transactions (IJITST), published (2020). DOI: 10.1504/IJITST.2020.10018455

2. [47] **K. EL HANDRI** and A. IDRISSI Parallelization of a new Top_{kWS} recommendation algorithm based on Spark cloud computing framework. 2020 IEEE systems journal. DOI 10.1109/JSYST.2020.3019368
3. **K. EL HANDRI** A. IDRISSI. A Top_{kws} Algorithm for Synthetics and Real datasets. International Journal of Artificial Intelligence IJAI. 2020. Accepted for publication.
4. [48] **K. EL HANDRI** and A. IDRISSI Étude comparative de Top_k basée sur l'algorithme de Fagin en utilisant des métriques de corrélation dans la qualité de service de Cloud Computing, In EGC 2019, vol. La revue Revue des Nouvelles Technologies de l'Information, RNTI-E-35, pp.359-360
5. [49] **K. El HANDRI** and A. IDRISSI. Correlations and Hierarchical Clustering investigation between weather and SARS-CoV-2. Recent Advances in Computer Science and Communications.2020. DOI 10.2174/2666255813999201109201006.
6. **K. El HANDRI** and A. IDRISSI, and, Map-Reduce Model for Top_{kWS} in Cloud Computing QoS Recommendation system. 2020. Manuscript in preparation for submission.

- International conference

1. [50] A. Idrissi, **K. EL HANDRI K**, H. Rehioui and M. Abourezq "Top_k and skyline for cloud services research and selec-tion system," in Proceedings of the International Confer-ence on Big Data and Advanced Wireless Technologies.ACM, 2016, p. 40
2. EL M. Mastour, **K. El HANDRI** and A. Idrissi : MISC'2018 conference "Advances in Intelligent Systems and Computing" Improved recommendation system based on combining approach using Top_k and k-means
3. [51] **K El Handri**, A. Idrissi. Étude comparative de Top_k basée sur l'algorithme de Fagin en utilisant des métriques de corrélation dans la qualité de service de Cloud Computing. International conference d'extraction et de gestion de connaissance (EGC) 2019: 359-360 https://editions-rnti.fr/?autid=1202142

- **Patent**:

1. **K. EL HANDRI** ,A.IDRISSI. "System collaboratif d'aide à la décision à base des recommandations multi critères" M. A . Pattent No. 50776. 3. Sep. 2020. Accepted for publication.

1.6 Thesis organisation

- Chapter 1: The first chapter was devoted to a general introduction talking about the background and context of the research study, then an overview of our contributions, ending with this thesis's organization.

- Chapter 2: This chapter provides an overview of state-of-the-art related to this thesis's recherche study. Then it exposes the principal challenges in the collaborative filtering and recommendation system fields, especially the exploitation of the MCDA methods with recommendation systems in the context of Big Data, which led us to present thoughts on Multi-Criteria Recommendation Systems (MCRS). The study is not exhaustive, but instead presents some works on the essential tools for processing and optimizing Top_k queries as one of the most exciting recommendation algorithms, which also gives the main objective of this thesis. Finally, this chapter contains a definition of some multi-criteria optimization methods, and a comparison of the most critical collaborative filtering model technologies is studied and presented in this chapter.

- Chapter 3: This chapter presents a new approach to dealing with Skyline refinement based on combining the two paradigms Top_k and Skyline. It proposed addressing the main drawbacks of both methods and combining their advantages while using MCDA techniques. The application of the approach above within this chapter concerns the Cloud Computing Service domain, mainly.

- Chapter 4: In the beginning, this chapter presents a generic recommendation system to respond to different scenarios of users' choice. The proposed method is based on a multi-objective optimization to help the decision-maker choose the best solution according to the most relevant information that reflects his needs. Furthermore, this chapter compares the proposed approach with others from state of the art based on Top_k algorithms. The comparison considers the different characteristics of Top_k query processing, which are discussed in Chapter 2, and finally, this comparison took into account both real and synthetic databases.

- Chapter 5: In this chapter, we expose the amelioration of Top_{kws} in a distributed context based on a parallel solution using the Map-Reduce of Hadoop and Spark RDD paradigms for combining the Top_k algorithm and the MCDA methods trough a new collaborative filtering approach. An evaluation of a similar process using synthetic and real datasets is further proposed while using the generic proposed solution's amelioration in the above chapter.

- Chapter 6: This chapter presents a general conclusion of the research work performed in this thesis and summarizes contributions and results. It likewise highlights remarkable significant suggestions for future research and prospects.

CHAPTER 2

STATE OF THE ART

This work focused on the primary two research areas that interact with the recommendation system field as the primary domain of this thesis. The filed are respectively: Top_k algorithm through recommendation systems and collaborative filtering (more specifically, the optimization of Top_k query processing and its combination with Skyline in sequential and parallel context). Then the MCDA area (and, more specifically, the Adapted MCDA techniques for recommendation systems). State of the art is still always related to Big Data, discussed above in the introduction and background section. More concretely, its central relationship with the two designated areas concerns various research fields, including the management of enormous data for recommendation system platforms. Therefore, the art state describes the Recommendation systems' current knowledge by analyzing similar or related published work. It provides a comprehensive overview of what has been done in the field related to the MCDA era and what will be further investigated to formulate the problems and hypothesis the thesis intends to address. This chapter also tries to analyze, compare, and evaluate and links different era of researches. While answering some questions like what is the basic principle that governs the operation of recommendation algorithms? What is a recommendation system using MCDA, and what are recommendation algorithms?

2.1 Recommendation system and collaborative filtering

Recommendation systems are technologies that help users find a set of new or appropriate items, usually by predicting the "rating" or indicator of decision which users would give to a topic they have not yet attended at [52]. Recommendation algorithms are based on the fact that there are essential dependencies between user-centered and item-centered activity.
This system can also be a decision-making system based on recommendation algorithms. For

Recommendation Approach	Recommendation Technique	
	Heuristic-based	**Model-based**
Content-based	Commonly used techniques: TF-IDF (information retrieval) Clustering Representative research examples: Pazzani & Billsus 1997 [53]	Commonly used techniques: Bayesian classifiers Clustering Decision trees Artificial neural networks Representative research example: Pazzani & Billsus 1997 [53] Zhang et al. 2002 [44]
Collaborative	Commonly used techniques: Nearest neighbor (cosine, correlation) Clustering Graph theory Representative research examples: Delgado & Ishii 1999 [54] Pennock & Horwitz 1999 [55] Sarwar et al. 2001 [56]	Commonly used techniques: Bayesian networks Clustering Artificial neural networks Linear regression Probabilistic models Representative research examples: Hofmann 2003 [57] Si & Jin 2003 [58]
Hybrid	Combining content-based and collaborative components using: Linear combination of predicted ratings Various voting schemes Incorporating one component as a part of the heuristic for the other Representative research examples: Tran & Cohen 2000 [59] Melville et al. 2002 [60]	Combining content-based and collaborative components by: Incorporating one component as a part of the model for the other Building one unifying model Representative research examples: Popescul et al. 2001 [61]

Table 2.1: Classification of recommendation systems [2]

example, a decider maker can be a doctor user interested in a disease documentary (which is our product or service in this case); The doctor is more likely to be involved in another documentary or an educational program than in an action movie. Let us item $i \in$ Items, which can be a service or a document as the example at the beginning, and user u, with $u \in$ Users and which can be a doctor or the other user of the service. The utility function that measures the suitability of recommendation is created between Items and Users while recommending the item i to user u is generally defined as R : Users $*$ Items $\rightarrow R_0$, which is usually described by non-negative integers or real numbers within a certain range [2]. It is assumed that this function is not known for the entire Users $*$ Items space but is only specified on a subset. Consequently, as part of the recommendation, we would like the system to approximate the utility function value for each item the user showed some interest in to be able to select a set of items that the user is really interested in so who ever is selling the item may recommend it to the user, aiming at an increase of income. Therefore, we would like every user $u \in$ Users to be able to:

- (a) Approximating the utility function $R_{(u,i)}$ for the $i \in$ Items item for which $R_{(u,i)}$ is not yet

determined

- (b) Select one or a set of elements i that will maximize $R_{(u,i)}$,

$$\forall u \in Users, i = \underset{i \in Items}{\arg\max} R(u,i) \tag{2.1}$$

This general definition incorporates the principal relation between User and Items in the context of recommendation. Therefore, recommender systems made significant progress over the last decade when numerous content-based, collaborative, and hybrid methods were proposed, and several industrial systems have been developed. However, despite all of these advances, the current generation of recommender systems surveyed in this paper still requires further improvements to make recommendation methods more effective in a broader range of applications. The work in [2] reviewed various limitations of the current recommendation methods and discussed possible extensions that can provide better recommendation capabilities. These extensions include, among others, the improved modeling of users and items, incorporation of the contextual information into the recommendation process, support for multicriteria ratings, and provision of a more flexible and less intrusive recommendation process.

Web users have become more and more accustomed to receiving a message tailored to their needs and desires, considering their satisfaction and introducing their preferences to display precisely the items most recommended for them according to their interests in real-time. For example, according to the search engine marketing agency and the technology company for online marks 360 i [2]: "Online purchases at the end of 2016 amounted to $300,000 per minute. A brand received an average of 350,000 "likes" per minute on Facebook." A simple explanation for this phenomenon is the intelligent use of Big Data tools and powerful recommendation system algorithms. Because the consumer now demands attention, that leaves little room for traditional marketing techniques. However, we start with this example of digital marketing to talk about recommendation systems since it is very much related to this area.
Nevertheless, this research area is also associated with cognitive science, approximation theory, information retrieval, forecasting theories, and management science links before it started to be an independent research area in the mid-1990s. From these years, the recommendation system has begun to focus on the problems that manage rating structures. The evaluation matrix model's result representing the dependency between users and items is used to make forecasts for the target users. The higher the number of rated items available to a user, the easier it is to make sound predictions about their future behavior. The improvements in recommendation systems incorporate new methods for representing user behavior and information about the recommended items. This improvement requires more advanced recommendation modeling methods. Like

[2] https://360i.com/

incorporating a variety of contextual information into the recommendation process, the use of multicriteria ratings, and the development of less invasive, resilient recommendation methods that rely on metrics that calculate accurately determine the performance of recommendation systems [2]. Besides, recommender systems are regularly classified into the following categories, based on how recommendations are given:

- Content-based recommendations: The user is recommended items similar to those he preferred in the past; It should be noted that content-based systems also use rating matrices in most cases, although the model generally focuses on the ratings of a single user instead of all users[62].
- Collaborative recommendations: the user has recommended items previously recommended by people with similar interests and preferences;
- Hybrid approaches: These methods combine collaborative and content-based methods. In particular, they recommend systems that predict the absolute values of the rankings that individual users would give to still-unknown articles [63].

From state of the art, another distinguishing between recommendation models is based on different characteristics. All models have a similar approach to it. This distinction really on which the basic models of recommendation systems work with two types of data [62], which are :
(i) user interactions with articles, such as ratings or buying behavior, and (ii) user attribute information and elements such as text profiles or relevant keywords.
Methods that use the former are called collaborative filtering methods, while methods that use the latter are called content-based recommendation methods. Note that content-based [2, 64] systems also use rating matrices in most cases. However, the model generally focuses on the ratings of a single user instead of all users. In knowledge-based recommendation systems, recommendations are based on explicitly specified user needs. Instead of using historical ratings or purchasing data, knowledge bases and external constraints create the suggested proposal. Furthermore, this literature distinction considers the hybrid method, which can help design techniques that can work more robustly in a wide variety of contexts. The basic idea behind collaborative filtering methods is that these unspecified rankings can be attributed because the observed orders are often positively correlated between different users and different items. For example, let's take two users in the above example named User1 and User2, who have very similar preferences. If the ratings, which both have specified, are very similar, their similarity can be identified by the underlying algorithm. In this situation, it is very likely that ratings in which the specified values' unique person are also alike. The mentioned similarity can be used to make inferences about incompletely specified values. Most collaborative filtering models focus on exploiting correlations between items or between users for the prediction process. Some models use both types of correlations. Besides, some models use carefully designed optimization techniques to create a

training model, in the same way, that a classifier makes a training model from labeled data. This model is then used to impute missing values in the matrix in the same way that a classifier imputes missing test labels. Two types of methods are commonly used in collaborative filtering, memory-based methods and model-based methods:
Netflix is a top-rated online video content rental service. Netflix has a recommendation service called Cinematic, which offers content to users based on user preferences. Netflix provided its data in October 2006 in a challenge called the Netflix Prize. Its objective was to improve the efficiency of the recommendation system. Narayanan and Shmatikov (2006)[65] showed that 96% of users could be uniquely identified by considering only up to 8 films evaluated by the user with the evaluation date [66]. In our work, we will use methods proposed by Netflix during the Netflix prize, such as the FunkSVD model.

2.1.1 Collaborative Filtering using Matrix Factorization and SVD based model

Collaborative filtering (CF) is the most general approach to designing a recommendation system and has been successfully used in many applications. However, CF is confronted with problems related to the sparsity of the rating matrix [67] due to increased Internet use and large-scale data production. These challenges are well supported by matrix factorization (MF) [68]. This unsupervised method [67] is one of the most effective dimensional reduction techniques in machine learning. The idea is to use latent factors to represent the user's preferences in a much smaller space.

$$\left[M\right] = \left[U\right]\left[W\right]\left[V\right]^T$$

that is

$$\begin{bmatrix} & & \\ & M & \\ & & \end{bmatrix} = \begin{bmatrix} & & \\ & U & \\ & & \end{bmatrix} \begin{bmatrix} & & 0 \\ & .. & \\ 0 & & \end{bmatrix} \begin{bmatrix} & & \\ & V & \\ & & \end{bmatrix}^T$$

Where superscript T is for the conjugate transpose of the corresponding matrix, if M coefficients are real, it is merely transposed. This expression involves: U, a unitary matrix (orthogonal if M is practical) of dimensions (m,n): its column vectors have a unit norm and are orthogonal by pairs: $U^TU = I_n$, where I_n is the identity matrix of dimension n. Its columns are constituted of the first eigenvectors U_m [67]. It was ordered according to decreasing values of the eigenvalues of matrix MM^T. Thus, the MF method based on Funk SVD [69] enables us to find a simple way to evaluate the recommendation engine and create a good U and V^T matrix.
Consequently, to make recommendations, even if there is a very sparse matrix. Unfortunately, Funk SVD is not suitable for use alone, as we may face a common problem in the recommendation

engine called "Cold Start Problem" [70]. This problem means that we cannot make recommendations for new users or new movies. A successful approach is to combine the Funk SVD with a ranking-based algorithm like Top_k query processing, k-dominating query processing, and content-based methods. The proposed solutions were moving in the same direction as our optimization approach. Where we use the Funk SVD model with Skyline and Top_k query processing using MCDA.

2.1.2 Recommender systems using MCDA

2.1.2.1 MCDA Methods

Recommender systems (RSs) have been successfully applied to alleviate information overload and assist users' decision-making. Multi-criteria recommender systems are one of the RSs that utilize users' multiple ratings on different aspects of the items (i.e., multi-criteria ratings) to predict user preferences. Traditional approaches treat these multi-criteria ratings as add-ons and aggregate them together to serve item recommendations. The authors in [71] proposes novel approaches that treat criteria preferences as contextual situations. More specifically, most studies from state of the art like [71, 72] believe that part of multiple criteria choices can be viewed as contexts. At the same time, they can traditionally use multi-criteria recommender systems. From state of the art, the study [71] compares the recommendation performance among three settings: 1) using all the criteria ratings traditionally, 2) treating all the criteria preferences as contexts and 3) utilizing selected criteria ratings. They try to use two real-world ratings dataset-based approaches to reveal that treating criteria preferences as contexts can improve item recommendations' performance, but they should be carefully selected. They prove that the hybrid model of using selected criteria preferences as contexts and the remaining ones in the traditional way as the overall conqueror in the experiments.
Interacting with a recommender system means making different decisions. The decision such as selecting a movie or a drug from a recommendation list, selecting specific feature values (e.g., the bandwidth of the network, latency of service) as criteria, selecting feedback features to be critiqued in a critiquing based recommendation gathering, or deciding a repair proposal for variable user choices when interacting with a knowledge-based recommender [73].In all these scenarios, users have to solve a decision task. Our work primarily focuses on Multi-criteria Decision Making in Recommender Systems; efficient human decision-making approaches in different recommendation scenarios. This relationship brings us to discuss the most critical used MCDA tools in the context of recommendation. Therefore,In this section we present and summarize the following elements of the popular MCDA methods, based on the State of the art and the work of Chen et al. [74].

Beyond the past forty years, many aggregation models have been developed, including MAUT, AHP, and Outranking. New methods or improvements on existing methods, such as Olson's (1996)

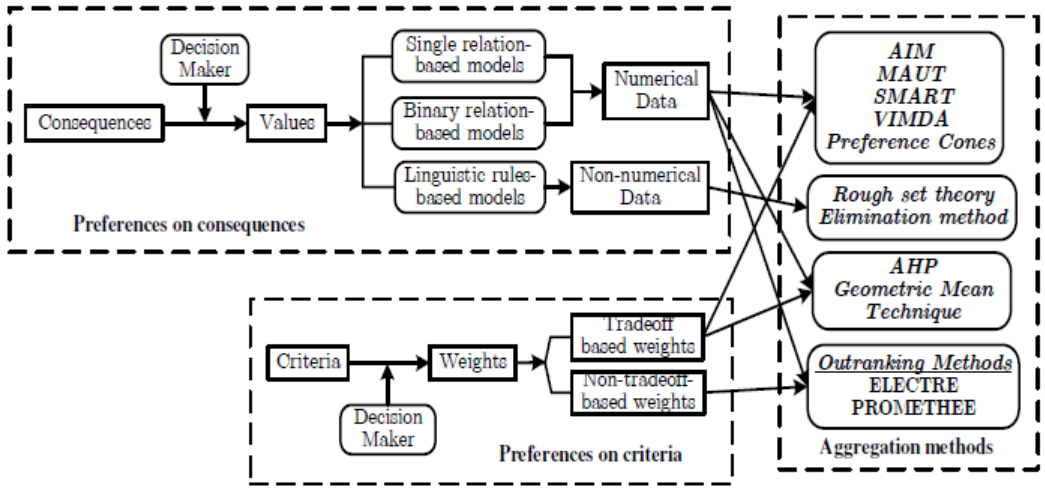


Figure 2.1: Global schema representing Preference Acquisition and Aggregation in MCDA

work, provided comprehensive literature.

- Value Production Methods There are three conventional approaches to generating values based on consequences: single alternative-based methods, binary alternative-based methods, and rule-based linguistic methods. Figure 2.1 elaborates on the mentioned methods.
  - single alternative-based methods
  - binary alternative-based methods
  - rule-based linguistic methods

- Weighting Methods Belton and Stewart (2002) review two kinds of weights: tradeoff-based weights and non-tradeoff-based weights.
  - Tradeoff-based weights: They highlight the “compensation” of values over criteria, which allows preference data to be compared since they are aggregated into a single representative evaluation.
  - Non-tradeoff-based weights: They do not permit direct tradeoffs across criteria; they are usually associated with outranking methods.

  Methods like Analytic Hierarchy Process (AHP), WSM, and geometric ratio weighting are integrated Methods of tradeoff-based weight categories, which suggests they continue from preference data and weight assessments to aggregated preferences to final results. These methods progress from preference data and weight assessments to aggregated choices to final results. These methods are primarily used in literature. Besides, outranking methods like ELECTRE family and PROMETEE family focus on weights (PROMETEE)

and do not implement procedures to obtain values. In contrast, other arrangements can generate weight information because they are based on peer comparisons of alternatives. The outranking relationship is made up of two indices: concordance and discordance. These two indices allow the introduction of vetoes and incompatibility in their implementation. Furthermore, thresholds and fuzzy measurements can be introduced to make the final recommendation for the studied issue.

- Methods of Aggregation In this part of the section, we explain the relationship between aggregation and other techniques in MCDA, showing in figure2.1, while given a general summary to demonstrate the nature of these three approaches. These three methods. Namely, Preference, Acquisition, and Aggregation. In the case of processes that operate cardinal preference. The Data and trade off-based weights include the aspiration-level interactive model (AIM), Multiattribute Utility Theory (MAUT), Simple Multi-Attribute Rating Technique (SMART), Visual Interactive Method for Decision Analysis (VIMDA), and Preference Cones. Methods that employ binary preference Data and tradeoff-based weights include Analytic Hierarchy Process (AHP), Geometric Mean Technique. Eventually, methods that employ binary preference data and non-tradeoff-based weights are explained above, Outranking Methods. However, Linguistic aggregation methods use just linguistic preference data. In counting, Weights are assigned to rules rather than criteria. These techniques include Rough Set Method and the Elimination Method.

2.1.3 MCDA Methods applied in Multi-objective optimization

The methods of multiobjective programming and multi-criteria decision analysis are tools to help the decision developed since 1900. The method of MCDA should be used to support decision-makers (DMs) [45] to make their decision better. It is for this reason that we find "decision analysis" "or" "decision aid" instead of" decision making." Nowadays, all systems that can be described by a mathematical model are optimized. The quality of results and predictions depends on the relevance of the model, the algorithm's effectiveness, and means for digital processing [75]. "The multiobjective optimization can be simply defined as the area that searches such a balance we cannot improve a criterion without damaging at least one of the other criteria" (Vilfredo Pareto). This balance is called the Pareto optimum [76] [77]. We wanted to showcase how such a method should help organize and synthesize the DMs' information to make better decisions. There are many mathematical methods of multi-criteria analysis to generate sets of the multiobjective program (MOPs) solutions, but they can be grouped into two approaches [75]: The first approach is based on utility theory and uses a prioraggregation criteria into a single criterion; the second approach is based on outranking methods [78] such as ELECTR I, III, IV, IS [79, 80], PROMETHEE I

and PROMETHEE II [79, 81]. In the same context, another distinction from the literature between Methods of secularization and methods nonscalarizing [82] In this thesis, we use The WSM, which will be discussed in the following chapters, especially we discussed the adaptation of this method in the context of multi-objective programming. However, from state of the art, we also use the ELECTRE (Elimination and Choice Translating Reality) IS method. The ELECTRE methods are the best known and most widely used outranking methods. They were introduced for the first time in 1965. The main objective was to remedy a problem that came from the MARSAN method (Method of Analysis, Research, and Selection of New Activities). However, the latter showed some limitations, and the ELECTRE I method was constructed to overcome them [83–85].
ELECTRE I is the oldest and simplest ELECTRE method. It is used in choice problems and requires that all criteria be expressed in numerical scales with the same ranges. Considering a set Alt of m variables and a set F of n criteria, we define for each criterion cr an evaluation function GCR and a weight kcr reflecting its importance in the decision-making process. To assert that an alternative a outperforms an alternative b, denoted a S b, we must first calculate the concordance index C(a,b) and the discordance index D(a,b). The concordance index measures the appropriateness of the arithmetic aSb as follows:

$$C(a,b) = \frac{1}{\sum_{cr=1}^{n} k_{cr}} \sum_{cr=1} k_{cr} \forall cr \text{ such as } g_{cr}(a) \geqslant g_{cr}(b) \tag{2.2}$$

This index is defined between 0 and 1. Similarly, the discordance index measures the opposition to the statement aSb as follows:

$$If \forall \text{ cr } g_{cr}(a) \geqslant g_{cr}(b) \text{ then } D(a,b) = 0$$
$$\text{if else } D(a,b) = \frac{1}{\delta} \max_{cr}[g_{cr}(b) - g_{cr}(a)]$$

Or δ is the maximum difference concerning the same criterion for two given alternatives. The outranking relation S is constructed by comparing the concordance and discordance indices. With a concordance threshold defined by c', and a defined discordance threshold d' as follows:

$$aSb \Leftrightarrow C(a,b) \geqslant c' \text{ et } D(a,b) \leqslant d' \tag{2.3}$$

However, real-world problems often consist of a heterogeneous set of criteria that may be contradictory and cannot always be expressed in numerical scales. Besides, in the context of real-world applications, this method suffers from the problem of imprecision, uncertainty, and preliminary determination. To remediate these problems, the ELECTRE IS method

has been defined. ELECTRE IS a generalization of ELECTRE I because it integrates the use of pseudo-criteria (Na and Werey, 2009) instead of objective criteria and introduces the vcr veto, the pcr preference threshold, and the qcr indifference threshold. In addition, the conditions of concordance and discordance change, and the latter is called the non-veto condition. A pseudo-criterion is a gcr function associated with two threshold functions: the indifference threshold qcr and the preference threshold pcr. The preference threshold pcr is a real-valued function such that, for two alternatives a and b in Alt, $pcr(gcr(a))$ is the minimum positive value of a score difference of type $gcr(a) - gcr(b)$ that could be compatible with the preference of a over b. The main difference between the veto threshold vcr and the discordance index D lies in the fact that vcr is correlated to the differences in preference between $gcr(a)$ and $gcr(b)$. In contrast, D has correlated to the absolute evaluation of a gcr criterion for a given Alt action. Moreover, the veto threshold is not necessarily associated with all criteria. t is assumed that:

$$\forall x \in \mathbb{R} q_{cr}(x) \leqslant p_{cr}(x) \leqslant v_{cr}(x)$$

The concordance index is formalized as follows:

$$C(a,b) = \sum_{cr=1}^{n} w_{cr} c_{cr}(a,b) \tag{2.4}$$

Where;

$$w_{cr} = \frac{k_{cr}}{\sum_{cr=1}^{n} k_{cr}} \text{ et } c_{cr}(a,b)$$

is defined as follows:

$$\begin{aligned} &\text{If } g_{cr}(b) - g_{cr}(a) > p_{cr}(g_{cr}(a)) \text{ alors } c_{cr}(a,b) = 0 \\ &\text{Else if } g_{cr}(b) - g_{cr}(a) \leqslant q_{cr}(g_{cr}(a)) \text{ alors } c_{cr}(a,b) = 1 \text{ ,} \\ &\text{Else } c_{cr}(a,b) = \tfrac{p_{cr}(g_{cr}(a)) + g_{cr}(b) - g_{cr}(a)}{p_{cr}(g_{cr}(a)) - q_{cr}(g_{cr}(a))} \end{aligned} \tag{2.5}$$

Thus, to validate the statement aSb, C(a,b) > c' and :

$$g_{cr}(a) + v_{cr}(g_{cr}(a)) \geqslant g_{cr}(b) + q_{cr}(g_{cr}(b)) \eta_{cr}$$

. The latter is known as the non-veto condition, or

$$\eta_{cr} = \frac{1 - C(a,b) - w_{cr}}{1 - c' - w_{cr}}$$

2.2 Top_k Algorithm through recommendation systems and Collaborative filtering

The Top_k algorithms are the most used algorithms in the recommendations system [86, 87],55,65]. Query processing Top_k is approached from various angles and links to many databases of research areas, including query optimization, query language, and indexing methods [79, 88, 89].

It is involved with obtaining the k objects that have the best overall score. In the following, we introduce a classification of different Top_k algorithms by presenting the various parameters that distinguish between the various categories of search Top_k query type, aggregate function, level of execution, and data access. The study in [87] presents a rich overview of the different techniques as shown in 2.2, that we review in this section.
Nevertheless, we have to rely on other related definitions to define the Top_k, such as the definition of the ranking function and aggregation concept, which will be detailed after and especially in the contribution sections, taking into account the most useful objective which allows a good understanding of the concept of Top_k and consequently of the algorithms of Top_k. However, a simple example of a Top_k request from the literature can be a request like:

SELECT Top_k in DataBase
Where Type = movie
Order by f (Name, actor, rating)

In the query above, the user is looking for a movie in the database (DataBase) according to preference criteria concerning movie name, actors, and rating, which are modeled by a ranking function f. The issue is not to reach the straightforward answer but to reach the answer in an optimal time. From a legal perspective, the Top_k problem addressed is applied to databases with a specific form. Specifically, it requires that each database considered has at most one quantitative attribute, and the entries should be ordered decreasingly according to it. From a theoretical point of view, such a requirement does not impose any restriction since every database can be decomposed as a set of databases holding such property.

2.2.1 Top_k algorithms characteristics in relational database

This section presents an overview of the different types of Top_k query processing from state of the art, which depends on many other criteria as shown in Figure 2.2

2.2.1.1 The Top_k Query Types

The first type is the request Top_k selection, which is the simplest and most used in the literature. This kind of query specifies the grading criteria, the value of k, possibly a function of filtering (selection) of objects, and the aggregation function to rank items according to the set standards. Each search criterion is evaluated by a predicate allowing for a local score for each object. A merge function (aggregation), generally monotonous, is used to calculate each object's overall score. Items k with the best scores are returned in the result. In this case, we find the NRA and TA algorithm [90, 91]; as well, there is the Upper algorithm in [92, 93]. The second type is the Top_k queries join in which the scores are assigned to the objects due to one or more join operations. As the selection queries, the Top_k join queries return the best k objects, which are

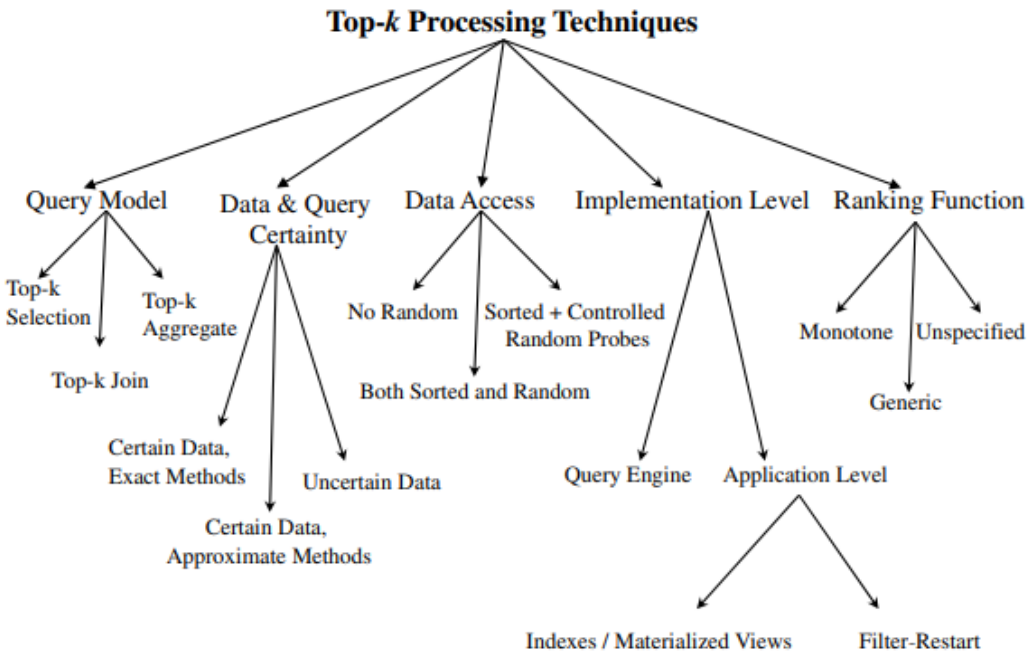


Figure 2.2: Classification of Top_k Query Processing Techniques [3]

always based on a fusion of function generally monotonous for establishing a final ranking of items. The third type is the Top_k queries aggregation in which the objective is to assign a score to a group of objects, not individually. A request Top_k aggregation returns the k best groups of objects.

2.2.1.2 Types of access to scores

Two types of access are distinguished: Sequential (sorted), to return whenever the next object in the list, in descending order of scores. Moreover, Direct (random), to produce the score of a given object. It will return the local score of o object in the source S.

2.2.1.3 Execution level

In this context, we distinguish two levels of integration processing of Top_k queries with the engine of the database: High level (application), Treatment ranking queries Top_k is considered here as an application on a layer of high-level outside the engine of the database. Alternatively, the Low level (the engine) covers all techniques that require adaptations and changes in the engine of the database to support the classification property.

2.2.1.4 The three sources to consult

S1 is a source sorted, accessible only by sequential access (S-source) to return whenever the next best object from the list in descending order of scores; SR-S2 is a source available by sequential and direct access The best-known algorithm in this category is the one proposed by Fagin et al. TA (Threshold Algorithm) [88, 91].S3 is an unsorted source (R-source) accessible by direct access. Each source has a different classification by the same objects. [90, 92, 93]. Data certainty: specific data and accurate result: this configuration is adopted by many works done for query processing Top_k. These techniques are based on some data to return the Top_k best quality. Specific data and approximate result: the search for an inaccurate output of a request Top_k from certain data is provided in the multi-criteria search [52]. It aims to reduce the cost of execution of Top_k queries. Uncertain data: in this category, we consider all the techniques treatment of Top_k queries based on probabilistic data.

2.2.1.5 Ranking function

As the name suggests, the ranking function shall allow ranking items to return the k best among them. It will enable aggregating the scores produced by each criterion by an overall score for each object [87, 92]. The vast majority of Top_k query processing techniques consider simple monotonous ranking functions (example: sum, min, max). Other methods use more sophisticated functions specific to address more complex problems, decision support, for example. Specific applications do not require classification of objects by relevance score (we generally talk about conflicting research criteria) [19], and in this case, using a ranking function is not necessary for managing this problem; the technique is known as Skyline [94],[95]. The processing techniques of Top_k queries aging Advanced can be classified according to the ranking function into two categories:

- With ranking function: The vast majority of treatment techniques Top_k queries consider a monotone ranking function. The monotonicity of this function optimizes the cost of research by providing guarantees on the evolution of candidates' scores. We believe this work is the most common context of the use of a monotonous ranking function.

- No ranking function: this category is mainly technical used as part of the Skyline [95] queries whose result is the set of objects not dominated by other objects according to conflicting criteria. The table 1 summarizes the properties of different processing techniques Top_k [87]. The Top_k algorithms are expensive, especially in a context where the result must be exact, i.e., we have sufficient guarantees that the objects returned in the final product had the highest scores among all existing. To reduce the cost of execution, we explore the multi-criteria analysis method.

Several articles have addressed the issue of processing Top_k requests in centralized database management systems [96],[97],[98],[99],[100]. One of the first examples of this works was presented in work [101] by Fagin. The author has given interesting preferment algorithms like Fagin algorithm (FA), threshold algorithm (TA), and No random access algorithm (NRA) [91] as we already sitter it as an example in the beginning. Also, he has provided a more improved algorithm presented in [88]. Indeed, FA and TA are intended for scenarios where sorted and random access costs are comparable, whereas NRA (as the name suggests) is used for dealing with the situation where random access is costly or is impossible [3].
However, different studies consider the data presentation distinction in a distributed area, indicating if the data are presented vertically or horizontally. TA then caught the attention of the authors of [86] by trying to improve some limitations, focusing on the vertical distribution of data to nodes, while the study [102] involves a horizontal distribution of data. Based on a shareholding scheme, the proposed algorithm gradually recovers the Top_k result, minimizing the number of servers polled and Data transferred. After that, the authors extend (Distributed Top_k Query Processing) DiTo to support limited data summaries. Related to the presented work, the same authors use the inherent relationship between Skyline and Top_k [103, 104] to effectively identify the servers that store the best k results. They then presented an extended version of DiTo.
Furthermore, they study it in a high-performance distribution environment. According to our knowledge, our approach remains the first work that uses a bi-objective adapted MCDA optimization in this combination of two paradigms, like Top_k and Skyline. Although it exists some results which are part of the critical relationship between Skyline and Top_k query processing, such as the study in [104] and [102] However, this approach is investigated by the use of the adapted weighted sum model (WSM). Therefore it was used as part of a parallel, and generic Research and Selection System presented in chapter 3.8. This approach either considers the Top_k and MCDA in a single step or combines the Topk MCDA hybridization and the Skyline paradigm in another scenario. Indeed, WSM is the most widely known multi-criteria decision analysis (MCDA) approach for evaluating multiple alternatives based on various decision criteria. Weight assignment is a complex task, especially if the number of measures is large and varied [105]. Another research focus in the field of Top_k query processing handles the preening data processing. For example, the authors Marian et al. in [106] were interested in evaluating Top_k based on sorted and random access as in [91] and [107] in which Trie-Phase Uniform Threshold" (TPUT) was designed. Later, this algorithm was improved by KLEE presented in [87], which approximately supports the Top_k recovery approach using the same TPUT mechanism. In addition, the algorithm uses another variant by applying only two trips. However, given the different access cost scenarios, several algorithms were proposed without random access, such as the NRA algorithm mentioned above and the MPro minimum probing algorithm[108]. A single figure

	Query model			Data query certainly			Data access			Implement Level		Ranking Function	
	Top-K Selections	Top-K Join	Top-K aggregate	Certain data , exact methods	Certain data, approx, methods	Uncertain data	No random	Both sorted and random	Sorted controlled random probes	Query engine level	Application level	Monotone	Generic
TA [91] Quick-Combine [107]	✓			✓				✓			✓	✓	
TA- approx [88]	✓				✓			✓			✓	✓	
NRA [91] Stream-Combine [79]	✓			✓			✓			✓	✓		
CA [91]	✓			✓			✓				✓	✓	
Upper Pick [92]	✓			✓					✓		✓	✓	
Mpro [109]	✓			✓				✓			✓	✓	
J* [110]		✓			✓		✓				✓	✓	
J* e-approx [110]		✓			✓		✓				✓	✓	
PREFER [111] Filtre-Restart [96] Onion Indice [97] LPTA [81]	✓	✓		✓			NA				✓	✓	
NRA –RJ [112]	✓			✓			✓			✓		✓	
Rank-Join [98]		✓		✓					✓	✓		✓	
Rank-SQL-μ operator [113]	✓			✓					✓	✓		✓	
Rankagger Operator [114]			✓	✓			✓			✓		✓	
TopX [115]	✓				✓			✓			✓	✓	
KLEE [87]	✓				✓		✓				✓	✓	
OPT* [116]	✓	✓		✓			NA			✓			✓
OPTU-TopK [117]	✓	✓					✓				✓	✓	
MS_TopK [118]	✓					✓	NA				✓	✓	
Top_{kws} [50] $MRTop_{kws}$ [1] $SPTop_{kws}$ [47]	✓	✓			✓	✓			✓		✓	✓	

Table 2.2: Different characteristics of Top_k algorithms

2.2.2 Parallel Top_k algorithms using Big Data infrastructure

Although these Top_k algorithms are sometimes efficient and ingenious when the research space becomes huge, each algorithm has its preference over other algorithms. Therefore, less sorted accesses lead to more comfortable to manage Top_k query processing to parallelize [119]. Existing solutions are parallelizations using already existing algorithms. Nevertheless, current research on recommendation systems is implemented using collaborative filtering (CF), where recommendations are based on users' historical behavior only, regardless of their knowledge of the field. There are two main approaches to CF: (i) the neighborhood approach and (ii) the latent factors approach; the second one is linked to our study based on the FunkSVD predictive model. We will discuss further in section 2.1.3. In addition to the use of CF techniques, the distributed recommendation system requires a parallel algorithm in a distributed environment. There is however, a growing interest in the parallelization of Top_k queries in various parallel environments recently [120–122]. All the existing approaches share the idea of partitioning the whole dataset into subsets, processing the subsets locally in parallel, and finally merging the results. The correctness of the approaches stems from the skyline operator's activity (Hose

Table 2.3: The added value of Apache Hadoop 3 above Apache Hadoop 2

Attributes	Hadoop2.x	Hadoop3.x
Handling Fault-tolerance	Through replication	Through erasure coding
Storage	consumes 200% in HDFS	Cosumes just 50%
Scalability	Limited	improved
File System	DFS, FTP and Amazon S3	All features plus Microsoft Azure Data lake File System
Manuel Intervention	Not needed	Not needed
Cluster Resource Management	Handled by YARN	Handled by YARN
Data Balancing	User HDFS balancer for this purpuse	User Intra-Data node balancer

and Vlachou, 2012a). Furthermore, in the Big Data era, the importance of Top_k treatment is paramount. It requires a scalable platform because it is practically impossible for users to inspect large sets of unclassified queries simply because of their extreme volume. Despite the importance of the problem, there is a lack of fully parallel algorithms that work effectively on a large scale and probably reveal the correct and exact results overall Map-Reduce framework. [123].Therefore for storing large volumes of data, distributed file systems are a possible solution. In the context of data analysis, a distributed model is advantageous both due to the ability to push computation to various nodes in a cluster and the scalability it providers [124].

Consequently, the open-source Hadoop and spark frameworks became synonymous with Big Data. The distributed file system is coupled with the Map-Reduce engine in Apache's Hadoop project [125]. Meanwhile, one of our contributions discussed in Chapter 5.4 uses Map-Reduce for paralleling our propped algorithm, which is a central component of the Apache Hadoop framework. We also operate another new element that is based on a similar principle and technical behind of Map-Reduce. This component is called the Apache Pig [126]. However, Pig is an abstraction above Map-Reduce. It is a tool and platform that is applied to analyze extensive flow data. It is commonly used with Hadoop; we can perform all data manipulation and operations in Hadoop using Pig. For using Pig, we need to manipulate a high-level language known as Pig Latin. This language offers various operators with which programmers can develop their functions for reading, writing, analyzing [127], and processing data.

In the contribution presented in Chapter 5.1, Pig will be used to enhance the Map-Reduce code, thanks also to the advantage given by Pig comparing it with Map-Reduce and which can be shown in the table 2.3. Therefore, to support Top_k queries in the Map-Reduce and also in Spark RDD Frameworks, the algorithms, in general, need to be implemented similarly and modify the sequential version to be suitably managed in the distributed era. In a part of the thesis contributions, we focused on parallelizing our proposed sequential $Top_{k_{WS}}$ algorithm. A parallel version of $Top_{k_{WS}}$ was studied, showing the variation influence of the database size and the criteria number on which the system users choose items. Then shed light on the impact of the number of user requirements giving different scenarios representing the weighting assigned to the function score. The authors also use the similar weighted voted techniques [128].

However, they present a parallel random forest (PRF) for extensive data on the Apache Spark

platform. The PRF algorithm is optimized based on a hybrid approach combining similar data. And parallel optimization of jobs. Further on, to improve the algorithm's accuracy for large, high-dimensional, and to manage the noisy data, the authors use a dimensional reduction approach in the training process and a weighted approach to voting in the prediction process before parallelization. Focusing on CF and task palatalization in a distributed environment, Panigran et al. [129] use Apache Spark to describe an effective parallel implementation of a user-oriented CF algorithm. The case in our approach, which applies dimensionality reduction techniques based on SVD, and Simon Funk's model, discuss how it can effectively solve the ample storage space and the high complexity in the traditional SVD.
However, the authors in [129] use an Alternating Least Square (ALS). Aleksandrova et al. proposed clustering techniques to overcome CF's limitations, including Sparsity and scalability of data, Seed-based Matrix Factorization (SMF) [106]. Both Hadoop and Apache Spark are frameworks for processing Big data but designed for different objectives. In particular, Hadoop forms the infrastructure of distributed data: extensive data collections are distributed among the many nodes that form a cluster of standard servers. This process means that we did not have to maintain expensive specialized material. Hadoop indexes and tracks data status, making it more efficient to process and analyze than before. Spark is a data processing tool. It allows us to perform various operations on distributed data collections but does not give their distributed storage. As usual, Spark is far ahead of MapReduce in performance since data is handled differently here. If MapReduce performs processing in a step by step mode, formerly Spark operates on the whole data set as a single entity. Spark performs all analytical operations in memory almost in realtime; therefore, the data is read from the cluster, the necessary procedures are performed, then the results are written to the cluster, after which the process is completed. With batch processing, Spark outperforms MapReduce by a factor of ten. When analyzing in memory by a hundred. ; This performance needs to be proved by commonly used measurements in this research field.

Moreover, regarding state of the art either in parallel or sequentially Top_k algorithm, the researchers need to evaluate systems' performance based on those algorithms. More preciously, they also need to assess the algorithms themselves in different contexts, such as size variations and dimensional variations. For doing that, various distance measures between Top_k lists are tools in the literature to realize this objective whit is the case of this thesis study.

2.2.3 Generally used Metrics for evaluating Top_k in Recommender systems

The metrics study in the information retrieval domain is ancient as the disciplines itself. However, it's impossible to compare a result without relying on some measure [130]. Despite several standard ways of measuring the quality information retrieval system, it seems that there is no well-studied and well-understood method for comparing Top_k lists for similarity purposes. Nevertheless, there are two main limitations: The first limit, these methods, instead of giving a relative measure of the distance they usually provide independent evaluations of the Top_k. The

other limitation, in the context of the word-wide-web and especially in a different context and that we are dealing with as a treatment domain example, there is often no clear notion of what ground truth is for information retrieval accuracy becomes more difficult [88].

- These observations lead to some questions: how do we define reasonable and meaningful distance measures between Top_k lists?
- What can be the suitable metric to evaluate the performance of Top_k list?

There are many different metrics used in the literature, from classical ones like Kendall's tau and Spearman's foot rule, two well-established distances between rankings. Non-classical ones like MAP and NDCG. Some of these distance measures are metrics, while others are not. Some metrics are also called correlation coefficients, which are used to measure the correlation between the values. By knowing the relationship between different assets, we can have a preventive idea of the Cloud Service System's directional risk and find support, which will diversify it.

2.2.3.1 Various distance measures between Top_klists.

The authors in [130] present an axiomatic approach. Demonstrating that there are several criteria that any evaluation of a metric should satisfy. That we want the sitter before we state the definitions of metrics on which we will be based to test our Top_k list.These criteria are as follows:

- **Richness.** The metric should support element weights, representing the relevance of a particular document or result to the query.
- **Simplicity.** One of the reasons metrics like Kendall's tau remains prominent, even when their shortfalls have been widely recognized, is their inherent Simplicity.
- **Generalization.** Notwithstanding the richness criteria, any proposed metric should collapse to a natural metric when the richer criteria do not play a role.
- **Basic properties.** The proposed evaluation measure should satisfy some basic properties that make it easier to reason about it. And, ideally, should follow the triangle inequality: for two permutations σ and τ , the metric M should satisfy $M_{(\sigma,\tau)} \leq M_{(\sigma)} + M_{(\tau)}$.
- **Correlation with other metrics.** Finally, a suite of metrics, all capturing the same effect differently, is much more powerful than a single new metric.

Let $[n] = 1 \ldots n$ be a universe of elements. Let S_n be the set of permutations on $[n]$ and for $\sigma \in S_n$, let $\sigma_{(i)}$ denote the rank of the element i. We have tree well-known metrics that evaluate the distance between two permutations $\sigma, \tau \in S_n$: The Spearman's foot rule distance $F_{(\sigma,\tau)}$,Pearsman's $rank(\sigma,\tau)$, and the Kendall's tau $K(\sigma,\tau)$.

Among the existing metrics, we have chosen those suitable for our problem and are frequently used in this field. In the following, we represent the formulas of each of these metrics. The three correlation coefficients (Pearson, Spearman, Kendall) are also used as metrics and which can be presented as three exceptional cases of the same formula, called Daniels formula [131]. We consider for every pair of individuals i, j two indices aij and bij the first one associated with the variable X. The second one related to the variable Y, for example, $aij = Xi - Xj$ and we define the following coefficient:

$$\frac{\sum\sum a_{ij} b_{ij}}{\sqrt{\sum\sum a_{ij}^2}\sqrt{\sum\sum b_{ij}^2}} \tag{2.6}$$

which varies between - 1 and + 1 from the Schwartz inequality [131].

Definition 1 The Pearson's coefficient is given by:
Taking $a_{ij} = X_i - X_j$ and $b_{ij} = Y_i - Y_j$. We find the coefficient r of Bravais-Pearson [132].

$$r = \frac{\sum\sum (X_i - X_j)(Y_i - Y_j)}{\sqrt{\sum\sum (X_i - X_j)^2}\sqrt{\sum\sum (Y_i - Y_j)^2}} \tag{2.7}$$

Definition 2 The Spearman's coefficient is given by:
Taking $a_{ij} = r_i - r_j$ and $b_{ij} = s_i - s_j$. We obtain The Spearman's coefficient according to the formula number 2.6 where r and s are the ranking ranks according to X and Y,

$$\rho = \frac{\sum\sum (r_i - r_j)(s_i - s_j)}{\sqrt{\sum\sum (r_i - r_j)^2}\sqrt{\sum\sum (s_i - s_j)^2}} \tag{2.8}$$

where the r and the s are the ranking ranks according to X and Y.
The simple formula for Spearman's coefficient is based on the difference between each pairs of ranks:

$$\rho = \frac{6\sum d_i^2}{n(n^2-1)} \tag{2.9}$$

where d_i is the differences between each pair of ranks (e.g., the ranks of corresponding values r_i and s_i, while n is the volume of the sample.

Definition 2 The Kendall's coefficient is given by:
Taking a_{ij} = sign of $x_i - x_j = \frac{x_i - x_j}{|x_i - x_j|}$ and b_{ij} = sign of $y_i - y_j$ where i < j have the same sign the pair is concordant, if have opposite signs the pair is discordant. We obtain The Kendall's coefficient:

$$\tau = \frac{\sum\sum \frac{x_i - x_j}{|x_i - x_j|} \frac{y_i - y_j}{|y_i - y_j|}}{\sqrt{\sum\sum \frac{x_i - x_j}{|x_i - x_j|}^2} \sqrt{\sum\sum \frac{y_i - y_j}{|y_i - y_j|}^2}} \tag{2.10}$$

while Spearman's rank correlation coefficient is like the Pearson correlation coefficient but computed from ranks, Kendall's tau τ correlation is rather a probability [132]. a simpler formula of Kendall's τ is given by:

$$\tau = \frac{2(C - D)}{n(n-1)} \tag{2.11}$$

In a sample of n observations it can be found $n(n-1)/2$ pairs corresponding to choices $1 \leqslant i < j \leqslant n$. where C is the number of concordant and D is the number of discordant.

2.2.4 Other used recommendation metrics

We employ two metrics for prediction accuracy, namely, Mean Absolute Error (MAE) and Root Mean Square Error(RMSE) are often used in collaborative filtering methods to measure the prediction accuracy. They are defined as formula 2.12 and formula 2.13:

$$MAE = \frac{1}{n} \Sigma_{i=1}^{n} \left| r_{ui} - r'_{ui} \right| \tag{2.12}$$

$$RMSE = \sqrt{\frac{1}{n} \Sigma_{i=1}^{n} \left(r_{ui} - r'_{ui} \right)^2} \tag{2.13}$$

Where r_{ui} represents the correct rating of user u for item i, Or u_i represents the predicted rating of user u for item i, and N represents the number of predicted values. The MAE value is calculated from the average difference between the item's accurate rating in the testing set and the predicted rating. The smaller the value is, the higher the prediction accuracy will be. The RMSE value is the root-mean-square of the deviation of the predicted value from the actual value to the number of observations. RMSE and MAE are both calculated separately for each user, and then the average is calculated for all users. Furthermore, in this thesis, we use the average error determined for all predictions. They are called GlobalRMSE and GlobalMAE. The effectiveness of recommendations can also be given by the well known IR standard measures recall, and precision [133]. We can define the recall of a Cloud Service recommender system as an example of the ratio of the number of relevant Cloud Services returned to the total number of pertinent Cloud Services for the user query in the collection, as shown in 2.14.

$$recall = \frac{|\{\text{relevant objects }\} \cap \{\text{ retrieved objects }\}|}{|\{\text{ relevant objects }\}|} \tag{2.14}$$

$$precision = \frac{|\{\text{ relevant objects }\} \cap \{\text{ retrieved objects }\}|}{|\{\text{ retrieved objects }\}|} \tag{2.15}$$

The precise ratio of the number of relevant objects that can be a Cloud Services as mentioned above 2.15 [134].

CHAPTER 3

COMBINING TOP$_K$-MCDA BASED ALGORITHM WITH SKYLINE FOR CLOUD COMPUTING SELECTION SYSTEM

This chapter is organized as follow: The first section section 3.2 presents a contribution introduction about the Cloud Computing services and the use of recommender systems in this field of recherche as it is a very promising domain for extracting data, and the second section 3.3 presents our contribution using the Top_K algorithms. In section 3.4 we give an overview of some TOP K algorithms using one of the MCDA methods, which is the weighted sum. In section 3.7, we offer the result of this work, and finally, we conclude in section 3.8.

3.1 Contribution 1: Combining Top$_k$ algorithm with Skyline based algorithm

3.2 Contribution introduction

Cloud Services remain one of the world's largest segments because it has experienced rapid growth published, making the service selection and recommendation a difficult task for users and service providers precisely, they are a significant growth opportunity for companies. For this reason, how to select the best and optimal services will be a considerable challenge. In this contribution, we present a new approach for cloud service selection. We use a Top_K query, which returns k services that dominate the largest number of benefits produced by the Skyline query. This request is based on a method that is among the most important and commonly used to compute scoring functions: The weighted sum method, also called linear. Our application is a vital decision support tool because it provides an efficient decision-making tool to help decision-makers to meet the most suitable services to their preferences. Also, it combines the advantages of Top_k

and Skyline queries without sharing their disadvantages. On the other hand, our results on real data show the relevance of this designed specific algorithm 1 and fully exploit the problem's characteristics.

3.3 Combining $\mathbf{Top_k}$-MCDA based algorithm with Skyline

Over the last years and with the emergence of concepts such as Big Data and cloud computing, IT organizations and individuals have increasingly adopted web services published to the cloud to have an agile solution for their business problems [135, 136]. As defined by the National Institute of Standards and Technology: "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [94, 137]. Moreover, information systems use different techniques that today exceed the database's interrogation and involve the end-user by considering their choices [138, 139]. This need has given birth to recommendation systems (RS) that are computer-based intelligent techniques to deal with finding appropriate products amongst various. It also remains an area with still high interest because, on the one hand, it is a costly research problem. On the other hand, it is due to the abundance of practical applications that help users provide personalized recommendations. [136, 138, 139]. Soon, as easy as it is to increase the volume and range of available Cloud services, it is equally challenging to find relevant best or optimal services among many different existing kinds. So this may present a significant problem and a great challenge[50]. In the same context, we will focus on the new approach based on a recommendation algorithm Top_K to optimize old results found by authors in [140–142]. This optimization will be achieved by interpreting the end-user preferences, using Multi-Criteria Decision Analysis Methods (MCDA), particularly the weighted sum [89].

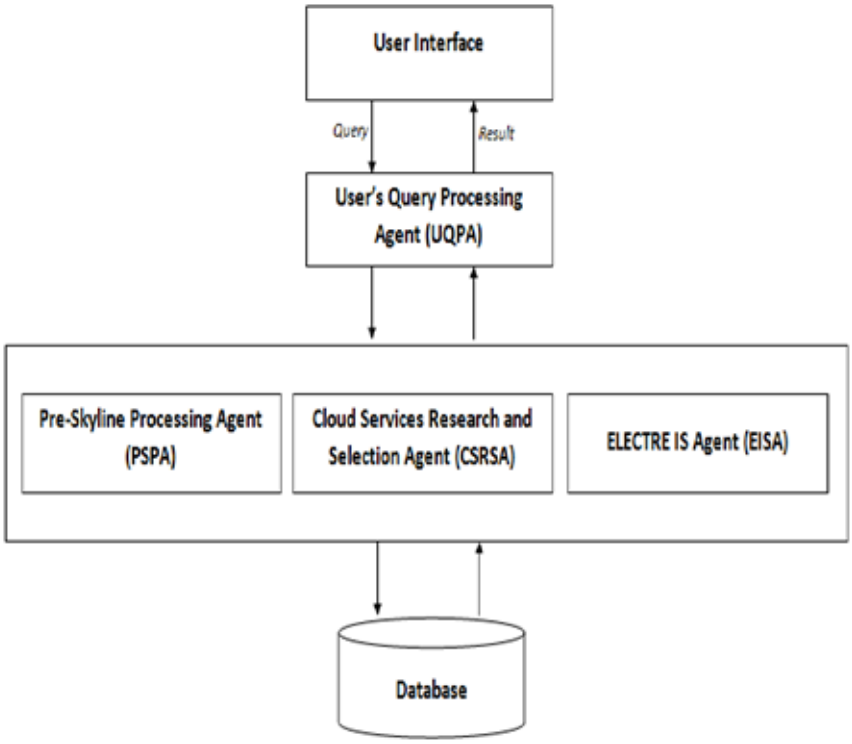
3.4 Top_K agent in the e-CSRSS

Nowadays, all systems that can be described by a mathematical model are optimized. The quality of results and predictions depends on the relevance of the model, the algorithm's effectiveness, and means for digital processing [75]. "The multi-objective optimization can be simply defined as the area that searches such a balance we cannot improve a criterion without damaging at least one of the other criteria" (Vilfredo Pareto). This balance is called the Pareto optimum. However, as mentioned in section 2.1.3, the method of MCDA should be used to help the decision-makers (DMs) to make their decision better. It is for this reason that we find "decision analysis" "or" "decision aid" instead of" decision making " [143]. We wanted to showcase how such a method should help organize and synthesize the information to the DMs to make better decisions and chose the best Cloud service according to the final user. There are many mathematical methods of multi-criteria analysis to generate sets of the multi-objective program (MOPs) solutions, but

they can be grouped into two approaches [75]: The first method is based on utility theory and uses a priori aggregation criteria into a single criterion. Where the second approach is based on outranking methods such as ELECTR I, III, IV, IS [80, 143], PROMETHEE I and PROMETHEE II [79, 81]. In the same context, another distinction from the literature between Methods of secularization and methods nonscalarizing [144] is that these approaches convert the MOP into a single objective program (SOP). This chapter will use an outranking approach presented by ELECTRE IS, which is used with the Skyline on the previous work [94, 95] in what we called the Cloud Services Recherche and Selection System (CSRSS). For instance the mentioned system, we added a new step based on the well-known secularization method [143, 144]. This method is called the weighted sum method (WSM), based on utility theory [145]. As explained above, it is also an MCDA method, which will be the ranking function of the Top_K algorithm for refining the Skyline. Therefore, for doing that, let us tell what the benefits of using the ELECTREIS combined with the Skyline. What can be the added value using another MCDA method combined with the Top_k algorithm for refining the Skyline? We should be noticed that the new system whit is an amelioration of the previous system, will be called the enhanced Cloud Services Recherche and Selection System (e-CSRSS).
It is also worth mentioning that the Skyline in the old and the new system uses the BNL (Block-Nested-Loops) algorithm, which will be described in the following. The BNL algorithm is based on the Basic-Nested-Loops algorithm. It calculates the list of Skyline by performing a pairwise comparison of the tuples and keeping the incomparable tuples in the main memory window. At each iteration, a tuple p is read from the set of input tuples and compared to the unmatched tuples in the main memory window [94],

> The choice of Top_k was used either as a post-processing step of the Skyline result or directly on the database. The first processing was motivated by the existing relationship between the Skyline and the Top_k requests. which has been discussed in the works [146], [147], and [148]. It was observed that the Top_1 object for any increasingly monotone function belongs to the Skyline set as described and proven in [147]. For more details, we present the proof as shown by the authors
>
> **Observation**
>
> The flowing paragraph represents an extracted proof from [147] which demonstrate this observation.
>
> **Proof**: Assuming a space D defined by n dimensions d_i, with i=1...n. Consider a point q that does not belong to the Skyline, but it is the top-1 for a query defined by an increasingly monotone function f. Then there exists another point p that dominates q, i.e. on each dimension $d(i)\in D$, $p([i])\leq q[i]$; and on at least one dimension $d(j)\in D$, $p[j]<q[j]$, and since f is increasingly monotone this leads to a contradiction, because q is the top_1, i.e.$f(p)>f(q)$ Consequently, the Top_1 object for any increasingly monotone function belongs to the Skyline.

Figure 3.1: The platform in the old CSRSS system



Let R^n and R^p be Euclidean vector spaces referred to as the decision space and the objective space. Let point $X \in R^n$ be a feasible set and let f be a vector-valued objective function [149]. $f : R^n \imath R^p$ composed of p real-valued objective functions $f = (f_1, \ldots, f_p)$, where $f_k =: R^n((\imath))R^p fork = 1 \ldots p$. A multi-objective program (MOP) is given by:

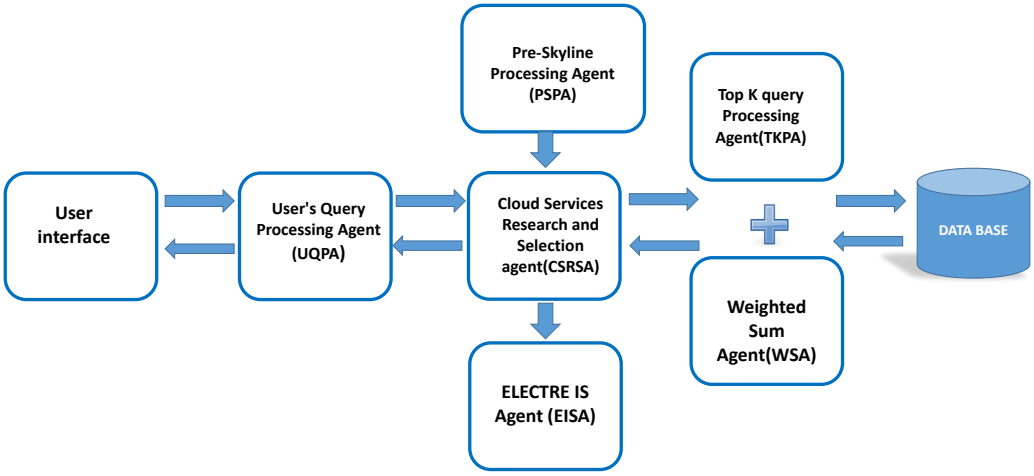
$$\text{Min}\left(\mathrm{f_1(x)}, \ldots, \mathrm{f_p(x)}\right), Subjecttox \in X \tag{3.1}$$

In the weighted sum approach a weighted sum of the objective functions is minimized as chowed in equation 3.2:

$$\min \sum_{k=1}^{p} w_1 f_k(x) \tag{3.2}$$

subject to $x \in X$, Where $w \in R^+$. Theorem 1 [144]
Let $w \in \mathrm{R}^+$. If $\hat{x} \in \mathrm{X}$. is an optimal solution of the problem 3.1 then $\hat{\mathbf{x}} \in \mathrm{X}$.
If $\hat{\mathrm{x}} \in \mathrm{X}$ is a unique optimal solution of problem 3.1 then $\hat{\mathbf{x}} \in \mathbf{X}_{\lambda \mathbf{E}}$ is an optimal solution of problem 3.1 then $\hat{x} \in X_E$.
Let $w \in R^+$ If $\hat{x} \in \mathrm{X}$ is an optimal solution of problem 3.1 then $\hat{x} \in \mathrm{X_{pE}}$.
Let the multi objective program (MOP) convex. A point $\hat{\mathbf{x}} \in \mathrm{X}$ is an optimal solution of problem 3.1 for some $w \in \mathrm{R}^+$ If and only if $\hat{\mathrm{x}} \in \mathrm{X_{pE}}$.

Figure 3.2: The platform in the new e-CSRSS system



Where the weights w_i, $i = 1$. p corresponding to objective functions satisfies the following conditions: $\Sigma_{i=1}^{p} w_i = 1, wi \geq 0, i = 1 \ldots$ p, and f $^{0}_{\text{k}}$(x) is normalized k -th objective function $f_k(x)$, k=1, . . . , p.
For the case of the linear weighted sum, we consider the MOO problem (1) with linear objective functions, having the next form:

$$f_i(x) = \sum (a_{ki} x_1), a_k \in R^+ \tag{3.3}$$

Therefore, normalized objective functions have the following forms:

$$f_k^0(x) = \frac{f_x(x)}{s_x} x_k = \frac{a_{ky}}{s_1} x_1 + \frac{a_{12}}{s_2} x_2 + + \cdots \frac{a_{kn}}{s_n} x_n \tag{3.4}$$

In which case the floating-point values S_k are evaluated in the following way:

$$S_k = \sum_{j=1}^{n} |a_{kj}| \neq 0 \tag{3.5}$$

3.5 Related works

The previous work's main contribution is to design an agent that uses both the Skyline and an outranking method to determine which cloud services meet the best user's requirements. The results show that this method helps select the 3634 most dominant objects from a database of 50 000 cloud services. It was considered that the search and selection of a cloud service is a choice problem based on the methods of outranking in order to better refine the results [3, 95, 141]

However, despite the importance of the impact of this work, we still have the number of a large choice provided by the system, which can disrupt the end-user to make the right decision. To solve this problem, we thought of reducing this number to facilitate the end-user task. Our approach is based on the (MCDA) methods [139], and introduces the use of a Top_k Agent, which will be exposed in the next section and which will ultimately act as a recommendation system. Several studies have been conducted to develop efficient algorithms and define variants for Skyline queries. So querying a set of multidimensional data with Skyline can lead to two cases:

- A significant number of responses is returned, which is generally uninformative,
- An insufficient number of responses is selected.

in both cases this can be confused from the standpoint of the user. Therefore, to remedy these problems, different Approaches have been introduced to refine and reduce the size of the Skyline by selecting him the most interesting points [150–154]. They propose metrics and criteria for identifying the most items satisfactory, selecting the Top_K answers. However, the scheduling criteria regarded May not match the needs of the user. The approach of k-dominance [150] proposes the concept of dominant-k, which determines a set T of k points, such that the sum of the cardinally of the sets of objects dominated by each element T, i.e.

$$\text{Sum} \sum_{\mathbf{p} \in \text{L}} |\text{Dom}(\{\mathbf{p}\})| \tag{3.6}$$

Represented all objects dominated by the object p), is maximized. However, this method does not take into account cases where an object can be dominated by several other items and can return items that are not the items Skyline. In the approach of Top_k representative Skyline points [155], the authors define a set L containing k points of the Skyline, called Top_k representative points of the Skyline, so that the sum of the carnality's of the sets of objects Skyline dominated every point of L is maximized. In other words, the sum

$$\sum_{\text{p} \in \text{L}} |\text{Dom}(\{p\})| \tag{3.7}$$

Where $Dom(\{p\})$ represents all the items dominated the Skyline. Moreover, this approach [91], is NP-complete in the case at least three dimensions. The approach of Top_k frequent points to refine a large Skyline, Chan et al [153] propose a metric called "Skyline frequency", to calculate the Top_k frequent items Skyline. For this, they measure, for each object p, the number of dimensions of sub-spaces where p is a point of the Skyline, and finally choose the k most frequent points. The major disadvantage of this metric is the fact that some of the most interesting points can be classified at the same level as other points much less interesting from the user request. Moreover, in practice, the proposed method is very expensive because it calculates Skylines 2^d - 1 subsets of a d-dimensional space. The approach to k-dominant Skyline points in [151] Chan et al. suggest a criterion, said k-dominance, which calculates the set of points (the Skyline) which

are not dominated by any other end in all of the k-dimensional sub-spaces. The approach is based on the diversity criterion. In [154], the concept of similarity Skyline was proposed, and its implementation is dedicated to case bases. It used to return the subset of maximally similar cases to a given query, using the dominance of Pareto [89, 143]. To reduce the number of results returned by the Skyline, the authors apply the criterion of diversity to select a subset (of size "k") of cases that is as diverse as possible. However, the returned responses can satisfy the less the user request on most of the dimensions (such as those best only on one dimension). Finally, this approach requires calculating the diversity of all subsets of size "k" possible, which is a costly task. The approach is based on the concept of fuzzy Skylines. In [152], the authors introduce the notion of a gradual Skyline. The latter results from a refinement of the original Skyline (i.e., obtained with the dominance relation Pareto), in which the membership of an element refined Skyline is expressed using a degree. Following the study of state of the art, we note that the preferences of consideration in user queries play an important role in the field of relational databases to improve the quality of results. Most approaches have used techniques to improve the performance and relevance of the pairings. However, we note that these methods are still costly to meet user preferences to improve the quality of results. To avoid this problem, we propose in the following section, taking into account user preferences within the database using a post-treatment Top_K algorithm that will refine Skyline's work. Figure 1 shows the pattern of our previous system [95, 141, 142].

3.6 Our contribution using the Top_K algorithm

In our case, we first decided to apply the weighted sum method as the data standardization process and by calculating the mean absolute deviation. So first calculating the average intensity for each dataset, and then calculating the average of the averages. This average was used as the basis for the computation of normalization factors that were consequently applied to each experiment as presented in equation 3.8; the average of all normalized data after that equaled the grand average given in equation 3.9.

$$s_i = \frac{1}{n}\left(\left|a_j - m_j\right| + \left|a_{2j} - m_j\right| + \cdots + \left|an_j - m_j\right|\right) \tag{3.8}$$

With the average:

$$\mathrm{m}_j = \frac{1}{\mathrm{n}}\left(\mathrm{a}_{1\mathrm{j}} + \mathrm{a}_{2\mathrm{j}} + \cdots + \mathrm{a}_{\mathrm{nj}}\right) \tag{3.9}$$

Then used for the calculation of Z scores [91], and, dividing that result by the Sj demonstrated by equation 3.9, according to the formula 3.10.

$$z_{ij} = \frac{a_{ij} - m_j}{s_j} \tag{3.10}$$

Obviously, in many practical problems, the objective functions are represented by various measure units (e.g. if f_1 is measured in byte,f_2 in seconds, etc). For these reasons, objective function normalization is required. It is obvious that now the coefficients have values from the segment [0, 1].

We denote that we now have the linear programming problem presented in equation 3.11 [89]:

$$Q_k(x) = \sum_k \frac{f_k(x)}{S_k} x_k = {}_1 \frac{a_{kt}}{s_1} x_1 + \cdots {}_n \frac{a_{kn}}{S_n} x_n \tag{3.11}$$

Subject to $x \in X$.

Thus, We propose applying the weighted sum method as a powerful core of the simple similarity approach that we will be introduced in this section. The principal objective of this method is to reflect the preference of the end-user properly. For this reason, we have developed in [94, 95], a system that is based on the principle of the Skyline. We then tried to improve our approach by applying outranking methods [156] to the Skyline results. Finally, we use the Top_K query at the option of selecting a service based on its importance. It is about finding the service or services that are the most powerful among the 3643 services and to rank the most relevant to least relevant (Top_K services). We then tried to improve our system by applying the weighted sum method. Figure 3.2 shows the pattern of our new system. As seen previously, the CSRSA uses the Skyline on the set of tuples returned by the PSPA to determine which Cloud services are in the Skyline and meet the user's preferences. Then we used an ELECTRE IS agent that applies the ELECTRE IS method on the Skyline list. To refine the results returned, we adjusted our prototype by adding a Top_K query processing agent TPKA 3.2 to do so, the agent uses the Top_K query algorithm as shown in 1.

It starts by reading an input tuple from a priority queue. Then it compares it to the Skyline and ELECTRE IS output list LS using the function Compare (Ls, item). The WSM agent is used for computing the score function, as we are already explaining in section 3. However, a response to the Top_K queries can be modeled as follows:

We denote a m = which is the n lists of data items such as each data item has a local score in each list, and the lists are sorted according to their local scores. Moreover, each data item has an overall rating, which is calculated based on its Local scores in all lists using a given scoring function. The problem is finding the Top_K items whose overall scores are the highest according to the formulas 3.16. If the condition is verified, this means that criteria are sometimes to maximize and sometimes to minimize. It also presents the score function using the WSM With Starting data:

- No criteria C_1, C_2, C_3 ... C_n.
- Vector weight (w_1 , w_2 , w_n) and w_i> 0.a$_{ij}$ = $u_i(Ai)$, cardinal utility function quotient.

These data represent the performance of each action on each of the criteria. Thus, at the end of all iterations, we keep only the items offering the best compromise in all the requirements

Algorithm 1 $Top_{k_{WS}}$

Ls: input PriorityQueue
$tlArray$: array of items
$Item$: input object for which we calculate its score function
k: the number of services returned by the algorithm
$TopList$: output list of the tuples forming the solution
Define Ls as priority queue based on $ScoreFunction$
function $ComputeTopK$
$returned$ =0
while $returned < k$ **do**
    $Ls \neq \emptyset$ $Ls \in$PriorityQueue result=Compare(Ls, item)
        $result < 0$
        select from Ls the object item with the maximum $ScoreFunction$
        Remove the head of this queue or returns null if this queue is empty
        $Ls.add(item)$
        Update $ScoreFunction(item)$, and update Ls accordingly
        $ScoreFunction(item)$ is completely known
        $Report(item, ScoreFunction(item))$
        $returned = returned+1$

    end while
return $TopList$
    end function

defined by the user. Furthermore, the final Top_K list is computed. In the next section, we present the algorithm's implementation and its performance and some illustration of its implementation. To assess the performance of our approach, we introduced the obtained results in the following section.

3.7 Results and discussion

The platform we used for the experiments is an HP workstation with a 2 GHz processor, 4 GB of main memory, Windows Server 2010 as the operating system, and MS SQL Server 2016 as DBMS. The algorithm is implemented using JAVA. We followed the resulting procedure in the experiment; it was as follows: First, we start with data Transformation:

1. Normalization of all a_{ij} order to maintain the proportionality between values.

2. Standardization of weight (the sum of weights = 1).

3. Implementation of the weighted sum method, and then taking into account that the subject to the decision is choosing a cloud, the user's weight values are given by the user(choice of decision-maker). However, in the experimentation, we give a fixed value to these weights.

Figure 3.3: An example of the 12 best services according to the criteria chosen by the user

TOP 12 Services

1 : service 1164 2525.71 1.97 291.71 93.68 479.1 822.71 293.41 1703.0 0.01 0.58

2 : service 1017 1072.76 4.38 117.86 1473.79 679.4 195.76 128.72 877.0 0.09 3.39

3 : service 2388 1073.31 9.93 0.58 2847.24 16.0 97.31 1.35 976.0 1.31 0.15

4 : service 1053 1073.68 9.78 0.81 3984.53 10.4 98.68 1.71 975.0 1.11 0.1

5 : service 2183 1325.38 2.64 64.22 1996.5 510.6 354.38 72.24 971.0 0.12 1.43

6 : service 1044 1555.39 6.74 64.35 1994.54 788.7 198.39 72.37 1357.0 0.12 3.84

7 : service 63 1076.01 3.64 367.66 180.24 996.0 227.01 386.83 849.0 0.05 4.32

8 : service 98 1076.0 9.88 1.89 3723.57 266.2 98.0 3.27 978.0 0.73 2.47

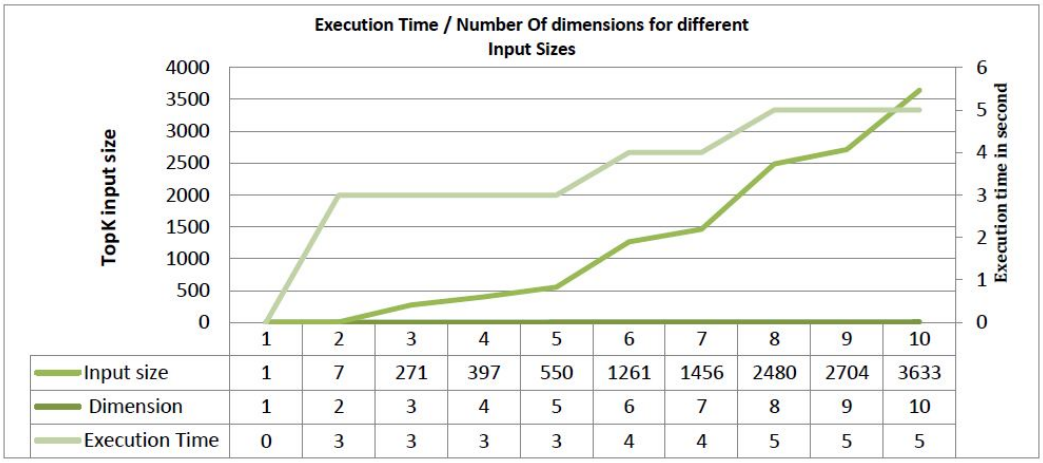
9 : service 2382 1076.29 9.32 0.34 3058.0 19.9 103.29 0.93 973.0 1.71 0.18

10 : service 1992 893.05 6.87 316.79 230.13 428.4 112.05 334.59 781.0 0.06 3.6

11 : service 238 20.8 6.34 64.81 111.3 876.1 2.8 72.86 18.0 0.12 95.91

12 : service 318 24.46 8.83 65.1 428.35 783.6 2.46 73.16 22.0 0.12 69.38

Figure 3.4: Execution Time/Number Of dimensions for different Input Sizes



	1	2	3	4	5	6	7	8	9	10
Input size	1	7	271	397	550	1261	1456	2480	2704	3633
Dimension	1	2	3	4	5	6	7	8	9	10
Execution Time	0	3	3	3	3	4	4	5	5	5

So we include more information in our datasets about maximizing or minimizing the criteria' values based on a binary label. In other words, we assign the label value for 0 or 1, respectively, if the criterion is to minimize or maximize. Service Consider the following 10 criteria:

Table 2. Decide for the choice of this experiment.

For having a coherent weight in WSM, which means normalized while dealing with the maximization and minimization of the given value for the criteria, it is necessary to consider only the criteria to maximize. However, the criteria and Availability, and Response time, for example, are minimized. A transformation of this data is then necessary to obtain the six

Criterion	Description	Min/Max =0/1li	weight
C1	Availability	0	0.1
C2	Bandwidth	1	0.3
C3	Dataloss	0	0.25
C4	Hard Drive	1	0.11
C5	Latency	0	0.12
C6	Ongoing cost	0	0.01
C7	Portability	1	0.02
C8	Acquisition	0	0.03
C9	Acquisition cost	0	0.04
C10	Response time	0	0.02

criteria to maximize. An appropriate transformation is as follows: We seek the maximum positive values back to search the minimum between the inverse of these values while keeping the same weighting. Therefore our function will score the sum of the Max weighted sum. This transformation continues with a single function that calculates a monk to be considered a maximized criterion. The criteria will be recognized C1 ', C3', C5 ', C6', C8 ', C9'.Then applying the method of Weighted Sums and is chosen as considering the example. While performing the following operation:

$$min(f_1(x),...,f_p(x)),x \in X \tag{3.12}$$

The function (3.13) shows the minimization of the weighted sum objective functions:

$$min \sum_{k=1}^{p} {}_k f_k(x), x \in X, and \ \in R^+ \tag{3.13}$$

Where the weights $_i, i \in \{1,...,p\}$ corresponding to objective functions satisfy the following conditions: $\sum_{i=1}^{p} {}_i = 1, {}_i \geqslant 0, i \in \{1,\ldots,p\}$.
In our case we consider the linear weighted sum, consequently, we consider the (MOP) presented in function (3.12) with linear objective functions, having the next form:

$$f_i(x) = \sum_{k=1}^{p} a_{ki} x_{ki}, a_{ki} \in R^+. \tag{3.14}$$

$$R(a_i) = \sum (w_j a_{ij}) \forall i \in [1,n] \tag{3.15}$$

With w_j represents the weights chosen by the user and the a_{ij} represents the values of the criteria to be maximized or minimized. Then performs the following operation: If the criteria is to maximize, the algorithm seeks the maximum at using function 5.3. If not, that is to say, if the criteria are to minimize, in this case, the algorithm considers the inverse of these values while keeping the same weighting, see the function 3.16:

Moreover, the aggregation consists of using the two (WS) functions with a small adaptation of the second function.

$$\begin{cases} max \sum (w_j a_{ij}) & \text{if } a_{ij} \text{ to be maximized} \\ max \sum (w_j 1/a_{ij}) & \text{if } a_{ij} \text{ to be minimized} \end{cases} \tag{3.16}$$

The weighted sum method, therefore, requires matching criteria and incorporates the influence of prior standardization. So the aggregation allows us to answer a bi-objective problem. This can be represented by the intersection of two monotonic and linear functions as follows:

$$f : max \sum (w_j a_{ij}) + max \sum (w_j (1/a_{ij})) \tag{3.17}$$

The monotony of f optimizes the cost of the research by providing guarantees on the evolution of the candidates' scores. The algorithm 1 considers this work in the most common context of using a monotonic ranking function. Based on this aggregate function the $Top_{k_{WS}}$ uses the lists of data items such as each data element to a local score in each list. The lists are sorted according to their local scores called scoreFunction.

Figure 3.3 presents the 12 best services according to the user's criteria: Top_{12}. Once this work is completed, we sort the resulting weighted sums in descending order, thereby realizing a ranking of services. The resulting classification depends on the normalization procedure, and it is essential to note that the decision-maker provides weights according to their personal preferences. Other values yield results entirely differently. We then measured the final solution's execution time and compared them to those obtained with the previous version (using only the Skyline with ELECTRE IS). The new algorithm's use proves its efficiency by determining the cloud services that best meet users' requirements, primarily by reducing the number of resulting cloud services. This reduction is explained by the fact that cloud services were considered difficult to compare because of the output size. When using the Skyline alone or with ELECTR-IS becomes easily comparable when using Top_K with the WSM, since the user weighs preference to the criteria. The result is reduced to the Top_{12} services object when calculating the solution to 10 criteria, and the response time was not increased. For this reason, we think that the developed representations for our system composed of Top_K agent and Skyline ELECTRE IS agent are not perfect, but it provides an excellent solution to have an efficient query evaluation and process and manage the size of Skyline points.
We resume the objective of our system in:
Firstly, the system's role is to return the most relevant answer to the user. Secondly, it aims to manage the size of Skyline points in the high dimensional. Finally and the third one to bridge between Skyline and Top_K query, while the bridge means here that the user can determine the number of results to retrieve in the Skyline query, which is coming from Top_K query specification: We need to use the advantage of the Top_k query and apply it on the Skyline query to have a

bridge between them. The bridge means allowing the user in the Skyline query to determine the number of Top_K most relevant answers to be returned. This behavior makes the Skyline query a weak concept and produces many incomparable result objects with increasing dimensions. The agents are applied in the real-life scenario when the user can specify the output size.

3.8 Conclusion

To better meet the user's needs without disturbing him by many service choices, we proposed improving the result returned by the Skyline algorithm and ELECTRE IS. The use of these two algorithms allows the search and the selection of Cloud services. Our approach is to apply the Top_K algorithm based on WSM on all tuples returned by the Skyline and ELECTRE IS. Experimental results show that our method can reduce the final solution's size, which the decision-maker chooses, without increasing the system response time. We can conclude that our approach gives promising results.

CHAPTER 4

OUR APPROXIMATION BASED ON $Top_{k_{WS}}$ AGAINST OTHER TYPES OF QUERIES PROCESSING

This chapter contains two main contributions: According to state of the art, we have chosen well-known $Topk$ algorithms that use matching algorithms behavior in our study. Thus, the first contribution in section 4.1 concerns a comparative investigation between our approach based on the adapted score function of the $Topkws$ algorithm and the Fagin Approach using Cloud Computing QoS dataset. Then, the second contribution in section 4.2 consists in comparison with other types of $Topk$ algorithms. This time, the algorithms with which the comparison was made use a different kind of access such as the sorted and the random access, controlled access and random access, and no-random access. Finally, the entire comparison took into account the algorithms' execution time, the degree of correlation between each dataset's studied parameters, and was also based on a statistical evaluation using the p-value.

4.1 Contribution 2: Our enhanced approach compared to Fagin Algorithm using Cloud Computing QoS dataset

4.1.1 Introduction to contribution 2

According to International Data Corporation (IDC), worldwide spending in the public's Cloud Computing will increase to $162B in 2020, attaining a 19% compound annual growth rate (CAGR) [140]. This tremendous growth is intended to meet the growing need for user support. However, Cloud Computing can improve business agility and productivity while increasing its efficiency and reducing its costs. So professionals are not satisfied with the way how to store Big data using Internet technologies. Still, they are trying to analyze this data in a subject to real business

needs. However, due to a large amount of digital data available, it is crucial to extract the data corresponding to the most recommended need for information; an application must express the end-users need [50, 138, 157]. Therefore, it is essential to classify the data elements according to their importance [158], and only the best data items are recovered. For these reasons, the Top_k algorithms have become increasingly used, especially in this context of recommendation, which requires interaction with the user. The approach represented in this contribution is carried out to refine the result found by the Skyline approaches, which were the main subject of the work exposed in [50, 95]. The setting up of this optimization is then based on the Cloud Service user's preferences interpretation by using multi criteria-analysis methods, and in particular, the Weighted Sum (WS). The injection of Multi-Criteria Decision Aiding (MCDA) in the heart of Top_k algorithm has given birth to the algorithm called "Top_k weighted sum", or $Top_{k_{WS}}$ [50].

In the beginning, we designed an agent that uses both the Skyline and outranking formalism to determine which Cloud Services better meet the user's requirements. The results show the effectiveness of this method for selecting the most dominant objects from a service's database [50, 94]. However, the defined Skyline returns a set of points that are not dominated by other issues in the datasets [141]. So, the large number of choices provided by the system makes the decision difficult for the end-user.

To solve this problem, we proposed a refinement of the result by optimizing the processing and using the $Top_{k_{WS}}$, especially about the monotony of the aggregation function chosen for the score calculation. So, the purpose of the hybridization with the "weighted sum" method is the filtering of a Cloud Services database whose elements are selected by the Skyline operator.

Finally, to evaluate its performance, we identified the Top_k algorithms most similar to our case from the state-of-the-art. We adopted Fagin's algorithm [91], as it is the best known in the context of Top_k based on a monotonic aggregation function considers databases have sorted elements.

4.1.2 The basic Fagin's algorithm

One of the main works in the context of Top_k algorithms is Fagin's algorithm [101] see algorithm 2. It is correct for monotonic aggregation functions. The cost of the middle-ware of the algorithm FA is $O(N(m-1)/mk1/m)$ if there are N objects in the database and if the orders in the sorted lists are probabilistic and independent [88]. The steps of the FA algorithm are shown below:

Algorithm 2 Fagin's Algorithm FA [101]

1: do sorted access in parallel to each of the m sorted lists L_i.
2: stop when there are at least k objects, each of which have been seen in all the lists. $R \in (L_i)$ that has been seen
3: retrieve all of its fields $x_1,...,x_m$ by random access.
4: compute $F(R) = F(x_1,...,x_m)$.
5: **return** the $topk$ data items

Hereafter, the details of the execution steps:
Step 1 (lines 1 and 2), paralleled, and sorted access to all the lists must be performed. Then all the data elements that have been seen are stored in a set R. To keep the traceability of data seen in all the lists, for each data element in the set R, it is also necessary to mark the lists in which they appear. If there are at least k data items which appeared in all the lists, sorted access can be stopped. In step 2: (lines 3, 4, and 5), for each data item which is in L, Fagin's Algorithm computes the score according to the aggregation function, since all the items are Top_k candidates. In order to do so, random access is performed for all the data items; consequently, the algorithm can calculate their score. Next, the Top_k data Elements are stored in a set of results R. They are taking into account replacing data items with a lower score with data items with a higher score. And finally, in step 3: (line 6), the set R contains the $topk$ data items, the algorithm returns the result set R.

4.1.3 Results and Discussion

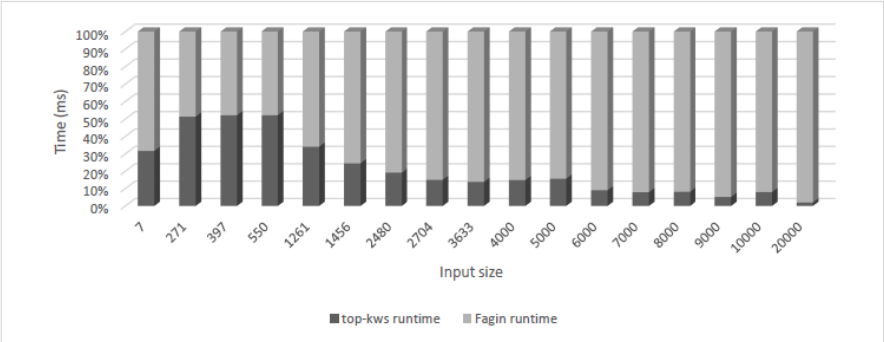
This section presents the results of our study experiments based on runtime evaluation and correlation metrics.

4.1.3.1 Performance evaluation using runtime comparison

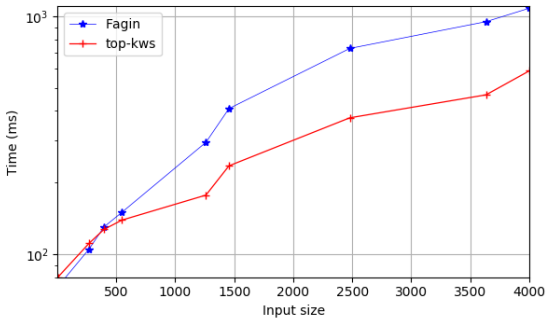
We use a Core i5 (2.70 GHz) PC with 8 GB of memory in the experiment. The $Top_{k_{WS}}$ and Fagin's algorithm are implemented in the JAVA environment. And the execution number is 10. Then the result is the average of the 10.
In the beginning, the system starts with the normalization of all the a_{ij} to maintain the proportionality between the values. And subsequently, get a ranking of cloud services in descending order. The use of $Top_{k_{WS}}$ algorithm proves its effectiveness in determining the Cloud Services that best meet users' needs and, in particular, by reducing the number of resulting Cloud Services. Besides, thanks to the weighted sum method, the $Top_{k_{WS}}$ gives, in a precise sense, the top k most recommended answers to the user. To ensure the $Top_{k_{WS}}$ performance, we study two important impacts. The above comparisons are significant:

1. Impact of input size (Skyline output):

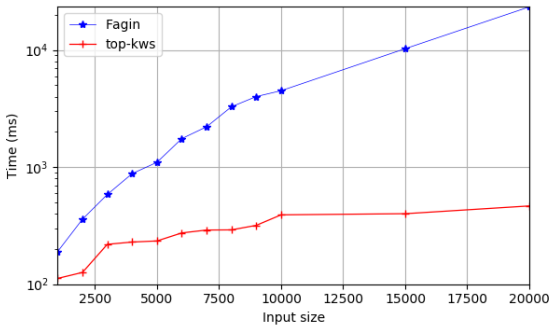
   For Fagin's algorithm, we distinguish between two types of access: The sorting access where the CSRSS obtains the quality of an object in a sorted list by sequentially proceeding from the list, and the random access where the CSRSS requests the quality of the item in a List and obtains it in a single step. In this part of the experiment, we evaluate the performance of $Top_{k_{WS}}$ with varying inputs size. The weights w are the same for all the criteria, the dimensions are either at 10 for Figures 4.1(a) and 4.1(b) or at 12 for Figure 4.1(c), and the number of k is 5.



(a) Runtime percentage comparison of $Top_{k_{WS}}$ and Fagin's algorithms according to different inputs size.



(b) Impact of inputs size according to 10 dimensions in small datasets.



(c) Impact of inputs size according to 12 dimensions in large datasets.

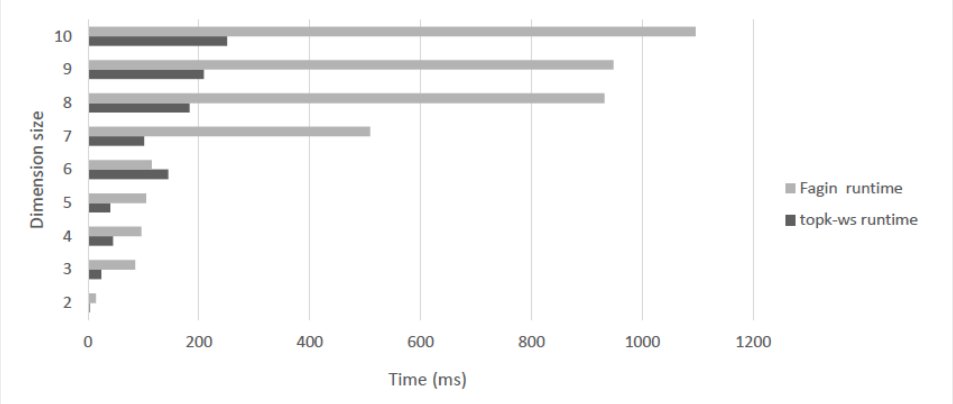
Figure 4.1: Impact of input size variation on the $Top_{k_{WS}}$ and Fagin's algorithm runtime.

As shown in Figure 4.1(a), Fagin's algorithm performs better than $Top_{k_{WS}}$ according to smalls inputs. However, with the growth of the database's size, the $Top_{k_{WS}}$ algorithm starts to be faster than Fagin's algorithm. It is found that the execution time percentage varies between 60% for 1261 inputs and more than 90% for 20000 in Figure 4.1(a). This remark remains valid in different dimensions. Figure 4.1(b) and Figure 4.1(c) show the variation of the inputs between a Dataset of 7 items and a Dataset of 4000 items, as shown in Figure 4.1(b), and between 1000 items and 20000 items, as shown in Figure 4.1(c). Both graphs represent respectively 10 and 12 dimensions. Unlike FA, which requires large buffers (the size of which can grow without limit as the size of the database grows [88]), $Top_{k_{WS}}$ requires only a small buffer, and the use of the PriorityQueue fixed size gives the better performance. Because it is based on a priority heap, and the elements are ordered in their natural order, allowing the PriorityQueue to have the time O(log(n)).

2. Impact of dimensional variation:

For this part of the experiment, we evaluate $Top_{k_{WS}}$ with varying dimensions. The weights values w are fixed for each criterion, the used dataset size is 3633 items, and the number of k is 5. Figure 4.2 shows the relevance of the $Top_{k_{WS}}$ algorithm, especially concerning its execution time, by varying the system dimensions, which presents here the different QoS criteria chosen by the end-user of the Cloud Service System. Also, Figure 4.2, depicts that the runtime increases according to the number of dimensions. Therefore, it was illustrated that runtime with seven dimensions is lower than that with the sixth dimension, according to $Top_{k_{WS}}$. On another side, from the sixth dimension, according to Fagin's algorithm, the runtime based on the same Skyline output has rapidly increased. This change incorporates the main drawback of Skyline outputs. In insets of high dimensional data, the number of Skyline points becomes large as the number of dimensions increases. In fact, for each Cloud Service s, it is more likely that there is another Cloud Service s such that s is better than s' or s' is better than s on the different subsets of dimensions. Suppose the

Figure 4.2: $Top_{k_{WS}}$ and Fagin' algorithm execution time according to the different dimensions



end-user of the CSRSS cared not just about the six dimensions, namely the PrixCS, RAM, hardDisk, bandwidth, latency, and response time, but also about the portability, risk data loss, and availability. In that case, most of the Cloud Services in the database may need to be included in the Skyline answer set. For each Cloud Service, there may be no Cloud Service dominated by all the criteria, even if it is dominated by many. A small number of outliers are expected (which is not due to an abnormal condition), mostly if the data are typically distributed [159]. Besides, many scientists' elimination of outliers is a practice poorly accepted until there are no more mathematical criteria to provide an objective method of rejecting values. Therefore, it is not easy to determine which dimensions should be retained. However, even though reducing Skyline's dimensions is a desirable solution [153], but using an efficient algorithm to handle the dimensionality problem proves that it is more appropriate in such a situation. Eventually, using $Top_{k_{WS}}$ instead of reducing the number of dimensions presents a collaborative filtering solution and approves the

combined approach's performance. After all, the $Top_{k_{WS}}$ outperforms Fagin's algorithm in dimensional variation according to runtime metric and by expressing the actual need of the end-user. To ensure this effectiveness, the following study builds as much as possible on essential correlation metrics.

4.1.3.2 Performance evaluation using correlation study

In this part, we analyze the performance instability when a request is executed in the Cloud Computing system. We investigate a correlation between some Cloud environment parameters—these parameters considered for us as being criteria in a bi-objective problem, which calculate the final scores. So the comparison consists of using the $Top_{k_{WS}}$ algorithm or Fagin's algorithm by choosing Cloud Services parameters according to the final user's requirement, such as the bandwidth level, Hard Disk, or the velocity, etc. Also, we examine we examine the various significance of the correlation parameter using statistical techniques. This analysis could lead to efficient or inefficient request execution. We consider that the query differs from one user to another since our system involves the user choice. So we consider the following scenario to carry out a test having appropriate conditions of comparison between the samples. For this test the user creates a top k query with $k = 5$, the number of Skyline dimensions is 10, and the user's choice represented by the following weights: $w_{ij} = \{0.1, 0.3, 0.25, 0.11, 0.12, 0.01, 0.02, 0.03, 0.04, 0.02\}$. To do so, we are focusing on the calculated correlation matrix through the use of our data Cloud Service for comparing the results of $Top_{k_{WS}}$ algorithm with those found using Fagin's algorithm. This calculation is based on the correlation coefficients Pearson, Spearman, and Kendall. It is important to note that if the data does not follow the normal distribution, in this case, it is recommended to use the non-parametric correlation, including the Spearman and Kendall tests. So, before performing the Pearson test, we ensured normality between the variables of the samples studied. The test was based on the Shapiro and Student test [160] to prove the normality between all the variables. The results given by this test proved the normality. It was, therefore, possible to apply the Pearson test. Tables from 4.1 to 4.6 represent the found results.

Table 4.1: Correlation matrix of Pearson r between the Cloud Services parameters for a query of top-5 Cloud Services using $Top_{k_{WS}}$.

	PrixCS	RAM	hardDisk	bandwidth	latency	ResponseTime	portability	risk	dataloss	availability
PrixCS	1.00	-0.86	0.85	-0.40	0.67	0.86	0.85	0.87	-0.21	-0.57
RAM	-0.86	1.00	-0.96	0.70	-0.72	**-0.96**	-0.96	-0.97	0.58	0.67
hardDisk	**0.85**	**-0.96**	1.00	-0.74	0.87	1.00	1.00	1.00	-0.69	-0.43
bandwidth	-0.40	0.70	-0.74	1.00	-0.53	-0.72	-0.74	-0.71	0.89	0.31
latency	0.67	-0.72	0.87	-0.53	1.00	0.87	0.88	0.85	-0.68	0.02
ResponseTime	0.86	-0.96	**1.00**	-0.72	0.87	1.00	**1.00**	**1.00**	-0.67	-0.45
portability	0.85	-0.96	**1.00**	-0.74	0.88	1.00	1.00	**1.00**	-0.69	-0.43
risk	0.87	**-0.97**	**1.00**	-0.71	0.85	1.00	1.00	1.00	-0.48	
dataloss	-0.21	0.58	-0.69	0.89	-0.68	-0.67	-0.69	-0.65	1.00	0.01
availability	-0.57	0.67	-0.43	0.31	0.02	-0.45	-0.43	-0.48	0.01	1.00

From Tables 4.1 and 4.2, it can be seen that:

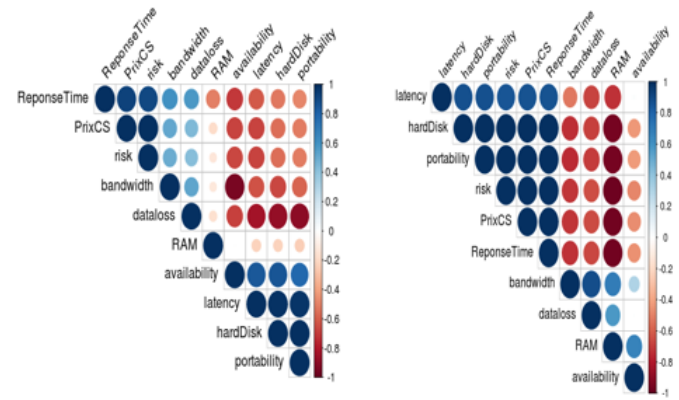
The Pearson correlation coefficient takes values from -1 to +1. If the value is equal to +1, the variables are perfectly linear and related to an increasing relationship. But if the value is equal to -1, the variables are perfectly linear, and they are related to a decreasing relationship. The value 0 shows that the variables are not linearly related to each other.
Concerning the $Top_{k_{WS}}$ algorithm, the values are positively correlated in general by comparing with those of Fagin's algorithm. It can be seen that all the parameters are very high coefficients:

Table 4.2: Corellation matrix of Pearson *r* between the Cloud Services parameters for a query of top-5 Cloud services using Fagin.

	PrixCS	RAM	hardDisk	bandwidth	latency	ReponseTime	portability	risk	dataloss	availability
PrixCS	1.00	-0.23	-0.89	0.91	-0.92	0.98	-0.88	1.00	0.89	-0.94
RAM	-0.23	1.00	**-0.07**	-0.15	-0.05	**-0.38**	-0.09	-0.21	-0.10	0.07
hardDisk	**-0.89**	**-0.07**	1.00	-0.94	1.00	-0.86	1.00	-0.89	-0.98	0.98
bandwidth	0.91	-0.15	-0.94	1.00	-0.94	0.92	-0.92	0.91	0.93	-0.99
latency	-0.92	-0.05	1.00	-0.94	1.00	-0.89	1.00	-0.92	-0.97	0.98
ReponseTime	0.98	-0.38	**-0.86**	0.92	-0.89	1.00	**-0.85**	**0.98**	0.89	-0.93
portability	-0.88	-0.09	**1.00**	-0.92	1.00	-0.85	1.00	**-0.88**	-0.98	0.96
risk	1.00	**-0.21**	**-0.89**	0.91	-0.92	0.98	-0.88	1.00	0.89	-0.94
dataloss	0.89	-0.10	-0.98	0.93	-0.97	0.89	-0.98	0.89	1.00	-0.96
availability	-0.94	0.07	0.98	-0.99	0.98	-0.93	0.96	-0.94	-0.96	1.00

Therefore, there is a strong redundancy between the Cloud Service parameters, which leads to a phenomenon called multicollinearity [131]. For example, if we take the values of the (Ram, hardDisk) and (Ram, Risk), their values are respectively (**-0.96** and **-0.97**) for $Top_{k_{WS}}$ and (**-0.07** and **-0.21**) for Fagin's algorithm, therefore, there is a strong redundancy between the Cloud Service parameters, which leads to a phenomenon called multicollinearity [131]. We can see that some values are closer as the case of (PrixCx and harddisk), with $\simeq$ **0,85**; we also notice that some values have a perfect correlation between the parameters according to the query of $Top_{k_{WS}}$. For example, in the columns of hardDisk and Risk parameters, the correlation reaches these

Figure 4.3: The correlogram of Pearson metric according to $Top_{k_{WS}}$ and Fagin's algorithm



The Correlogram between QoS parameters of top 5 Cloud service according to Fagin' algorithm

The Correlogram between QoS parameters of top 5 Cloud service according to topk-ws algorithm

maximum values concerning RespenseTim and Portability, and it is equal to **1**. As against Fagin's

algorithm, these best values vary between **-0.85** and **-0.98**. Figure 4.3 of correlogram proves this performance.

Table 4.3: Correlation matrix of Spearman rho between the Cloud Services parameters for a query of top-5 Cloud services using $Top_{k_{WS}}$.

	PrixCS	RAM	hardDisk	bandwidth	latency	ReponseTime	portability	risk	dataloss	availability
PrixCS	1.00	**-0.90**	0.10	-0.30	0.10	0.90	0.10	0.90	-0.10	-0.70
RAM	-0.90	1.00	-0.30	0.60	0.00	-1.00	-0.30	-1.00	0.30	0.90
hardDisk	0.10	-0.30	1.00	-0.90	0.70	0.30	1.00	0.30	-1.00	-0.10
bandwidth	-0.30	0.60	-0.90	1.00	-0.40	-0.60	-0.90	-0.60	0.90	0.50
latency	0.10	0.00	0.70	-0.40	1.00	0.00	0.70	0.00	-0.70	0.40
ReponseTime	0.90	-1.00	0.30	-0.60	0.00	1.00	0.30	1.00	-0.30	-0.90
portability	0.10	-0.30	1.00	-0.90	0.70	0.30	1.00	0.30	-1.00	-0.10
risk	0.90	**-1.00**	0.30	-0.60	0.00	1.00	0.30	1.00	-0.30	-0.90
dataloss	-0.10	0.30	-1.00	0.90	-0.70	-0.30	-1.00	-0.30	1.00	0.10
availability	-0.70	**0.90**	-0.10	0.50	0.40	-0.90	-0.10	-0.90	0.10	1.00

From Table 4.3 and 4.4, It can be seen that:
For the $Top_{k_{WS}}$ algorithm, only 32% of the values have a higher correlation coefficient than FA. Despite that, in some distinct cases, such as the RAM parameter, we find that the $Top_{k_{WS}}$ algorithm has very remarkable coefficients compared to FA. Their values are respectively: (**0.90**, **-1**, **-1**) for $Top_{k_{WS}}$, and (**-0.07**, **-0,28**, **0,04**) for FA. Therefore, since the Spearman's coefficient

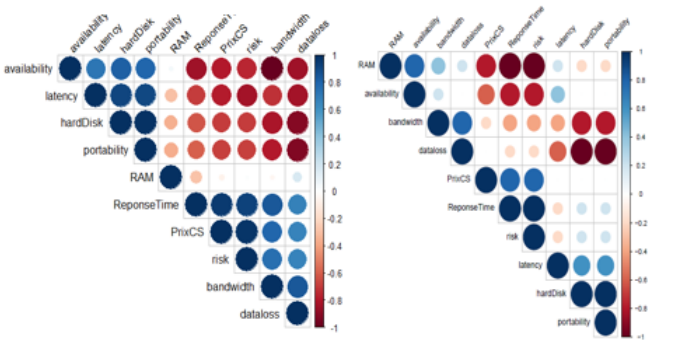
Table 4.4: Correlation matrix Spearman ρ between the Cloud Services parameters for a query of top-5 Cloud services using Fagin.

	PrixCS	RAM	hardDisk	bandwidth	latency	ReponseTime	portability	risk	dataloss	availability
PrixCS	1.00	**-0.07**	-0.70	0.79	-0.78	0.95	-0.68	0.98	0.66	-0.79
RAM	-0.07	1.00	-0.36	-0.04	-0.30	-0.28	-0.37	0.02	0.16	0.04
hardDisk	-0.70	-0.36	1.00	-0.82	0.92	-0.62	0.99	-0.70	-0.92	0.82
bandwidth	0.79	-0.04	-0.82	1.00	-0.72	0.84	-0.79	0.76	0.84	-1.00
latency	-0.78	-0.30	0.92	-0.72	1.00	-0.70	0.90	-0.83	-0.83	0.72
ReponseTime	0.95	**-0.28**	-0.62	0.84	-0.70	1.00	-0.60	0.93	0.67	-0.84
portability	-0.68	-0.37	0.99	-0.79	0.90	-0.60	1.00	-0.68	-0.93	0.79
risk	0.98	0.02	-0.70	0.76	-0.83	0.93	-0.68	1.00	0.66	-0.76
dataloss	0.66	0.16	-0.92	0.84	-0.83	0.67	-0.93	0.66	1.00	-0.84
availability	-0.79	**0.04**	0.82	-1.00	0.72	-0.84	0.79	-0.76	-0.84	1.00

is used for assessing how well an arbitrary monotonic function could describe the relationship between two Cloud Service's parameters. We then can conclude that according to our approach, there's a less monotonous relationship between the Cloud QoS's parameters than that of FA, as illustrated by the figure 4.4.
Based on Table 4.5 and 4.6, It can be seen that:
For the Kendall metric, our approach outperforms FA by 46%, which represents approximately half of the correlation coefficient values.

However, Kendall's tau τ correlation coefficient is considered equivalent to the Spearman correlation coefficient. While the Spearman rank correlation coefficient is like the Pearson correlation coefficient computed from ranks purpose, Kendall tau τ correlation rather Represents a probability of observing the nice concordant and discordant pairs. The correlogram shown in

Figure 4.4: The correlogram of Spearman metric according to $Top_{k_{WS}}$ and Fagin's algorithm



The Correlogram between QoS parameters of top 5 Cloud service according to Fagin' algorithm

The Correlogram between QoS parameters of top 5 Cloud service according to topk-ws algorithm

Table 4.5: Correlation matrix of kendall τ between the Cloud Services parameters for a query of top-5 Cloud services using $Top_{k_{WS}}$

	PrixCS	RAM	hardDisk	bandwidth	latency	ReponseTime	portability	risk	dataloss	availability
PrixCS	1.00	-0.80	0.00	-0.20	0.00	0.80	0.00	0.80	0.00	-0.60
RAM	-0.80	1.00	-0.20	0.40	0.20	-1.00	-0.20	-1.00	0.20	0.80
hardDisk	0.00	-0.20	1.00	-0.80	0.60	0.20	1.00	0.20	-1.00	0.00
bandwidth	-0.20	0.40	-0.80	1.00	-0.40	-0.40	-0.80	-0.40	0.80	0.20
latency	0.00	0.20	0.60	-0.40	1.00	-0.20	0.60	-0.20	-0.60	0.40
ReponseTime	0.80	-1.00	0.20	-0.40	-0.20	1.00	0.20	1.00	-0.20	-0.80
portability	0.00	-0.20	1.00	-0.80	0.60	0.20	1.00	0.20	-1.00	0.00
risk	0.80	-1.00	0.20	-0.40	-0.20	1.00	0.20	1.00	-0.20	-0.80
dataloss	0.00	0.20	-1.00	0.80	-0.60	-0.20	-1.00	-0.20	1.00	0.00
availability	-0.60	0.80	0.00	0.20	0.40	-0.80	0.00	-0.80	0.00	1.00

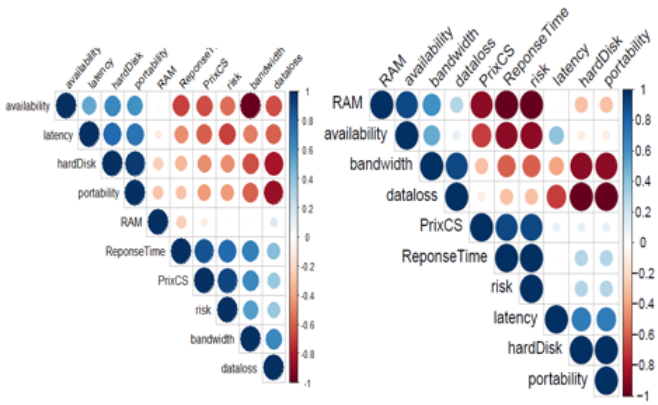
Table 4.6: Correlation matrix of Kendall τ between the Cloud Services parameters for a query of top-5 Cloud services using Fagin.

	PrixCS	RAM	hardDisk	bandwidth	latency	ReponseTime	portability	risk	dataloss	availability
PrixCS	1.00	-0.11	-0.47	0.64	-0.60	0.87	-0.42	0.91	0.38	-0.64
RAM	-0.11	1.00	-0.24	-0.02	-0.11	-0.24	-0.29	-0.02	0.16	0.02
hardDisk	-0.47	-0.24	1.00	-0.64	0.78	-0.33	0.96	-0.47	-0.82	0.64
bandwidth	0.64	-0.02	-0.64	1.00	-0.51	0.69	-0.60	0.56	0.64	-1.00
latency	-0.60	-0.11	0.78	-0.51	1.00	-0.47	0.73	-0.69	-0.60	0.51
ReponseTime	0.87	-0.24	-0.33	0.69	-0.47	1.00	-0.29	0.78	0.42	-0.69
portability	-0.42	-0.29	0.96	-0.60	0.73	-0.29	1.00	-0.42	-0.87	0.60
risk	0.91	-0.02	-0.47	0.56	-0.69	0.78	-0.42	1.00	0.38	-0.56
dataloss	0.38	0.16	-0.82	0.64	-0.60	0.42	-0.87	0.38	1.00	-0.64
availability	-0.64	0.02	0.64	-1.00	0.51	-0.69	0.60	-0.56	-0.64	1.00

Figure 4.5 illustrates this comparison. Besides, there's an equivalent between concordance and discordance relationship between the Cloud QoS's parameters for both $Top_{k_{WS}}$ and FA algorithm. If the concordance between the two rankings is perfect and the two rankings are the same, the coefficient has value 1. If the discordance between the two rankings is perfect, and one ranking is the reverse, the coefficient has value -1. For all other arrangements, the value lies between -1 and 1, and increasing amounts imply increasing concordance between the rankings. If the rankings

are independent, the coefficient has value 0. Thus, Kendall's coefficient is used to assess and

Figure 4.5: The correlogram of kendall metric according to $Top_{k_{WS}}$ and Fagin's algorithm



The Correlogram between QoS parameters of top 5 Cloud service according to Fagin' algorithm

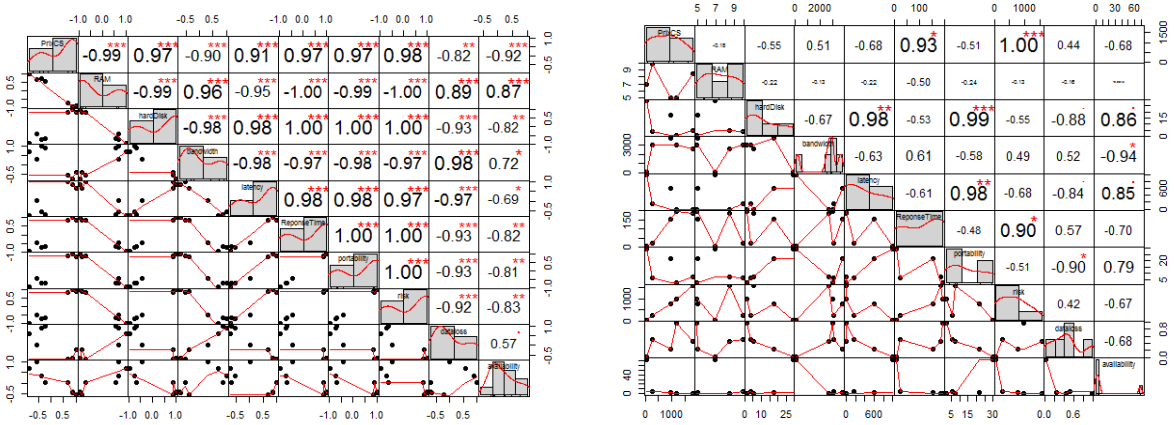
The Correlogram between QoS parameters of top 5 Cloud service according to topk-ws algorithm

test correlations between non-interval scaled ordinal Cloud Services parameters. We then can conclude that our approach result is very close to Fagin's algorithm result, which is proved by Figure 4.5.

4.1.3.3 Discussion of the Results according to the significance of the correlation level

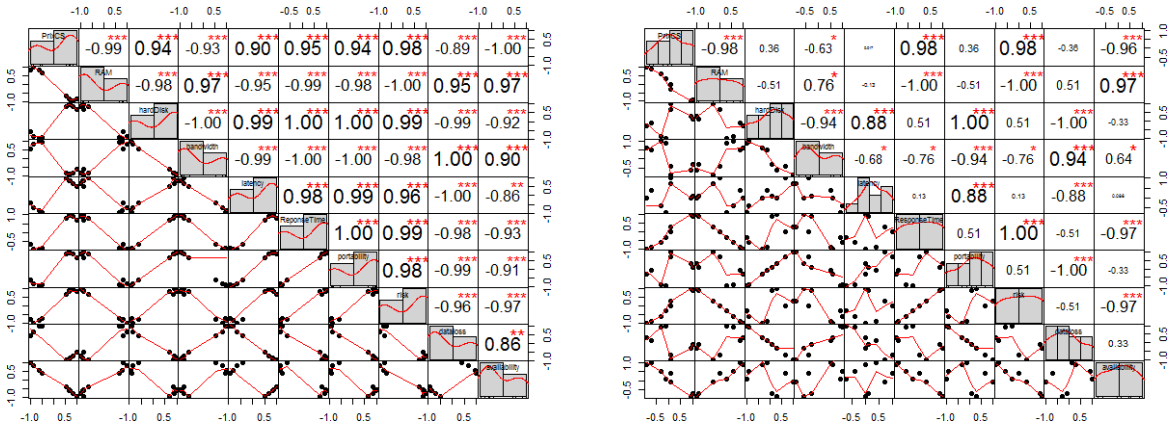
To better analyze these correlations, we tried to use the p-value [161]. This statistical parameter shows the level of part of contribution is marked by " $*$ ", a common practice in many areas to report p-value with the star attached to indicate P <0.05 and two stars to indicate P <0.01. Three stars were used to indicate P <0.001. Whether for the coefficients: r of Pearson, τ of Kendall, or ρ of Spearman. The test is then applied to the correlation results obtained by our approach compared to the result obtained by Fagin's algorithm. This test concerns all pairs of possible variables in an example request of Top_k with k=5 from the Database of Cloud Service system. Figure 4.6(a) and 4.6(b) show the test.

- For Pearson coefficient significance:
  We assume the following hypothesis is represented by the null hypothesis H_0 vs. the alternative view H_1.
  H_0: r = 0 (there is no correlation between the Cloud QoS parameters).
  H_1: r < > 0 (Cloud QoS parameters are correlated). For 5% of the significance level, p-value < 0.05 means that there is evidence to reject the null hypothesis in favor of the alternative hypothesis. In other words, there is a statistically significant linear relationship between the Cloud Service QoS parameters. The histograms of the variables are shown on the diagonal. The asterisks indicate the significance levels of the correlations as shown in figures 4.6(a)



(a) $Top_{k_{WS}}$ significance graph. (b) Fagin significance graph.

Figure 4.6: The significance graph of Pearson coefficient.

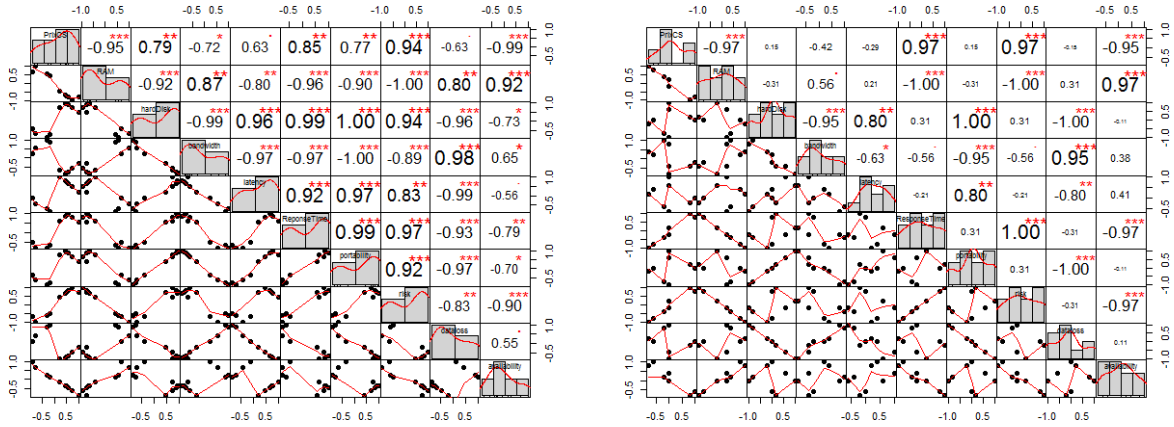


(a) $Top_{k_{WS}}$ significance graph. (b) Fagin significance graph.

Figure 4.7: The significance graph of Spearman coefficient.

and 4.6(a). Pearson test application shows that the relationship can be modeled in a rather linear way. Our approach shows more linearity using p-value, comparing it to Fagin's algorithm, which approves the correlation test in the study of the previous subsection.

- For Spearman coefficient significance presented in figures 4.7(a) and 4.7(b): We compare the observed ρ with published values for different levels of significance (e.g., 0.05, 0.01...). The proposed solution is to know the importance of a specific range or less than an absolute value. This method is tested by applying the Student t-test at a significance level of 5%. It



(a) $Top_{k_{WS}}$ significance graph.

(b) Fagin significance graph.

Figure 4.8: The significance graph of Kendall coefficient.

can be seen that our approach performs Fagin's algorithm and shows more monotonicity on the graphs. Wich was not clear by using only the correlation coefficient of Spearman.

- Finally, for Kendall coefficient significance in figures 4.8(a) and 4.8(b): The null hypothesis vs. the alternative view for Kendall τ correlation coefficients is:
  H_0: $\tau = 0$ (there is no correlation between the two QoS parameters). H_1: $\tau <> 0$ (the two parameters are correlated or are concordant). So, from the graphs: It can be seen that we are dealing with a discordant association between variables in the cases of tests carried out at Fagin's algorithm contrary to our approach, which shows more concordance on the graphs.

4.1.4 Conclusion

In this contribution, we discussed the different types of Top_k algorithms before presenting the evolution of our approach based on the $Top_{k_{WS}}$ algorithm for refining the Skyline. We show how the algorithm works, not only to mention the significant improvement in the runtime given by our algorithm compared to Fagin's algorithm, but we also tried to compromise between a good runtime and a good quality of results found. This trade-off is achieved by exceeding Fagin's runtime algorithm by applying it to different dimensions and the dataset's different sizes. Even more, the quality of our approach's result, in some cases, is considered attractive. In other cases, it remains comparable according to the three best correlation metrics like can be seen in Table 4.7 taking into account the following notations the Correlation coefficient of Pearson is written by CCP, and the Correlation coefficient of Spearman is given by CCS, while CCK marks the Correlation coefficient of Kendall. and the comparison was taking into account tree p-value test of correlation significance. Finally, we prove that this approach found a balance

	CCP	CCS	CCK
$Top_{k_{WS}}$	72%	36%	46%
FA	28%	64%	54%

Table 4.7: Comparison between $Top_{k_{WS}}$ and FA based on the three correlation metrics.

between the recommendation quality and the runtime measurements. Thanks to using the WSM, which improved the performance of the Top_k query. In our next contributions, we try to adopt the $Top_{k_{WS}}$ algorithm to show that it can perform case match queries for sorted lists with the threshold algorithm. We will also present the parallelization pf this algorithm by adopting it to a distributed environment based on Hadoop's Map-Reduce Framework using Big Data technologies. In another work, we are also trying to project this problem in the Mobile Ad-hoc Networks (MANET) based on the results presented in [162, 163].

4.2 Contribution 3: Our approach compared to NRA and TA in other application domains using the GRSS

Recommendation Systems (RS) based on collaborative filtering is the most commonly used Information technology in the last decade. RS's processing behavior can be found on different approaches, such as Decision Making (DM) support and preferences query processing. These systems have been widely used in many Internet activities, mainly to overcome information overload and for many other purposes. Some of these include e-commerce sites, web page searching, e-learning, and Cloud Computing Services. Also, research has been conducted on the use of RS in some sport management Systems. This contribution presents a performance evaluation of the $Topk_{ws}$ recommendation algorithm, applying on a new RS called Generic Research and Selection System (GRSS) based on the Skyline and the Top_k query processing. The conceived algorithm, which used an adapted Multi-criteria Decision Aiding (MCDA) method, was applied to different research eras in the discussed contribution, namely the Cloud Computing Services and Sport management System. The algorithm shows to be more and more efficient. Extensive experiments based on correlation study toward both real and synthetic datasets demonstrate the efficiency and scalability of this algorithm compared with other best-known algorithms in this field.

4.2.1 Introduction to contribution 3

In the last decade, several works proposed their RSs for Top_k query processing. An important issue in such systems is faced by the Big Data size and dimensionality, with a sufficient level of quality. The principal purpose of RS is to help each user identify the ideal results of a manageable size. As such an example, let consider an RS based on the most required videos of each user during the football World Cup event [1],[2]. The given statistics show that some users needed to follow

the match between the finalist, while others worried about seeing a movie or a cartoon video, accordingly. They depend on the user's interest and feedback. The users find their requirements acting on their systems. Therefore, a subset of the most relevant answers instead of all solutions is given using the recommendation query. Not far from this example, and to provide an accurate ranking that is explicitly applied to the impact of the 2018 FIFA World Cup matches [164, 165]. RS's Man-of-the-match prediction is also a fascinating and seductive research problem, not least because of its complications, the effort it requires, and unexpected results. Hence, a football match depends on various factors, characters, and irregular situations. As a result, it is excruciating and hard to predict football matches' correct and accurate results. Such research requires a multi-criteria decision-making approach to predict a ranking that is explicitly applied to the 2018 FIFA World Cup football matches' outcomes. The GRSS has proven that correlations results between the different criteria are significantly better than those of the other algorithms. However, from state of the art, the Top_k [50] and the Skyline [95] are considered the most popular used questions in information retrieval and query preference field. We present in this contribution an extensive work about $Top_{k_{WS}}$ algorithm based on multi-criteria decision support. This algorithm was being used by applying the Cloud Service Research and Selection Section (CSRSS) shown in figure 5.5, and that was being applied on the Cloud Computing Quality of Service (CCQoS) dataset presented in table 4.8. Then we compare the presented algorithm with the Threshold Algorithm (TA) and the No Random-access Algorithm (NRA) [91], [88]. The performance evaluation of the studied algorithm, which is the $Top_{k_{WS}}$, is performed using two subsystems of recommendation in both synthetics and real datasets. The subsystems above, namely the Recommender System number 1 (RS1) using the CCQoS dataset and Recommender System number 2 (RS2) using the FIFA dataset, are the component of a comprehensive System named the Generic Research and Selection System (GRSS), which is an improved approach of the CSRSS presented in [1]. This contribution is structured as follows: In section 4.2.2, we expose the related works. In section 4.2.3 and 4.2.4, we present our approach using two subsystems of recommendation. In Section 4.2.5, we extend experimental results based on real and synthetics datasets. Section 4.2.6 presents a brief discussion of the result achieved during these experiments. Then, the conclusion and some future work were given in section 4.2.7.

4.2.2 Connected Works

Collaborative filtering and query processing methods have expressed significant evolution from the past decade. These tools are several used in recommendation systems that require innovative platforms [166] to incorporate the user comportment [167]. Besides, easy access to information and descriptions of desired services (e.g., restaurants, hotels, department stores, and travel destinations) has led to many decision support systems and recommendation services. The most

[1] //www.fifa.com/worldcup/news/russia-2018-most-engaging-fifa-world-cup-ever
[2] www.kaggle.com/mathan/fifa-2018-match-statistics

used algorithms for query processing like Top_k recommendation algorithm for refining the Skyline [168] query for multi-dimensional data can lead either to a very large or to an insufficient number of responses. Indeed, these results can confuse the choice of user [50]. Therefore, before citer, the connected works that use the Skyline and the Top_k algorithm in decision aiding and RSs let us shed light on the objective of using this paradigm. Meanwhile, the Skyline, also referred to as Maximum in computational geometry or Pareto in business management, is important for many applications where multi-criteria decision making is needed.

Let have P a set of points n. Each the point p of d real-valued attributes can be described as a point $(p[1], p[2], \ldots, p[d]) \in \mathbb{R}^d$ where $p[i]$ is the ith attribute of p.

Now, let's assume that $p = (p[1], p[2], \ldots, p[d])$ and $p' = \left(p'[1], p'[2], \ldots, p'[d]\right) \in \mathbb{R}^d$, p dominates p′ if for each i, $p[i] \leq p'[i]$ and for at least one i, $p[i] < p'[i] (1 \leq i \leq d)$.

The Skyline is determined as the set of points P that are not dominated by another point of P [169]. Meanwhile, the Skyline performs the most important points or the optimal Pareto solutions of the data set. These Skyline points cannot dominate each other [170]. Nevertheless, some first works have been reported introducing in [171] for the first time the principle of dominating query in a dynamic context. A generalization conceived this approach of dominating queries concept into multi-dimensional datasets. In other studies, authors chose to combine the Skyline and Top_k paradigms in the context of distributed decision Systems. In another context, handling the Top_k problem was considered as the several employed in this domain, such as the amelioration of Fagin's algorithm. The studies in [91] show that The Threshold Algorithm (TA) is like the FA algorithm with some improvements. It is an approximation function used to find the optimal degree in all cases and uses less buff space to stop earlier. Algorithm 3 presents the TA as shown in [91]. Furthermore, another enhancement of FA is given by the No random access algorithm (NRA).

Moreover, this algorithm is used for dealing with the situation where the Random Access (RA) is very expensive than the Sorted Access (SA) or is impossible [91] like can be seen in algorithm 4. To the best of our knowledge, our method used to combine both paradigms, namely, Top_k and Skyline, is the first work that uses a Bi-objective Weighted Sum (BWS) to handle the Skyline and Top_k dominating query problem. In the previous work, the combination of Top_k and Skyline was used in synthetic dataset [1], in which the comparison with the Fagin algorithm was based on correlation study and runtime measurement of this approach shows its performance in the Cloud Computing Service domain. Consequently, the presented work aims to evaluate the used algorithm in a general context using the GRSS in the other application domain and using FIFA 2018 as a real dataset. However, sport management needs to take into account different properties [167]. Therefore we adopted the conceived algorithm while using users' choice scenarios for responding top Man of the match according to these characteristics. In practice, match statistics are never available before the match; this leads to using previous match statistics for having an accurate prediction. According to the authors in [172], this approach reflects that

informative match prediction can be made. The authors used the predicted statistics, which was calculated using a comprehensive method called: The Generalized Attacking Performance (GAP) Ratings. Another research in the field of sport management [165] was based on the Generalized Fuzzy Technique for Order of Preference by Similarity to Ideal Solution. (TOPSIS) as an MCDM tool. The TOPSIS method was developed for the first time by Hwang and Yoon in 1981 [173, 174], and the Fuzzy TOPSIS used in the discussed work to foresee accurate ranking and applied to the fallouts of FIFA 2018 world [1] cup soccer matches explicitly. The match statistics have been used up to quarter-finals to make better estimates for the upcoming games. Our work remains similar to the work mentioned above due to the use of the most widely used MCDA methods, particularly in sport management. However, our approach differs in that we use recommendation algorithms such as Top_k and Skyline to benefit from their combination in addition to hybridization with MCDA methods in two different application domains.

Algorithm 3 TA

Do parallel SA on all m lists object x seen under SA in a list
fetch its scores from other lists by RA
compute overall score
if |Buffer| < k **then**
    add x to Buffer
elseif score(x) <= $k-th$ score in buffer
    replace bottom of buffer with (x, $score(x)$)
    toss
    Stop when threshold <= $k-th$ score in buffer
    Threshold := t (worst score seen on L_1,...,L_m)
end if

Output the Top_k objects & scores (in buffer)

Algorithm 4 NRA

Do SA on all lists in parallel
Stop when there are at least k objects, each of which have been seen in all the lists
while depth d **do**
    Maintain worst scores x_1, ..., x_m,
    x any object seen in lists $1,...,i$
    Best(x) = t(x_1, ..., x_i, x_i+1, ..., x_1)
    Worst(x) = t(x_1, ..., x_i, 0, ..., 0)
    Top_k contains k objects with max worst scores at depth d
    Break ties using Best. $M = k-th$ worst score in Top_k
    Object y is viable if $Best(y) > M$
    Stop when Top_k contains >= k distinct objects and no object outside Top_k is viable
end while
Return Top_k

4.2.3 Our Approach

Although that the combined approach of Top_k and Skyline processing presented in chapter 2.2.4 manages the Skyline drawbacks. Still, the processing step shown in figure 5.5, cannot always be a suitable solution because it did not cover the whole of the possible recommendation scenarios. However, there are still a few scenarios in which the user can determine his ranking function, where the Skyline step is not required. The preprocessing step can increase the recommendation runtime. Consequently, this remark drives us to study the presented approach in the general case and provide a completely comprehensive system solution for these paradigms. The GRSS,

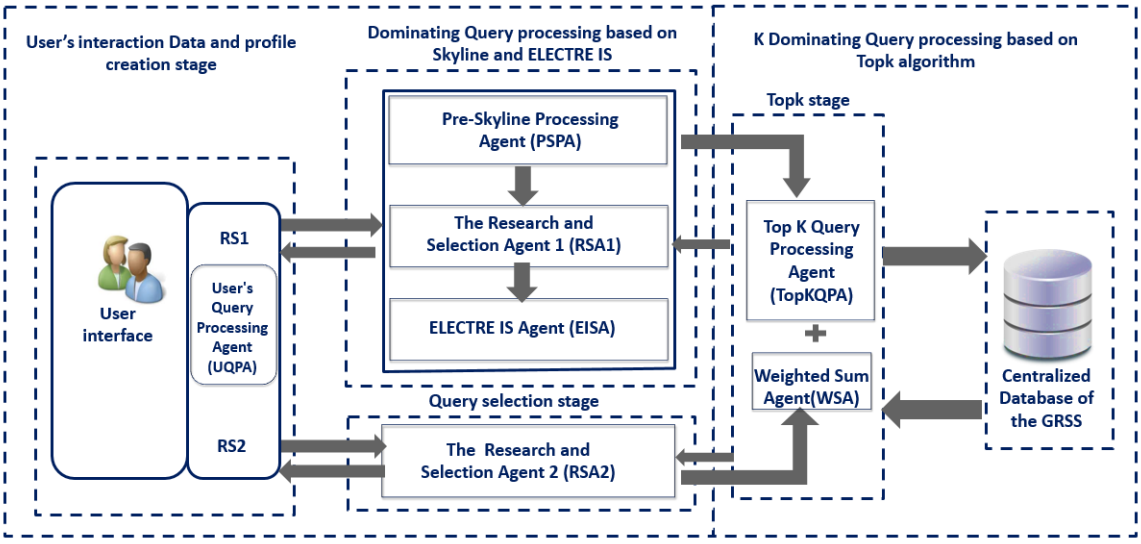


Figure 4.9: Generic Research and Selection System (GRSS) based on Skyline and Top_k

considered in figure 4.9, consists of the design of two subsystems that operate respectively in two stages. In the following we give an illustration of the formalism:

1. System for refinement of Skyline dominating Query:

   In the first stage, the combined approach uses the advantages of both paradigms. The aim is handling the problems coming respectively from Top_k used over a Big Data size, which creates an expensive sort of Top_k query, and the pain coming from the Skyline on high dimensionality. The Pre-Skyline Processing Agent (PSPA) provides the results from the database for the CSRSA and manages the Skyline Operator. Finally, the best requirements Cloud Services are returned, and their dimensions are stored as tuples. This stage was presented as a collaborative filtering system by using the Skyline agent, the ELECTRE IS agent [1, 51], and Top_k agent to better optimize the result size and response time of the request, [1, 50, 51] while maintaining efficiency and accuracy of the obtained results.

2. System for Top_k dominating Query:

In the second stage, we use the $Top_{k_{WS}}$ algorithm directly on the datasets to show the effectiveness of this algorithm, especially when the Skyline is not required. The importance of the applied query processing was incorporated into multi-objective behavior based on an adapted Multi-Criteria Decision-Aiding approach.
Finally, the processing of the GRSS uses the standard treatment of Users Query Processing Agent (UQPA), which uses either the Top_k alone or combine the Skyline and the Top_k Query together for the Research and Selection Agent. In the following, we explain the Top_k and the Weighted Sum Method Agents in detail, and we present the $Top_{k_{WS}}$ algorithm.

4.2.4 Bi-objective Weighted Sum Method (BWSM) [1]

Let R^n and R^p be Euclidean vector spaces referred to as the decision space and the objective space. We denote $X \in R^n$ as a feasible set and f as a vector-valued objective function.
$f : R^n \longrightarrow R^p$ composed of p real-valued objective functions $f = (f_1, \ldots, f_p)$, where $f_k : R^n \longrightarrow R^p$ for $k \in \{1, \ldots, p\}$ A multi-objective program (MOP) is given by:

$$min(f_1(x), ..., f_p(x)), x \in X \tag{4.1}$$

The function 4.2 shows the minimization of the weighted sum objective functions:

$$min \sum_{k=1}^{p} {}_k f_k(x), x \in X, and \ \ \in R^+. \tag{4.2}$$

We name the Bi-objective Weighted Sum as BWS to designate the method presented in [50] and [1] instead of WS, which is used as a linear weighted sum aggregation function. Consequently, we consider the (MOP) with linear objective functions, having the following form:

$$f_i(x) = \sum_{k=1}^{p} a_{ki} x_{ki}, a_{ki} \in R^+. \tag{4.3}$$

$$R(a_i) = \sum (w_j a_{ij}) \forall i \in [1, n] \tag{4.4}$$

With w_j represents the weights chosen by the user, and the a_{ij} represents the values of the criteria to be maximized or minimized. Therefore, If the criteria are to maximize, the algorithm seeks the maximum at using function. If not, as can be seen in function 4.5, the Algorithm 3 considers the inverse of these values.

$$\begin{cases} max \sum (w_j a_{ij}) & \text{if } a_{ij} \text{ to be maximized} \\ max \sum (w_j 1/a_{ij}) & \text{if } a_{ij} \text{ to be minimized} \end{cases} \tag{4.5}$$

As shown in function 4.6, the BWS has been used as an adaptation of the general WS. It represents the aggregation function. This monotonic and linear function allows us to answer a bi-objective

problem while optimizing the research cost by providing guarantees on the evolution of the candidates' scores [1].

$$f : max\sum(w_j a_{ij}) + max\sum(w_j(1/a_{ij})) \tag{4.6}$$

The $Top_{k_{WS}}$ uses the monotonic ranking score. Based on the aggregation function, which is denoted by the scorefunction in algorithm 5.

Algorithm 5 $Top_{k_{WS}}$ in the GRSS

```
Ls: input PriorityQueue
(the input is the result of the Skyline and ELECTRE IS algorithm in RS1)
tlArray: array of items
Item: input object which will calculate its score function
k: the number of objects returned by the algorithm
TopList: output list of the tuples forming the solution
Define Ls as priority queue based on ScoreFunction
function ComputeTopK
returned =0
while returned < k do
    Ls ≠ ∅ Ls ∈PriorityQueue
    result = Compare(Ls, item)
    if result < 0 then
        Select from Ls the object item with the maximum ScoreFunction using equation (4.5)
        Remove the head of this queue or returns null if this queue is empty
        Ls.add(item)
        Update ScoreFunction(item), and update Ls accordingly
    else
ScoreFunction(item) is completely known
        Report(item, ScoreFunction(item))
        returned= returned+1
    end if
end while
return TopList
end function
```

To evaluate requests' performance, we distinguish from the state-of-the-art, the most used metrics between rankings in this field. Hence we use the correlation study based on Kendall, Spearman coefficients, and their correlation significance coefficients, respectively, based on the p-value. The similarity between two users is based on their ratings of items that both users have rated [2].

4.2.5 Experiment results

In the experiment, we use a Core i5 (2.70 GHz) PC with 8 GB of memory. Algorithms are implemented in JAVA environment.

Table 4.8: The used configuration for the experiments.

dataset Num	dataset Type	Size (items)	Dimension
Dataset1	Synthetic (CS QoS) [103]	50000	4 - 10
Dataset2	Real (FIFA Predict) [4]	128	4 - 10

1. Experiment results based on synthetic datasets:

   The approach was used on the Cloud Service QoS (CS QoS), which designs the dataset1 in Table 4.8 using Algorithm 5. The denoted algorithm was compared with the Fagin algorithm (FA) in the study presented in [1] using the same dataset. The comparison showed our approach's efficiency according to data size and dimensionality variation, which was approved by runtime measurement and six recommendation metrics between rankings. Furthermore, an extended evaluation is given on this experiment based on comparing other algorithms, namely the TA and NRA, using the dataset1 and dataset2. The studies [50] and [1] contains more details about the dataset1.

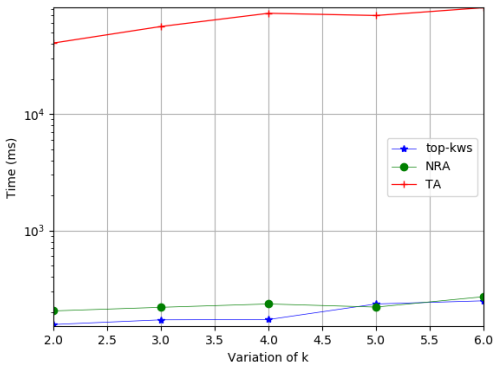


Figure 4.10: Response time variation of the $Topk_{ws}$, TA and NRA algorithm according to k variation using synthetic dataset

2. Experiment results based on real life datasets: For dataset2, we use the FIFA Predict 2018 Man of the Match found on 4.10 [4] shows an extracted example from the used dataset. It should be noticed that each row contains a team's performance details. As an example, a match between Morocco and 'Iran' on 14th June 2018 has two rows; one for 'Morocco' and another one for 'Iran.'

   (1) Goal Scored (GS): Number of goals scored by this team

(2) Ball Possession % (Percentage) (BallP): Amount of time ball was in control by the team

(3) Attempts: Number of attempts to score a goal

(4) On-Target: Number of shots on-target

(5) Off-Target: Number of shots that went off-target

(6) Blocked: Number of opponent team's attempts blocked by the team

(7) Corners: Number of corner shots used

(8) Off-sides: Number of off-site events

(9) Free Kicks: Number of free-kicks used

(10) Fouls Committed (FoulsC): Number of fouls committed by the team members

Date Match Date	Team Playing Team	Opponent Opponent Team	Goal Scored Number of goals scored by this team	Ball Possession % Amount of time ball was in control by the team	Attempts Number of attempts to score goal
17-06-2018 6% 25-06-2018 6% Other (23) 88%	France 5% Croatia 5% Other (30) 89%	France 5% Croatia 5% Other (30) 89%	0 6	25 75	3 26
14-06-2018	Russia	Saudi Arabia	5	40	13
14-06-2018	Saudi Arabia	Russia	0	60	6
15-06-2018	Egypt	Uruguay	0	43	8
15-06-2018	Uruguay	Egypt	1	57	14
15-06-2018	Morocco	Iran	0	64	13
15-06-2018	Iran	Morocco	1	36	8
15-06-2018	Portugal	Spain	3	39	8
15-06-2018	Spain	Portugal	3	61	12
16-06-2018	France	Australia	2	51	12
16-06-2018	Australia	France	1	49	4

Figure 4.11: Example of a part of dataset2 [4]

4.2.6 Discussion

In the beginning, we analyze the performance instability when the end-user executed a request of Top_k with k=5 in the GRSS. We use dataset1 for the RS1 and the dataset2 for the RS2. We distinguish which parameters from these datasets will be to maximize and which one of them will be to minimize. We evaluate with a scenario of identifying k = 5 Top_k objects over user-specified preferences on four dimensions on a range of ten dimensions. After that, we investigate if there exists a significant correlation between the parameters. The test was applied while considering our system involves user choice for answering a bi-objective problem. Then, we evaluate the runtime of recommendation according to k variation, as can be seen in figure 4.10, by comparing the $Top_{k_{WS}}$ algorithm with NRA and TA algorithms respectively, responding to the final user's requirement. According to dataset1, the results show that $Top_{k_{WS}}$ outperforms the NRA and

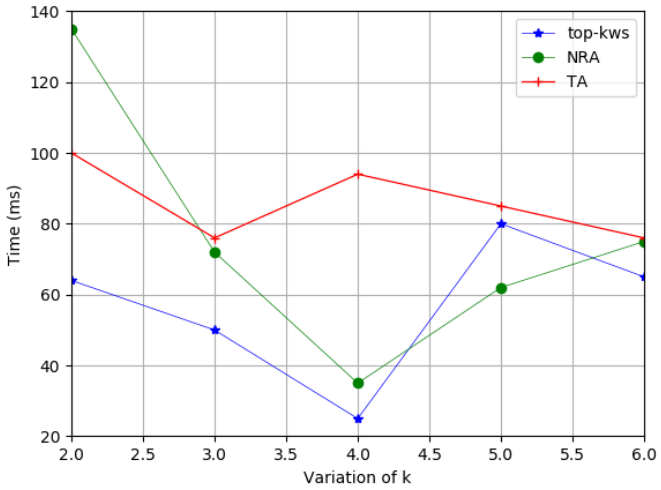


Figure 4.12: Response time variation of the $Topk_{ws}$, TA and NRA algorithm according to k variation using Real dataset

TA algorithms. Nonetheless, while the TA has a considerable runtime expressed, the runtime of $Top_{k_{WS}}$ remains very close to the NRA than that of TA. On another side, according to the dataset2 and the RS2. The k variation influence more slowly the gap between the algorithm's runtime. Although, the $Top_{k_{WS}}$'s runtime still inferior to NRA and TA algorithm. We noticed that the $Top_{k_{WS}}$ and NRA reach their best runtime in k=4. For studying the Spearman and Kendall metrics, we employ ten dimensions in the used datasets. The test was treating all pairs of possible variables in an example request of Top_k. At the same time, we were having the k=5 to both datasets of the system. For the dataset1: the criteria in our formalism include seven that must be minimized, which are the Availability, Data loss, Latency, Ongoing cost, Risk, RAM, and Response time. Besides, the criteria that must be maximized are Bandwidth, Hard Drive, and Portability. For the dataset2, we used six criteria that must be maximized: the Goal Scored, Ball possessing, Attempts, On-Target, and Blocked. Moreover, four criteria must be minimized: Off-Target, Corners, Off-sides, Free Kicks, and Fouls Committed. The Spearman coefficient estimates how strong a monotonic function could represent the relationship between two Cloud Service's parameters using dataset1 and the FIFA prediction parameters using dataset2. From figure 4.13(a),4.13(b),4.13(c), 4.15(a),4.15(b), and 4.15(c) it can be seen that: The coefficient values are from -1 to +1. If the value is +1, this tells that the variable's relationship is perfectly monotonous and is related to an accumulating correlation. However, if the value is -1, this explains that the relationship of variables is perfectly monotonous, related to a decreasing relationship. However, when the value is equal to 0, it indicates that the variables are not related. On the other hand, the correlogram's right legend shows the correlation coefficients and the intensity of the corresponding color. According to the presented graphs, the comparison between the three algorithms using

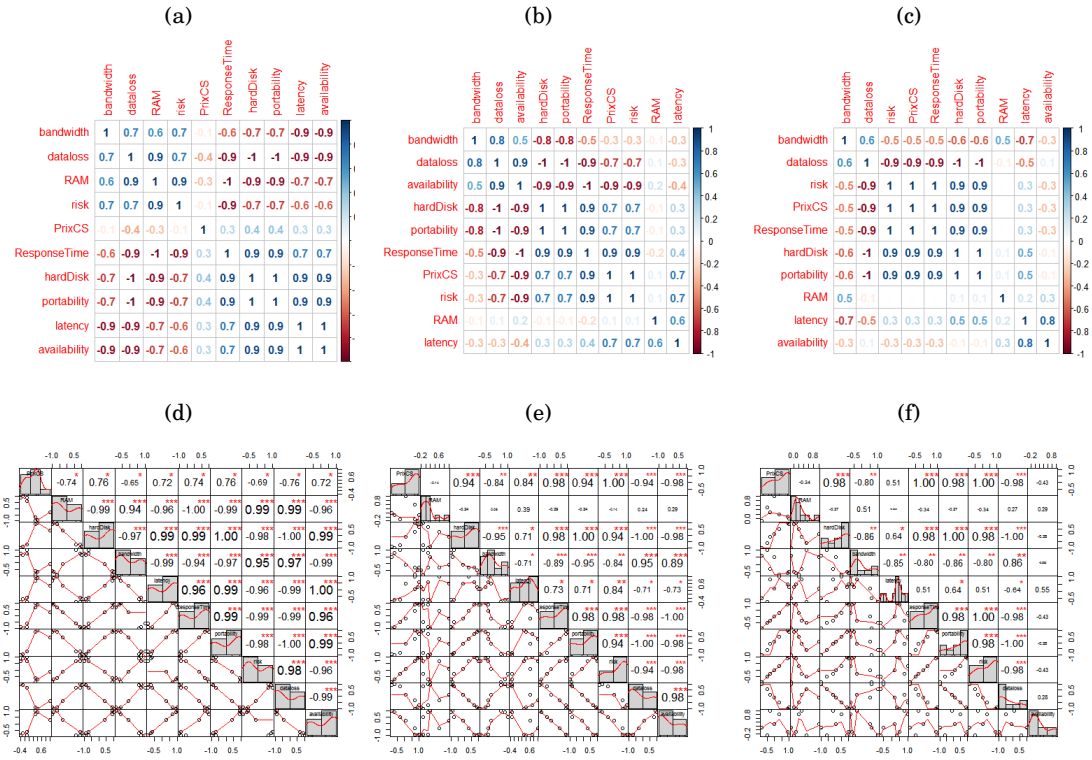


Figure 4.13: (a) Spearman Metric for Cloud Services Data using Topkws algorithm. (b) Spearman Metric for Cloud Services data using the TA algorithm. (c) Spearman Metric for Cloud Services data using the NRA algorithm. (d) The P-value for significance graph using Spearman coefficient Metric for Cloud Services data applying Topkws. (e) P-value for significance graph using Spearman coefficient Metric for Cloud Services data applying TA algorithm. (f) P-value for significance graph using Spearman coefficient Metric for Cloud Services data applying NRA algorithm.

the Spearman metric shows a more monotonous relationship according to $Top_{k_{WS}}$ than TA and NRA. The seem remark is for the Kendall metric. We noticed that Kendall metrics deal with the concordance and the discordance relationship between the parameters. While Spearman's metric, as mentioned above, study the monotonicity relationship. For more clarity of the use of the correlations coefficients matrix, we present an example of the Correlation matrix of Spearman and Kendall coefficients presented respectively in Table 4.9 and 4.10 between the 2018 FIFA parameters for a query of the top-5 man of the match using Top_{kWS}.

Furthermore, to completely analyze the given results, the $Top_{k_{WS}}$'s superiority was improved by the p-value [161] of Spearman and Kendall metrics, presented in figure 4.13(d), 4.13(e), 4.13(f),4.16(d), 4.16(e), and 4.16(f). In which The histograms of the variables shown on the diagonal. And the asterisks indicate the significance levels of the correlations. Moreover, the aforementioned statistical parameter shows the level of significance of the correlation coefficients. It is marked on the graph by " $*$ ", to indicate $P < 0.05$, two stars to indicate $P < 0.01$, and three

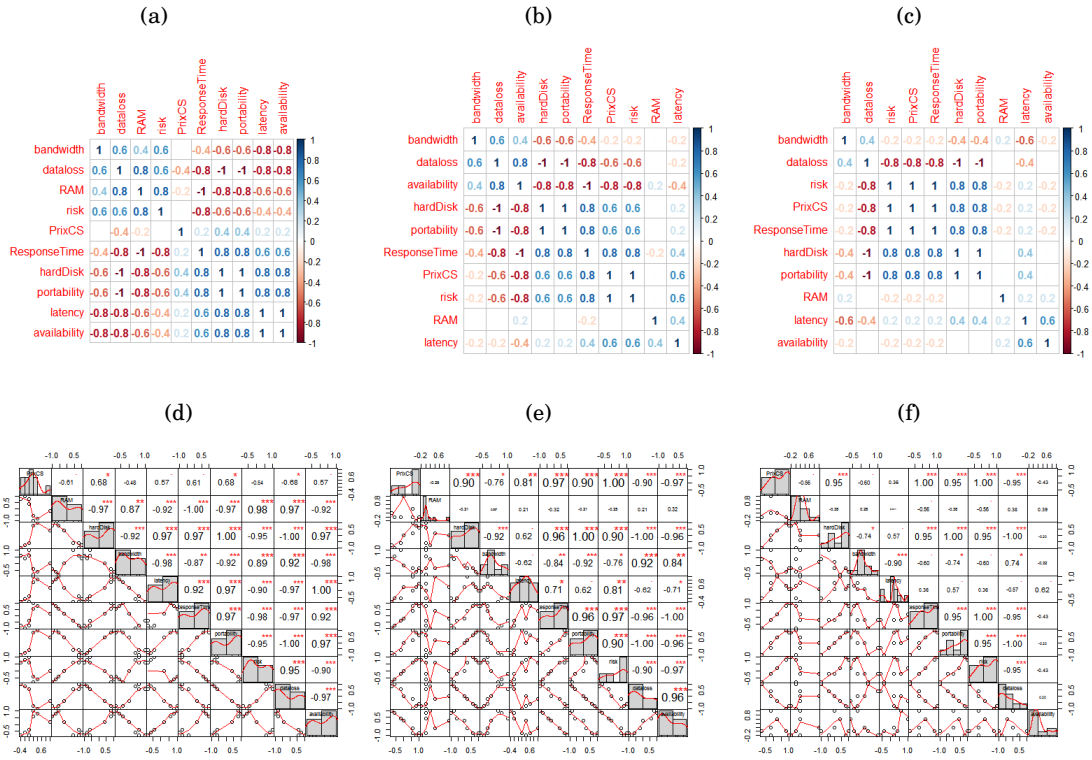


Figure 4.14: (a) Kendall Metric for Cloud Services Data using Topkws algorithm. (b) Kendall Metric for Cloud Services data using the TA algorithm. (c) Kendall Metric for Cloud Services data using the NRA algorithm. (d) The P-value for significance graph using the Kendall coefficient Metric for Cloud Services data applying Topkws. (e) P-value for significance graph using Kendall coefficient Metric for Cloud Services data applying TA algorithm. (f) P-value for significance graph using Kendall coefficient Metric for Cloud Services data applying NRA algorithm.

stars were practiced to indicate $P < 0.001$. Whether for the coefficients: ρ of Spearman, or the τ of Kendall. Meanwhile, the test is applied to our approach's correlation results compared to the given results by TA and NRA algorithms according to the dataset 1 and 2.

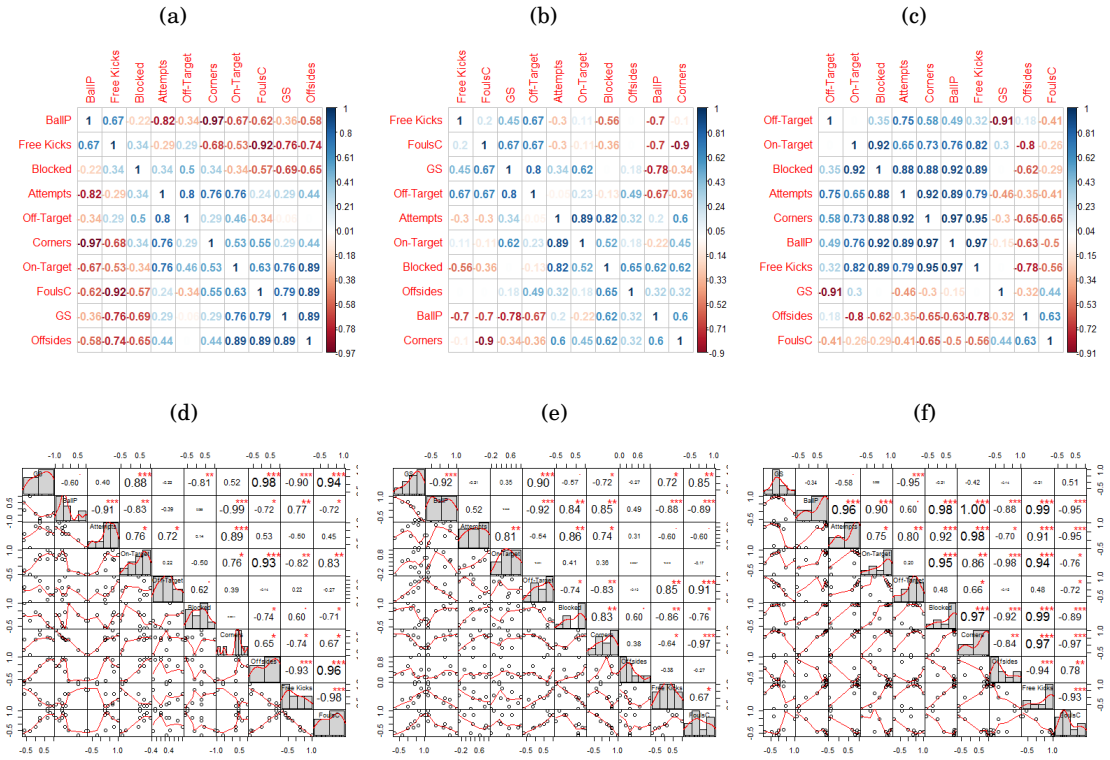


Figure 4.15: (a) Spearman Metric for FIFA man of the match 2018 Data using Topkws algorithm. (b) Spearman Metric for FIFA man of the match 2018 using the TA algorithm. (c) Spearman Metric for FIFA man of the match 2018 data using the NRA algorithm. (d) The P-value for significance graph using Spearman coefficient Metric for FIFA man of the match 2018 data applying Topkws. (e) P-value for significance graph using Spearman coefficient Metric for FIFA man of the match 2018 data applying TA algorithm. (f) P-value for significance graph using Spearman coefficient Metric for FIFA man of the match 2018 data applying NRA algorithm.

4.2.7 Conclusion

Whether in the field of sports management or Cloud Computing Service Selection, the improved approach based on the RS1 and RS2 shows the significant performance of the given runtime by Top_{kWS} algorithm compared to TA and NRA algorithm to respond to user requirements. Moreover, We try to obtain a trade-off between a good runtime and a good quality of found results. It is also worth mentioning the overall improvement of the GRSS as a general solution. Consequently, the quality of our approach results is considered attractive according to runtime evaluation and four correlation metrics. In the next chapter, We will propose a parallelization of this algorithm by utilizing it in a distributed context based on Hadoop and Spark while using more Big datasets.

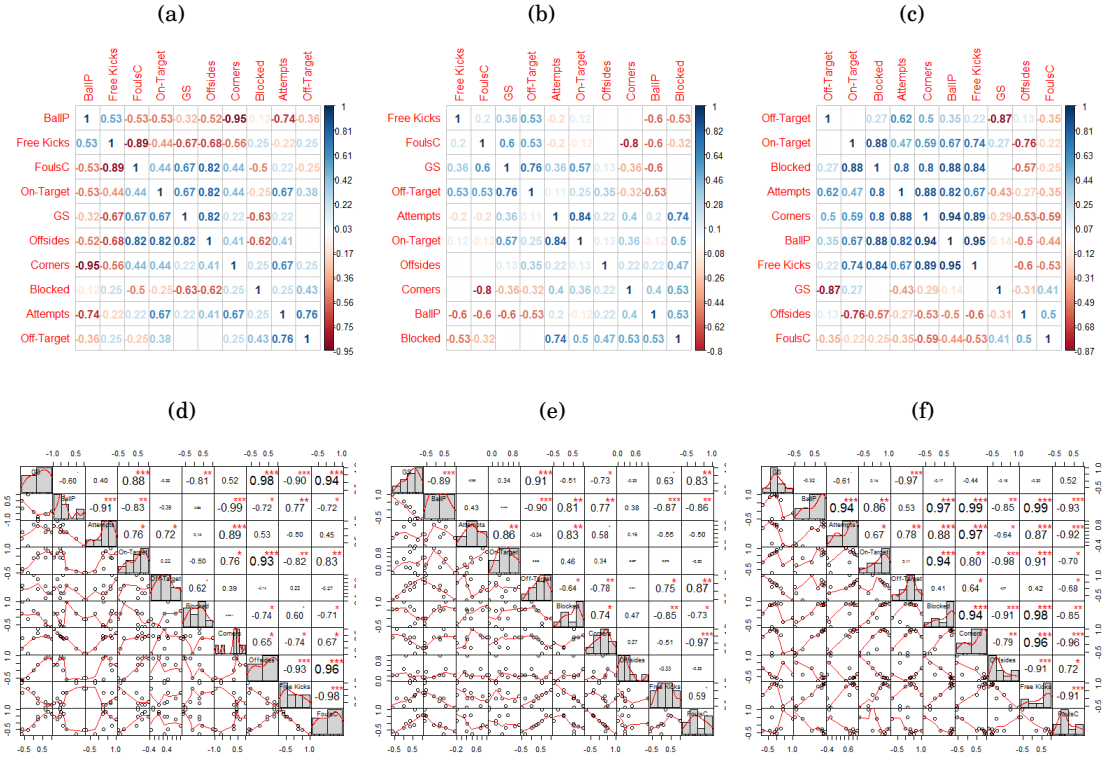


Figure 4.16: (a) Kendall Metric for FIFA man of the match 2018 data using Topkws algorithm. Asterisks show multiple sites of RH initiation in a single root hair cell (indicated by the arrows). (b) Kendall Metric for FIFA man of the match 2018 data using the TA algorithm. (c) Kendall Metric for FIFA man of the match 2018 data using the NRA algorithm. (d) The P-value for significance graph using Kendall coefficient Metric for FIFA man of the match 2018 data applying Topkws. (e) P-value for significance graph using Kendall coefficient Metric for FIFA man of the match 2018 data applying TA algorithm. (f) P-value for significance graph using Kendall coefficient Metric for FIFA man of the match 2018 data applying NRA algorithm.

Table 4.9: Example of Spearman's correlation coefficients between the criteria of the 2018 FIFA dataset based on Topkws using k=5.

	GS	BallP	Attempts	On-Target	Off-Target	Blocked	Corners	Offsides	Free Kicks	FoulsC
GS	1.00	-0.36	0.29	0.76	-0.06	-0.69	0.29	0.89	-0.76	0.79
BallP	-0.36	1.00	-0.82	-0.67	-0.34	-0.22	-0.97	-0.58	0.67	-0.62
Attempts	0.29	-0.82	1.00	0.76	0.80	0.34	0.76	0.44	-0.29	0.24
On-Target	0.76	-0.67	0.76	1.00	0.46	-0.34	0.53	0.89	-0.53	0.63
Off-Target	-0.06	-0.34	0.80	0.46	1.00	0.50	0.29	0.00	0.29	-0.34
Blocked	-0.69	-0.22	0.34	-0.34	0.50	1.00	0.34	-0.65	0.34	-0.57
Corners	0.29	-0.97	0.76	0.53	0.29	0.34	1.00	0.44	-0.68	0.55
Offsides	0.89	-0.58	0.44	0.89	0.00	-0.65	0.44	1.00	-0.74	0.89
Free Kicks	-0.76	0.67	-0.29	-0.53	0.29	0.34	-0.68	-0.74	1.00	-0.92
FoulsC	0.79	-0.62	0.24	0.63	-0.34	-0.57	0.55	0.89	-0.92	1.00

Table 4.10: Example of Kendall's correlation coefficients between the criteria of the 2018 FIFA dataset based on Topkws using k=5.

	GS	BallP	Attempts	On-Target	Off-Target	Blocked	Corners	Offsides	Free Kicks	FoulsC
GS	1.00	-0.32	0.22	0.67	0.00	-0.63	0.22	0.82	-0.67	0.67
BallP	-0.32	1.00	-0.74	-0.53	-0.36	-0.12	-0.95	-0.52	0.53	-0.53
Attempts	0.22	-0.74	1.00	0.67	0.76	0.25	0.67	0.41	-0.22	0.22
On-Target	0.67	-0.53	0.67	1.00	0.38	-0.25	0.44	0.82	-0.44	0.44
Off-Target	0.00	-0.36	0.76	0.38	1.00	0.43	0.25	0.00	0.25	-0.25
Blocked	-0.63	-0.12	0.25	-0.25	0.43	1.00	0.25	-0.62	0.25	-0.50
Corners	0.22	-0.95	0.67	0.44	0.25	0.25	1.00	0.41	-0.56	0.44
Offsides	0.82	-0.52	0.41	0.82	0.00	-0.62	0.41	1.00	-0.68	0.82
Free Kicks	-0.67	0.53	-0.22	-0.44	0.25	0.25	-0.56	-0.68	1.00	-0.89
FoulsC	0.67	-0.53	0.22	0.44	-0.25	-0.50	0.44	0.82	-0.89	1.00

CHAPTER 5

PERSONALIZED TOP-KWS PROCESSING: FROM CENTRALIZED TO DISTRIBUTED SYSTEM

This chapter presents two main contributions. The first one consists of designing a parallel modulation of the Top_k algorithm using the Map-Reduce paradigm; it is presented as the fourth contribution proposed in section 5.1 of this thesis. A second contribution in section5.2 presents another modeling using Spark RDD technologies and machine learning techniques to improve and develop the recommendation system in the context of hybrid recommendation. Besides, it discusses the main limitations of the above algorithms.

5.1 Contribution 4: Parallel Top_{kws} using Map Reduce Model

Various studies that are concerned with the parallelism of Top_k algorithms attempt to use the Map-Reduce Framework see figure 5.1, as it is the same state in our solution proposal. However, in this figure, we show that to finalize the implementation of Map-reduce, we take the reducer's key-value pair and write it in the file by record writer. By default, it separates the key and the value by a tab and each record by a linefeed character. We can configure it to get a more productive output format. But the final data is still written to the HDFS.

5.1.1 Map-Reduce Framework for Top_{kws}

In this work, we propose the algorithm 6 and 7 of parallel Top_{kws} computations based on this paradigm for refining the skyline result, which will be called: $MRTop_{kws}$.
Suppose an m-dimensional dataset $Ds\{d_1, d_2, \cdots, d_m\}$. We assume that the dataset is distributed into n subsets $\{Ds_1, Ds_2, \cdots, Ds_n\}$ in different locations. The algorithm starts by partitioned each distributed dataset vertically. Then, each partition was forwarded to the map workers (mappers).

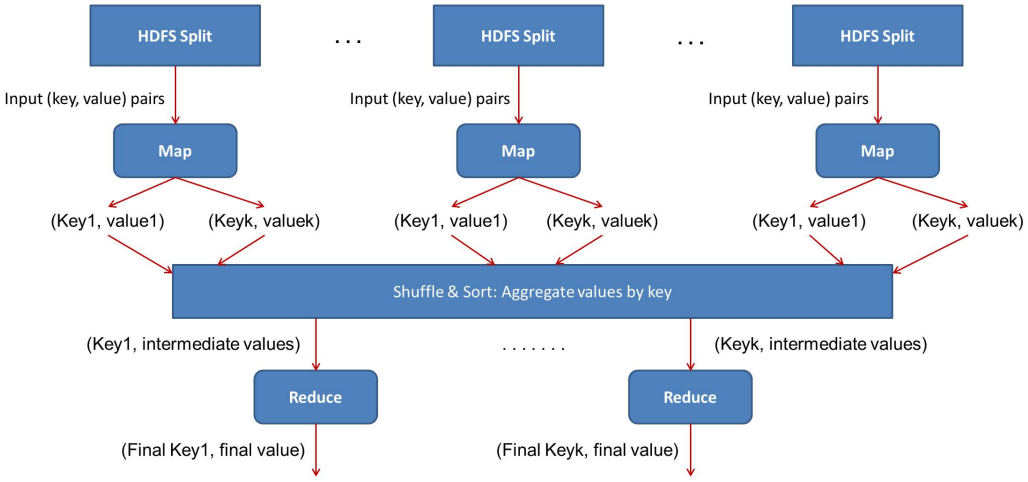


Figure 5.1: Map-Reduce work in Hadoop[5].

Algorithm 6 $MRTop_{kws}$**: Map phase**

1: **input**: Dsi
2: **Output**: records of Ds: key, value
3: $Map(key, value)$
4: verify normalization of criteria
5: $String = nexTokenizer(valuetostring)$
6: Emit $(key, value)$

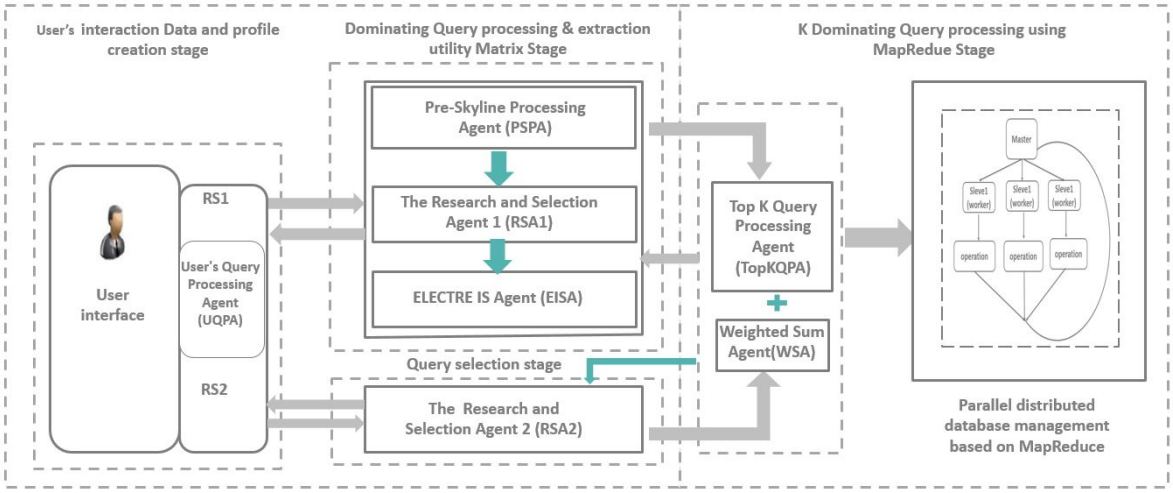


Figure 5.2: The e-CSRSS based on parallel database using Map-reduce framework

The Map-Reduce algorithm takes input a set of values, representing Cloud service criteria to replace them in the form: (key, value) for it to be processed by the Mapper and then by the reducer.

Algorithm 7 $MRTop_{kws}$**: Reduce phase**

1: **input**: A subset of key key_1, key_2,... with the associated sets of val_1,val_2,... resulting from the mapper phase
2: **Output**: Top_kScore for value key
3: $Reduc(key, value)$//with key sorted in descending order of scores
4: and with w_{ij} the user input and ,$label$ l $\in \{0,1\}$
5: int $l = -1$
6: **for** $(val_i : values)$ **do**
7: $++l$
8: **if** $(label[l] == 0)$ **then**
9: $Score+ = w_i[l] * (1/val)$
10: **else**(label[l]!=0)]
11: $Score+ = w_i[l] * (val)$
12: $Score+ = 0$
13: **end if**
14: **end for**
15: $res.set$(Score)
16: **Emit**(key, res)
17: **skyp** a objects with score
18: **Emit** (key ,Top_kScore)

- **Mapper phase** At line 5 of algorithm 6, a StringTokenizer is used to split the string obtained from a transformation. For each word received, we create a pair whose key is the word, and the value is worth the offset of the line. For each data slicing, Hadoop creates a Map task that will execute the map function developed accordingly. Each data slicing is processed by only one task Map.

  It should be noted that it is not the data transported to the program, but the reverse. Hadoop will try to find the closest node containing the data to transfer the Map function. It is a data Locality Optimization.

- **Reducer phase.** Hadoop will launch the Reduce tasks until all tasks are completed. The input of the Reduce function corresponding to the output of the Map functions set.
  Each Reduce task generates an output file to be stored, this time in the HDFS file system. Then the Reducer calculates the scores associated with keys and outputs the k pairs with the highest scores. To implement that, we check if the criteria are to be minimized or to be maximized. Then we calculate the weighting to be assigned to each label that is 0 or 1 and, we calculate the final score. After applying our score function to have a precise decision about the first K recommended cloud services, we still have to schedule our output and select the services by fixing an integer k. Our goal is always to detect the Top_k Cloud service with a higher degree of importance according to the user choice that mains the Cloud services, which have higher scores.

To understand our algorithm's modeling, here is a Figure 5.3 that illustrates the use of the data injected in an Hdfs input copy of our CSV file that contains the services and their associated scores. However, the mapper reads a record in the form of a couple <key, value> with :

- Value of type String (it is a line in the file).
- Key of type LongWritable (it is the Id of the service, and it is the line offset).

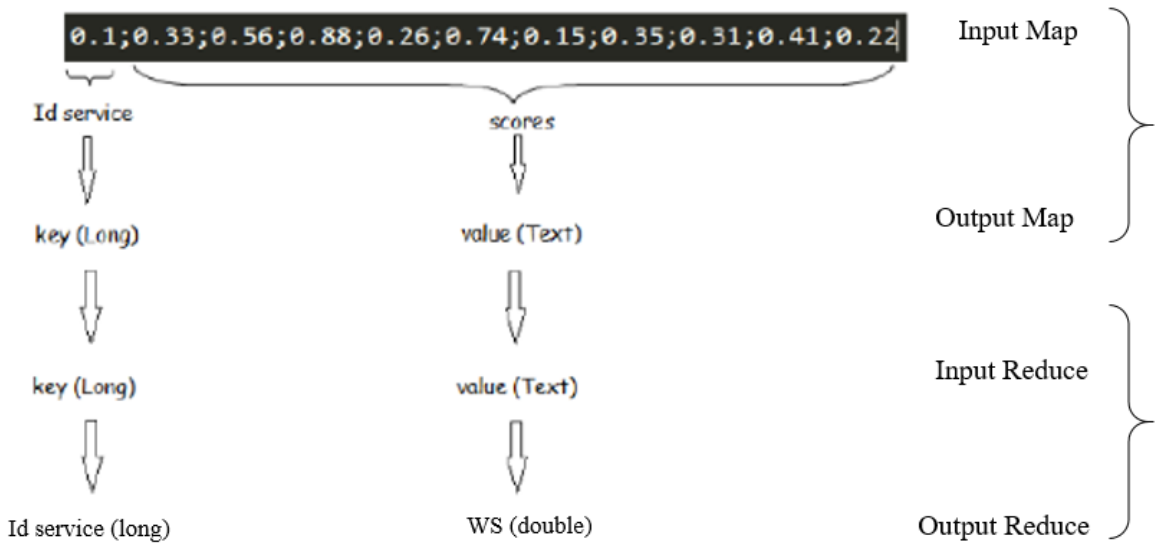


Figure 5.3: The weights given by the user assigned to each service in parallel

After applying our weighted sum function to have a precise decision about the first k recommended services, we still have to schedule our output and select the services by setting an integer k. The choice of Pig instead of Map-Reduce and especially in this phase was not random. Still, the data flow, the processing difficulty concerning the values in Map-Reduce, and the complexity of the program were our constraints. Here we will load our Map-Reduce output as a table to be processed with the Pig Latin language thanks to the communication between the two components of the Hadoop platform. Then apply a simple query to order and limit the selection of the K items. Prizing that this query has a very detailed Map Reduce java code behind it. Still, thanks to the Pig Latin its becomes something straightforward.The language of this platform is abstracted from the Java programming language Map-Reduce and is placed at a higher level of abstraction.

5.1.2 Results and discussion

For this experiment we use Hadoop 3.0.0.0 with java 8

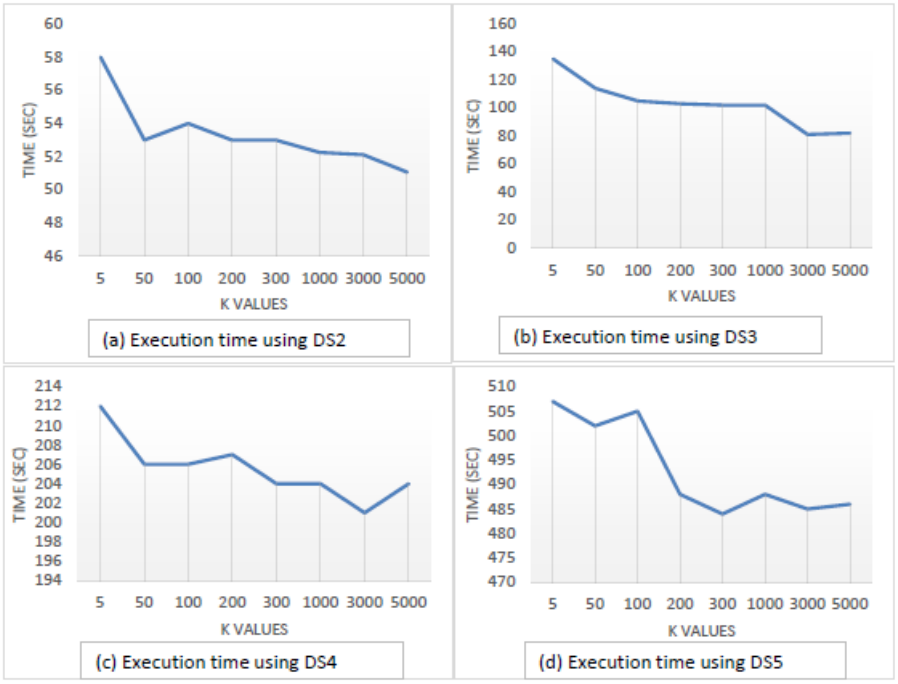


Figure 5.4: runtime of $MRTop_k$ for Skyline refining problem in (CSRSS)

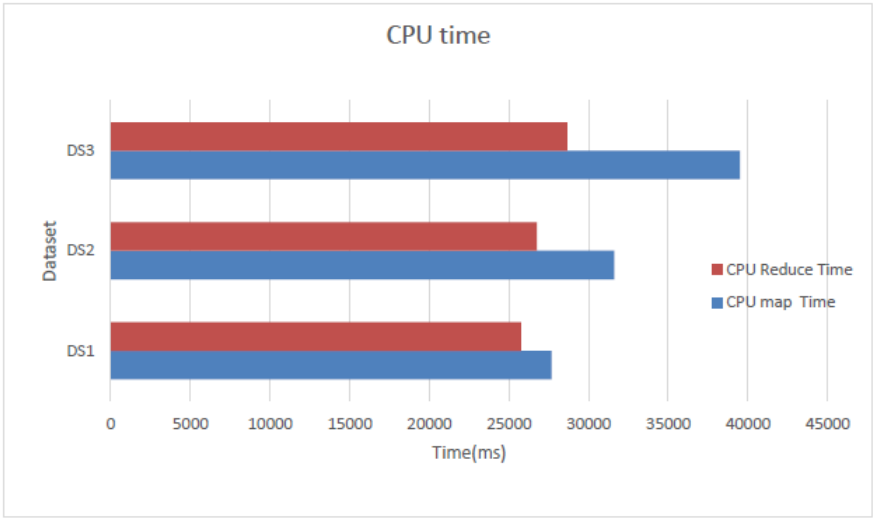


Figure 5.5: runtime of $MRTop_k$ for Skyline refining problem in (CSRSS)

Service	key	value (score)
S1	137794	8266108.906
S2	70573	1374702.158
S3	99828	333200.887
S4	160649	238354.308
S5	142991	174565.898
S6	37883	74508.398
S7	135662	67738.272
S8	119152	31215.643
S9	9118	28001.248
S10	128721	16254.917

Table 5.1: $MRTop_k$final output while using Pig and Map-Reduce for k=10

datasets	input values	input size
DS1	3633	200,6 KB
DS2	50000	2 ,7 MB
DS3	100000	5,5 MB
DS4	200000	11,1 MB
DS5	500000	27,7 MB

Table 5.2: Dataset used for scalability study

5.1.3 Conclusion

In this work, we tried to model our Top_{kws} algorithm using Map-Reduce as the main component of the Hadoop Framework. This modeling was done by combining Map-Reduce and Pig. The latter is also based on the Map-Reduce paradigm. However, Pig ensures a more significant acceleration of 10 times faster than Map-Reduce. So the addition of Pig gives us high-level processing of data flow and simplicity of intercommunication with Hadoop. In the part of this chapter, we will present an other modeling based on Spark and benefit from the technology of the Spark Streaming and Machine learning library, compare our algorithm with the distributed version of the Top_k algorithm like the Fagin algorithm,for being able to deals with the most common recommendation problems like the cold start problems and using other bigger datasets

5.2 Contribution 5: Parallel Top_{kws} using Spark RDD and Machine Learning based Model

This contribution is presented as follows: In section 5.3, we expose our contribution by using the RDD paradigm for combining the Top_{kws} algorithm and the MCDA methods trough a new collaborative filtering approach. In Section 5.5, we present an evaluation of the parallel approach using synthetic and real datasets. Finally, in section 5.6, we conclude this contribution while giving some perspective.

5.3 Modeling Top_{kws} using Spark RDD

The RDDs contribution aims to parallelize the $Top_{k_{WS}}$ algorithm using the open-source Hadoop and spark framework in the Cloud suggested system. We provide service recommendations based on user decisions. The application of these technologies has given rise to the Parallel version of $Top_{k_{WS}}$, called "**SP**ark $\underline{Top_k}$-**W**eigted **S**um" ($SPTop_{k_{WS}}$). The work introduces a new hybrid solution to traditional user-based CF methods that combines MCDA [105, 175], and Top_k query processing to improve its recommendation accuracy. Extensive experiment results indicate the superiority of $SPTop_{k_{WS}}$, showing that parallel algorithms can be significantly more efficient than their sequential counterparts $Top_{k_{WS}}$, mainly, by proving its significant advantages over other algorithms in terms of the recommendation accuracy and prediction evaluation. However, in the previous work [1], $Top_{k_{WS}}$ presented a step of Skyline refinement using MCDA techniques. Indeed the result was positively influenced by the Skyline that reduced the number of choices given to the user. Although this important advantage denotes the combination of both paradigms and notwithstanding the parallel solution presented by RDDs. The problem of sparsity generated by user ratings remains existing that requires good preprocessing on a large scale. To remedy the challenges above, we have introduced a preprocessing based on the Funk SVD model based on Matrix Factorization Machine Learning techniques. The main contributions in this chapter are summarized as follows:

1. An optimization approach is proposed to improve the recommender system's scalability, which includes a dimension reduction approach. Sparsity is addressed by applying an associative retrieval framework and related distribution algorithms to explore transitive associations between users through their past requirements and feedback. The reduction is based on the Funk SVD model [46] in the training process and an adapted bi-objective weighted sum method in the prediction process.

2. A novel parallel algorithm $SPTop_{k_{WS}}$ is proposed using RDD to improve the $Top_{k_{WS}}$ algorithm performance, while combining Top_k-parallel recommendation processing and parallel MCDA optimization.

3. Improvement of Generic Research and Selection System presented in [1] is proposed by combining the Skyline and Top_k dominating query processing and by using a hybrid recommendation based on MCDA and CF approaches.

5.4 The proposed approach using the Machine learning tools: Funk SVD based model

The creation of hybrid recommendation systems involves implementing separate systems based on collaboration and content [176, 177]. We can then have two different scenarios. Practically

combining the results obtained from the individual recommendation systems into a single final recommendation [178]. Using one of the individual recommendations at any time, choose to use the appropriate one than the others based on a quality recommendation measure. In the present case, the approach distinguishes between two different scenarios. However, it uses a generic recommendation system consisting of two subsystems shown in Figure 5.6, each subsystem being considered a hybrid recommendation system based on the MCDA and CF methods.
The purpose of the first is the refinement of the Skyline and ELECTRE IS (MCDA outranking method). Refinement uses dominating query processing applying by Top_k, and the CF techniques applying by funk SVD model. It, however, combines the advantages of the Skyline and Top_k algorithms into the first common subsystem called Recommendation System number 1 (RS1). The main topic of the RS1 is the management of problems emanating from Top_k and Skyline. Namely, the problem of Top_k used on large data size, which creates a kind of expensive Top_k query, and the problem originating of the Skyline used on high dimensionality. Moreover, the Pre-Skyline Processing Agent (PSPA), manages the Skyline operator and the MCDA ELECTRE IS algorithm according to the best requirements of the Top_k algorithm. detailed in the previous work [1, 48, 50, 51, 95, 179]. Now that their dimensions are stored in tuples. This hybrid recommendation system optimizes the result's size and the response time of the request while maintaining the efficiency and accuracy of the results obtained by the Research and selection Agent (RSA1). As mentioned above, RS1 manages Skyline and the Top_k drawbacks and uses the main advantages of the combined paradigms. Despite this, the RS1 did not cover all possible recommendation scenarios. There are still some scenarios where the user can determine the user's ranking function, where the Skyline step is not necessary. And therefore, the elimination of the Skyline can lead to a reduction in terms of recommendation runtime. This remark motivates us to generalize the System while providing a complete equilibrium system solution called GRSS by adding a Recommendation System number 2 (RS2). Afterwards, RS2 is used for Top_k's dominant queries that directly use the $Top_{k_{WS}}$ algorithm in case of centralised database and $SPTop_{k_{WS}}$ in case of distributed database as figured in 5.6. The processing of applied requests has been incorporated into the use of a subsequent step by employing the best-known secularization technique [180] presented by the weighted sum method (WSM) and based on utility theory [181]. Afterward, we adopted this method as a bi-objective weighted Sum Method (BWSM), which is the score function for sequential and parallel Top_k algorithms detailed in what follows.

5.4.1 Top_k and Funk SVD model

To use the SVD model for $Top_{k_{WS}}$ recommendation, we implemented the well-known matrix factorization algorithm as introduced by Simon Funk [69]. The model was trained on the dataset using the Funk SVD algorithm. It implements a stochastic gradient descent optimization technique. However, The U(user) and V(item) factor matrices are initialized at small values and cropped to k features. Each feature is trained until convergence, while the convergence value

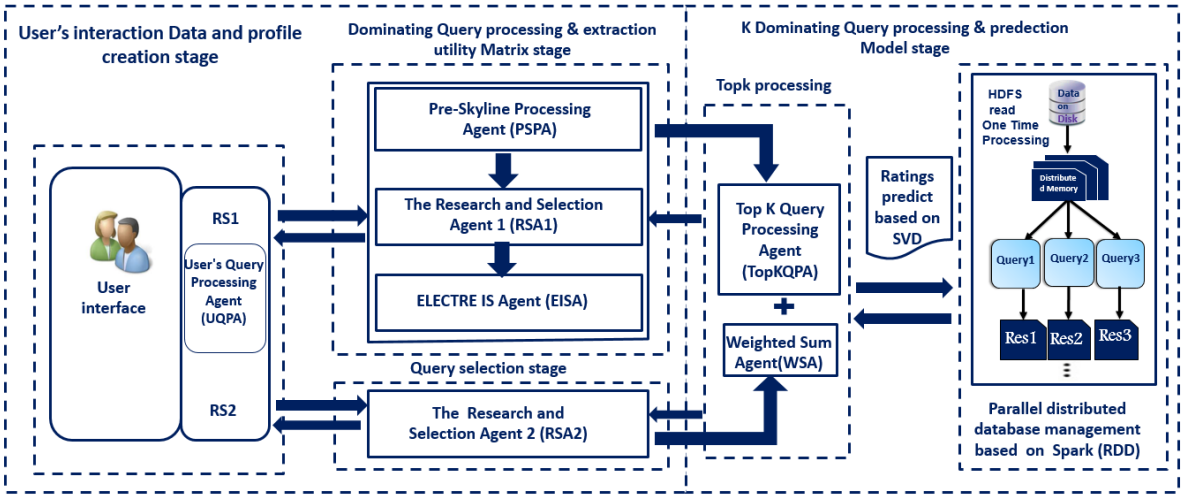


Figure 5.6: Parallel in GRSS using distributed database and SVD-based model

has to be specified by the user, by configuring the steps argument. The algorithm predicts r_{ui} and calculates the error as presented in the algorithm 8 [182]: The attribute η represents the

Algorithm 8 Stochastic Gradient Decent (SGD)

$e_{ik} \leftarrow r_{ik} - u_i^t v_k$
$u_i \leftarrow u_i - \eta(e_{ik} v_{ki} - \beta u_i)$
$v_k \leftarrow v_k + \eta(e_{ik} u_i - \beta v_k)$

learning rate, while β corresponds to the regularization term's weight. Subsequently, the algorithm verifies if the argument biases are $TRUE$, the biases are computed to update the features and generate predictions. Then, we create the attribute matrix of the element. Afterward, we use the traditional similarity cosine-based similarity calculation formula 5.1. For doing that, we use a similarity algorithm based on a k-nearest neighbor item-based "IBKNN". This heuristic is introduced to find a set of "closest neighbors" for each user and, at the same time, simplify the scoring procedure. In fact, given two items a and b, as rating vectors and . The method computes the cosine similarity as:

$$sim(\vec{a}, \vec{b}) = cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|} \tag{5.1}$$

The $Top_{k_{WS}}$ uses the monotonic ranking score. Based on the aggregation function, which is expressed by the scoreFunction in algorithm 9.

Algorithm 9 $Top_{k_{WS}}$ using FukSVD

Ls: input PriorityQueue
$tlArray$: array of items
M, the user-item ratings matrix. Each $M_{u,i}$ element represents a rating and is either in N or empty.
u, the user that the recommendations computation is doing for him.
n, the number of recommendations to return.
Sim(u,v), a user defined function which computes the cosine similarity between two users see **equation: 5.1**
$Item$: input object which will calculate its score function generated using FunkSVD from **algorithm: 8**
k: the number of items returned by the algorithm
$TopList$: output list of the tuples forming the solution
Define Ls as priority queue based on $ScoreFunction$
function $ComputeTopK$
$returned$ =0
while $returned < k$ **do**
    $Ls \neq \emptyset$ $Ls \in PriorityQueue$
    $result = Compare(Ls, item)$
    if $result < 0$ **then**
        Select from Ls the object item with the maximum $ScoreFunction$ as presented in **equation 3.16 and 4.6**
        Remove the head of this queue or returns null if this queue is empty
        $Ls.add(item)$
        Update $ScoreFunction(item)$, and update Ls accordingly
        $Report(item, ScoreFunction(item))$ and
        $returned$= returned+1
    end if

end while
return $TopList$
end function

5.4.2 The proposed parallel $SPTop_{k_{WS}}$ using Spark RDD and parallel SVD Model

Let R^n and R^p be Euclidean vector spaces referred to as the decision space and the objective space. We denote $X \in R^n$ as a feasible set and f as a vector-valued objective function. $f : R^n \longrightarrow R^p$ composed of p real-valued objective functions $f = (f_1, \ldots, f_p)$, where f_k: $R^n \longrightarrow R^p$ for $k \in \{1, \ldots, p\}$ A multi-objective program (MOP) is given by:

$$min(f_1(x), ..., f_p(x)), x \in X \tag{5.2}$$

The function (5.3) gives the minimization of the weighted sum objective functions:

$$min \sum_{k=1}^{p} {}_k f_k(x), x \in X, and \ \in R^+. \tag{5.3}$$

Where the weights $_i, i \in \{1,...,p\}$ corresponding to objective functions satisfy the following conditions: $\sum_{i=1}^{p} {}_i = 1, {}_i \geqslant 0, i \in \{1,\ldots,p\}$. We represent the Bi-objective Weighted Sum as BWS to indicate the method presented in [1] instead of WS, for using this linear weighted sum aggregation function. Hence we consider the (MOP) presented in function (5.2) with linear objective functions, as: Several studies that are interested in the parallelism of Top_k algorithms try resorting to

Algorithm 10 $SPTop_{k_{WS}}$

1: **Input**:
2: $Mu*$, an RDD representing the user-item ratings matrix, keyed on the user index. Each Mu*, element represents a rating generating by FunkSVD model.
3: $NearItems$, the Top_k Nearest Items for each item in M, as described in 2.1.1. It is initialized as a broadcast variable.
4: K, the number of item recommendations to return. $ScoreFunction(n;V;NearItems;K)$ a user-defined function that calculates the item recommendations concerning each user n based on the weighted sums approach presented in section 5.4.1.
5: $Map(f,(n,V))$ and $GroupByKey(n,V)$ the (n,V) denotes an RDD object and f is a user-defined function.
6: **Output**: Top_K Score for value key, itemRecs, the Top_k item recommendations for the user u.
7: ***function***GroupedItemRatings(M)
8: $UIR \leftarrow GroupByKey(Mu*)$ // where UIR is User Item Ratings
9: **Emit** UIR
10: **End function**
11: **Emit** (key ,Top_K Score)
12: ***End function***
13: ***function***TopKRecommendations(UIR,NItems,K) // where NItems is the Nearest Items
14: IRecs ← Map(ScoreFunction(NItems, K), UIR) // where IRecs is the Recommended Item
15: **Emit** IRecs
16: ***End function***

working of the MapReduce Framework or Spark RDD based Framework, as is the case in our suggested solution. In this work, we propose a new algorithm in 10. This algorithm is a parallel $Top_{k_{WS}}$ computations for refining the Skyline result in RS1 and for generating the Top_k request directly in RS2. The algorithm will be called: $SPTop_{k_{WS}}$. As with the sequential approach, the first step in calculating the similarity of parallel users is to obtain a matrix M based on the evaluation matrix of the user element. Using a parallel funk SVD model in the Spark machine learning library, MLlib[183].

The SGD is considered easy to parallelize. Since the processing of a notation updates the U and V rows, special consideration must be given to avoid changing the same rows at the same time. Then, once the element similarity matrix has been calculated, the next step is to calculate the k most important recommendations for each user. It is done by iterating the history of interactions between each user's elements and calculating the bi-objective weighted sum score function for the items related to each element. In this case, we initialize a SparkContext object and use Spark's textFile operation to read the data in a parallelized collection as well as Map(key, value)

and Reduce, and GroupBykey functions. After specifying the number of partitions to distribute the data. We then construct M by mapping each element of our data set and collecting the corresponding pairs of users and ratings of that element. We call the cache to request Spark to keep the RDD in memory of the worker's nodes to improve performance.

5.5 Result and performance evaluation.

In this section, we will perform an empirical study to investigate the effectiveness and efficiency of $Top_{k_{WS}}$ and $SPTop_{k_{WS}}$ as part of an extensive research in this work. Hence, we first present the experimental configurations and then discuss synthetic and real data sets' testing results.

5.5.1 Experimental setup

Sequential algorithms were implemented in a JAVA environment on a Core i5 (2.9 GHz) PC with 8 GB of memory while the parallel algorithm was implemented using Scala on the Spark IBM cluster. Therefore, we have created an Apache Spark and Apache Hadoop clusters, and we use data in IBM Cloud Object Storage. We use the Default Node size - 4vCPU, 16GB RAM, 2 x 300GB HDFS disk, and we build a total of the Spark cluster that contains three virtual machines: one is the primary node (master), the others are the slaves (workers). The clusters have the following Components: Apache Spark 2.3.2, Hadoop 3.1.1, and Java version 1.8.0 25 from OpenJDK. Each slave node could run simultaneously two maps or reduce tasks. Each experiment was repeated twice, and the average time was considered.

5.5.2 The used datasets

In the experiments, we evaluate sequential and parallel approaches to synthetic and real data sets. The respective parameters of these datasets are reported in Table 5.3. Respectively. The dataset is divided into five roughly equal parts "folds" (using Cross-validation with the number of folds equal to 5). In turn, each of the k parts is used as a test set. The rest (in other words, the union of the four different positions) is used for training.

5.5.3 Evaluation metrics

To evaluate the prediction accuracy, we employ two metrics, MAE and RMSE, which are discussed in the state of the art chapter. RMSE and MAE are both calculated separately for each user, and then the average is calculated for all users. Moreover, the average error is determined for all predictions. They are then called GlobalRMSE and GlobalMAE. The presented recommender systems' effectiveness is also measured by recall and precision in this part of the results.

Table 5.3: Description of the used datasets

Datasets	Inputs-values	Input-size	Description	Dimensions
MovieLens	29517	1,06 MB	It is a real dataset from Learning from Sets of Items in Recommender Systems. This dataset contains results from a survey about the users' ratings on sets of movies. More details about the dataset are present in the research paper "Learning from Sets of Items in Recommender Systems", published in [184] The data were collected on movielens.org between February and April 2016.	6
Fifa 2018	128	58KB	It is a dataset from FIFA Predict 2018 Man of the Match found on https://www.kaggle.com/mathan/fifa-2018-match-statistic [4]	27
CCQoS1 presented by DS1, DS2, and DS3	10^2,10^3,10^4	(1.9, 20.3 and 213)KB	It is a dataset used in the previous work, and it represents Skyline outputs of RS1 [1] in the previous chapters.	4
CCQoS2 denoted by DS4, DS5, DS6, and DS7	10^5,10^6,10^7,10^8	(22.7, 237) MB and (1.7, 2.3) GB	It is a synthetic datasets generated with our proper generator created using python while simulating characteristics of CCQoS real dataset presented in the previous work maintaining the same distribution of the mentioned dataset.	4 and 12

5.5.4 performance evaluation based on runtime measurement

In this part, we analyze the runtime of the compared algorithms, respectively, while studying K's effect, i.e., the number of Top_k objects requested. Then, the impact of the dataset volume and dimensions. Moreover, we study the impact of the given weighting for different scenarios.

1. **The effect of the datasets' variation :** We use seven different datasets of varying sizes for this experiment, designated from $DS1$ to $DS7$ of Cloud Computing Quality of Service (CCQoS) data using RS1. For all scenarios, we used K = 5 in the experiment, which represents the number of K user-defined queries. The η default value is 0.001. The β default

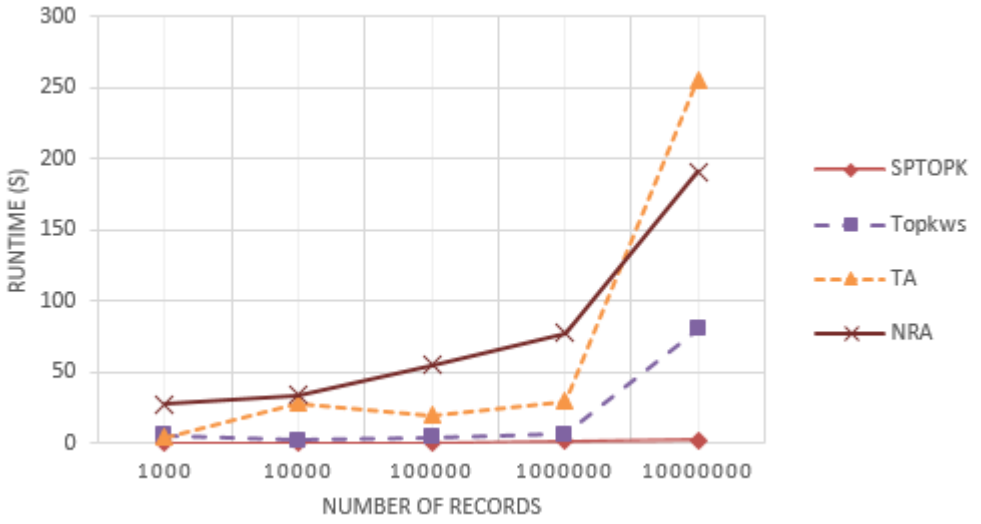


Figure 5.7: impact of the variation of the different number of records on the algorithms runtime.

value is 0.0001. The steps default value is k-fold while, k=5. 5.7, shows that the runtime of the $SPTop_{k_{WS}}$ algorithm is very short when compared to its later version and the other algorithms, namely TA and NRA. For example, in Figure 5.7, we are almost far from noticing the curve representing $SPTop_{k_{WS}}$, as it shows a minimal influence compared to the other algorithms. However, between 10000 and 1000000 records, the execution time of $Top_{k_{WS}}$, TA, and NRA increase, respectively, with the number of records. It should also be noted that this increment of the response time of the $Top_{k_{WS}}$ algorithm remains stable up to 10000000 records from which it starts to increase rapidly. This growth in time is mainly affected by the large volume of data. We evoke that our objective was to parallelize this algorithm to exploit it on a large scale. Let's immediately investigate how this algorithm works in a fully distributed environment with more massive datasets. Figure 5.8 shows the impact of large datasets on $SPTop_{k_{WS}}$, where runtime slowly increases with the data volume from DS1 to DS5. Besides, when considering more massive Datasets such as 1.7 GB (DS6) and 2.3 GB (DS7), the increase in runtime is beginning to be more significant, especially for the biggest Dataset of 17.1 GB (DS8) for which the runtime achieved more than 113 (s), due to the resources required to manage essential data in a distributed cloud infrastructure. Moreover, Much of the memory exhaustion problems were caused by processing large aggregations [**?**], which increases the time to access the score. We try to enable GPU scheduling and defining the GPU mode on hosts in our future works. Including using dynamic allocation technics and configurations for faster execution of Spark applications. Also, from another angle, we seek to address this problem by using a more refined parallel optimization of the User-Defined Aggregation Function (UDAF) in $SPTop_{k_{WS}}$.

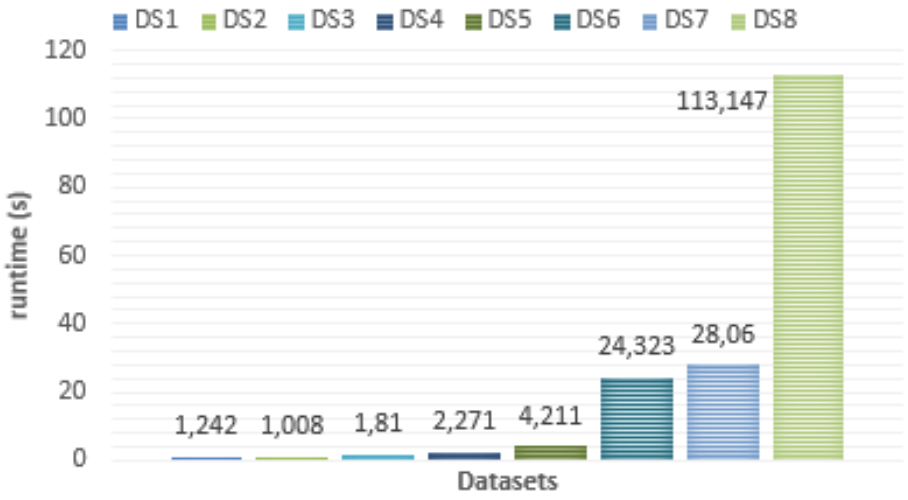


Figure 5.8: Impact of large data set on $SPTop_{k_{WS}}$ using Spark

2. **The effect of k variation**:
   With varying Top_k requests defined by the user, the proposed algorithms show the best execution time by comparing them with other algorithms based on different data. In the case of CCQoS, $SPTop_{k_{WS}}$ gets the result of the query three times faster than the other algorithms. $SPTop_{k_{WS}}$ is followed by $Top_{k_{WS}}$, NRA than TA. On the other hand, the $Top_{k_{WS}}$ runtime has similar behavior to that of NRA and TA, especially for the CCQoS DS6 data presented in Figure 5.9(a). Also, the TA experienced its worst value in K= 2, from which it starts to decrease respectively with the increase in k. This unstable performance can be explained by the main disadvantage of the sequential Top_k algorithms in a large database. When addressing the case of RDD operation in Figure 5.11, $SPTop_{k_{WS}}$ increases its runtime simultaneously by the number of k with a more monotonous attitude due to the known improvement through the Big data framework. Essentially. It is reasonable to assume that the processing time increases as k increases because all algorithms need to evaluate more data items.

3. **The effect of the number of dimensions**:
   We analyze the dimensional variation of 2 to 5 dimensions with the FIFA 2018 dataset, MovieLen dataset, and DS6 (see 5.3 for more details about the datasets). And the weighting is set with equiprobable numbers for each dimension. Related to these algorithms, $SPTop_{k_{WS}}$ followed by $Top_{k_{WS}}$ evaluate fewer Data elements, which allows them to evolve well. We can notice that they effectively reduce overhead calculation costs and significantly

exceed other algorithms when d ≥ 3 5.10(a). On the other hand, when d is small, mainly d = 2, the number of data required by $SPTop_{k_{WS}}$ to calculate its scores is small, especially for FIFA 2018, as shown in 5.10(b) and MovieLen data as showed in 5.10(c). But for TA and NRA, when we use RS1, which is the case in this part of the evaluation, the system gives results of Skyline to be managed by algorithms as a dominating query processing, which always has an influence when it comes to large dimensionality (the case of the FIFA 2018 data dimensions) 5.10(c), because in this case, the Skyline produces extreme values that are difficult to manage.

4. **The effect of weighting variations**:
   To obtain more generalized results than the previous study that uses only fixed weights, they have the same probability. We added another scenario in which the user chooses a different weight for each criterion. For example, for selecting a cloud service with four dimensions, which are respectively hard disk, latency, RAM, and price, the user can weight w = 0.25 for each criterion. In contrast, another user can have a different weight, such as 0.4 for the hard disk, 0.3 for latency, 0.2 for RAM, and 0.1 for price. On a large scale, the use of dataset (DS6) with dimensions ranging from d = 2 to d = 6 $SPTop_{k_{WS}}$ is generally faster than when we used a fixed weight starting from d = 3, as shown in 5.12(a). We notice that the execution time increases, respectively, while increasing the number of dimensions from d = 4, and it returns to decrease from d = 5. This variation in execution time is due to the increase in data size, which means that the Spark cluster requires a longer time, and the data processing rate decreases significantly. When the data size is larger than the cache size, Spark must replace the disk data with cached data, resulting in a degradation of execution time. Nevertheless, variable weights have similar incremental behavior. On the other hand, in the case of FIFA 2018 5.12(b) and MovieLen 5.12(c) with d ≥ 3, the algorithm is faster if the weighting changes; this is due to the time needed to handle variable weighting, which is even more critical than in the case of equiprobable weighting (fixed parameters w).

5.5.5 Performance evaluation based on prediction metrics:

This experience is based on the MAE and RMSE evaluation. The given results are presented respectively in 5.5, and 5.6, and 5.7. Knowing that the number of dimensions was fixed in 4 by varying the weights according to the user's choice. According to the three datasets represented in the last column of each algorithm, the mean values returned by the algorithms are returned. Accordingly, depending on the 5-fold variation, we can mention that $SPTop_{k_{WS}}$ obtained the best MAE, RMSE, Global MAE, and Global RMSE in a Cloud QoS dataset (of 10^6 items) (see Table 5.5), followed by TA (the best MAE and RMSE) and $Top_{k_{WS}}$ (the best RMSE and GlobalMAE), then the NRA. We want to mention that the average error values are indicated in bold. It is

between 38% and 66%. However, for FIFA 2018 dataset shown in Table 5.6, we noted the very closeness between the first three algorithms. Particularly $SPTop_{k_{WS}}$ (having the best value in k-fold =4, which presents a prediction accuracy of about 75% having the best errors average value in general of about 25% in k= 4) followed by $Top_{k_{WS}}$ and NRA then TA. In the MovieLens dataset presented Table 5.7, the TA algorithm and $SPTop_{k_{WS}}$ have the best and very closed according to all metrics while $SPTop_{k_{WS}}$ with a small Average of MAE and RMSE between 30% and 50%, which has its best value, followed by NRA, and it combine finally, the $Top_{k_{WS}}$. In general, except for NRA, the best benefits of algorithms are found when k-fold is between 3 and 4, which explains why these algorithms react better, increasingly with their training. Regardless, the NRA exceeds our approach's performance in some Metrics cases of the first and the second datasets, although TA sometimes takes over in the third dataset. Nevertheless, our approach's quality of algorithms remains comparable, either giving better results than that of TA and NRA in most cases, especially using large data volumes (DS5).

5.5.6 Performance evaluation based on recommendation metrics

Table 5.4: precision-versus-recall average according to the different Datasets

		Algorithms			
datasets	Metric	SPTop_{kws}	Top_{kws}	NRA	TA
Cloud QoS	AvgPrec (AP)	0.617	0.52	**0.21**	**0.26**
	AvgRecall (AR)	**0.86**	0.62	0.58	0.74
FIFA 2018	AvgPrec (AP)	0.48	0.60	0.92	0.92
	AvgRecall(AR)	0.80	0.68	**0.96**	0.84
MovieLens	AvgPrec (AP)	**0.82**	0.92	0.75	0.64
	AvgRecall (AR)	0.92	0.83	0.86	0.78

In this experiment, we evaluate the accuracy based on precision and recall measurements. They are measured on each fold. we consider k-fold number as k= 5, while the average of precision and recall are denoted respectively by (AvgPrec) and (AvgRecall) as presented in 5.4.
From this table, it can be seen that the precision of $SPTop_{k_{WS}}$ is 62% in the CCQoS dataset (DS5),48% in FIFA 2018 dataset, and 82% in MovieLens dataset. This algorithm outperforms all the other algorithms except NRA in FIFA dataset. Followed by $Top_{k_{WS}}$ that has an average value of 52% in CCQoS dataset, 60% in FIFA 2018, and finally 82% in MovieLens dataset. However, $Top_{k_{WS}}$ outperforms NRA and TA according to CCQoS dataset and in MovieLens dataset, but it is not the case according to FIFA 2018 dataset.

Now let us analyze the recall results. It can be seen that all algorithms have a good recall between 60% and 92% exept for NRA in the case of CCQoS. We remark that the given algorithms have a similar precision and recall order according to the datasets' size. The $SPTop_{k_{WS}}$ operates better in a large dataset thanks to the Spark scalability. It is followed by $Top_{k_{WS}}$. The NRA and TA have their worst precision values in a large dataset (DS5), respectively 21% and 26%. Nevertheless, in the dataset having a small size, they perform well NRA has its best value of 96% in the FIFA 2018 dataset.

Table 5.5: Performance evaluation according to CCQoS dataset and k-fold variation

Algorithm	k-fold	Evaluation metrics			
		MAE	RMSE	globalMAE	globalRMSE
SPTop_kws	1-fold	0.4529721	0.4815557	0.4529721	0.5604104
	2-fold	0.4398974	0.5019997	0.439897	0.5722369
	3-fold	0.4224660	0.4764471	0.3921200	0.4958319
	4-fold	0.3892030	0.4152863	0.3892030	0.4673963
	5-fold	0.5419766	0.5609317	0.6067095	0.6679329
	Average	**0.4493030**	**0.4872441**	**0.4561804**	**0.5527617**
Top_kws	1-fold	0.5305688	0.5623369	0.4948864	0.5737175
	2-fold	0.5572842	0.5572842	0.5572842	0.6099304
	3-fold	0.4211315	0.4335493	0.4613080	0.5719759
	4-fold	0.7776578	0.8082888	0.7786263	**0.9233383**
	5-fold	0.4491020	0.4665050	0.4256779	0.4953223
	Average	**0.5471489**	**0.5655928**	**0.5435566**	**0.6348569**
NRA	1-fold	0.8392115	0.8523775	0.7742273	**0.8626721**
	2-fold	0.6145218	0.6314360	0.6479524	0.6902829
	3-fold	0.5697128	0.5889957	0.5598569	0.6274208
	4-fold	0.3054167	0.3141103	0.3333656	0.5071920
	5-fold	0.3899480	0.4073660	0.3701223	0.5531957
	Average	**0.5437622**	**0.5588571**	**0.5371049**	**0.6481527**
TA	1-fold	0.5177311	0.5225600	0.5183760	0.6059585
	2-fold	0.7511018	0.7720389	0.6961532	0.8198908
	3-fold	0.4858580	0.4928537	0.4650058	0.5273599
	4-fold	0.4445216	0.4445222	0.3833107	0.4500604
	5-fold	0.5284641	0.5522513	0.5842491	0.6922502
	Average	**0.5455353**	**0.5568452**	**0.5294190**	**0.6191040**

Table 5.6: Performance evaluation according to FIFA dataset and k-fold variation

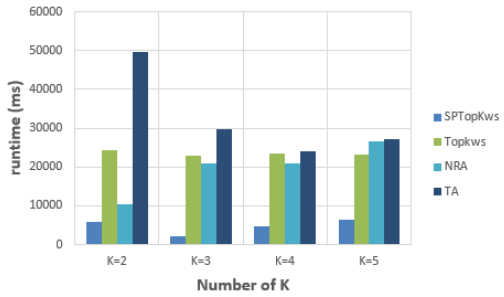
		Evaluation metrics			
Algorithm	k-fold	MAE	RMSE	globalMAE	globalRMSE
SPTopkws	1-fold	0.4935072	0.5447378	0.4935072	0.6229914
	2-fold	0.4329948	0.4446138	0.4021093	0.4839211
	3-fold	0.4591033	0.4704956	0.4095816	0.4748044
	4-fold	**0.2532359**	0.2716580	0.2587456	0.3524250
	5-fold	0.6819742	0.6827952	0.6971563	0.7135454
	Average	**0.4641631**	**0.4828601**	**0.4522200**	**0.5295375**
Top_kws	1-fold	0.4096374	0.4295725	0.4065968	0.4790669
	2-fold	0.5413649	0.5683439	0.5554488	0.6073628
	3-fold	0.5994933	0.6275359	0.5994933	0.6663208
	4-fold	**0.2801580**	0.2933537	0.3139222	0.3557099
	5-fold	0.3827683	0.4218604	0.3827683	0.4492793
	Average	**0.4426844**	**0.4681333**	**0.4516459**	**0.5115479**
NRA	1-fold	0.2727140	0.2885488	0.2727140	0.3526827
	2-fold	0.4151324	0.4378241	0.4151324	0.4636970
	3-fold	0.5479008	0.5567818	0.5479008	0.6088120
	4-fold	0.3756854	0.4471211	0.3756854	0.4805530
	5-fold	0.5399272	0.5734732	0.5399272	0.6264307
	Average	**0.4302720**	**0.4607498**	**0.4302720**	**0.5064351**
TA	1-fold	0.5191103	0.5289941	0.4500224	0.5242637
	2-fold	0.7410020	0.8278465	0.7410020	0.8546511
	3-fold	0.4744016	0.4887852	0.4186899	0.4832489
	4-fold	0.5610251	0.5884906	0.4986651	0.6028782
	5-fold	0.5094359	0.5327206	0.5291337	0.6439283
	Average	**0.5609950**	**0.5933674**	**0.5275026**	**0.6217940**

Table 5.7: Performance valuation according to MovieLens dataset and k-fold variation

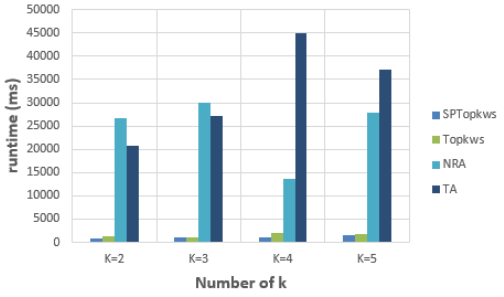
		Evaluation metrics			
Algorithm	k-fold	MAE	RMSE	globalMAE	globalRMSE
SPTopkws	1-fold	0.3294688	0.3544146	0.3294688	0.3717615
	2-fold	0.3871408	0.3979953	0.3871408	0.4621384
	3-fold	**0.2943676**	0.3261459	0.2943676	0.3568519
	4-fold	0.5132226	0.5921562	0.5132226	0.6315264
	5-fold	0.3254896	0.3429484	0.3254896	0.4195962
	Average	**0.3699379**	**0.4027321**	**0.3699379**	**0.4483749**
Topkws	1-fold	0.4748850	0.5117958	0.4748850	0.5714326
	2-fold	0.5110591	0.5437837	0.5110591	0.6094091
	3-fold	0.4798185	0.5213363	0.4798185	0.5525684
	4-fold	0.4060955	0.4690757	0.4060955	0.5226337
	5-fold	0.4981506	0.5451497	0.4981506	0.6361740
	Average	**0.4740017**	**0.5182282**	**0.4740017**	**0.5784436**
NRA	1-fold	0.3590184	0.4211800	0.3590184	0.4568762
	2-fold	0.4310697	0.4786529	0.4310697	0.5477215
	3-fold	0.4222542	0.4362956	0.4222542	0.5101311
	4-fold	0.4051761	0.4505279	0.4051761	0.4677384
	5-fold	0.3423374	0.3982740	0.3423374	0.4170881
	Average	**0.3919712**	**0.4369861**	**0.3919712**	**0.4799111**
TA	1-fold	0.3341137	0.3636124	0.3341137	0.4406306
	2-fold	0.3324039	0.3682521	0.3324039	0.4191887
	3-fold	0.3888290	0.4430565	0.3888290	0.5002412
	4-fold	0.4455065	0.4805483	0.4455065	0.5394343
	5-fold	0.3267538	0.3677336	0.3267538	0.4427215
	Average	**0.3655214**	**0.4046406**	**0.3655214**	**0.4684433**

5.6 Conclusion

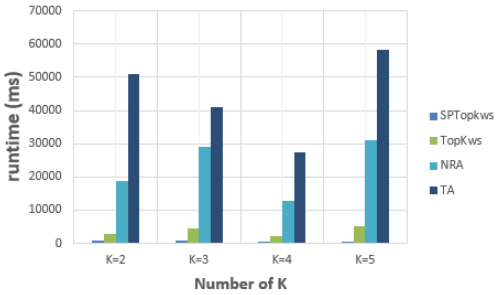
This chapter presents a new recommendation system combining the MCDA field and Top_k dominating query processing techniques, particularly the MCDA tools used to model user preferences and the collaborative filtering technique to identify the unknown elements preferred by each user. We demonstrate this proposed methodology as a distributed recommendation system and test its performance with real and synthetic data. Also, through a comparative study with other approaches using different Top_k algorithms, we prove that the paralleling of $Top_{k_{WS}}$ efficiently using the high-level programming model provided Apache Spark-based on Funk's SVD are an integral part of an improved recommendation process. Wich is also the case with Hadoop with Map-Reduce Framework. We aim to use a fully distributed recommender system while using more cluster numbers and bigger datasets as a prospect. Furthermore, we try to apply deep learning techniques rather than machine learning in the prediction stage.



(a) Cloud QoS data (DS6).

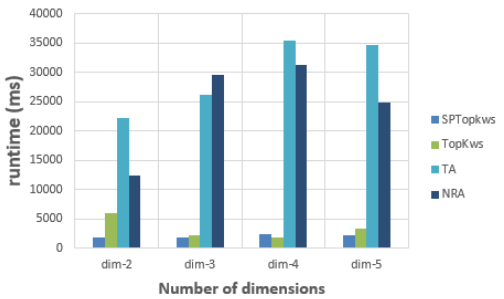


(b) FIFA data.

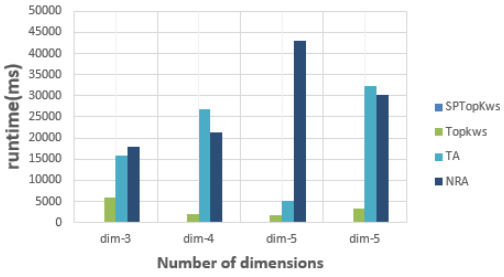


(c) MovieLen data.

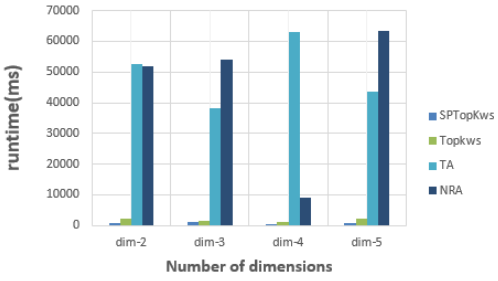
Figure 5.9: The impact of k variation on the algorithms runtime.



(a) Cloud QoS dATA (DS6).



(b) FIFA data.



(c) MovieLen data.

Figure 5.10: The impact of dimensional variation on the algorithms runtime.

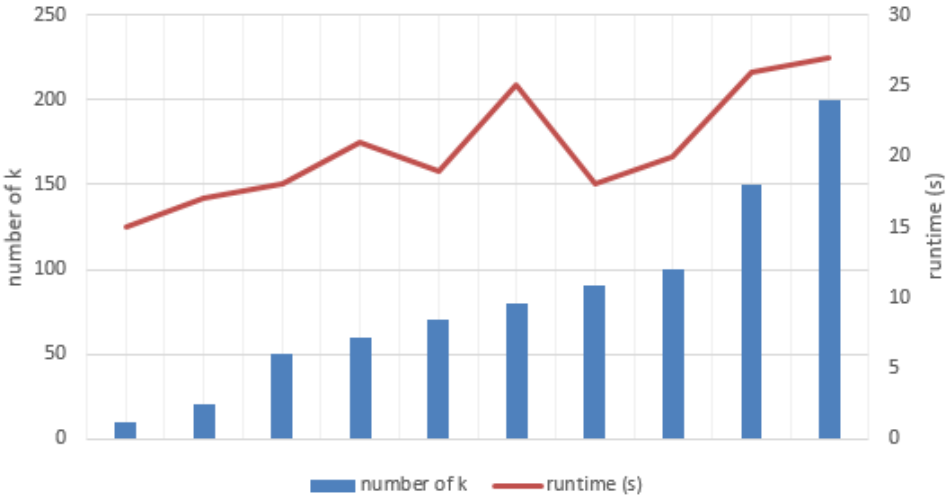
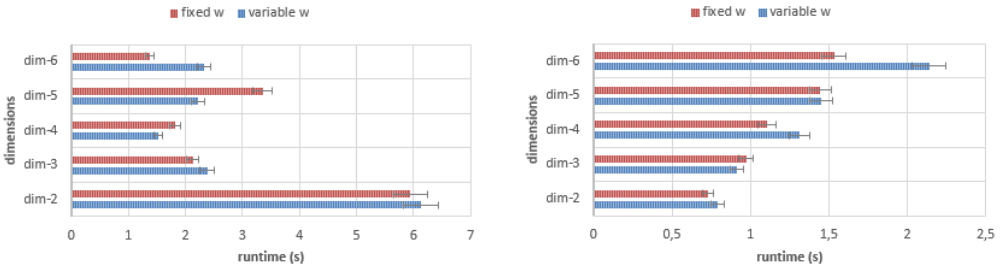
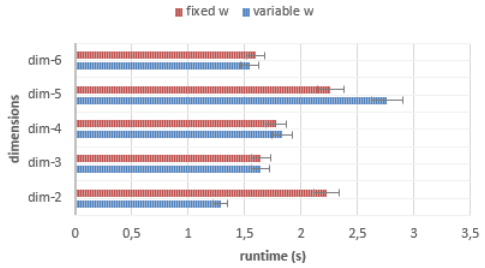


Figure 5.11: runtime of $SPTop_{k_{WS}}$ Using k variation.



(a) Cloud QoS data (DS6).

(b) FIFA data.



(c) MovieLen data.

Figure 5.12: The impact of weighting variation on the SPTop_{kws} algorithm runtime.

CHAPTER 6

GENERAL CONCLUSION AND PROSPECTS

In this chapter, we present a review of the work we have done and a set of openings and perspectives for this work. The chapter is composed of two sections the first one 6.1 is about a determination, respectively, of the given contribution. In contrast, the second section 6.2 is around the top prospects of this work.

6.1 Conclusion

With the advent of the Web and technological developments, the mass of data to be exploited and analyzed has become very voluminous. Thousands of decisions are made every day. If some of them seem self-evident, others are harder to make in the face of the deluge of data we are currently experiencing, especially when it comes to deciding on a critical area such as healthcare. The recommendation is a growing line of research to help us in this decision-making process. Indeed, decision making is a complex cognitive process that aims to select an action (or other) among different alternatives. This process is based on selection criteria, an analysis of the stakes and options, and a final choice. More concretely, one of the steps in the decision-making process is to confront a set of data and information by exploring them to make a relevant decision. Since the beginning of the nineties, with the development of Web technologies, communications, and the Internet of Things, data and information have increasingly invaded us. Eventually, when a person has to decide, they may find themselves overwhelmed by so much data and information that they will not know what to choose. Computer techniques to facilitate exploring this Big Data exist in Information Retrieval, in Business Intelligence, or Data Mining. In Business Intelligence, the means, tools, and methods allow us to collect, consolidate, model, and restore data to provide decision support and an overview of the data.

For example, data warehouses are used to collect, order, and store information from operational databases, thus providing a basis for decision support. On the other hand, data mining is about extracting knowledge from large amounts of data, i.e., finding impressive structures based on predefined criteria and raising as much experience as possible from them. While decision-making informatics enables a fact to be established and explained, data mining allows points to be classified and predicted to a certain extent or even clarified. Despite their respective interests, the adequacy to the data mining problem and their performance, the techniques used in Information Retrieval, Business Intelligence, and Data mining require decision-makers to have a minimum of information or initial knowledge as well as an idea of the direction in which to orient their data mining (sometimes limited to particular application domains).

This kind of decision aid is a recommendation. Practically is based on Computer techniques to facilitate the search and retrieval of relevant information exist. The one we focused on in this thesis is the multi-criteria recommendation based on the Top_k algorithm.

Therefore, before describing our contributions, we began in the second chapter. Introducing the main area directly related to this thesis and the notions evoked through the main stages of development that have known the location of recommendation weighing the three generations so far, as the central area of the study carried out. Then we modeled our problem by a query optimization problem based on a formulation of the recommendation problem while presenting the basic notions, state of the art, and the measures helping to evaluate the quality of our approaches.

In chapter 3, we started by presenting our first contribution, which links two main query and selection paradigms. The Skyline and Top_k justify the choice of the combination to be made and propose an improvement of a new Cloud service search and selection system thanks to the multi-criteria analysis methods discussed in the first chapter.

However, to evaluate these methods, we were led in chapter 3 to compare this approach with the different techniques that use the procedures discussed and treat the Databases in different ways of access to the score and the different types of data used, especially concerning the Top_k algorithms. For this purpose, we have proposed two contributions that compare the work with Fagin's approach using respectively three algorithms to compare our systems, which is respectively: FA TA and NAR. This work has been carried out in two other contributions. A second contribution was proposed in which we compared the same system presented in Chapter 2 with the Fagin approach. In contrast, in the third contribution, we proposed an improvement of the Top_k algorithm by the Top_kws algorithm, which is based on an adapted MCDA method while improving the global system by a Generic Search and Selection System, which was applied on synthetic and real Databases. The comparison was studied using response time and correlation

measures to investigate the quality of the results found and to evaluate the new recommendation system.

Despite the performances achieved by our approaches, we found that we are still confronted by the vast mass of data generated by the Big data phenomenon, which requires special processing based on new modeling and infrastructure. To solve this problem, in the fourth chapter, we have modeled our proposed algorithm in a parallel and distributed context based on the Map-Reduce and Pig Framework to use parallelism and exploit the data. If we solved the problem of parallelism and massive data management, or even more, we proposed the management of data in real-time, thanks to Pig. But other limitations are still existing and are limitations of the second generation of recommendation systems, such as the problem of cold start and hollow matrix generated by the users who are, in our case, decision-makers of a decision-making system. For the fifth contribution, we have parallelized Spark's algorithm as a new Big data technology. This choice was because the spark tool outperforms Hadoop Map-Reduce and helps form the MLib library that allows us to use collaborative filtering methods and very sophisticated machines learning that solve the Gold start problem. The improvements proposed in this thesis, both in terms of algorithms and data, have been exploited in specific areas such as Cloud Computing in chapter 3 and sports management and Cloud Computing in chapter 4, and the two other domains plus data and movies chapter 5.

6.2 Prospects

- We proposed a multi-criteria recommendation system based on MCDA methods. It would be interesting to improve the procedures that we have chosen to hybridize with the Top_k algorithms by other MCDA methods more adapted to the new proposed system. The MCDA methods, such as the TOPSIS Fazy TOPSIS methods, can encourage results because using a generalization of bi-objective formalization with other optimization structures to have a multi-objective formalization. This formalization can be generalized to other optimization structures.

- Secondly, the use of advanced automatic learning techniques, such as the canonical correlation analysis method, Gradient boosting, knowing that it will be an improved version of Gradient descent proposed with the Funk SVD model.
  Using other models to achieve this matrix factorization approach compared with Funk SVD and solving Cold-Start Problem with different Matrix Factorization techniques for the prediction phase, for extending our work presented in [49].

- The case of the processing of parallel or distributed queries: We have studied a parallelization of Top_k queries in a centralized and distributed context. It would be judicious to adapt the proposed approaches which combine Top_k with the skyline in the framework

of parallel or distributed databases. We issued especially for the Skyline query processing phase because it is already made for Top_k query processing.

- Using deep learning techniques for the machine learning phase instead of machine learning methods used in our approach.

BIBLIOGRAPHY

[1] K. E. Handri and A. Idrissi, "Comparative study of topk based on fagin's algorithm using correlation metrics in cloud computing qos," *International Journal of Internet Technology and Secured Transactions*, vol. 10, no. 1-2, pp. 143–170, 2020.

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge & Data Engineering*, no. 6, pp. 734–749, 2005.

[3] I. F. Ilyas, G. Beskales, and M. A. Soliman, "A survey of top-k query processing techniques in relational database systems," *ACM Computing Surveys (CSUR)*, vol. 40, no. 4, pp. 1–58, 2008.

[4] F. A. 2018-08-03, "Predict fifa 2018 man of the match, www.kaggle.com/mathan/fifa-018-match-statistics."

[5] I. T. R. Institute, "big data kernel description," 2019.

[6] R. E. Hostler, V. Y. Yoon, Z. Guo, T. Guimaraes, and G. Forgionne, "Assessing the impact of recommender agents on on-line consumer unplanned purchase behavior," *Information & Management*, vol. 48, no. 8, pp. 336–343, 2011.

[7] S. Schelter, A. Palumbo, S. Quinn, S. Marthi, and A. Musselman, "Samsara: Declarative machine learning on distributed dataflow systems," in *NIPS Workshop MLSystems*, 2016.

[8] Splunk, "Les données encore trop peu utilisées pour prendre des décisions, www.lemondeinformatique.fr/actualites/lire-sante-le-chantier-monumental-de-la-convergence-des-si-hospitaliers-77852.html."

[9] A. Podviezko, "Use of multiple criteria decision aid methods in case of large amounts of data," *International Journal of Business and Emerging Markets*, vol. 7, no. 2, pp. 155–169, 2015.

[10] W. Y. C. Wang and Y. Wang, "Analytics in the era of big data: The digital transformations and value creation in industrial marketing," 2020.

[11] Statista, “Data created worldwide 2010-2025, www.statista.com/statistics/871513/worldwide-data-created.html.”

[12] S. Gault, *Improving MapReduce Performance on Clusters*.
PhD thesis, 2015.

[13] M. Meeker and L. Wu, “Internet trends 2018,” 2018.

[14] G. du Big Data, “2016. l’annuaire de référence à destination des utilisateurs,” 2015.

[15] N. Khan, M. Alsaqer, H. Shah, G. Badsha, A. A. Abbasi, and S. Salehian, “The 10 vs, issues and challenges of big data,” in *Proceedings of the 2018 International Conference on Big Data and Education*, pp. 52–56, 2018.

[16] A. M. S. Osman, “A novel big data analytics framework for smart cities,” *Future Generation Computer Systems*, vol. 91, pp. 620–633, 2019.

[17] M. F. Uddin, N. Gupta, *et al.*, “Seven v’s of big data understanding big data to extract value,” in *Proceedings of the 2014 zone 1 conference of the American Society for Engineering Education*, pp. 1–5, IEEE, 2014.

[18] I. Yaqoob, I. A. T. Hashem, A. Gani, S. Mokhtar, E. Ahmed, N. B. Anuar, and A. V. Vasilakos, “Big data: From beginning to future,” *International Journal of Information Management*, vol. 36, no. 6, pp. 1231–1247, 2016.

[19] F. Tekiner and J. A. Keane, “Big data framework,” in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1494–1499, IEEE, 2013.

[20] D. E. O’Leary, “Artificial intelligence and big data,” *IEEE intelligent systems*, vol. 28, no. 2, pp. 96–99, 2013.

[21] C. Lam, *Hadoop in action*.
Manning Publications Co., 2010.

[22] H. Karau, A. Konwinski, P. Wendell, and M. Zaharia, *Learning spark: lightning-fast big data analysis*.
" O’Reilly Media, Inc.", 2015.

[23] T. White, *Hadoop: The definitive guide*.
" O’Reilly Media, Inc.", 2012.

[24] H. Guo, M. F. Goodchild, and A. Annoni, *Manual of Digital Earth*.
Springer Nature, 2020.

[25] V. Rubin and T. Lukoianova, “Veracity roadmap: Is big data objective, truthful and credible?,” *Advances in Classification Research Online*, vol. 24, no. 1, p. 4, 2013.

[26] E. A. Balas, M. Vernon, F. Magrabi, L. T. Gordon, J. Sexton, *et al.*, "Big data clinical research: Validity, ethics, and regulation.," in *MedInfo*, pp. 448–452, 2015.

[27] A. Kumar, S. Dubey, M. Arshad, S. Saxena, S. K. Sinha, P. Dixit, and A. Arjaria, "Enhanced cloud data storage security by using hadoop," *Available at SSRN 3564061*, 2020.

[28] L. W. Cong and Z. He, "Blockchain disruption and smart contracts," *The Review of Financial Studies*, vol. 32, no. 5, pp. 1754–1797, 2019.

[29] F. Almeida, "Big data: Concept, potentialities and vulnerabilities," *Emerging Science Journal*, vol. 2, no. 1, pp. 1–10, 2018.

[30] B. ÖzÇakmak, A. ÖzbİIen, U. YavanoĞlu, and K. CİIn, "Neural and quantum cryptography in big data: a review," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 2413–2417, IEEE, 2019.

[31] A. Lambshead, "The importance of standards in a time of innovation [report from the standards vice president]," *SMPTE Motion Imaging Journal*, vol. 123, no. 8, pp. 7–7, 2014.

[32] A. S. Fiaz, N. Asha, D. Sumathi, and A. S. Navaz, "Data visualization: Enhancing big data more adaptable and valuable," *International Journal of Applied Engineering Research*, vol. 11, no. 4, pp. 2801–2804, 2016.

[33] R. S. Segall and G. Niu, "Big data and its visualization with fog computing," in *Cognitive Analytics: Concepts, Methodologies, Tools, and Applications*, pp. 341–377, IGI Global, 2020.

[34] J. V. Gautam, H. B. Prajapati, V. K. Dabhi, and S. Chaudhary, "A survey on job scheduling algorithms in big data processing," in *2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–11, IEEE, 2015.

[35] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer, "Enterprise data analysis and visualization: An interview study," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2917–2926, 2012.

[36] M. Mahrt and M. Scharkow, "The value of big data in digital media research," *Journal of Broadcasting & Electronic Media*, vol. 57, no. 1, pp. 20–33, 2013.

[37] B. Kayyali, D. Knott, and S. Van Kuiken, "The big-data revolution in us health care: Accelerating value and innovation," *Mc Kinsey & Company*, vol. 2, no. 8, pp. 1–13, 2013.

[38] J. Berryhill, K. K. Heang, R. Clogher, and K. McBride, "Hello, world: Artificial intelligence and its use in the public sector," 2019.

[39] M. Kaptein and P. Ketelaar, "Maximum likelihood estimation of a finite mixture of logistic regression models in a continuous data stream," *arXiv preprint arXiv:1802.10529*, 2018.

[40] K. Tannir, *Optimizing Hadoop for MapReduce*.
Packt Publishing Ltd, 2014.

[41] P. Riyaz and S. M. Varghese, "A scalable product recommendations using collaborative filtering in hadoop for bigdata," *Procedia Technology*, vol. 24, pp. 1393–1399, 2016.

[42] Q. Zhang, J. Lu, and Y. Jin, "Artificial intelligence in recommender systems," *Complex & Intelligent Systems*, pp. 1–19, 2020.

[43] M. Leyva-Vázquez, F. Smarandache, and J. E. Ricardo, "Artificial intelligence: challenges, perspectives and neutrosophy role.(master conference)," *Dilemas Contemporáneos: Educación, Política y Valore*, vol. 6, no. Special, 2018.

[44] Z. Zhang, L. Zhang, S. Croll, and M. Chopp, "Angiopoietin-1 reduces cerebral blood vessel leakage and ischemic lesion volume after focal cerebral embolic ischemia in mice," *Neuroscience*, vol. 113, no. 3, pp. 683–687, 2002.

[45] J. Pino, J. Dreiling, C. Figgatt, J. Gaebler, S. Moses, C. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, *et al.*, "Demonstration of the qccd trapped-ion quantum computer architecture," *arXiv preprint arXiv:2003.01293*, 2020.

[46] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[47] K. El Handri and A. Idrissi, "Parallelization of topk algorithm through a new hybrid recommendation system for big data in spark cloud computing framework," *IEEE Systems Journal*, 2020.

[48] M. Rousset and L. Boudjeloud-Assala, eds., *Extraction et Gestion des connaissances, EGC 2019, Metz, France, January 21-25, 2019*, vol. E-35 of *RNTI*, Hermann-Éditions, 2019.

[49] k. EL handri and A. Idrissi, "Correlations and hierarchical clustering investiga-tion between weather and sars-cov-2," *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science)*, 2020.

[50] A. Idrissi, K. Elhandri, H. Rehioui, and M. Abourezq, "Top-k and skyline for cloud services research and selection system," in *Proceedings of the International Conference on Big Data and Advanced Wireless Technologies*, p. 40, ACM, 2016.

[51] K. El Handri and A. Idrissi, "Étude comparative de topk basée sur l'algorithme de fagin en utilisant des métriques de corrélation dans la qualité de service de cloud computing.," in *EGC*, pp. 359–360, 2019.

[52] G. Adomavicius and J. Zhang, "Improving stability of recommender systems: a meta-algorithmic approach," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 6, pp. 1573–1587, 2014.

[53] M. Pazzani and D. Billsus, "Learning and revising user profiles: The identification of interesting web sites," *Machine learning*, vol. 27, no. 3, pp. 313–331, 1997.

[54] J. Delgado and N. Ishii, "Memory-based weighted majority prediction," in *SIGIR Workshop Recomm. Syst. Citeseer*, p. 85, Citeseer, 1999.

[55] X. N. Lam, T. Vu, T. D. Le, and A. D. Duong, "Addressing cold-start problem in recommendation systems," in *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pp. 208–211, 2008.

[56] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.

[57] A. Hofmann, "Sampling mantle heterogeneity through oceanic basalts: isotopes and trace elements," *TrGeo*, vol. 2, p. 568, 2003.

[58] L. Si and R. Jin, "Flexible mixture model for collaborative filtering," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 704–711, 2003.

[59] T. Tran and R. Cohen, "Hybrid recommender systems for electronic commerce," in *Proc. Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, Technical Report WS-00-04, AAAI Press*, vol. 40, 2000.

[60] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," *Aaai/iaai*, vol. 23, pp. 187–192, 2002.

[61] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Generative models for cold-start recommendations," in *Proceedings of the 2001 SIGIR workshop on recommender systems*, vol. 6, 2001.

[62] C. A. CHARU, *RECOMMENDER SYSTEMS: The Textbook*.
SPRINGER, 2018.

[63] K. Wong, "Final year project report," *University of Glasgow*, 1990.

[64] C. C. A. I. T. Watson *et al.*, *Recommender Systems: The Textbook*.
Buku Digital, 2016.

[65] J. Mothe, Y. Pitarch, and E. Gaussier, "Big data: le cas des systèmes d'information," *Ingénierie des Systèmes d'Information*, vol. 19, no. 3, pp. 9–48, 2014.

[66] H.-H. Lee and W.-G. Teng, "Incorporating multi-criteria ratings in recommendation systems," in *2007 IEEE International Conference on Information Reuse and Integration*, pp. 273–278, IEEE, 2007.

[67] D. Bokde, S. Girase, and D. Mukhopadhyay, "Matrix factorization model in collaborative filtering algorithms: A survey," *Procedia Computer Science*, vol. 49, pp. 136–146, 2015.

[68] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proceedings of KDD cup and workshop*, vol. 2007, pp. 5–8, 2007.

[69] S. Funk, "Netflix update: Try this at home," 2006.

[70] X. Guo, S.-C. Yin, Y.-W. Zhang, W. Li, and Q. He, "Cold start recommendation based on attribute-fused singular value decomposition," *IEEE Access*, vol. 7, pp. 11349–11359, 2019.

[71] Y. Zheng, "Situation-aware multi-criteria recommender system: using criteria preferences as contexts," in *Proceedings of the Symposium on Applied Computing*, pp. 689–692, 2017.

[72] Y. Zheng, "Criteria chains: a novel multi-criteria recommendation approach," in *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pp. 29–33, 2017.

[73] L. Chen, M. De Gemmis, A. Felfernig, P. Lops, F. Ricci, and G. Semeraro, "Human decision making and recommender systems," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 3, no. 3, pp. 1–7, 2013.

[74] S.-H. Chen and W.-T. Lin, "Analyzing determinants for promoting emerging technology through intermediaries by using a danp-based mcda framework," *Technological Forecasting and Social Change*, vol. 131, pp. 94–110, 2018.

[75] E. Løken, "Use of multicriteria decision analysis methods for energy planning problems," *Renewable and sustainable energy reviews*, vol. 11, no. 7, pp. 1584–1595, 2007.

[76] F. Helff, L. Gruenwald, and L. d'Orazio, "Weighted sum model for multi-objective query optimization for mobile-cloud database environments.," in *EDBT/ICDT Workshops*, 2016.

[77] A. Jaszkiewicz and J. Branke, "Interactive multiobjective evolutionary algorithms," in *Multiobjective optimization*, pp. 179–193, Springer, 2008.

[78] A. Kumar, B. Sah, A. R. Singh, Y. Deng, X. He, P. Kumar, and R. Bansal, "A review of multi criteria decision making (mcdm) towards sustainable renewable energy development," *Renewable and Sustainable Energy Reviews*, vol. 69, pp. 596–609, 2017.

[79] J. Guntzer, W.-T. Balke, and W. Kießling, "Towards efficient multi-feature queries in heterogeneous environments," in *Proceedings International Conference on Information Technology: Coding and Computing*, pp. 622–628, IEEE, 2001.

[80] A. Nafi and C. Werey, "Aide à la décision multicritère: introduction aux méthodes d'analyse multicritère de type electre," *Module d'ingénierie financière, ENGEES*, vol. 2010, 2009.

[81] G. Das, D. Gunopulos, N. Koudas, and D. Tsirogiannis, "Answering top-k queries using views," in *Proceedings of the 32nd international conference on Very large data bases*, pp. 451–462, 2006.

[82] J. Figueira, S. Greco, and M. Ehrgott, *Multiple criteria decision analysis: state of the art surveys*, vol. 78.
Springer Science & Business Media, 2005.

[83] B. Roy, "The outranking approach and the foundations of electre methods," in *Readings in multiple criteria decision aid*, pp. 155–183, Springer, 1990.

[84] J. C. Leyva-Lopez and E. Fernandez-Gonzalez, "A new method for group decision support based on electre iii methodology," *European journal of operational research*, vol. 148, no. 1, pp. 14–27, 2003.

[85] M. Marzouk, "Electre iii model for value engineering applications," *Automation in Construction*, vol. 20, no. 5, pp. 596–600, 2011.

[86] R. Akbarinia, E. Pacitti, and P. Valduriez, "Best position algorithms for top-k queries," in *Proceedings of the 33rd international conference on Very large data bases*, pp. 495–506, VLDB Endowment, 2007.

[87] S. Michel, P. Triantafillou, and G. Weikum, "Klee: A framework for distributed top-k query algorithms," in *Proceedings of the 31st international conference on Very large data bases*, pp. 637–648, VLDB Endowment, 2005.

[88] R. Fagin, R. Kumar, and D. Sivakumar, "Comparing top k lists," *SIAM Journal on discrete mathematics*, vol. 17, no. 1, pp. 134–160, 2003.

[89] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and multidisciplinary optimization*, vol. 41, no. 6, pp. 853–862, 2010.

[90] S.-w. Hwang and K. C.-c. Chang, "Optimizing top-k queries for middleware access: A unified cost-based approach," *ACM Transactions on Database Systems (TODS)*, vol. 32, no. 1, p. 5, 2007.

[91] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," *Journal of computer and system sciences*, vol. 66, no. 4, pp. 614–656, 2003.

[92] N. Bruno, L. Gravano, and A. Marian, "Evaluating top-k queries over web-accessible databases," in *Data Engineering, 2002. Proceedings. 18th International Conference on*, pp. 369–380, IEEE, 2002.

[93] E. Triantaphyllou, "Multi-criteria decision making methods," in *Multi-criteria decision making methods: A comparative study*, pp. 5–21, Springer, 2000.

[94] M. Abourezq and A. Idrissi, "Introduction of an outranking method in the cloud computing research and selection system based on the skyline," in *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, pp. 1–12, IEEE, 2014.

[95] A. Idrissi and M. Abourezq, "Skyline in cloud computing," *Journal of Theoretical and Applied Information Technology*, vol. 60, no. 3, 2014.

[96] N. Bruno, S. Chaudhuri, and L. Gravano, "Top-k selection queries over relational databases: Mapping strategies and performance evaluation," *ACM Transactions on Database Systems (TODS)*, vol. 27, no. 2, pp. 153–187, 2002.

[97] Y.-C. Chang, L. Bergman, V. Castelli, C.-S. Li, M.-L. Lo, and J. R. Smith, "The onion technique: indexing for linear optimization queries," in *ACM Sigmod Record*, vol. 29, pp. 391–402, ACM, 2000.

[98] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid, "Supporting top-k join queries in relational databases," *The VLDB Journal—The International Journal on Very Large Data Bases*, vol. 13, no. 3, pp. 207–221, 2004.

[99] I. F. Ilyas, W. G. Aref, A. K. Elmagarmid, H. G. Elmongui, R. Shah, and J. S. Vitter, "Adaptive rank-aware query optimization in relational databases," *ACM Transactions on Database Systems (TODS)*, vol. 31, no. 4, pp. 1257–1304, 2006.

[100] C.-M. Chen and Y. Ling, "A sampling-based estimator for top-k selection query," in *Proceedings 18th International Conference on Data Engineering*, pp. 617–627, IEEE, 2002.

[101] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," in *Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '01, (New York, NY, USA), pp. 102–113, ACM, 2001.

[102] A. Vlachou, C. Doulkeridis, and K. Nørvåg, "Distributed top-k query processing by exploiting skyline summaries," *Distributed and Parallel Databases*, vol. 30, no. 3-4, pp. 239–271, 2012.

[103] K. El handri and A. Idrissi, "Comparative study of topk based on fagins algorithm using correlation metrics in cloud computing qos, 2018," *International Journal of Internet Technology and Secured Transactions (IJITST), (in press)*, 2018.

[104] D. Amagata, T. Hara, and M. Onizuka, "Space filling approach for distributed processing of top-k dominating queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1150–1163, 2018.

[105] A. R. H. Witwit, *Non-weighted aggregate evaluation function of multi-objective optimization for knock engine modeling*.
PhD thesis, Universiti Utara Malaysia, 2017.

[106] A. Marian, N. Bruno, and L. Gravano, "Evaluating top-k queries over web-accessible databases," *ACM Transactions on Database Systems (TODS)*, vol. 29, no. 2, pp. 319–362, 2004.

[107] U. Güntzer, W.-T. Balke, and W. Kießling, "Optimizing multi-feature queries for image databases," 2000.

[108] M. Badr and D. Vodislav, "A general top-k algorithm for web data sources," in *International Conference on Database and Expert Systems Applications*, pp. 379–393, Springer, 2011.

[109] K. C.-C. Chang and S.-w. Hwang, "Minimal probing: supporting expensive predicates for top-k queries," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pp. 346–357, ACM, 2002.

[110] A. Natsev, Y.-C. Chang, J. R. Smith, C.-S. Li, and J. S. Vitter, "Supporting incremental join queries on ranked inputs," in *VLDB*, vol. 1, pp. 281–290, 2001.

[111] V. Hristidis, N. Koudas, and Y. Papakonstantinou, "Prefer: A system for the efficient execution of multi-parametric ranked queries," *ACM Sigmod Record*, vol. 30, no. 2, pp. 259–270, 2001.

[112] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid, "Joining ranked inputs in practice," in *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pp. 950–961, Elsevier, 2002.

[113] C. Li, K. C.-C. Chang, I. F. Ilyas, and S. Song, "Ranksql: query algebra and optimization for relational top-k queries," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pp. 131–142, 2005.

[114] C. Li, K. Chen-Chuan Chang, and I. F. Ilyas, "Supporting ad-hoc ranking aggregates," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 61–72, 2006.

[115] M. Theobald, R. Schenkel, and G. Weikum, "An efficient and versatile query engine for topx search," in *Proceedings of the 31st international conference on Very large data bases*, pp. 625–636, 2005.

[116] Z. Zhang, S.-w. Hwang, K. C.-C. Chang, M. Wang, C. A. Lang, and Y.-c. Chang, "Boolean+ ranking: querying a database by k-constrained optimization," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 359–370, 2006.

[117] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang, "Top-k query processing in uncertain databases," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 896–905, IEEE, 2007.

[118] C. Re, N. Dalvi, and D. Suciu, "Efficient top-k query evaluation on probabilistic data," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 886–895, IEEE, 2007.

[119] M. S. H. Im, "Parallel top-k query processing using mapreduce,"

[120] X. Li, Y. Wang, X. Li, and Y. Wang, "Parallel skyline queries over uncertain data streams in cloud computing environments," *International Journal of Web and Grid Services*, vol. 10, no. 1, pp. 24–53, 2014.

[121] A. Shanbhag, H. Pirk, and S. Madden, "Efficient top-k query processing on massively parallel hardware," in *Proceedings of the 2018 International Conference on Management of Data*, pp. 1557–1570, ACM, 2018.

[122] X. Ding, C. Yan, and Y. Zhao, "Parallel processing of top-k dominating queries on incomplete data," in *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, pp. 1785–1791, IEEE, 2018.

[123] M. Saouk, C. Doulkeridis, A. Vlachou, *et al.*, "Efficient processing of top-k joins in mapreduce," in *Big Data (Big Data), 2016 IEEE International Conference on*, pp. 570–577, IEEE, 2016.

[124] S. P. Ahuja and B. Moore, "State of big data analysis in the cloud," *Network and Communication Technologies*, vol. 2, no. 1, p. 62, 2013.

[125] C. Ji, Y. Li, W. Qiu, U. Awada, and K. Li, "Big data processing in cloud computing environments," in *Pervasive Systems, Algorithms and Networks (ISPAN), 2012 12th International Symposium on*, pp. 17–23, IEEE, 2012.

[126] A. Saxena, A. Chaurasia, N. Kaushik, and N. Kaushik, "Handling big data using mapreduce over hybrid cloud," in *International Conference on Innovative Computing and Communications*, pp. 135–144, Springer, 2019.

[127] K. Meena and J. Sujatha, "Reduced time compression in big data using mapreduce approach and hadoop," *Journal of Medical Systems*, vol. 43, no. 8, p. 239, 2019.

[128] J. Chen, K. Li, Z. Tang, K. Bilal, S. Yu, C. Weng, and K. Li, "A parallel random forest algorithm for big data in a spark cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 919–933, 2016.

[129] S. Panigrahi, R. K. Lenka, and A. Stitipragyan, "A hybrid distributed collaborative filtering recommender engine using apache spark," *Procedia Computer Science*, vol. 83, pp. 1000–1006, 2016.

[130] R. Kumar and S. Vassilvitskii, "Generalized distances between rankings," in *Proceedings of the 19th international conference on World wide web*, pp. 571–580, ACM, 2010.

[131] G. Saporta, *Probabilités, analyse des données et statistique*.
Editions Technip in Paris, French, 2006.

[132] S.-D. Bolboaca and L. Jäntschi, "Pearson versus spearman, kendall's tau correlation analysis on structure-activity relationships of biologic active compounds," *Leonardo Journal of Sciences*, vol. 5, no. 9, pp. 179–200, 2006.

[133] M. Junker, R. Hoch, and A. Dengel, "On the evaluation of document analysis components by recall, precision, and accuracy," in *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318)*, pp. 713–716, IEEE, 1999.

[134] T. Zain, M. Aslam, M. Imran, and A. Martinez-Enriquez, "Cloud service recommender system using clustering," in *2014 11th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pp. 1–6, IEEE, 2014.

[135] L. Columbus, "Roundup of cloud computing forecasts and market estimates, 2016," *Retrieved May*, vol. 1, p. 2019, 2016.

[136] C. C. Aggarwal, "An introduction to recommender systems," in *Recommender systems*, pp. 1–28, Springer, 2016.

[137] P. Mell, T. Grance, *et al.*, "The nist definition of cloud computing," 2011.

[138] R. Karim, C. Ding, A. Miri, and M. S. Rahman, "Incorporating service and user information and latent features to predict qos for selecting and recommending cloud service compositions," *Cluster Computing*, vol. 19, no. 3, pp. 1227–1242, 2016.

[139] M. Spruit, R. Vroon, and R. Batenburg, "Towards healthcare business intelligence in long-term care: an explorative case study in the netherlands," *Computers in Human Behavior*, vol. 30, pp. 698–707, 2014.

[140] L. Columbus, "Roundup of cloud computing forecasts, 2017," *Forbes, April 29th*, 2017.

[141] M. Abourezq and A. Idrissi, "A cloud services research and selection system," in *2014 International Conference on Multimedia Computing and Systems (ICMCS)*, pp. 1195–1199, IEEE, 2014.

[142] A. Idrissi, H. Rehioui, A. Laghrissi, and S. Retal, "An improvement of denclue algorithm for the data clustering," in *2015 5th International Conference on Information & Communication Technology and Accessibility (ICTA)*, pp. 1–6, IEEE, 2015.

[143] M. Ehrgott, "Multiobjective optimization," *Ai Magazine*, vol. 29, no. 4, pp. 47–47, 2008.

[144] S. Greco, J. Figueira, and M. Ehrgott, *Multiple criteria decision analysis*. Springer, 2016.

[145] R. Caillet *et al.*, "Analyse multicritère: Étude de comparaison des méthodes existantes en vue d'une application en analyse de cycle de vie," tech. rep., CIRANO, 2003.

[146] A. Vlachou, C. Doulkeridis, K. Norvag, and M. Vazirgiannis, "Skyline-based peer-to-peer top-k query processing," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pp. 1421–1423, IEEE, 2008.

[147] A. Vlachou, C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis, "On efficient top-k query processing in highly distributed environments," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pp. 753–764, ACM, 2008.

[148] S. S. Bhowmick, C. E. Dyreson, C. S. Jensen, M. L. Lee, A. Muliantara, and B. Thalheim, "Database systems for advanced applications 19th international conference, dasfaa 2014, bali, indonesia, april 21-24, 2014. proceedings, part ii," in *Conference proceedings DASFAA*, p. 22, Springer, 2014.

[149] M. Abourezq and A. Idrissi, "Database-as-a-service for big data: An overview," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 7, no. 1, 2016.

[150] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "An optimal and progressive algorithm for skyline queries," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pp. 467–478, ACM, 2003.

[151] A. Hadjali, O. Pivert, and H. Prade, "On different types of fuzzy skylines," in *International Symposium on Methodologies for Intelligent Systems*, pp. 581–591, Springer, 2011.

[152] C.-Y. Chan, H. Jagadish, K.-L. Tan, A. K. Tung, and Z. Zhang, "Finding k-dominant skylines in high dimensional space," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 503–514, ACM, 2006.

[153] C.-Y. Chan, H. Jagadish, K.-L. Tan, A. K. Tung, and Z. Zhang, "On high dimensional skylines," in *International Conference on Extending Database Technology*, pp. 478–495, Springer, 2006.

[154] E. Hüllermeier, I. Vladimirskiy, B. P. Suárez, and E. Stauch, "Supporting case-based retrieval by similarity skylines: Basic concepts and extensions," in *European Conference on Case-Based Reasoning*, pp. 240–254, Springer, 2008.

[155] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang, "Selecting stars: The k most representative skyline operator," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pp. 86–95, IEEE, 2007.

[156] D. Ben-Arieh, "Multi-criteria decision making methods: A comparative study," 2002.

[157] K. K. Kaboré, F. Sèdes, O. Sié, and A. Peninou, "Implementing the information access assistant service for an evaluation," *International Journal of Internet Technology and Secured Transactions*, vol. 6, no. 1, pp. 25–42, 2015.

[158] P. Jomsri, S. Sanguansintukul, and W. Choochaiwattana, "Citerank: combination similarity and static ranking with research paper searching," *International Journal of Internet Technology and Secured Transactions*, vol. 3, no. 2, pp. 161–177, 2011.

[159] D. Ruan, G. Chen, E. E. Kerre, and G. Wets, *Intelligent data mining: techniques and applications*, vol. 5.
Springer Science & Business Media, New Rochelles NY, 2005.

[160] N. M. Razali, Y. B. Wah, *et al.*, "Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests," *Journal of statistical modeling and analytics*, vol. 2, no. 1, pp. 21–33, 2011.

[161] R. L. Wasserstein, N. A. Lazar, *et al.*, "The asa's statement on p-values: context, process, and purpose," *The American Statistician*, vol. 70, no. 2, pp. 129–133, 2016.

[162] A. Idrissi and F. Yakine, "Multicast routing with quality of service constraints in the ad hoc wireless networks," *Journal of Computer Science*, vol. 10, no. 9, pp. 1839–1849, 2014.

[163] M. Abourezq, A. Idrissi, and F. Yakine, "Routing in wireless ad hoc networks using the skyline operator and an outranking method," in *Proceedings of the International Conference on Internet of things and Cloud Computing*, p. 37, ACM, 2016.

[164] S. Burer, "Robust rankings for college football," *Journal of Quantitative Analysis in Sports*, vol. 8, no. 2, 2012.

[165] M. Saeed, M. Saqlain, and M. Riaz, "Application of generalized fuzzy topsis in decision making for neutrosophic soft set to predict the champion of fifa 2018: A mathematical analysis," *Journal of Mathematics (ISSN 1016-2526)*, vol. 51, no. 8, pp. 111–126, 2019.

[166] E. Ott, N. Hayashi, and M. Fukuda, "Method and system for using smart tags and a recommendation engine using smart tags," Apr. 5 2007.
US Patent App. 11/424,966.

[167] B. Vaziri, S. Dabadghao, Y. Yih, and T. L. Morin, "Properties of sports ranking methods," *Journal of the operational research society*, vol. 69, no. 5, pp. 776–787, 2018.

[168] J. Liu, L. Xiong, J. Pei, J. Luo, H. Zhang, and W. Yu, "Group-based skyline for pareto optimal groups," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[169] A. K. Kalavagattu, A. S. Das, K. Kothapalli, and K. Srinathan, "On finding skyline points for range queries in plane.," in *CCCG*, 2011.

[170] J. Liu, L. Xiong, J. Pei, J. Luo, and H. Zhang, "Finding pareto optimal groups: Group-based skyline," *Proceedings of the VLDB Endowment*, vol. 8, no. 13, pp. 2086–2097, 2015.

[171] E. Tiakas, G. Valkanas, A. N. Papadopoulos, Y. Manolopoulos, and D. Gunopulos, "Processing top-k dominating queries in metric spaces," *ACM Transactions on Database Systems (TODS)*, vol. 40, no. 4, p. 23, 2016.

[172] E. Wheatcroft, "Forecasting football matches by predicting match statistics," *arXiv preprint arXiv:2001.09097*, 2020.

[173] G.-H. Tzeng and J.-J. Huang, *Multiple attribute decision making: methods and applications*. CRC press, 2011.

[174] Y.-J. Lai, T.-Y. Liu, and C.-L. Hwang, "Topsis for modm," *European journal of operational research*, vol. 76, no. 3, pp. 486–500, 1994.

[175] Y. Chen, "Multiple criteria decision analysis: classification problems and solutions," 2006.

[176] P. Thakkar, K. Varma, V. Ukani, S. Mankad, and S. Tanwar, "Combining user-based and item-based collaborative filtering using machine learning," in *Information and Communication Technology for Intelligent Systems*, pp. 173–180, Springer, 2019.

[177] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin, "Combing content-based and collaborative filters in an online newspaper," 1999.

[178] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," *Artificial intelligence review*, vol. 13, no. 5-6, pp. 393–408, 1999.

[179] M. Abourezq and A. Idrissi, "Integration of qos aspects in the cloud service research and selection system," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 6, pp. 111–122, 2015.

[180] L. Paquete and T. Stützle, "Stochastic local search algorithms for multiobjective combinatorial optimization: A.," *Handbook of Approximation Algorithms and Metaheuristics: Methologies and Traditional Applications*, vol. 1, 2018.

[181] C. Lin, F. Gao, and Y. Bai, "An intelligent sampling approach for metamodel-based multi-objective optimization with guidance of the adaptive weighted-sum method," *Structural and Multidisciplinary Optimization*, vol. 57, no. 3, pp. 1047–1060, 2018.

[182] D. C. Anastasiu, E. Christakopoulou, S. Smith, M. Sharma, and G. Karypis, "Big data and recommender systems," *Novtica: Journal of the Spanish Computer Scientist Association,(240)*, 2016.

[183] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.

[184] M. Sharma, F. M. Harper, and G. Karypis, "Learning from sets of items in recommender systems," *arXiv preprint arXiv:1904.12643*, 2019.