Exploring the reuse of past search results in information retrieval

Claudio Gutiérrez-Soto

▶ **To cite this version:**

Claudio Gutiérrez-Soto. Exploring the reuse of past search results in information retrieval. Information Retrieval [cs.IR]. Université Paul Sabatier - Toulouse III, 2016. English. NNT : 2016TOU30034 . tel-01589001

HAL Id: tel-01589001
https://theses.hal.science/tel-01589001

Submitted on 18 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *17/05/2016* **par :**

CLAUDIO GUTIÉRREZ SOTO

Exploring the Reuse of Past Search Results in Information Retrieval

---

JURY

MOHAND BOUGHANEM	Professeur, IRIT, Université Paul Sabatier, Toulouse	Examinateur
JEAN-PIERRE CHEVALLET	Maître de conférences - HDR, LIG, Université Grenoble Alpes	Rapporteur
BRIGITTE GRAU	Professeure, LIMSI - ENSIIE, Évry	Rapporteur
GILLES HUBERT	Maître de conférences - HDR, IRIT - Université Paul Sabatier, Toulouse,	Directeur
CHRISTIAN SALLABERRY	Maître de conférences - HDR, LIUPPA - Université de Pau et des Pays de l'Adour	Invité

École doctorale et spécialité :
MITT : Image, Information, Hypermedia
Unité de Recherche :
Institut de Recherche en Informatique de Toulouse (UMR 5505)
Directeur de Thèse :
Gilles HUBERT
Rapporteurs :
Brigitte GRAU et *Jean-Pierre CHEVALLET*

Exploring the Reuse of Past Search Results in Information Retrieval

Claudio GUTIÉRREZ SOTO

Acknowledgement

Firstly, I would like to express my sincere gratitude to my advisor Gilles Hubert for the continuous support of my Ph.D study and related research, for his patience and motivation. His guidance helped me in all the time of research and writing of this thesis.

Besides my advisor, I would like to thank the rest of my thesis committee : Brigitte Grau, and Jean-Pierre Chevallet, whose remarks during the review process were also very helpful and detailed. For writing the thesis, I also received data, comments and advice from Josiane Mothe.

I want to particularly thank my good friend Arturo Curiel, with who shared the challenges of PhD program, also for your help when I was with health problems.

In addition, I would like to thank my friend Rémi Dubot, and Chantal Morand for their help and support.

I should also like to thank UBB (Universidad del Bío-Bío), in particular to my friend Manuel Crisosto and the dean Benito Umaña for their support.

Last but not least, I thank my parents, Marcia and Willy, my sisters Ingrid and Marcia, and my daughters Bárbara and Cynthia, for their constant support before and during my years at IRIT.

Résumé

La recherche d'informations (RI) concerne l'obtention d'éléments (habituellement documents) de nature non structurée (habituellement du texte) qui satisfait un besoin d'information, dans de grandes collections de documents. Un système de recherche d'information (SRI) a pour objectif de représenter et de stocker de grandes quantités d'informations, pour faciliter et accélérer l'identification des informations pertinentes estimées pour une requête de l'utilisateur. Les deux processus principaux mis d'un SRI sont l'indexation et l'appariement. Le processus d'indexation vise à représenter les représentations des documents, de manière efficace non seulement pour le stockage, mais aussi pour l'accès. Le processus d'appariement vise à estimer si un document est pertinent pour une requête exprimée par un utilisateur. Cette mise en correspondance est généralement représentée par un score. Lorsque le processus est appliqué, un ensemble de documents est retourné à l'utilisateur sous forme de liste classée par score décroissant. Bien que les systèmes de RI aient émergé dans les années 1940, des améliorations sont vraiment apparues dès la fin des années 1950. Les améliorations en RI les plus importantes sont liées à l'évaluation des SRI. La communauté RI a bénéficié notamment de collections d'évaluation, notamment au travers de l'initiative TREC, qui organise chaque année un atelier. Ces ateliers ont offert aux chercheurs la possibilité de mesurer l'efficacité de leur système et de comparer les approches.

De nombreuses approches en RI traitant de l'indexation, de fonctions d'appariement, de modèles formels, et de retour de pertinence ont été proposées. Cependant, peu d'approches tirent avantage des recherches effectuées précédemment par d'autres utilisateurs. Les recherches passées constituent pourtant une source d'information utile pour les nouveaux utilisateurs (nouvelles requêtes). Par exemple, un utilisateur intéressé par un nouveau sujet pourrait bénéficier des recherches antérieures menées par les utilisateurs précédents intéressés par le même sujet. En raison de l'absence de collections ad-hoc de RI, à ce jour il y a un faible intérêt de la communauté RI autour de l'utilisation des recherches passées. En effet, la plupart des collections de RI existantes sont composées de requêtes indépendantes. Ces collections ne sont pas appropriées pour évaluer les approches fondées sur les requêtes passées parce qu'elles ne comportent pas de requêtes similaires ou qu'elles ne fournissent pas de jugements de pertinence.

Par conséquent, il n'est pas facile d'évaluer ce type d'approches. En outre, l'élaboration de ces collections est difficile en raison du coût et du temps élevés nécessaires. Une alternative consiste à simuler les collections. Par ailleurs, les documents pertinents de requêtes passées similaires peuvent être utilisées pour répondre à une nouvelle requête. De nombreuses contributions ont été proposées portant sur l'utilisation de techniques probabilistes pour améliorer les résultats de recherche. Des solutions simples à mettre en œuvre pour la réutilisation de résultats de recherches peuvent être proposées au travers d'algorithmes probabilistes. De plus, ce principe peut également bénéficier d'un clustering des recherches antérieures selon leurs similarités. Ainsi, dans cette thèse un cadre pour simuler des collections pour des approches basées sur les résultats de recherche passées est mis en œuvre et évalué. Quatre algorithmes probabilistes pour la réutilisation des résultats de recherches passées sont ensuite proposés et évalués. Enfin, une nouvelle mesure dans un contexte de clustering est proposée.

Abstract

Information retrieval (IR) is obtaining material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collections (usually stored on computers). Aiming to facilitate and accelerate the determination of the estimated relevant information for a user query, an information retrieval system (IRS) has as purpose to represent and stores large amounts of information. Two main processes can be found in a common IRS : indexing and matching. Indexing process corresponds to the representation and storing of documents, which should be efficient not only for storage but also for access. Matching intends to estimate whether a document is relevant according to a query issued by a user. This matching is usually represented via a score. When matching process is applied, a set of documents is returned to the user as a ranked list by decreasing score. Although IR systems emerged in the late 1940s, improvements really appeared from the late 1950s. The most important IR improvements are related to evaluation of IRS, which dates back from the late 1950s. The IR community have notably benefited from evaluation collections. A particular example is provided by the TREC community, which annually organize a conference and proposes tracks to support IR researches. The TREC evaluation campigns have offered to the researchers the opportunity to measure system effectiveness and compare approaches.

A wide range of approaches in IR exist dealing with indexing, matching functions, formal models and relevance feedback. Nevertheless, few approaches take advantage from searches performed previously by users. Past searches provide a useful source of information for new users (new queries). For example, a user searching about a new subject could benefit from past searches led by previous users about the same subject.

Due to the lack of ad-hoc IR collections, to this date there is a weak interest of the IR community on the use of past search results. Indeed, most of the existing IR collections are composed of independent queries. These collections are not appropriate to evaluate approaches rooted in past queries because they do not gather similar queries due to the lack of relevance judgments. Therefore, there is no easy way to evaluate the convenience of these approaches. In addition, elaborating such collections is difficult due to the cost and time needed.

Thus a feasible alternative is to simulate such collections.

Besides, relevant documents from similar past queries could be used to answer the new query. This principle could benefit from clustering of past searches according to their similarities. Two major categories of clustering can be easily identified : static clustering and post-retrieval clustering. On one hand, static clustering is the traditional application of the cluster method on a document collection. On the other hand, post-retrieval clustering includes information from the query into document clustering. Typically, similarity functions such as cosine distance, are used in static clustering. Nevertheless, these functions do not consider the specific context under which the similarity of two objects is judged.

On the other hand, a large assortment of contributions exist dealing with the use of techniques and probabilistic algorithms with the aim to improve results of retrieval process. Two major types of research can be easily categorized, learning techniques and optimization. Such approaches can imply high resources in computational time and human resources. By contrast, simple solutions to implement and represent can be proposed through randomized algorithms.

Thus, in this thesis a framework to simulate ad-hoc approaches based on past search results is implemented and evaluated. Four randomized algorithms to improve precision are proposed and evaluated, finally a new measure in the clustering context is proposed.

Table of contents

List of Figures

List of Tables

Chapter 1

Introduction

Résumé : Chapitre 1

La littérature en recherche d'information (RI) rassemble une grande variété de contributions qui traitent de différents aspects comme l'indexation,les modèles formels ou encore les retours de pertinence. Cependant, peu de contributions cherchent à tirer avantage des recherches effectuées précédemment. Les recherches effectuées précédemment par d'autres utilisateurs peuvent être une source d'information utile pour de nouvelles recherches. En outre, les documents pertinents de recherches passées peuvent être intéressants pour de nouvelles recherches. Dans la littérature RI, diverses approches sont liées au Web et s'appuient sur l'utilisation d'historiques de requêtes. La plupart des approches se concentre sur les requêtes répétitives. Cependant, les collections issues du Web ne sont pas utilisables pour évaluer les approches fondées sur les requêtes passées car elles ne rassemblent pas de recherches similaires effectuées par différents utilisateurs et pour lesquelles des jugements de pertinence (vérité terrain) sont fournis.

La simulation constitue donc une alternative pour construire des environnements appropriés pour analyser les avantages des approches s'appuyant sur les résultats de recherche passés. Les contributions de cette thèse sont basées sur l'utilisation de documents issus des résultats fournis par un système de recherche d'information (SRI) en réponse aux requêtes passées les plus proches. Le but est de capitaliser sur les résultats obtenus par d'autres utilisateurs pour répondre à une nouvelle recherche.

La réutilisation des recherches passées suppose le stockage des requêtes soumises par les utilisateurs ainsi que les résultats retournés pour celles-ci. Pour une nouvelle requête soumise au SRI, une première approche de réutilisation se base sur la recherche d'une requête stockée similaire et la sélection de documents

issus du résultat retourné pour celle-ci. Typiquement, des fonctions de similarité telles que Jaccard ou Cosinus sont utilisables pour mesurer la similitude entre les requêtes (Salton and McGill, 1983). Une seconde approche consiste à se baser sur des groupes (clusters) de requêtes similaires. Pour une nouvelle requête soumise au SRI, il s'agit dans ce cas de trouver un groupe de requêtes similaires et de sélectionner des documents issus des résultats retournés pour les requêtes de ce groupe.

La motivation principale de cette thèse a été d'étudier l'intérêt d'utiliser les résultats de recherche passées effectuées par d'autres utilisateurs pour construire et améliorer les résultats de nouvelles recherches d'informations. À cette fin, et en tenant compte des questions mentionnées précédemment, les principales contributions de cette thèse sont les suivantes : 1) Un cadre de simulation qui permet de construire un environnement ad hoc pour évaluer les approches fondées sur la réutilisation de résultats passés de requêtes similaires. 2) Un ensemble d'algorithmes probabilistes pour sélectionner des documents issus du résultat de la requête la plus proche pour répondre à une nouvelle requête. 3) Une approche utilisant un clustering des requêtes passées et leurs résultats, basée sur une nouvelle mesure de similarité (baptisée Query-Document Similarity Measure (QDSM)). Le résultat d'une nouvelle requête est alors construit à l'aide des documents des résultats passés issus du cluster le plus proche.

Cette thèse est organisée en sept chapitres (y compris le présent chapitre). Un aperçu du contenu des chapitres suit. Dans le chapitre 2, les principaux concepts de recherche d'information sont introduits, en se concentrant sur les questions qui sont pertinentes pour les travaux rapportés dans cette thèse. Le but est d'établir une terminologie de base et la couverture des questions qui sont utilisés dans les chapitres suivants. Dans le chapitre 3, une présentation détaillée des travaux liés à l'utilisation des résultats de recherche passés en RI existants dans la littérature est réalisée, et notamment sur le Web. Le chapitre 4 décrit une approche pour la simulation de collections de documents, de requêtes et de jugements des utilisateurs. Un ensemble de résultats expérimentaux, destinés à montrer la possibilité de simuler des collections variées, sont présentés et analysés. Dans le chapitre 5, après une revue bibliographique sur l'utilisation d'approches probabilistes en RI, une étude de différents algorithmes probabilistes pour la sélection de documents dans les résultats de recherche passés est présentée. Le chapitre 6 s'intéresse ensuite à une approche de clustering pour la réutilisation des résultats de recherche passés. Il examine préalablement les travaux existants liés au clustering en RI. Enfin, le chapitre 7 fait le bilan des principales contributions présentées dans cette thèse introduit les perspectives de travaux futurs envisagées.

Introduction

Information Retrieval (IR) involves tasks such as organization, storage and search of information. These tasks are carried out with the purpose to provide relevant information for users. User requirements are capitalized through queries, where it is desirable to find information that fill their expectation. It is feasible to find different sources of information such as other sources (i.e., video and audio), XML files to mention a few. Usually, information is spread out in a document or a set of documents (document collection). On the other hand, the relevance about a document or a set of documents according to is provided by users. The latter is well-known as judgments from users, thus given a query and a set of documents related to the query, each document can be classified by users as relevant or non-relevant according to the relevance of this document with respect his/her information need. The document set, a set of queries and their judgments of users is called by IR community a *test collection*. Nowadays, information is embodied in thousands and million of documents, by which the automation of the tasks previously mentioned is needed. Thus, an Information Retrieval System (IRS) provides support to the user in information searching in a document collection. An IRS provides a set of ranked documents which are related to the query submitted by a user. Commonly, the list of documents provided by IRS is ranked by decreasing score. This score represents the similarity between documents and the query computed by the IRS (similarity can be applied between queries as well as between documents). Nevertheless, all the retrieved documents do not satisfy the user needs. In simple words, not all documents which appear in the list are relevant for the user. Furthermore, the position of relevant documents in the list is very important for the user. Therefore, a good situation is when relevant documents appear together and at the top of the list (i.e., relevant documents should not be widely dispersed). This property is related to precision in IR. In order to illustrate this, we can suppose the following scenario : for a query q, there is a subset of relevant documents srd for a document collection. An IRS S_1 supplies a list of documents where srd appears at the top of the list. In a similar way, another IRS S_2 yields a list of documents where srd appears at the bottom of the list. When we compare both systems, S_1 provides a better precision than S_2 because the same srd appears at the top. An important and difficult challenge for an IRS is to provide the best precision. Nowadays, the IRSs most widely used are Web Search Engines (WSEs). WSEs respond to millions of queries per day on collections which involve billions of documents. Moreover, with the sustained growth in the number of documents, the task to find relevant documents for a query submitted by a user can becomes unprofitable. In summary, the most relevant challenge for an IRS is to supply the best precision possible.

A large assortment of contributions in IR, which address different pers-

pectives such as matching functions, indexing, formal models and relevance feedback can be found in the literature. Nevertheless, few of these contributions take advantage from previously performed searches. Past searches can be a useful source of information for new searches. Furthermore, relevant documents from past searches can be profitable, because these documents can be used new searches. Most of the approaches In the IR literature rely on the use of historical queries on the Web, most of which are supported on repetitive queries (Gutiérrez-Soto and Hubert, 2013, 2014). The low level of interest on the use of past queries may be understandable because of the lack of suitable IR collections. Indeed, most of the existing IR collections are composed of repetitive queries. These collections are not usable to evaluate approaches based on past queries since they do not gather similar queries for which ground truth relevance judgments are provided. Thus, an alternative is to build suitable environments to analyse the advantages of approaches relying on past search results using simulation (Gutiérrez-Soto and Hubert, 2013). Therefore, a contribution of this thesis is the use of simulation to give an ideal environment based on past queries. It is essential to highlight that all contributions of this thesis are based on the use of relevant documents obtained from the most similar past query with the aim to improve precision for some IRS, and one way is using clustering of past queries.

Clustering and the inverted file are data structures mainly used to build the indexes and access to documents. Inverted files support efficient answering to keyword queries meanwhile clustering allows to build groups of similar documents. Clustering in IR have been used to improve efficiency and effectiveness of IR systems (IRSs). In general, two major categories of clustering are easily identifiable : static clustering and post-retrieval clustering. On one hand, static clustering is the traditional application of the cluster method on a document collection. On the other hand, post-retrieval clustering includes information from the query into the clustering of documents. Query clustering is a type of post-retrieval clustering, which is devoted to find similar queries in a cluster. Similarity between queries is related to the overlapping among query terms. Typically, similarity functions such as Jaccard or cosines are used to measure the similarity between queries (Salton and McGill, 1983). Nevertheless, these functions do not consider the specific domain (i.e., specific domain is additional information used to calculate the similarity) under which the similarity of two objects is judged. Thereby, the queries with their documents can be stored in a query-document clustering in order to provide a context. Thus, another contribution of this thesis is a clustering of past queries along with their relevant documents, where relevant documents from a similar past query are used to answer a new query. Besides, it is important to consider approaches that promote answers to the queries in reasonable times not only in the design or searching of efficient mechanism to improve precision of IRSs, (i.e., involved time in a task of data mining to assign the best IRS, which improve precision for a specific type of query) but also in involved execution time for IRS in order to provide documents for a given query.

A wide range of approaches in IR are devoted to improving the list of retrieval documents for particular queries. Among these approaches we can find solutions that involve efficient assignments of IRSs to respond to certain types of queries, by applying data mining techniques (Bigot et al., 2011). Nonetheless, some tasks of data mining can imply not only long periods of time, but also a high cost in money (Gray and Watson, 1998). On the other hand, solutions that involve an exhaustive analysis of all possible alternatives to find the best answer to a query (i.e., the best precision for each type of query), can be found in the IR domain. Prior solutions correspond to approaches based on learning techniques (e.g., neural networks, genetic algorithms and machine support vectors). However, these approaches should imply a high cost in learning time as well as diverse convergence times when the used datasets are heterogeneous (Nopiah et al., 2010). Additionally, characteristics such as the scopes where these types of algorithms are applied and the performance achieved in different environments, are complex to address (Kearns, 1989). An interesting solution should be finding a function called "oracle", which will be able to guess that documents are relevant for a specific query submitted to an IRS (i.e., "oracle" supplies *srd*). This means that "oracle" provides the judgments of users on the document collection for a query q. Thus, if "oracle" could be used in S_1 and S_2, both IRSs should provide the same precision using *srd* (this is, "oracle" provides *srd* at the top of the lists for both systems S_1 and S_2). Nevertheless, finding this function is a hard task. The most important assumption used in this thesis, is that the relevant documents tend to appear at the top of the result list. Therefore, it is possible to suppose that for the "oracle1" function (for the system S_1) and the "oracle2" function (for the system S_2), there is a high probability that the most relevant documents appear at the top of the list (i.e., the first document that appear at the top of the list has higher probability to be relevant than the last document in the list). Finally, the last contribution of this thesis is a set of randomized algorithms, which select relevant documents according to their position in the result list from the most similar past query. These algorithms do not require learning time and provide an acceptable precision.

To sum up, three problems can be easily identified :

1. Approaches which take advantage from past search results are based on repetitive queries rather than similar queries. In addition, exiting IR collections are not suitable to evaluate approaches based on past search results. The construction of such collections implies high cost not only in time but also in effort (Sanderson, 2010; Alonso and Mizzaro, 2012).
2. The relationship between associated documents with similar past queries has not been extensively exploited (Gutiérrez-Soto and Hubert, 2013). Traditionally, query clustering has been applied by using similarity functions such as Jaccard or cosines among queries. Nonetheless, the specific context in which these function are applied is not considered.

3. Usually, approaches that provide the best answer to a query, involve an exhaustive analysis of all possible IRSs (i.e., it is given when the approach considers several queries and several IRSs that respond to these queries). Typically, these approaches are based on machine learning or data mining, which can imply a high cost in computation and human resources.

The main motivation of this thesis has been to research different treads to improve precision using past search results. To that end, and taking into account the issues previously mentioned, the major contributions of this thesis are :

1. A framework for simulation that built an ad-hoc environment to evaluate approaches under the domain of similar past results.
2. A set of randomized algorithms, which are easy to implement and improve precision using similar past search results.
3. A new similarity measure (query-sensitive similarity measures (QSSM)) on query-document clustering, which allows to store past queries in clusters, and responds to new queries using relevant documents from the most similar past query.

1.1 Thesis Statement

The goal of this work is to improve precision for new queries, taking advantage from relevant documents associated to similar past queries.

To achieve this goal, this work designs and implements experimental ad-hoc environments and algorithms to exploit the use of past queries. Thus, the objectives of this research can be divided in three major areas : a) *simulation of collections*, b) *Monte Carlo algorithms to improve precision*, and c) *query-document clustering* (Gutiérrez-Soto, 2014a).

1.1.1 Simulation of Document Collections, Queries and Judgments of Users

It is difficult to find suitable collections to evaluate approaches based on past queries. Moreover, internal processes of some IRSs such as stemming, stopping, tuning among others, do not provide clear perception about the performance of algorithms in the same experimental condition. Thus, an objective of this work is to provide ad-hoc environments to evaluate approaches based on past queries.

1.1.2 Randomized Algorithms

Several approaches are devoted to improve precision. Nevertheless, some involve high cost in resources such as execution time, and human resource time. This

research aims to design, implement and evaluate algorithms based on previous search results, which do not require learning time.

1.1.3 Query-Document Clustering

Commonly, clusters in the IR context are used to store documents (Tombros and van Rijsbergen (2004)). Nonetheless, storage of queries with their relevant documents can provide a search context using query-sensitive similarity measures (QSSM). To that end, this research studies, provides, and evaluates a new measure of similarity taking into account the context to exploit the use of past search results and enhance precision for new queries.

This work has produced the following results :

International Journal

— *On The Reuse of Past Searches in Information Retrieval :Study of Two Probabilistic Algorithms.* International Journal of Information System Modeling and Design, 6(2), 71-90, April-June 2015 (Gutiérrez-Soto and Hubert, 2015).

International Conferences

— *Evaluating the Interest of Revamping Past Search Results.* International Conference on Database and Expert Systems Applications (DEXA 2013), Prague, Czech Republic, August 2013 (Gutiérrez-Soto and Hubert, 2013).

— *Randomized Algorithm for Information Retrieval Using Past Search Results.* In Research Challenges in Information Science (RCIS 2014), Marrakesh, Morocco, May 2014 (Gutiérrez-Soto and Hubert, 2014).

— *Probabilistic Reuse of Past Search Results*, International Conference on Database and Expert Systems Applications (DEXA 2014), Munich, Germany, September 2014 (Gutiérrez-Soto and Hubert, 2014)

— *Simulation, Randomized and Clustering Algorithms for Information Retrieval.* In Proceedings of the Doctoral Consortium (RR 2014), The 8th International Conference On Web Reasoning And Rule Systems, Athens, Greece, September 2014 (Gutiérrez-Soto, 2014a).

National Conferences

— *Taking Advantage from Past Search Results Through a Probabilistic Algorithm*, Spanish Conference on Information Retrieval (CERI 2014), A Coruña, Spain, June 2014 (Gutiérrez-Soto, 2014b).

Others

— *Randomized Algorithms to Improve Precision in Information Retrieval*, Doctoral Seminar (DocToMe), IRIT Université Paul Sabatier, Toulouse, France, March 2014.

1.2 Thesis outline

This thesis is organised into 7 chapters (including the present chapter). An outline of the contents of the remaining chapters follows.

Chapter 2 : In this chapter, I discuss some of the main concepts of information retrieval, focusing on issues that are relevant to the experimental works reported in this thesis. The purpose is to establish a basic terminology and coverage of issues that are used in the following chapters.

Chapter 3 : This chapter provides a detailed review of past works. The first part is about the use of past search results in IR. The second part is how similar past queries are used on the Web.

Chapter 4 : I define the domain about the simulation of document collection, query collection and judgements of users. A set of experimental results are presented and analyzed.

Chapter 5 : In this chapter, a bibliographical review about probabilistic approaches in IR is presented. Besides, I give the mathematical support for all randomized algorithms analyzed in this thesis.

Chapter 6 : In this chapter, a review of previous works related with clustering in IR is presented, in special post-retrieval clustering in the similar past search results.

Chapter 7 : In chapter 7, I report the main contributions that this work made, and I also point some issues for future work that will follow this thesis.

Chapter 2

Information Retrieval

Résumé : Chapitre 2

Plusieurs modèles de RI ont été développés, qui définissent la manière dont les documents et les requêtes sont représentés, mais aussi la façon dont le processus d'appariement document requête est effectué. Les modèles les plus répandus en RI sont le modèle booléen, le modèle vectoriel (Vector Space Model) (Salton, 1971; Salton et al., 1975), le modèle probabiliste (Robertson et al., 1981), et le modèle logique (Van Rijsbergen, 1986).

Le modèle vectoriel, très largement utilisé dans la littérature, permet de représenter des documents et des requêtes selon des vecteurs pondérés de termes, sur la base des fréquences de termes dans les documents. Plusieurs mesures de similarité entre représentations des requêtes et des documents ont été proposées comme le très utilisé cosinus entre les vecteurs de termes de la requête et du document. Ce modèle a été choisi comme base pour le travail expérimental réalisé dans cette thèse.

De plus, il est important de mentionner que la recherche abordée dans cette thèse est basée sur l'information textuelle dans des documents non structurés. Dans ce chapitre, tous les concepts utiles pour fournir le contexte nécessaire pour comprendre cette recherche sont fournis progressivement. La section 2.2 présente les principes de représentations des documents et des requêtes dans un SRI. La section 2.3 décrit le processus d'appariement entre requêtes et documents, et la section 2.4 introduit l'évaluation des SRI. Enfin, la section 2.5 fournit un résumé introductif sur l'utilisation des résultats de recherches passées.

Information Retrieval

Information Retrieval (IR) is the branch of computer science, which is concerned with the organisation, structuring, analysis, storage, and searching of information. Several definitions of IR can be found in the literature. Baeza-Yates and Ribeiro-Neto (1999) supply the following :

"... the IR system must somehow *interpret* the contents of the information items (documents) in a collection and rank them according to a degree of relevance to the user query. This *interpretation* of a document content involves extracting syntactic and semantic information from the document text ... "

From the previous paragraph, three important aspects should be considered in that definition. First of all, there is a user with an information need, a document collection with which this requirement is compared, and finally the list of documents provided by an IRS in response to the query. Thus, an IRS aims to supply a set of documents which satisfies the information need of a user. A user who wants to look for information, expresses his requirement through a query, which is submitted to the system. Aiming to process this query, the IRS runs an internal representation of it. Once formed, this query representation, is matched with the document collection. As a result, a ranked list of documents is provided to the user. Therefore, the user can check the final list of documents and if documents are not enough relevant for him/her, he/she can submit a new query or reformulated his/her query.

Several IR models have been developed, where it is possible to find not only a description of how documents and queries are represented but also the way in which the document-query matching process is carried out. The most widespread models in IR are the Boolean, The Vector Space (Salton, 1971; Salton et al., 1975), The Probabilistic (Robertson et al., 1981), and The Logical (Van Rijsbergen, 1986).

The vector space model represents documents and queries according to vectors, based on the frequencies of terms in documents. The similarity measure between a query and a document corresponds to the cosine between the query terms and the document terms. This model has been the baseline for experimental work on document clustering and corresponds to the model, which was used to carry out the experimental phase in this thesis.

The first IRSs appear with the purpose to support the automated searching of library material by users. As a response to the exponential growth of the information available in an electronic format, the IRSs broadened their spectrum in other scopes. Thus, a wide range of researches relying on heterogeneous types of information can be found in the literature.

Some researches concern passage-based information retrieval (Salton et al., 1993; Liu and Croft, 2002). Other researches based on XML, study techniques which allow the retrieval of information segments from structured data (Fuhr et al., 2005; Fuhr and Lalmas, 2007). Thus the World Wide Web has become the most popular media used by people to look for information. IRSs have been developed on the internet as Web Search Engines (WSEs), which index a big amount of information and promote a simple way to provide information access to users.

Nowadays, a great deal of work have been carried out with the aim to exploit features that characterise web pages such as HTML structure, Web page popularity, and hyperlinked structures among others (Bharat and Henzinger, 1998; Kleinberg, 1999). In Broder (2002), a taxonomy which classifies web queries in three major categories, navigational, informational, and transactional is presented. Roughly speaking, navigational queries concerns how users can reach a particular web-page or document. In the second category, information queries supply a user with information related to a particular topic. In this domain the user can be interested in more than a single document. Lastly, transactional queries tackle at locating services with which the user should interact.

On the other hand, it is important to mention that the research addressed in this thesis is based on textual information, which is simulated or stored in a document format. Indeed, all objects used in this thesis do not have a hyperlinked structure. Besides, it is relevant to point out that the purpose of this chapter is not to provide a general overview about the basic concepts involved in IR research. Basic concepts related to IR can be found in the books, widely used and cited by the research community (Baeza-Yates and Ribeiro-Neto, 1999; Rijsbergen, 1979; Sparck Jones and Willett, 1997a).

In this chapter, all concepts that are relevant to provide the necessary background to understand this research are provided gradually. Section 2.2 presents the representations of documents and queries in an IRS. Section 2.3 describes the matching process among the query and documents, and Section 2.4 exposes the evaluation of an IRS. Finally, Section 2.5 provides an introductory summary about the use of past search results.

2.1 Representing documents and queries

Generally, documents are transformed by an IRS from the original form to an internal representation, this process is called indexing. The purpose of this process is to provide a representation of the information as accurate as possible. To achieve this goal, a set of indexing features are assigned for each document. The most relevant features for a document correspond to a list of

words, which allow to discriminate between them. These are well-known as terms. Indeed, a document is not only represented by this list of terms, but also accessed by terms which belong to its list. In order to obtain a higher level of representation, which allows to retrieve phrasal units, or the use of linguistic, semantic and knowledge-based methods ; complex characteristics should be involved in the indexation process (Lewis and Jones, 1996). In the domain of this thesis, representations of documents are provided by lists of terms, which are extracted from documents.

A normalization process takes place before indexing. The goal of this process is to provide only relevant terms. For example, words with high frequencies in the document (stop-words) will not be considered in the indexing (Rijsbergen, 1979). Stop-words are well-known words such as articles (e.g., the) and prepositions (e.g., in, at). The main advantage of this process is to reduce the text volume up to 50 percent. Another process before indexing consists in removing the suffixes from the remaining words of the input text. To that end, a stemming algorithm is applied to reduce words to a common root form (known as stem). For example, the stemming algorithm reduces the words 'cardiovascular', 'cardiology' to the word cardio, which will be in the vocabulary of indexed terms.

Aiming to obtain terms which are representative of a document against other documents that belong to the collection (at the vocabulary of indexed terms), it is necessary to strive in the presence of non-frequent terms in contrast to terms which appear in all documents. For achieving this goal, the concept of inverse document frequency weighting (IDF) was introduced in (Jones, 1972). Thus, the weight of a term in a document is increased if it appears more often in that document. By contrast, the weight of a term in a document decreases if it appears frequently in other documents. Therefore, for a document collection which contains N documents (we can suppose a collection with N documents), if the term i occurs in n_i documents, then the *idf* weight of a term is given by $log(\frac{N}{n_i})$. A term weighting function corresponds to the combination of the *tf* and *idf* weights, which is commonly known as *tf-idf* weight (Salton (1971)) :

$$w_{ij} = \frac{log(freq_{ij}+1)}{log(length_j)} log(\frac{N}{n_i})$$

w_{ij} = *tf-idf* weight of term i in document j
$freq_{ij}$ = frequency of term i in document j
$length_j$ = length (in terms) of document j
N = number of documents in the collection
n_i = number of documents that term i is assigned to

An overview of various weighting schemes as well as evaluation measures in the IR domain are given by Salton and Buckley (1987). It is important to highlight that in the experiments related to simulation the stemming process have been omitted. I made this choice because in every experiment an ideal environment is built. Thus, the stemming process is not necessary because every term is

unique, therefore there is not a common root among the terms. In the same way, stopping (i.e., removing stop-words) is discarded because the terms used are assumed to be representative terms. In summary, it can be seen as the general application of the same stemming process and stopping in all experiments.

Three major categories of data structures are used by IRSs to store terms, lexicographic indices (indices that are sorted), clustered file structures, and indices based on hashing. Nevertheless, data structures most used by WSEs correspond to the inverted file (Ouksel, 2002). This structure corresponds to a list, which contains representative terms from document collection, thus the terms which belong to the query are matched with keyword (terms) that are in the list. Hence, it is feasible to immediately locate all documents in the collection that contain this keyword (Rijsbergen, 1979). Clustered file structures will be exposed in detail in chapter 6.

2.2 Query operations

An IRS has as main objective to support the user in the searching of those documents that can satisfy his information need. Information requirements of a user are expressed in a manner that can be understood by the IRS. This manner to express the user requirements is denominated "query".

The formulation of a query can be carried out for some IRSs through the use of boolean operators. An example of this kind of query could be : ((Simulation AND Algorithms) NOT Randomized). The main disadvantage for boolean IRSs is that their results are few intuitive for non-experienced users, hence the formulation of an ad-hoc query can be ineffective (Sparck Jones and Willett, 1997a)). As a consequence, these systems have been replaced by systems which provide the query formulation through of natural language without the use of specific operators. These systems rely on best-match or similarity searching, which is calculated for each document as well as for each query. The matching between documents and queries is presented in Section 2.3.

In the same way that documents, the queries are processed prior to be indexed, that means, the lexical processing and term-weighting are executed by IRSs.

2.3 Matching between documents and queries

Through the boolean model, an IRS finds the subset of documents from the whole collection, in which every document that belongs to the subset has at least one term from the query submitted to the IRS, which are presented to

the user in an unranked way. On the other hand, it is feasible to find systems which have better methods of comparison and provide the outcome according to a relevance score about the pertinence of the document regarding the query. Thus an IRS supplies ranked list of documents, sorted in decreasing way to the user.

In the vector space model (Salton and McGill, 1986), both documents and queries are represented as vectors in a multidimensional space. The spatial representation corresponds to the indexed terms of the document collection.

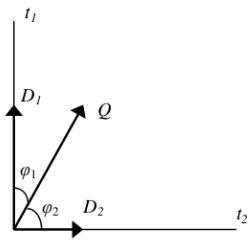

FIGURE 2.1: Document and query representations in the vector model

An example of vector model is presented in Figure 2.1. In this model a document and query are represented. In this example, the vector space is displayed in two dimensions. Thus, the dimension corresponds to the terms t_1 and t_2. Documents D_1 and D_2 are represented in the space using each document term weights as coordinates. Weights correspond to term frequency weights : $D_1 = t_1,\ t_1$ and $D_2 = t_2$. On the other hand, the query Q is represented as $Q = t_1,\ t_1,\ t_2$. Besides, the angles between the vectors of $D_1 - Q$ and $D_2 - Q$ are presented in the Figure 2.1.

In this space, measures to quantify the similarity among queries and documents can be defined. The best match between a particular query and set of documents corresponds to the closest document regarding the query according to the similarity measure. A wide range of formulas dealing with distance measures can be found in the IR literature. A detail of most of them can be found in books such as (Rijsbergen, 1979; Salton and McGill, 1986).

A simple way to compare a document regarding a query is by counting the number of common terms between them. In this way, we can assume that both are represented as vectors of length n (where n is the number of terms in the collection). Thus, we have the following measure :

$$sim(D, Q) = \sum_{i=1}^{n} D_i Q_i \tag{2.1}$$

This measure is known as the *coordination level* matching function. By contrast,

other measures of similarity can be normalized according to the length of the document and the query.

The most widely used measure by the IRSs corresponds to the cosine. The reason of its popularity is due to the geometric interpretation of the vector model.

$$sim(D,Q) = \frac{\sum_{i=1}^{n} D_i Q_i}{\sqrt{\sum_{i=1}^{n}(D_i)^2 \sum_{i=1}^{n}(Q_i)^2}} \tag{2.2}$$

The cosine measure is the function that provides the angles between the vectors of the document and the query (see equation 2.2), and whose value ranges are comprised between 0 (the vectors of the documents and the query form an angle of 90°, i.e., they are dissimilar) and 1 (the vectors of the document and the query form an angle of 0°, i.e., they are equal). In Figure 2.1, the similarity between documents D_1 and D_2 is 0, because the angle between the two vectors is 90°. It means that they do not have terms in common.

2.4 Evaluation of IR systems

A large assortment of methodologies deals with the evaluation of IRSs. The evaluation task involves an intrinsic complexity because it implies issues from different research areas such as cognition, statistics, experimental design, system design, human computer interaction, among others. In this section, an outline of evaluation issues with emphasis on central aspects which comprise the core of this thesis are reported.

Several aspects on the IR process can be evaluated. Such aspects should be related, among others, with the speed of an IRS, the interfaces and the level of end-user interaction, the format of information presented to the users. Nonetheless, the most relevant focus of this thesis corresponds to the evaluation of the number of relevant documents provided by an IRS in response to a user query. The most often used measures of effectiveness correspond to precision and recall. Precision represents the fraction of documents retrieved that are relevant, meanwhile that recall is the proportion of relevant documents that have been retrieved. According to Figure 2.2, precision and recall can be defined as :

$$Precision = \frac{(number\ of\ relevant\ documents)\ AND\ (retrieved\ documents)}{(number\ of\ retrieved\ documents)}$$

$$Recall = \frac{(number\ of\ relevant\ documents)\ AND\ (retrieved\ documents)}{(number\ of\ relevant\ documents)}$$

Commonly, these measures are expressed in a range between 0 and 1, although they also can be expressed in percentages as in (Chowdhury, 2010).

From the previous definitions, it is necessary to know the total number of relevant documents in a collection to compute the recall. Nevertheless, it is not always possible to calculate this measure, because it involves not only a considerable amount of effort but also time. On the other hand, collections which allow to calculate these measures can be commercial as well as freely available. This type of document collection can be found together with a set of queries and judgments from users (about the relevance of a document with respect to a query). Through the use of these collections, the IR researchers have the opportunity to evaluate their approaches with empirical results and compare their results with other systems and approaches.

In Figure 2.2, a trade-off between recall-precision (R-P) graph is shown. That means, precision and recall are related in an inverse way. Every time that precision is increased, recall is decreased and vice-versa. This is represented in almost all IR textbooks and handbooks.

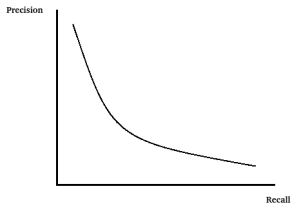

FIGURE 2.2: A recall-precision graph

With the constant increases of large collections, specially with the appearing of the Text Retrieval Conferences (TREC) (Harman, 1993), which contain thousands and millions of documents (it involves the use of many gigabytes of disk space), it has become impossible to provide exhaustive judgments about the pertinence of documents (Tombros et al., 2002).

Aiming to tackle this problem, a technique extensively used and well-known in these cases is denominated *pooling* (Harman, 1993). The core of this technique is rooted in the combination of the top-ranked documents from several IRSs, where a particular query is submitted in each of them. Thereby, the judgments of users correspond to a combined set from IRSs. We can deduce that this technique is effective, when retrieved relevant documents are representative of all relevant documents available in the IRSs (Harman, 1993).

The type of evaluation stated previously relies on judgments provided by some expert judges, based on topical (or algorithmic) relevance. In (Schamber et al., 1990; Barry, 1994), the relevance corresponds to a multidimensional concept, which involves just one dimension. Inspired by this work, other di-

mensions about the concept of relevance through the evaluation methodologies with respect to the utility provided by IRSs, have been presented in (Borlund and Ingwersen, 1997; Reid, 2000).

It is important to highlight that the conclusions drawn from empirical results of IR researches are an issue that involves a large number of factors such as the choice of suitable measures of performance, the statistical validation of empirical results, among others. To address these issues, a methodology to obtain scientific inference from IR experiments have been proposed (Keen, 1992).

2.5 Past Search Results

The use of past search results to respond to a new search is not recent. Several approaches dealing with the use of historical queries to enhance the results of search, can be found in the IR literature (Fitzpatrick and Dent (1997) ; Hust (2004) ; Cetintas et al. (2011)). Repetition (i.e., repetition or repeated query, is a query which has been submitted previously in IRS) and query reformulation (i.e., reformulation or modified query, is a query which add or delete terms from a query submitted previously in IRS) have achieved considerable success with the exploitation of available information inside web files. Log files have been studied extensively by considering repeated queries and query reformulation (i.e., by adding terms or by deleting terms). Furthermore, historical queries (i.e., historical queries are queries that have been submitted previously in IRS) have been used in similar communities in the domain of recommender systems. On the other hand, approaches in IR, which are not based on log files, also have studied the use of similar queries with the aim to improve new search. Nevertheless, both approaches are mainly devoted to exploit repeated queries than similar queries.

In the following chapter, I provide the domain of using the past search results to improve information retrieval.

Chapter 3

Related work on the use of past searches

Résumé : Chapitre 3

Une grande variété d'approches visait à par tirer avantage de requêtes préalablement soumises existe dans la littérature. Plusieurs études ont révélé qu'une partie non négligeable des requêtes soumises aux moteurs de recherche sont des requêtes similaires (par exemple, deux requêtes q et q' sont similaires si elles ont plusieurs termes en commun) et même des requêtes répétées (par exemple, une requête q répétée est une requête qui a été préalablement soumise au SRI). Par exemple, Jansen et al. (2000), lors de l'analyse de logs de requêtes du moteur Excite 97, ont constaté que, pour un utilisateur donné, 35 % des requêtes étaient des requêtes uniques, 22 % étaient de requêtes modifiées, et 43 % correspondaient à des questions identiques. En ce qui concerne la façon dont les utilisateurs redéfinissent leurs requêtes, 34,7 % des modifications correspondaient à des substitutions de mots. De plus, 19 % des modifications étaient des ajouts d'un seul mot et 9,5 %, des ajouts de deux mots. Les résultats ont montré que les utilisateurs préfèrent redéfinir les requêtes en ajoutant des termes. Par ailleurs, les propriétés temporelles des requêtes ont été analysées par Beitzel et al. (2004). La principale conclusion de cette étude souligne que « les requêtes reçues pendant les heures de pointe sont plus semblables les unes aux autres que leurs homologues soumises en dehors des heures de pointe ». Teevan et al. (2007) ont exploré les questions de répétition de recherche à l'aide de logs de requêtes Yahoo. Leurs analyses ont révélé de très fréquentes recherches répétées et clics répétés. Elles ont révélé également que 7 % des requêtes répétées ont été soumises par différents utilisateurs. Une conclusion est qu'il est possible de prédire les requêtes de

navigation et les résultats susceptibles d'être cliqués. Ces études soutiennent l'idée que les requêtes préalablement soumises pourraient être utilisées dans le traitement de requêtes similaires nouvellement soumises. Un premier type de travail exploite les requêtes passées pour améliorer l'efficacité (par exemple, le temps de traitement d'une requête) du traitement des nouvelles requêtes. La plupart de ces travaux concernent les requêtes répétées, traitant de la mise en cache pour des moteurs de recherche (Xie and O'Hallaron, 2002; Brown et al., 1994; Jónsson et al., 1998; Markatos, 2001; Luo et al., 2000; Saraiva et al., 2001; Baeza-Yates and Saint-Jean, 2003; Lempel and Moran, 2003; Fagni et al., 2004, 2006). Cependant, ce type d'approche ne traite pas le même problème que l'approche proposée dans cette thèse, car l'objectif est d'accélérer l'accès à des résultats tandis que cette thèse traite de l'amélioration de la précision des résultats en réutilisant les requêtes passées. En revanche, peu de recherches ont été menées visant à bénéficier de la réutilisation des requêtes passées similaires pour améliorer la qualité des résultats de nouvelles requêtes. Plusieurs raisons peuvent expliquer le manque de recherche sur les requêtes passées similaires, parmi lesquelles le manque de collections de RI appropriées. Typiquement, une collection appropriée pour l'évaluation des systèmes de RI est composée d'un ensemble de documents, un ensemble de requêtes, et un ensemble de documents pertinents pour chaque requête (par exemple, les documents pertinents fournis reposent sur les jugements des utilisateurs). Cependant, la plupart des collections disponibles, qui ont été construites pour les campagnes d'évaluation de RI telles que TREC, CLEF et NTCIR, n'offre que des ensembles de requêtes très différentes. Par conséquent, ce type de collections n'est pas utilisable pour évaluer des approches dans le contexte de la réutilisation des requêtes passées pour améliorer la recherche pour de nouvelles requêtes. D'autres collections, basées sur les fichiers de logs des moteurs de recherche, comprennent les requêtes similaires, mais en général correspondant à des reformulations soumises par le même utilisateur au cours d'une session de recherche, et sans fournir de jugements de pertinence. Ces collections sont destinées à l'étude de la détection de session ou la recherche d'information au cours de sessions utilisateur plutôt que des requêtes ponctuelles. En outre, la construction de ces collections implique d'importants coûts mais aussi du temps (Sanderson, 2010; Gutiérrez-Soto and Hubert, 2013). Cependant, plusieurs approches utilisant des requêtes antérieures existent dans la littérature. Elles peuvent être séparées en deux types :

— les approches reposant sur les sessions utilisateur, où les utilisateurs sont enregistrés dans le système. Dans une session utilisateur, l'identification de données est possible, cette information implique l'identification de l'utilisateur, l'historique des requêtes et des documents récupérés, la date, la durée de la session (dans le temps), entre autres. Les approches de personnalisation sont généralement basées sur les sessions utilisateur.

— les approches pour les utilisateurs non connectés, ce qui implique l'utilisation d'un système anonyme. Sans l'aide de sessions utilisateur, l'information disponible est plus limitée. L'approche présentée dans cette thèse est de ce

type.

Ce chapitre est organisé comme suit. La section 3.1 présente les approches reposant sur les requêtes passées au sein d'une session utilisateur. La section 3.2 expose les approches exploitant les requêtes passées sans session utilisateur et enfin la section 3.3 conclut le chapitre.

Introduction

A large assortment of approaches interested in taking advantage from queries previously submitted exists in the literature.

Several studies revealed that a non-negligible part of the queries submitted to search engines are similar queries (i.e., two queries q and q' are similar if they have several terms in common) and even repeated queries (i.e., a repeated query q is a query, which has previously been submitted to the IRS). For instance, Jansen et al. (2000), analyzing Excite97 query log, found that, for a given user, 35% of queries were unique queries, 22% were modified queries, and 43% corresponded to repeated queries. Regarding how users redefine their queries, 34.7% of modifications corresponded to word substitutions ; meantime 19% of modifications were single word additions and 9.5% double word additions. The results showed that users prefer to redefine queries by adding terms. Continuing the trend, temporal properties of queries were analyzed by Beitzel et al. (2004). The main conclusion of this study points out that "*The queries received during peak hours are more similar to each other than their non-peak hour counterparts*". Teevan et al. (2007) explored issues of re-finding using Yahoo query log. Their analyses revealed that repeated searches and repeated clicks were very common. They revealed also that 7% of repeated queries were issued by different users. A conclusion was that it is possible to predict which queries are navigational and what results are likely to be clicked. Such studies support the idea that queries previously submitted could be used in processing similar queries newly submitted.

A first type of work takes advantage of past queries to improve efficiency (i.e., time to process a query) when processing new queries. Most of these works concern repeated queries, dealing with caching approaches for search engines (Xie and O'Hallaron, 2002; Brown et al., 1994; Jónsson et al., 1998; Markatos, 2001; Luo et al., 2000; Saraiva et al., 2001; Baeza-Yates and Saint-Jean, 2003; Lempel and Moran, 2003; Fagni et al., 2004, 2006). However, this type of approaches does not address the same problem as the approach proposed in this thesis since it intends to accelerate access to results while this thesis deals with improving effectiveness (i.e., result accuracy) by reusing past queries. For example, Xie and O'Hallaron (2002) analyzed several engine query logs, where mostly the repetition and locality of queries take place. Although an experimental approach of caching is not provided, this work suggests that any caching technique for search engines should take into account the popularity of queries as well as the locality of repetitions. Other approaches propose two-level caches for search engines. The first level corresponds to the caching of result sets, which avoids disk accesses. The second level based on caching inverted lists decreases the I/O costs (Saraiva et al., 2001; Baeza-Yates and Saint-Jean, 2003).

By contrast, little research has been carried out aiming at dealing with advantages of reusing similar past queries to improve retrieval effectiveness for new queries. Several reasons can explain the lack of research on similar past queries, among them the shortage of suitable IR collections. Typically, a collection suitable for evaluation of IR systems is composed of a set of documents, a set of queries, and a set of relevant documents for each query (i.e., relevant documents are provided based on users' judgments). However, most of the available collections, which were built for IR evaluation campaigns such as TREC, CLEF and NTCIR, provide sets of dissimilar queries only. Consequently, this type of collections is not usable to evaluate approaches in the domain of reusing past queries to improve retrieval effectiveness for new queries. Other collections, based on log files of search engines, comprise similar queries, but usually corresponding to query reformulations submitted by the same user during a search session, and without providing user relevance judgments. Such collections are intended for studying session detection or information retrieval over user sessions rather than one-time queries.

However, several approaches using past queries can be found in the literature. They can be divided in two types :

— Approaches rooted in user sessions, where users are logged in some system. In a user session, the data identification is possible, this information involve the user identification, the historical record of queries and retrieved documents, the date, the session duration (in the time) among others. Approaches related to personalization and customization are usually based on user sessions.
— Approaches for which users are not logged, which implies the use of a system anonymously. Without using user sessions, the available information is more limited than approaches based on user sessions. The approach presented in this thesis is of this type.

The remaining of this chapter is organized as follows. Section 3.1 presents the approaches taking advantage of past queries through user sessions. Section 3.2 exposes the approaches taking advantage of past queries without user sessions and finally section 3.3 concludes the chapter.

Related Work

3.1 Approaches taking advantage of past queries through user sessions

In the last decades, two trends of personalization research have appeared. One has its origin in the document space in the area of Personalized Information Retrieval (PIR) meanwhile the other has emerged from the hypertext space in the branch of Adaptive Hypermedia (AH). PIR is based on the search results and focuses mainly on personalization of relevant information by varying traditional document ranking algorithms. Personalization techniques involved in PIR, tend to typify users with simplified characters (usually rooted in historic interests), by which it is feasible to calculate an efficient way personalized ranked lists. Otherwise, Adaptive Hypermedia (AH) tackles the challenge of distortion for content retrieval and presentation by considering several characteristics. Such challenges are denominated "dimensions", which consider either user objectives or prior knowledge. The dimensions previously mentioned make possible personalized result compositions and adaptive navigations. Steichen et al. (2012) present a survey whose aim is to investigate the main techniques and their impacts on the use of PIR and AH technologies. To analyze these techniques several activities in the retrieval process are examined : query adaptation, adaptive retrieval, adaptive result composition and presentation. The conclusions in line with this thesis, and related to past queries, concern query adaptation and adaptive retrieval. For query adaptation, systems either require user involvement such as ad-hoc relevance judgments or they create user models involving keyword and category classifications. Finally, in adaptive retrieval, user past searches are used to disambiguate a query, where the terms can be matched to several categories. However, the approach presented in this thesis differs from those described in this survey because this thesis reuses the relevant documents obtained from the most similar past query aiming to respond a new query (i.e., a new query is a query that has not been submitted in an IRS).

Other approaches focus on the query level. Fonseca et al. (2003) proposed an automatic method to produce suggestions of related queries (i.e., related queries are similar queries with respect to the queries submitted previously in an IRS) based on previously submitted queries. To achieve this goal, an algorithm of association rules was applied on log files. In addition, a simple method of query expansion using related queries was experimented. Experiments carried out on popular search engine in Brazil showed accuracy of suggestions produced and superior results for the expanded queries compared with the initial ones. However, Baeza-Yates et al. (2004) claimed that there is no easy way to calculate the

real effect of approaches founded on association rules. It is mainly due to the complexity to determine the successive queries that belong to the same session (i.e., for the same user). Despite this work is related to the approach presented in this thesis, it differs since it uses related query definitions only and not their associated results. However, it could be used to suggest related queries of which retrieved results could be used in this thesis approach.

More recently, Bedi and Chawla (2010) followed the same idea by proposing a framework of personalized web search founded on agents and clustering. To provide a personalized search on the web, agents based on information retrieval systems accomplish their purpose by clustering the query sessions of users applying information scent. Query disambiguation is addressed through hub and authority recommendations by using user profiles, information scent, and clusters of query sessions on the web. Disambiguation is performed by tightly coupled multi-agent systems. Among some agents used in this framework, the interface agent provides related queries in the same domain. The historical query selected by the user is used to find the cluster corresponding to the most similar information need. Coordination agent mines the query session database, where the information is gathered from user agents. Aiming to group similar query sessions in cluster/subclusters, a clustering algorithm is applied. Three domains are addressed in the experimental environment - Academics, Entertainment and Sport. The dataset is built by deploying the clicks of anonymous users. The number of URLs in the dataset was 2995. The dataset was preprocessed obtaining 595 query sessions. The similarity between two query sessions was calculated using the cosine measure. The final results show a better precision for trained queries than untrained queries. Nevertheless, it is important to point out that no explanations are given on how the judgments of users were obtained, and which precision was exactly calculated. This work is in line with our approach variant based on clustering but differs in using clustering based on query definitions only.

Still at the query level, Cui et al. (2003) proposed a method to improve retrieval effectiveness for new queries through automatic query expansion. The method expansion bases on log files to extract correlations between query terms and terms of relevant documents. The central idea of the method is that the terms of documents often selected for the same queries are strongly related to the terms of these queries. Implicit relevance judgments are extracted from user logs assuming that a document chosen by a user is "relevant". Consequently, relationships between query terms and document terms are used to expand new queries. A set of experiments, which take into consideration both long and short queries, enables to appreciate that the log-based query expansion method provides important improvements regarding effectiveness. Final results show that query expansion is more effective for short queries than for long queries. As this work, this thesis reuses past relevant results to improve effectiveness of new queries (i.e., new queries are queries that have not been submitted in an IRS). However, this work uses terms from past relevant documents to expand a new submitted query while our approach proposes to reuse past relevant documents

to respond to a new query.

When it comes to retrieval results, Shen et al. (2005) proposed models based on implicit feedback information to enhance precision during a search session performed by a user. Implicit feedback information is given by queries and click-through history in an active session (i.e., the set of queries is provided by the same user, in the same domain). Bayesian interpolation has been applied to rerank some documents, which have not been seen by a user. The TREC AP data have been used to create a test collection with the aim to evaluate quantitatively the model. Empirical results showed that the use of implicit feedback can improve the final results. Differing from this work, this thesis proposes an approach (not considering user sessions) that reuses the past queries submitted by all the users and not only by the user who submits a new query.

Eventually, Dou et al. (2007) promoted a framework in which it is possible to evaluate large-scale personalized search. In this framework, click-through data recorded in query logs are used to simulate user experience in web search. Furthermore, clicking decisions are employed as relevance judgments to evaluate precision. A new measure denominated click entropy of query is defined. Lower click entropy implies that most of the users agree on a small number of web pages. Therefore, personalization is not necessary in these cases. By contrast, large click entropy implies that many web pages were clicked for the query. This corresponds to two cases : First, a user selects several pages because his/her query has not been satisfied and therefore the query corresponds to an informational query. In this case, personalization can help the user filtering the most relevant pages by making use of historical selections. Second, different users made different selections for this query, which implies that the query is ambiguous. In such cases, personalization can be used to provide different web pages to each individual. Experimental scenarios were based on 12 days of MSN query logs with the aim to evaluate five personalized search strategies. Empirical results showed that click-based personalization strategies provide good results. Although this work seems to be more far from this thesis than the previous ones, it could be interesting to complete our approach by estimating the interest of reusing some past queries.

3.2 Approaches taking advantage of past queries without user sessions

The approach proposed in this thesis is in line with the approaches presented in this section, as it does not exploit the notion of search sessions and their possible additional data about users.

For instance, Raghavan and Sever (1995) proposed to use a set of persistent past optimal queries (and their retrieved results) (i.e., persistent past optimal queries, are stored queries in an IRS which split relevant documents and non-

relevant document using a linear classifier). They identified to possible uses : to answer a new query with the retrieval result associated to a similar persistent query when it exists, or to reformulate a new query replacing it by an optimal persistent one. Not addressing these issues, Raghavan and Sever (1995) focused on a preliminary common issue, which concerns the choice of similarity measures between queries. The authors claim that the queries can be seen as elements of the function space according to retrieval functions, which are defined on the document space. In other words, the comparison between two queries should be based on the relationship of their retrieval outputs. As a result, a new measure of similarity between two queries is proposed, based on result rankings of two queries and experimented for the second aforementioned issue. The experimental study on the Cranfield collection was not conclusive, however the result showed that the hypothesis held when the query base has a small number of persistent queries. Although this work does not address the same issue as this thesis, it is important since it sheds light on the issue tackled in thesis, related to reusing directly past results to answer a new query.

Fitzpatrick and Dent (1997) described experiments grounded on past queries as source of evidence for automatic query expansion. The main assumption is that the top documents recovered from result lists of similar past queries are a good source to improve automatic query expansion. In addition, a new query similarity metric is proposed, which compares overlap between result lists of two queries. Weights used for a position in the result lists are determined by the likelihood to find a relevant document in that position. Experimental scenarios on TREC collections compared performance of automatic feedback based on similar past queries with standard top-document feedback, and without feedback. Empirical results showed superior performance of the approach based on past queries. As in this work, this thesis is based on the assumption that result lists of similar past queries are a good source to improve effectiveness for new queries. However, it proposes to use directly past results to build new results rather than performing query expansion and then a full retrieval process.

In a different manner, Hust (2004) proposed a work close to the approach presented in this thesis in terms of motivation, goal, and reused past elements. He proposed a collaborative approach of query expansion based on past queries (and their associated results) submitted by different users. He developed and experimented various expansion methods (e.g., sum of selected relevant documents, linear combination of past queries, and query term reweighting) based on the relevant documents of the most similar queries for query expansion of a new query. Experiments did not enable to establish strong conclusion about effectiveness of new expansion methods. An explanation claimed by Hust is a possible inappropriateness of the collections used, not existing more appropriate ones, which is still the case. Despite being close to our approach, a main difference is that this work uses terms from past relevant documents at the query formulation level while our approach intends to study the interest of reuse past relevant documents to respond directly to a new query.

Another similar approach with respect to this thesis was presented by Cetintas et al. (2011) in the domain of distributed information retrieval. In distributed IR a unified search interface is used for multiple search engines of distributed text information sources. In this domain, resource selection is an important component. Cetintas et al. (2011) proposed an approach of resource selection based on similar queries and their associated past results. The approach estimates the utilities of available information sources by combining results of past queries with respect to similarities between a particular query and all past queries. Pointing out the lack of available collections comprising similar queries to evaluate their approach, Cetintas et al. (2011) proposed to generate simulated similar past queries from traditional TREC sets of queries, by extracting titles of some top-ranked documents or removing some terms of the queries. This work is close to this thesis as it is based on similar past queries (using cosine as in our approach) and the ranking of their associated results. In addition, evaluation used generated similar queries by simulation. However, this work takes place in a different domain, which is distributed IR. The approach aims at improving distributed IR through resource selection. Query generation by simulation only concerns query formulation since the approach proposed for resource selection does not exploit relevance judgements. A contribution of this thesis concerns simulation of relevance judgements on results associated to similar queries.

More recently, Song and Myaeng (2012) proposed a method that relies on a novel term weighting. Their main assumption was that the term role in accumulated retrieval sessions in the past has an importance in the retrieval process, which cannot be ignored. It takes into account the availability of past retrieval results, by exploiting the queries, their retrieved documents, and their relevance judgments. More specifically, the method considers the rankings and similarity values of the relevant and non-relevant documents to compute a term evidential weight. In addition, a measure of discrimination called discrimination power (DP) is proposed to apply a new term weighting scheme for new queries, based on past queries and their retrieved results. Experimental results show that the proposed term weighting scheme improves traditional $tf-idf$. This work differs from other ones previously presented by applying a new term weighting scheme to process a new query instead of reformulating or expanding the initial query. However, as them it applies a full retrieval process for the new query while the approach presented in this thesis intends to build the result for a new query directly from past results.

3.3 Conclusions

In this chapter, a review of the state-of-the-art about approaches that take advantages on the use of past searches have been analyzed. Several studies revealed that a non-negligible part of the queries submitted to search engines are similar queries, mainly repeated or reformulated queries (i.e., reformulated

queries are modify queries)during search sessions performed by the same user, but also similar queries submitted by different users.

An important research part is dedicated to repeated queries. In particular, various approaches deal with caching. However, this type of approaches has been shortly presented since not closely related to the issues tackled in this thesis. Thus, approaches of caching focus on repeated queries with the aim to improve efficiency while this thesis deals with similar queries submitted by different users and is effectiveness oriented.

However, the literature comprises several approaches using past queries, which are related to this thesis and which were presented in this section. They can be separated in two types : approaches taking advantage of past queries through user sessions and approaches taking advantage of past queries without user sessions. The main differences between these two types of approaches are :

— Approaches that exploit user sessions intend to improve new queries submitted by a user by using historical queries and/or results and/or interaction information (e.g., clicks) about the same user ;
— Approaches that do not use user sessions intend to improve new queries submitted by any user by using all past queries and/or past results, including those performed by other users. A perceptible problem for this type of approaches is the lack of suitable collections to truly evaluate their effectiveness. This thesis takes place in this category.

The issues tackled by these two types of approaches by using elements of past searches concern :

— Measure query similarities. These approaches propose new similarity measures to find queries similar to a given one based not only on query definition as usual by also based on query results. However, these measures require processing an initial retrieval for a new query to find similar past queries.
— Automatically expand or reformulate new queries. These approaches propose to use past searches to modify the new submitted query before performing a full retrieval process.
— Change matching function or ranking for new queries. These approaches use past searches to rerank an initial result returned for a new query or modify term weightings to process the new query.

This thesis follows the same motivation and goals of such approaches, but differs from the way to reuse past searches. However, it studies the interest of proposing a different approach that use similar past queries, and more particularly past results, to build results of new queries. The approach rather acts at the result level than at the query level. In fact, in chapter 5, several algorithms which use this scheme are presented. In addition, a problem noticed in related approaches of the literature is the difficulty to evaluate such approaches, due to the lack

of suitable collections. To tackle this issue, this thesis proposes a method approach to build such collections through simulation. This method is presented in Chapter 4.

Chapter 4

Simulation framework for evaluation of IR approaches reusing past searches

Résumé : Chapitre 4

Comme évoqué dans le chapitre 3, plusieurs approches visent à bénéficier de requêtes précédemment soumises, la plupart d'entre elles utilisant les requêtes répétées pour améliorer les temps de réponse. Ces approches sont principalement liées à la mise en cache. Un deuxième type d'approches vise à étendre ou modifier les requêtes pour améliorer les résultats de recherche, en particulier pour les utilisateurs connectés à un serveur, à savoir, au travers des sessions utilisateur. Avec les sessions utilisateur davantage d'informations concernant l'utilisateur, telles que ses préférences, et son comportement, sont stockées et peuvent donc être exploitées pour de nouvelles requêtes soumises par le même utilisateur. D'autres approches proposent de nouvelles fonctions d'appariement (basées principalement sur les clics) ou de nouvelles pondérations des termes. Néanmoins, quelques approches (Hust, 2004; Cetintas et al., 2011) ont étudié directement la réutilisation des documents pertinents retournés pour les requêtes passées similaires. Une raison peut être un manque de collections de RI appropriées pour évaluer ces systèmes. La plupart des travaux connexes (présentés dans le chapitre 3) sont basés sur des collections de logs, qui ne fournissent pas les jugements de pertinence associés aux documents retournés pour une requête spécifique. De telles approches visent à proposer une liste de documents susceptibles d'être cliqués par l'utilisateur. Contrairement aux collections de logs, les collections de RI traditionnelles sont composées d'un ensemble de documents, un ensemble de requêtes, et un ensemble de jugements

de pertinence des utilisateurs sur les documents retournés pour les requêtes. La création d'une collection de RI appropriée est très coûteuse en temps et efforts, bien que des solutions pour réduire le coût ont été proposées (Sanderson, 2010; Alonso and Mizzaro, 2012). Une solution envisagée pour l'évaluation des approches de RI est la simulation (Azzopardi et al., 2011). Cependant, peu de travaux traitent de simulation en RI, principalement en RI interactive. Deux catégories d'approches peuvent être distinguées :

— les approches qui simulent les utilisateurs, dans le but d'explorer des stratégies de recherche, actions ou étapes fournies par les utilisateurs, ou d'analyser comment les requêtes sont soumises par les utilisateurs.
— les approches utilisées pour évaluer ou concevoir des systèmes de RI.

En outre, à notre connaissance, aucune simulation de collection complète de RI n'a été proposée pour des approches de réutilisation de requêtes et résultats passés. Un travail proche vis-à-vis de cette thèse, est proposé par Cetintas et al. (2011), utilisant une simulation de requêtes à partir d'une collection TREC dans le cadre de la RI distribuée. Cependant, une limite de ce travail est que les jugements de pertinence ne sont pas considérés.

Ainsi, l'un des principaux objectifs de cette thèse, est d'évaluer l'impact des approches basées sur les requêtes passées similaires, ce qui n'a été que très peu étudié. Sans collection ad hoc, une alternative possible pour étudier les approches qui réutilisent les requêtes passées est la simulation (Gutiérrez-Soto and Hubert, 2013), incluant non seulement les requêtes ou les documents, mais également les jugements de pertinence, sans interaction avec l'utilisateur (Huurnink et al., 2010).

La principale contribution de ce chapitre est un cadre qui simule des collections, offrant des environnements ad-hoc pour évaluer les approches reposant sur l'utilisation de requêtes passées similaires. Une première série d'expériences a été réalisée pour évaluer la possibilité de créer différents types de collections.

Le chapitre est organisé comme suit. Dans la section 4.2, les travaux connexes sur les approches qui traitent de la simulation en RI sont présentés. Dans la section 4.3, notre cadre de simulation est décrit et la section 4.4 introduit l'utilisation des résultats de recherche passées. Dans la section 4.5, des expériences analysant la faisabilité de simuler différents types de collections sont présentés. Enfin, la section 4.6 rapporte les conclusions finales.

4.1 Introduction

As stated in chapter 3, several approaches aims at taking advantages from previously submitted queries, most of them benefiting from repeated queries to improve response times. These approaches are mainly related to caching. A second type of approaches aims at expanding or modifying queries to improve retrieval results, in particular for users logged to a server, i.e., through user sessions. With user sessions more information about the user, his/her preferences, and his/her behavior are stored and thus can be exploited for new queries submitted by the same user than without user sessions. Other approaches apply modifications on ranking functions (i.e., which are based primarily on clickthrough) and/or on internal weighting schemes of terms. Nevertheless, few approaches (Hust, 2004; Cetintas et al., 2011) studied directly the reuse of relevant documents returned for similar past queries. A reason may be a lack of suitable IR collections to evaluate such systems. Most of related work (presented in Chapter 3) are based on log collections, which do not provide relevance judgments of users associated to the documents returned for a specific query. Such approaches aims at proposing a list of documents susceptible to be clicked by the user. Unlike log collections, classical IR collections are composed of a set of documents (D), a set of queries (Q), and a set of relevance judgments of users (RD) on documents (i.e., a group of expert users determines if a document is relevant or non-relevant for a given query) for the queries that belong to Q. The usual creation of an appropriate IR collection is very costly in time and efforts, though solutions to reduce the cost have been proposed (Sanderson, 2010; Alonso and Mizzaro, 2012). An alternative considered for evaluation of IR approaches is simulation (Azzopardi et al., 2011). However, few works deal with simulation in IR, mostly in interactive IR (IIR). Two categories of approaches can be distinguished :

— Approaches that simulate users, with the aim to explore search strategies, actions or step provided by users, or how queries are submitted by users.
— Approaches used to evaluate or design IR systems.

Furthermore, as far as we know, none proposed simulation of complete IR collection for approaches reusing past retrieval queries and results. Indeed, the most similar work with respect to this thesis, is proposed by Cetintas et al. (2011), which uses a traditional TREC collection in the domain of distributed IR. However, a drawback of this work is that no relevance judgments are considered.

Thus, one of the main objectives in this thesis, is to evaluate the impact of approaches in the domain of similar past queries, which has not been deeply studied. Without ad-hoc collections, a feasible alternative to study approaches that reuse similar past queries is simulation (Gutiérrez-Soto and Hubert, 2013), including not only queries or documents but also relevance judgments, without user interaction (Huurnink et al., 2010).

The main contribution of this chapter is a framework, which simulates collections, providing ad-hoc environments to evaluate approaches relying on the use

of similar past queries. A first series of experiments were carried out to evaluate the feasibility of creating different kinds of collections.

The chapter is organized as follows. In section 4.2, related work about approaches which deal with simulation in IR is presented. In section 4.3, our simulation framework is described and section 4.4 introduces retrieval using past search results. In section 4.5, experiments analyzing the feasibility to simulate different kinds of collections are reported. Finally, section 4.6 reports final conclusions.

4.2 Related Work on simulation in IR

A wide range of approaches which deal with interactive information retrieval (IIR) can be found in the IR literature, where several of these approaches simulate users with the purpose to explore search strategies and how queries are submitted by users.

In the domain of IIR and evaluations, White et al. (2005) presented an evaluation of relevance feedback (RF) algorithms by using searcher simulations that emulate the interaction of searchers. Six different models (e.g., Binary Voting model, WPQ-based models) applying query modifications were evaluated. The authors point out that there is not a standard way to evaluate term selection models that require complex or copious searcher interaction with result interfaces. The simulation-based approach did not model factors such as types of users, search experience, or types of information needs. Various strategies were used to model simulated searchers, such as only viewing relevant/non relevant documents, or viewing all relevant or non relevant information. To operate effectively, implicit feedback models should handle different retrieval situations. To achieve this goal, models were tested in extreme and pre-modelled situations. In extreme situations only relevant or non-relevant paths are traversed. Pre-modelled situations assume that searchers try to interact with relevant information, but they also accept inevitably to view non-relevant information. This assumption was based on the intuition about how searchers generally interact with search systems. As this thesis, this work proposed to use simulation for evaluating different RF systems to avoid soliciting human subjects to interact with these systems. However, differing from this thesis, this work has different objectives requiring to simulate various scenarios of user interactions.

Still in the domain of IIR, Kelly (2009) presented an overview about Wizard of Oz studies. Wizard of Oz studies are inspired by the well-known film/book with the same title. In such studies, researchers project the characteristic of "grand" systems that they wish to study. On one hand, users have the perception that they are interacting with a real system, meanwhile one or more researchers provide the functionality. The objectives about Wizard of Oz studies are serving as proof-of-concept and providing an idea about how things should happen in

ideal circumstances. Wizard of Oz studies correspond to system simulations, but users can also be simulated with the aim to exercise systems. The idea behind simulated users is that they can also represent different actions or steps, which should be considered for real users when interacting with an IR system. An advantage of simulated users is that IIR evaluations can be conducted quickly and with less cost. As in this work, we claim that preliminary evaluations of IRSs can be carried out more early and inexpensively through simulation. It is important to highlight that this thesis does not fit into the same domain. In this thesis, users are not directly simulated and therefore their actions or steps are not studied.

Continuing the trend, Yue and Joachims (2009) proposed an approach for on-line optimization of IRSs. They presented an on-line learning framework relying on pairwise comparisons, which can learn in real-time from observed user behaviour in search engines and other information retrieval systems. They used simulation for experiments based on a real Web Search dataset. The idea was to simulate users issuing queries for training, by sampling from queries in the dataset. For each query, the competing retrieval functions produce rankings, where subsequently the "user" randomly prefers one ranking over the other. This dataset was used in the first step to simulate the user behaviour over on-line learning setting. Although this work does not tackle the same challenges as this thesis, this work is related with regard to simulation of users preferring a result ranking to another. However this thesis intends to propose a method to simulate relevance judgments on documents constituting a result list rather than preference between result lists.

More recently, Azzopardi et al. (2011) proposed to consider the search process in IR as an economic issue. The interaction process between a user and a system can be modelled as a series of inputs (queries, assessments, among others), which produces an output (utility/gain from finding relevant items). Although the production process is not exactly the same as the search process, the relevant documents found provide users some utility or gain. This work use simulation to explore a set of possible search strategies, which could be used by users. Specifically, search sessions are simulated considering a number of queries per topic. Aiming to obtain a desired level of Cumulative Gain utility, a sequence of interactions is provided for a greedy best-first approach to select the subset of required queries. In order to evaluate this approach, a set of simulations on three TREC collections were carried out taking into account several combinations of inputs in the production of relevance. Results were analysed to determine the relationship between the total Cumulative Gain obtained during a search session and the number of queries issued as well as the number of documents assessed per query. A conclusion was that this relationship can be represented mathematically by the Cobbs-Douglas production function. Such as in this work, in this thesis queries are simulated. However, this thesis differs from this work by not considering the notion of user session, as well as sequences of interactions in order to provide required queries. Furthermore, the exploration of various search strategies is not considered as an objective in this thesis. An objective

of this thesis and presented in this chapter is to simulate different kinds of IR collection suitable for evaluation of system reusing past searches.

Apart from IIR, few work on simulation in IR can be found in the literature. A work close to the approach presented in this chapter was proposed by Cooper (1973) who proposed a simulation model for designing and evaluating IRSs. The simulator enabled building specified collections of documents and analysing the effect of changes in query characteristics on the quantity of output produced by IRSs. The simulator is composed of five parts : a thesaurus generator, a pseudo-document generator, a pseudo-query generator, search routines, and evaluation routines. Several parameters can vary for thesaurus generation (e.g., vocabulary size, word frequency distribution), document generation (e.g., number of documents to be generated), query generation (e.g., mean and standard deviation of the length of a query). The search and evaluation routines used in the simulation study were quite simple. The search routine measured the extent to which the queries matched the documents, and the evaluation routine tested how many documents were retrieved for a given threshold. A conclusion of this work was that the proposed simulation model provides a limited but useful framework for the evaluation of information retrieval systems. However, a limit of this model is that relevance judgements are not considered, and thus usual IR evaluation is not feasible. Following the same idea, this thesis proposes a framework for IR collection situation, in which it is feasible to build words (i.e., vocabularies), documents, and then queries. Our framework extends functionalities enabling to build documents and queries under different probability distributions (i.e., words can be chosen under different probability distributions). Furthermore, the proposed framework enables generating relevance judgments of users. So, several retrieval scenarios can be adequately evaluated.

In addition, extending a previous work, Tague and Nelson (1981) proposed a general model and simulation algorithms for bibliographic retrieval systems. Bibliographic retrieval systems are characterized by random behaviour in the sense that the relationship between queries and documents to answer must be described probabilistically rather than deterministically. The proposed model relies on various parameters such as probability functions for the distribution of terms over queries and documents, and the distribution of relevance, given a query, over documents. The model aims at being used to evaluate both the effectiveness (utility) and efficiency (complexity) of bibliographic retrieval systems. Replicative validity (i.e., produced data match already acquired data) were tested using two bibliographic retrieval test collections, Cranfield and Medlars. Although the experiments showed that simulated data appeared to fit to the real data in a reasonable way, the results led to significant differences w.r.t recall-precision evaluation. However, this model must be regarded as an interesting preliminary work. Similar to this work, the proposed framework in this chapter of this thesis relies on different parameters to build documents and queries under different probability distributions. Nevertheless, the set of terms, the number of documents and number of queries are instantiated at the beginning of each simulation. Our framework relies on well-known laws of information science used

in various research fields such as Digital Libraries and Information Retrieval (Chen and Leimkuhler, 1986; Sparck Jones and Willett, 1997b; Schaer, 2013). For example, in our framework the judgments of users are simulated using the Bradford's law.

Finally, as presented in Chapter 3, a work close to this thesis in the sense that it is based on the reuse of past results but differing in the objectives, was proposed by Cetintas et al. (2011). Pointing out the lack of available collections comprising similar queries to evaluate their approach, they proposed to generate simulated similar past queries from traditional TREC sets of queries, by extracting titles of some top-ranked documents for an initial query or removing some terms of the queries. However, an important drawback of this work is that relevant judgments were not considered in the proposed approach.

4.3 Simulation Framework

As previously introduced in both Chapter 3 and the previous section, few work aims at reusing past queries and past results to improve new queries. Maybe it is due to the lack of suitable collection for evaluation with respect to this thesis. Maybe it also contributes to the absence of new collection creation for the evaluation of such systems. Although simulation is seen as an interesting alternative for evaluation of IR approaches (Azzopardi et al., 2011), rare approaches were proposed for full IR collections. Furthermore, none could respond to our needs with regard to evaluation of the reuse approach proposed in this thesis.

From our point of view, the same relevant documents are usually answers to similar queries. However, all relevant documents are not necessarily relevant for two similar queries. Our assumption is that most of relevant documents for a past query could be relevant for a new similar query, and the judgments of users on past results should be exploited.

This section presents a framework that allows simulating traditional IR collections, which are formed by a set of documents, a set of queries, and the judgments of users on the documents. Therefore, one of the main objectives of this framework is to provide an ideal environment to evaluate approaches based on similar past queries, where both sets of queries (i.e., which are similar between them) having their own judgments of users. It is important to mention that this framework is used in Chapter 5 to evaluate randomized algorithms in the domain of similar past queries, because it supplies a clean experimental environment (i.e., stemming, stop-word removal are omitted) and robust (i.e, final results do not depend on particular characteristics of a system). The building of this framework comprises three steps : the creation of documents and queries, the simulation of users' judgments, and the retrieval using past queries. This framework relies on well-known laws of information science, such as Zipf's and Bradfor's laws, used in various research fields such as Digital Libraries and

Information Retrieval (Chen and Leimkuhler, 1986; Sparck Jones and Willett, 1997a; Schaer, 2013).

Typically, in traditional collections a document has a title and a body (i.e., which contains some kinds of information that can be useful for a user). Both title and document body are formed by a set of terms (i.e., words which form the document). Usually, document terms provide semantics about some particular subject. For example, a document about the 2010 Chile earthquake can contain terms such as plate, Nazca, disaster, geology, and Chile, but geology and Chile can belong to other documents with different information such as geology of Chile. Thus, the sets plate, Nazca, disaster, earthquake and geology, Chile are denominated topics, where each topic expresses a particular subject. In the last example, the first document has terms from the second topic. However, it is not rare in the real world to have documents containing terms from different topics. Furthermore, using specific terms in a document (i.e., terms related to a particular topic) it is possible to determine the topic it deals with and thus decide if a document is relevant or non-relevant for a specific query (i.e., the judgments of users are applied to the documents according to the relevance for a query). Therefore, in this framework various topics are considered and a document can have terms from different topics such as it happens in the real world.

Like the documents, also queries are formed by terms. The judgments of users determine if a document is or non-relevant for a specific query. The simulation of documents, queries and judgments of users are specified in sections 4.3.2 and 4.3.3. In the following section, some definitions and notations are provided.

4.3.1 Definitions and notations

Broadly speaking, each document is composed of terms, which are obtained from English alphabet. A set of terms that has a semantic relation between them is denominated topic. In a topic there are not repeated terms and the intersection between topics is empty. Similar to the documents, the past and new queries are composed of terms, however for each type of query, terms are chosen in a different way (*see section 4.3.2*).

Definition 1. Let To be a topic composed by terms, where a term is a finite set of letters that belongs to the English alphabet. In other words, To is composed of words, for example car, wheel, brake among others.

a) $\forall t_i \in To$, t_i is unique.

Definition 2. Let d be a document conformed by a set finite terms that belongs to different topics.

Definition 3. Let D be a finite set of documents.

Definition 4. Let q be a query with a finite number of terms, which is built using terms from a particular document.

Definition 5. Let $DB = D \cup Q$ be an IR dataset, composed of a set of documents D and a finite set of past queries Q.

a) $\forall d \in D \wedge \forall q \in Q$, both d and q are unique.

By this definition, we have a collection of documents D, and a finite collection of past queries Q.
The point a) indicates that there are either equal documents nor equal queries.

Definition 6. Let Q' be a finite set of new queries, such as every query is unique..

Definition 7. Let $V_N(q)$ be a set of N retrieved documents given q.

Definition 8. Let $A(q)$ be the set of all the relevant documents retrieved for the query q, such as $A(q) \subseteq V_N(q)$.

Definition 9. Let $A'(q)$ be the set of all the non-relevant documents retrieved for the query q, such as $A'(q) \subseteq V_N(q)$.

Definition 10. Given a query q, such as $q \in Q$, $p(q) = (q, V_N(q))$ is defined as the profile of the query q, which is a pair composed of the query and its retrieved documents.

Definition 11. Let $P = \bigcup p(q)$ be the finite set of all the query profiles stored in an IRS.

Definition 12. Given a query q', such as $q' \in Q'$. $R_p(q', P)$ corresponds to the set of retrieved documents from the most similar past query in P according to a similarity measure (e.g., cosine).

Definition 13. $\partial : R_p(q', P) \rightarrow A(q')$ is a function, which assigns the most relevant documents to a new query q', such as $q' \in Q'$ (see *Definition 10* and *Definition 12*).

In the framework, the simulation is carried out as follows. First of all, the terms are built and each term is unique. Subsequently, the terms are split in topics, whereby there are not common terms between topics. From the terms, the documents are built. The past queries are obtained from the documents and new similar queries are obtained from past queries. The judgments of users are simulated using the Bradford's law. The creation of documents, queries and the judgments are explained in detail in the following section.

4.3.2 Creation of documents and queries

In this first step, the English alphabet to build a set of terms is used. Each term is composed of letters of this alphabet. This set of terms can be split in subsets

called topics in order to represent different subjects (i.e., a topic is a set of terms, which belong to a specific area such as biology, chemistry, physic to mention a few). Each letter is chosen using uniform distribution with the purpose to build a term. Thus, each term is unique. In addition, each document is defined according to all the topics. In order to build a document, topics are selected using either the exponential or Zipf distribution and then terms constituting the document are chosen using uniform distribution. Thus, a document is constructed with terms from one topic mainly but not exclusively. Past queries are built from documents. To built a past query, a document is chosen under uniform distribution. The terms that constitute the query are chosen from the document under uniform distribution. It is important to emphasize that the intersection among past queries is empty, that is, they do not have terms in common. New queries are then built from past queries. For each past query a new query is built, either by changing or adding another term. Thus, the most similar query for the new query is its corresponding past query.

4.3.3 Simulating relevant judgments

In order to simulate the decision given by a user about if a document is either relevant or not relevant for a given query, Zeta distribution have been implemented. Zeta distribution gives a discrete approach of Bradford's law. Bradford's law says, among the production of journal papers, there is an heterogeneous number of papers where the most relevant papers are in few journals, while a few number of relevant papers are spread on a high quantity of journals. In our case, for a given query, it means that the most relevant documents should be at the top of the list because they are the most similar with respect to the query (this is, the most relevant papers are in few journal), while a few relevant documents should be spread at down of the list document given a query.

On the other hand, the hypothesis is that for two very similar queries a and b, when a document is relevant for a query, it could be also relevant for the other query. In an intuitive way, there is a subset of common relevant documents for both queries. It does not implies that all relevant documents for query a, are relevant documents for query b. Therefore, when measuring effectiveness, for instance with precision at ten retrieved documents (P@10), precision for query a is not necessarily the same as for query b. With the objective to simulate this scenario, Zeta distribution is used to determine relevant documents over a subset of common documents between similar queries. Afterwards, and with the purpose to have all relevant documents for each query, Zeta distribution is applied again on list of documents conserving the relevant documents from the subset of common documents. For example in Figure 4.1, relevant documents for each query have 1, non-relevant documents have 0. First, both list of documents are retrieved, from the intersection (d_3, d_5, and d_6), relevant documents common to both queries are computed by using Zeta distribution. Afterwards, relevant documents are calculated again for each

query by preserving the relevant documents of the intersection (d_3 and d_5). Finally, P@10 for query q is not necessarily the same that for query q'.

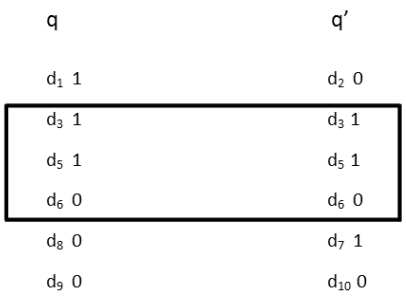

FIGURE 4.1: Obtaining judgments from users

4.4 Retrieval using past queries

In simple words, the basic idea behind our approach is to incorporate to the IRS every query with its set of associated documents (query along with its list of documents, which are part of the answer of this query). The system has not only the set of documents but also the queries submitted by users (past queries) with the set of associated documents. At the beginning there are just documents without the queries, but every time a query is given in the system, it is aggregated with its documents to the system. When a new query is submitted, first, it is checked and compared with the past queries, which are in the system. From the most similar past query, the most relevant documents can be obtained. Our method consists of two parts. First, the query executed is stored with its documents. Second, each new query is checked and compared with the past queries in the system, thus if there is a similar past query in the system, the relevant documents of this past query are retrieved. Relevant documents are used to respond the new query.

Method in detail

In a more precise way, originally there is a set of documents D. The set of queries Q is empty. At the beginning the first query q_1 is given by user, it is checked only with all documents in D. Then N documents are retrieved for q_1. Now, this set of documents is associated with the query q_1 in a new set, and this set is added to the system. From that Q set is no more empty.

When a new query is submitted to the system, first, it is checked and compared with past queries from the system. If it exists a query enough

similar in the system, then a set of documents are obtained. At the same time, every query must be checked and compared in the traditional way with the documents of the system. From both, it is possible to compare our approach with traditional retrieval.

4.5 Experiments

4.5.1 Design of Information Retrieval Benchmark

A typical IR collection is composed of three sets : documents, queries and relevance judgments per query (i.e., documents considered as relevant or non-relevant) (Voorhees and Harman, 2005). Aiming to build a test collection, two steps are carried out (Gutiérrez-Soto and Hubert, 2013). The first step aims at creating terms, documents, and queries. Both Heap's law and Zipf's law have been considered to build document collections. Heaps' law indicates that a document with size $O(n)$ (where n is the number of terms) has a vocabulary with size (O^{β}), where $0 < \beta < 1$. Therefore, a simulated document can represent a document written in English of $O(n^2)$ terms (Navarro et al., 2000; Silva de Moura et al., 2000). We assume that both processes, elimination of stop words and stemming were carried out. Furthermore, according to the terms that compose each document, documents belong to several subjects (i.e., several topics). Zipf's law (Poosala, 1997; Zipf, 1949) is applied to simulate the distribution of the frequencies of words in the vocabulary (to select the terms from topics). Like Zipf's law, exponential distribution also have been used as an alternative for selecting terms from topics. Otherwise, past queries are created from documents and new queries are built from past queries. Finally, the Bradford's law is used to simulate relevant judgments provided by users about document relevance for a specific query (Garfield, 1980).

4.5.2 Experimental Environment

The experimental environment was set as follows : the length of a term $\mid t \mid$, was between 3 and 7. Uniform distribution is used to establish the length. The number of terms $\mid T \mid$ was 700 in each experiment. The number of terms for each document can be between 15 and 30. According to Heaps' law (Heaps, 1978), it was possible to represent documents between 300 to 900 words in this case. The number of topics used in each experiment was 7. Each topic is formed by 100 terms. When a document was built, terms of other topics were chosen using either Exponential distribution or Zipf distribution. Whereby, most words, which composed a document were chosen from a specific topic. The numbers of documents in each experiment were 700, 1400, 2100, 2800 and 3500.

On the other hand, terms for a query were between 3 and 8. Query terms were chosen from a particular document. Both terms and documents were chosen using uniform distribution to build the past queries. We built 15 past queries. From the set of past queries, 15 new queries were built. Finally, the number of queries in each experiment was 30.

In order to simulate user judgments, when documents were retrieved given a query q, Zeta distribution have been applied with the purpose to represent the Bradford's law (Garfield, 1980; Zerchaninova, 2008). Zeta distribution was applied, on the 30 most similar documents with respect to the query q, with parameters $2, 3$ and 4, for each experiment. Zeta distribution was applied as follows : After the new query q' was obtained from past query q, the most similar documents for q' was obtained. Afterwards, the common documents for both queries were obtained, and Zeta distribution was used over the set of common documents. It means, there was a subset of relevant documents not only for q' but also q. Eventually, Zeta distribution was applied over q' (other relevant documents could be added to q'), and the relevant documents from the set of common documents were preserved.

4.6 Empirical Results

In this section, three experiments are reported. In, first and second experiments, exponential distribution have been used to build the collection of documents D. In the third experiment, Zipf distribution has been applied to build D. Additionally, we used the Student's Paired t-Test (paired samples) over each average P@10 (our approach with respect to traditional retrieval (using cosine)) for each set of documents (700, 1400, 2100, 2800 and 3500). Preliminary results report the average P@10 over all the queries.

Experiment 1. Exponential distribution with parameter equal to 1.5 was used to build the dataset D. When using zeta distribution with parameter s = 2 for relevance, our approach improved for 26 over 30 queries (see figures 4.2, 4.3, 4.4 and Table 4.1). Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00128 for the set of 700 documents. When using zeta distribution with parameter s = 3, the approach improved for 24.4 over 30 queries on average over the five sets of documents. The highest p-value for the Student's t-test being 0.00056 for 3500 documents. When using zeta distribution with parameter s = 4, the approach improved for 21.4 over 30 queries. Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00732 for 1400 documents.

Experiment 2. Exponential distribution with parameter equal to 1.0 was used to build the dataset D. Zeta distribution with parameter s = 2 for relevance, our approach improved for 26.2 over 30 queries. The average P@10

was improved (see figures 4.5, 4.6, 4.7 and Table 4.2). Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00002 for 1400 documents. Using zeta distribution with s = 3, the approach was improved for 21.6 over 30 queries. Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.01306 for 3500 documents. For zeta distribution with parameter s = 4, the approach was improved for 21.6 over 30 queries. Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00122 for 2800 documents.

Experiment 3. Zipf distribution with parameter equal to 1.6 was used to build the dataset D. Zeta distribution with parameter s = 2 for relevance, our approach was improved for 24.8 over 30 queries. The average P@10 improvement was +25.50. Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00005 for 700 documents. Zeta distribution with parameter s = 3, the approach was improved for 22.6 over 30 queries. The average P@10 was improved (see figures 4.8, 4.9, 4.10 and Table 4.3). Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00034 for 1400 documents. For zeta distribution with parameter s = 4, the approach was improved for 24.8 over 30 queries. Statistical significance was reached in all cases, the highest p-value for the Student's t-test being 0.00031 for 2100 documents.

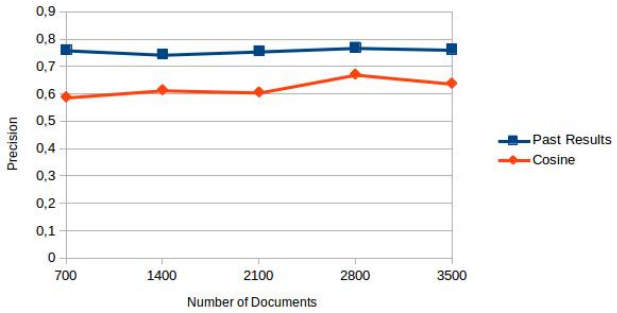

FIGURE 4.2: Simulation with Exponential Distribution 1.0 (for D) and Zeta Distribution 2 (for judgments)

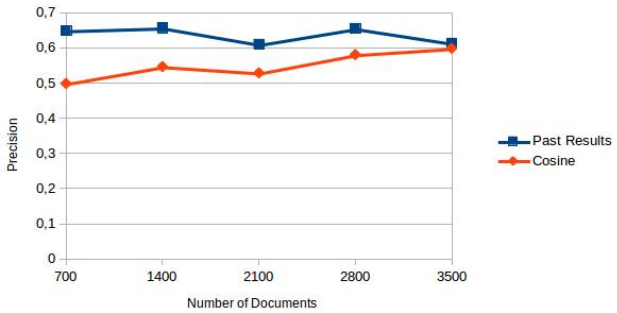

FIGURE 4.3: Simulation with Exponential Distribution 1.0 (for D) and Zeta Distribution 3 (for judgments)

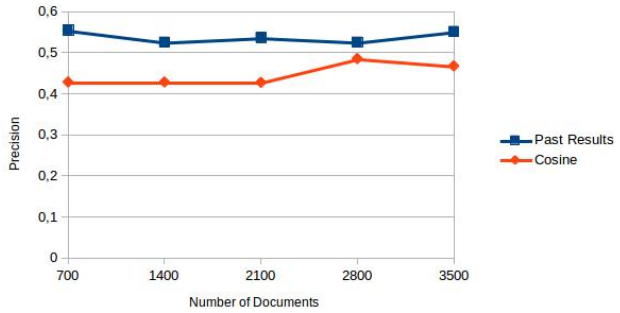

FIGURE 4.4: Simulation with Exponential Distribution 1.0 (for D) and Zeta Distribution 4 (for judgments)

TABLE 4.1: Simulation with a collection D based on exponential distribution using $\theta = 1.0$ – Comparison of the two approaches (i.e., *Past Results* and *Cosine*) according to average P@10 (over 30 queries)

		Approaches	
Zeta Distribution	D Size	Past Results	Cosine
$S = 2$	700	0.761	0.589
	1400	0.745	0.615
	2100	0.757	0.607
	2800	0.770	0.672
	3500	0.763	0.639
$S = 3$	700	0.649	0.499
	1400	0.657	0.547
	2100	0.610	0.529
	2800	0.655	0.581
	3500	0.613	0.559
$S = 4$	700	0.555	0.429
	1400	0.526	0.429
	2100	0.537	0.428
	2800	0.526	0.486
	3500	0.552	0.468

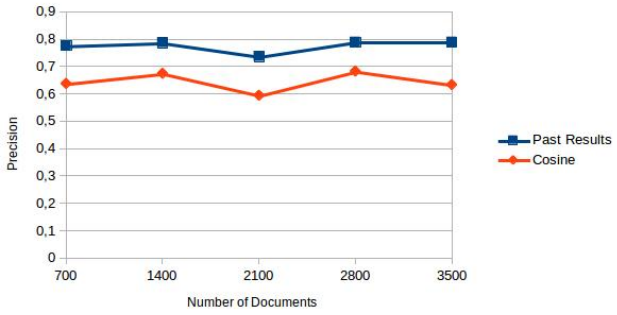

FIGURE 4.5: Simulation with Exponential Distribution 1.5 (for D) and Zeta Distribution 2 (for judgments)

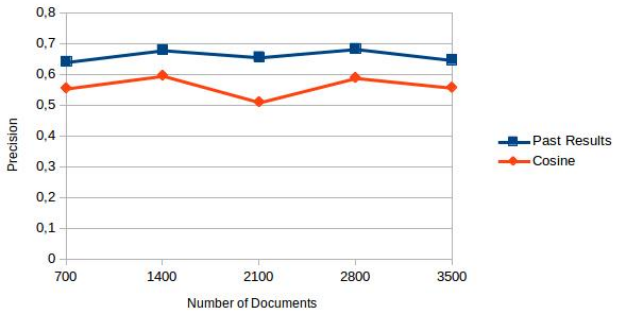

FIGURE 4.6: Simulation with Exponential Distribution 1.5 (for D) and Zeta Distribution 3 (for judgments)

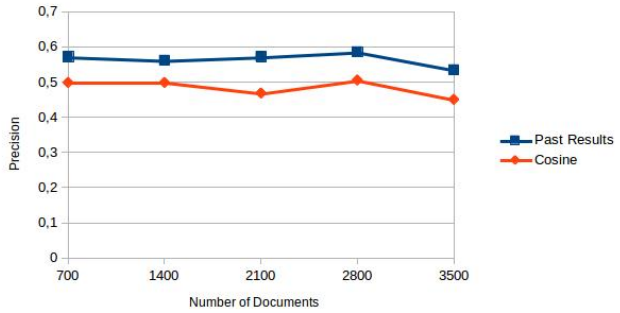

FIGURE 4.7: Simulation with Exponential Distribution 1.5 (for D) and Zeta Distribution 4 (for judgments)

TABLE 4.2: Simulation with a collection D based on exponential distribution using $\theta = 1.5$ – Comparison of the two approaches (i.e., *Past Results* and *Cosine*) according to average P@10 (over 30 queries)

		Approaches	
Zeta Distribution	D Size	Past Results	Cosine
$S = 2$	700	0.776	0.638
	1400	0.787	0.675
	2100	0.737	0.595
	2800	0.789	0.683
	3500	0.789	0.634
$S = 3$	700	0.642	0.556
	1400	0.680	0.597
	2100	0.657	0.511
	2800	0.684	0.590
	3500	0.648	0.558
$S = 4$	700	0.572	0.499
	1400	0.562	0.499
	2100	0.572	0.469
	2800	0.586	0.506
	3500	0.535	0.451

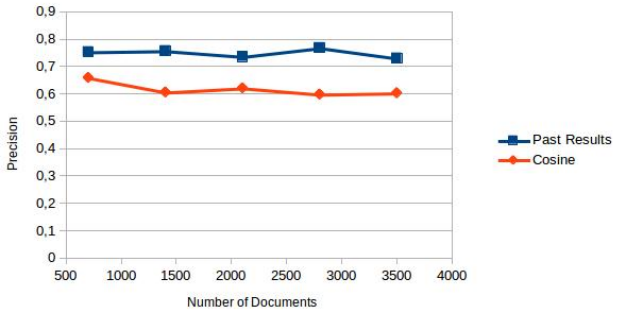

FIGURE 4.8: Simulation with Zipf Distribution 1.6 (for D) and Zeta Distribution 2 (for judgments)

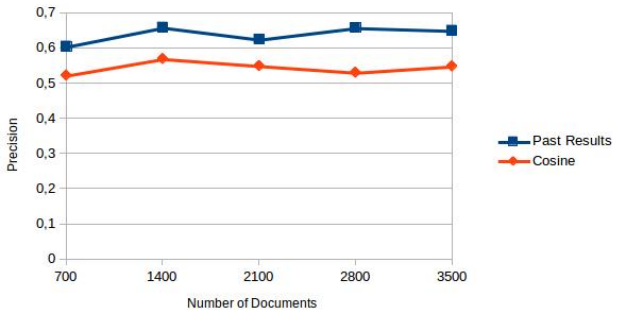

FIGURE 4.9: Simulation with Zipf Distribution 1.6 (for D) and Zeta Distribution 3 (for judgments)

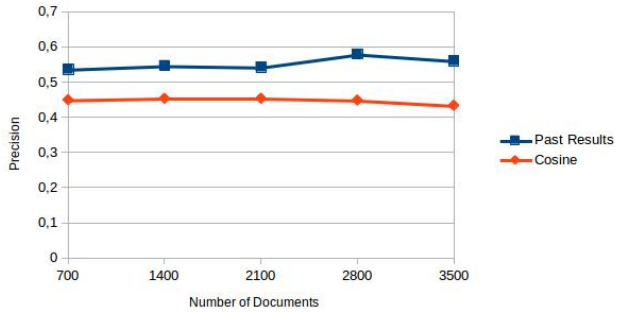

FIGURE 4.10: Simulation with Zipf Distribution 1.6 (for D) and Zeta Distribution 4 (for judgments)

TABLE 4.3: Simulation with a collection D based on Zipf distribution using $\lambda = 1.6$ – Comparison of the two approaches (i.e., *Past Results* and *Cosine*) according to average P@10 (over 30 queries)

		Approaches	
Zeta Distribution	D Size	Past Results	Cosine
$S = 2$	700	0.754	0.660
	1400	0.758	0.607
	2100	0.737	0.622
	2800	0.769	0.599
	3500	0.731	0.604
$S = 3$	700	0.605	0.523
	1400	0.659	0.570
	2100	0.625	0.550
	2800	0.658	0.531
	3500	0.650	0.549
$S = 4$	700	0.537	0.450
	1400	0.547	0.454
	2100	0.543	0.454
	2800	0.580	0.449
	3500	0.561	0.434

Discussion

It is relevant to emphasize that Zipf distribution was used with value 1.6, in order to simulate frequency distributions of the words on topics. Although, in more precise way Zipf distribution with values between 1.4 and 1.8 are used to simulate the distribution of the frequencies of words on vocabulary (Poosala, 1997; Zipf, 1949). In our case, it has been applied to select, terms of the topics (we have considered the number of the topics like vocabulary). We argued this decision, because in a document of real life, it is feasible to find terms used in other fields (i.e., in other topics). It has been the main argument by which, we have used not only the Zipf distribution but also the exponential distribution to build the collections of documents.

In addition, the objective of fixing different parameter values in Zeta distributions (S = 2, 3 and 4) to simulate the judgments of users was to analyse how this function influences average P@10. According to tables 4.1, 4.2 and 4.3 and figures from 4.2 to 4.10, when the parameter S is increased in Zeta distribution, the results shows that both average P@10 decreases for both traditional retrieval (cosine) and our approach (past results). This phenomenon can be observed in all the configurations used to build the collections of documents.

The results reported in tables 4.1, 4.2 and 4.3 and figures 4.2 to 4.10 show that there is a quite stable average P@10 for both approaches, when varying the number of simulated documents for a given configuration of distribution parameters. Thus, it seems not necessary to test simulations with more generated documents to expect other system behaviors.

Both approaches used to test the simulated collections have homogeneous effectiveness according to average P@10 over all the queries. However, performances differ from one collection to another supporting the idea that the different collections could be used as different scenarios.

4.7 Conclusions

Faced with the lack of suitable collections for evaluation of IRSs based on past queries and past results, an alternative to create an IR collection in a traditional costly way lies in simulation. Related work on simulation in the domain of IR has first been presented. Approaches using simulation, which can be found in the IR literature, take place mainly in the field of interactive information retrieval (IIR) These approaches intend to simulate users' interactions in a retrieval process, aiming to explore search strategies and how queries are submitted by users. Apart from IIR, few work on simulation in IR can be found in the literature, and none usable for evaluating the approach presented in the next chapter of

this thesis.

Then, an approach of framework to produce various collections to support evaluation of IRSs based on past queries and past results has been described. Simulated collections are generated as usually in IR, i.e., composed of a set of documents, a set of queries, and relevance judgments on documents. Therefore, one of the main objectives of this framework was to provide an ideal environment to evaluate approaches based on similar past queries and results, where exist similar queries with their own judgments of users. The framework supplies a clean experimental environment (i.e., stemming, stop-word removal are omitted) and robust (i.e, final results do not depend of particular characteristics of a system). Configurable simulations of documents, queries, and users' judgments are based on well-known laws of information science, such as Zipf's and Bradfor's laws, used in various research fields such as Digital Libraries and Information Retrieval (Chen and Leimkuhler, 1986; Sparck Jones and Willett, 1997a; Schaer, 2013).

The proposed framework was tested to simulate various collections. Several scenarios have been simulated under different probability distributions to build the collections of documents and determine the relevant documents given a query. To build the documents, exponential distribution with parameters 1.0 and 1.5, was applied. Similarly, Zipf distribution was employed to build the documents with parameter 1.6. The main reason was to select terms from different topics (different fields, i.e, chemistry, computer science, biology to mention a few), which compose a document. On the other hand, Zeta distribution have been used to represent the Bradford's law, which allows to simulate the judgment users. Three parameters for Zeta distribution have been applied in the experiments ($S = 2$, $S = 3$ and $S = 4$), aiming to cover a wide range of applications on this distribution. Experiment results showed homogeneous effectiveness of the two tested approaches according to average P@10 over all the queries. Furthermore, performances differ from one simulated collection to another supporting the idea that the different collections could be used as different scenarios.

The simulated collections were used in experiments presented in Chapter 5 to evaluate randomized algorithms in the domain of similar past queries and compared effectiveness with a traditional IR method.

Chapter 5

Probabilistic Approaches in IR for reusing past results

Résumé : Chapitre 5

De nombreux travaux ont porté sur l'amélioration des résultats pour des requêtes particulières. Plusieurs approches apportent des solutions impliquant une sélection efficace des systèmes pour répondre à certains types de requêtes, en appliquant des techniques de fouille de données par exemple. Cependant, certaines tâches de fouille de données nécessitent un coût élevé ainsi que du temps lorsque les jeux de données utilisés sont hétérogènes. En outre, il est possible de trouver des approches qui impliquent une analyse exhaustive de toutes les solutions possibles pour donner la meilleure réponse à une requête (par exemple, la meilleure précision pour chaque type de requête). D'autres solutions correspondent à des approches basées sur des techniques d'apprentissage (par exemple, les réseaux de neurones, les algorithmes génétiques et les machines à vecteurs de support). Les données de base utilisées pour les approches reposant sur les techniques d'apprentissage, sont composées de «requêtes passées et leurs documents avec jugements des utilisateurs». Bien que les approches d'apprentissage et d'optimisation ne peuvent pas être réellement considérées comme relatives à «l'utilisation des résultats de recherche passés », elles ne sont pas très différentes. En effet, elles impliquent d'analyser les relations entre les requêtes déjà soumises à un système (requêtes passées) et leurs documents retournés (résultats antérieurs). En outre, de bons résultats reflètent une adequation compréhension entre les algorithmes et les données passées (« requêtes passées » et leurs « jugements de pertinence ») utilisés pour l'apprentissage.

En outre, il est fréquent que deux SRI, qui utilisent la même collection fournissent des résultats différents pour un ensemble de requêtes. L'idée est que

chaque SRI ne connaît pas la distribution de probabilité des jugements de pertinence sur les documents pour une requête donnée (c.-à-d., comment la fonction de distribution de probabilité affecte la pertinence des documents obtenus pour une requête). Cependant, il est possible d'apprécier de façon empirique que les documents les plus pertinents ont tendance à apparaître en haut de la liste résultat. Ainsi, pour un SRI, si nous connaissons les documents retournés et les jugements de pertinence pour une requête, et nous avons une nouvelle requête très similaire à cette requête, alors nous pouvons utiliser les documents pertinents associés pour répondre à la nouvelle requête. Ainsi, l'hypothèse la plus importante utilisée dans le cadre de cette thèse, est que les documents pertinents ont tendance à apparaître en haut de la liste des résultats (à savoir, le premier document qui apparaît en haut de la liste a une probabilité plus élevée d'être pertinent que le dernier document de la liste). Un type d'algorithme probabiliste, utilisé lorsque les entrées de l'algorithme sont non-déterministes, correspond aux algorithmes de Monte Carlo et Las Vegas. Ainsi, les algorithmes de Monte Carlo sont utilisés dans cette thèse pour attribuer une probabilité de pertinence en fonction de la position d'un document pertinent dans la liste obtenue à partir de la requête passée la plus proche.

Il est important de souligner qu'il existe une différence importante entre les méthodes de Monte Carlo (ou des expériences de Monte Carlo) et les algorithmes de Monte Carlo. Les méthodes de Monte Carlo comprennent une large gamme d'algorithmes de calcul, qui sont basés sur un échantillonnage aléatoire répété pour obtenir des résultats numériques. En général, ces méthodes sont utilisées en Physique pour simuler des systèmes avec de nombreux degrés de liberté, tels que les fluides et les structures cellulaires. En outre, les méthodes de Monte Carlo sont utilisées dans des problèmes mathématiques, telles que l'optimisation, l'intégration numérique, et la génération de tirage dans une distribution de probabilité. Quelques approches de RI ont appliqué des méthodes de Monte Carlo. Par exemple plusieurs d'entre elles sont liées à l'optimisation (Puolamäki et al., 2005; Cemgil and Kappen, 2011; Roitman et al., 2014), d'autres se sont concentrées sur un échantillonnage aléatoire (Alexandrov et al., 2003). Cependant, ces travaux ne sont liés ni à des algorithmes randomisés, ni avec les requêtes passées et leurs résultats, qui constituent le sujet de cette thèse. Ce genre d'approche ne sera donc pas détaillé dans cette thèse.

Différentes des approches probabilistes basées sur des techniques d'apprentissage et d'optimisation, qui peuvent nécessiter des ressources élevées, en temps mais aussi en coût, dans cette thèse un ensemble d'algorithmes randomisés qui ne nécessitent pas d'apprentissage et fournissent une précision acceptable sont présentés. Ces algorithmes sélectionnent les documents pertinents en fonction de leur position dans la liste des résultats de la requête passée la plus proche. À notre connaissance, aucune approche similaire n'a été proposée dans la littérature. Comme évoqué dans les chapitres précédents, cela peut être dû à un manque de collections appropriées pour l'évaluation. Par conséquent, quelques travaux connexes seront présentés dans ce chapitre, surtout basés sur l'apprentissage, qui peuvent être vus comme une sorte de réutilisation des requêtes

passées.

Ce chapitre vise à fournir une vue d'ensemble sur les algorithmes randomisés. En outre, quatre algorithmes randomisés qui constituent une importante contribution de cette thèse, ainsi que leurs résultats empiriques sont présentés. Le chapitre est organisé comme suit. La section 5.2, présente les travaux connexes sur les approches probabilistes en RI en utilisant des requêtes passées pour l'apprentissage ou l'optimisation. Dans la section 5.3, un aperçu sur les algorithmes de Las Vegas et Monte Carlo est introduit. Dans la section 5.4, quatre algorithmes randomisés sont proposées pour la réutilisation de requêtes passées. Dans la section 5.5, les résultats empiriques utilisant les algorithmes proposés sont analysés. Enfin, à la section 5.6, les conclusions sont présentées.

5.1 Introduction

A large range of work has focused on improving the list of retrieval documents for particular queries. Several approaches provide solutions that involve effective selection of systems to respond to certain types of queries (e.g., difficult queries, are queries which provide poor results, in other words a bad precision), by applying data mining techniques for example. Nevertheless, some tasks of data mining require high cost in money as well as time when the used datasets are heterogeneous. Besides, it is possible to find solutions that involve an exhaustive analysis of all possible alternatives to give the best answer to a query (i.e., the best precision for each type of query). Prior solutions correspond to approaches based on learning techniques (e.g., neural networks, genetic algorithms and support vector machines). Baseline data used for approaches relying on learning techniques, are composed of "past queries and their documents with judgments of users". Despite the fact that all steps applied in machine learning and optimization cannot be seen as "the use of past search results", these are not very different. Indeed, it implies to analyze the relationship between queries previously submitted in a system (past queries) and their retrieved documents (past results). Furthermore, good results reflect a deep comprehension between algorithms and past data ("past queries" and their "judgments of users") used for training.

In addition, different two IRSs, which use the same collection C can provide different results (i.e., the list of documents obtained for each IRS is different and as consequence precision is different) for a set of queries Q. The idea is that each IRS does not know the probability distribution of relevance judgments on D for a given query that belongs to Q (i.e., how the probability distribution function assigns the relevance to the documents obtained as result of a query). However, it is feasible to appreciate empirically that the most relevant documents tend to appear at the top of results. Thus, for an IRS if we know C for example, and we have Q', which is very similar to Q, then we can use the associated relevant documents to answer queries of Q'. Thereby, the most important assumption used in the domain of this thesis, is that the relevant documents tend to appear at the top of the result list (i.e., the first document that appear at the top of the list has higher probability to be relevant than the last document in the list). A kind of probabilistic algorithms, that are used when algorithm inputs are non-deterministic, correspond to Monte Carlo and Las Vegas algorithms. Thus, Monte Carlo algorithms are employed in this thesis to assign a probability of relevance according to the position of a relevant document in the list obtained from the most similar past query.

It is important to point out that there is an important difference between Monte Carlo methods (or Monte Carlo experiments) and Monte Carlo algorithms. Monte Carlo methods comprise a wide range of computational algorithms, which are based on repeated random sampling to obtain numerical results. Commonly, these methods are used in physics to simulate systems with

many coupled degrees of freedom, such as fluids, disordered materials, strongly coupled solids, and cellular structures. Furthermore, Monte Carlo methods are used in mathematical problems, such as optimization, numerical integration, and generating draws from a probability distribution. Few approaches in IR have applied Monte Carlo methods, for example several of them are related to optimization (Puolamäki et al., 2005; Cemgil and Kappen, 2011; Roitman et al., 2014) ; while others are focused on random sampling (Alexandrov et al., 2003). Nevertheless, it is relevant to emphasize that these works are related neither to randomized algorithms nor with past queries, which are applied in this thesis.

Different to the probabilistic approaches based on learning techniques and optimization, which can involve high resources not only in time but also cost, in this thesis a set of randomized algorithms that do not require learning time and provide an acceptable precision are presented. These algorithms select relevant documents according to their position in the result list from the most similar past query.

This chapter aims to provide an overview about randomized algorithms. Furthermore, four randomized algorithms which constitute an important contribution of this thesis along with their empirical results are presented. The chapter is organized as follows. Section 5.2, presents related work about probabilistic approaches in IR using past queries for learning or optimization. In section 5.3, an overview about Las Vegas and Monte Carlo algorithms is introduced. In section 5.4, four randomized algorithms are proposed. In section 5.5, empirical results using the proposed algorithms are analysed. Finally, in section 5.6, conclusions are presented.

5.2 Related Work

The literature of IR is crammed with different contributions that use techniques and probabilistic algorithms to improve results of a retrieval process. Roughly speaking, two major types of research can be easily distinguished : learning techniques and optimization. Bayesian Networks and variants are widely used probabilistic techniques. Although bayesian networks provide a high-level representation for a probability distribution over a set of variables, which model the problem domain, an understandable and compact representation of the involved variables in the problem are needed. Generally, this representation is obtained as the result of a data mining process. Whereby, when the data mining process is complex, it can involve time and considerable effort. Furthermore, a poor representation of variables can supply disappointing results. Other approaches rely on probability estimations, for example, of term frequencies or document relevance.

Traditional approaches based on Bayesian Networks build a network by using

documents. In contrast, Indrawan et al. (1996) came up with a model where Bayesian Networks are composed of two entities, evidence or cause and hypothesis or effect. Thus, retrieving a relevant document corresponds to calculating a conditional probability. Therefore, a set of relevant documents is obtained by ranking the belief value on the query node in decreasing order. Preliminary results on CACM, ADI and MED collections showed promising performance of the system. Despite the fact that Bayesian networks are not used in this thesis, we also apply probability to recover relevant documents. This probability is not a conditional probability, it is rather an assigned probability for the randomized algorithms according the position in the result list, which is obtained from the most similar past query.

In the domain of text categorization, Joachims (1997) presented a probabilistic variant of the Rocchio classifier ($tf - idf$ algorithm) denominated PrTFIDF. PrTFIDF provides a new perception about the vector space model by using a descriptor for every document, the theorem of total probability, and the Bayes' theorem. Empirical results on six categorization tasks using Usenet articles showed that the two probabilistic methods Bayes and PrTFIDF supply performance improvements of up to 40 % reduction of error rate on five of the six tasks. The author pointed out that PrTFIDF not only provides a better theoretical understanding, but also has a good performance. The approach presented in this thesis differs from this work both in terms of objectives (i.e., reuse of past results vs text categorization), and because it proposes an additional process applicable to any initial IR model. Besides, the Bayes' theorem is not applied in this thesis. However, this work provides a source of inspiration about the use of probabilities in the IR domain.

In a more usual ad-hoc retrieval process, Hiemstra (1998) proposed a new probabilistic model. This model has as assumption that documents and queries are defined by ordered sequences of single terms, which is well-known in the field of statistical natural language processing. A new probabilistic interpretation of $tf-idf$ term weighting is proposed. Both documents and queries are modeled as compound events through an ordered sequence of events. Experimental results on Cranfield collection showed that this model outperforms the vector space model. Different to this research, which consider probability at the term weighting level, this thesis deals with document relevance probability. Furthermore, the approach proposed in this thesis can be applied to any IR model.

Blei et al. (2003) designed a simple hierarchical Bayesian approach for modeling document collections and other large-scale data collections. The model assumes that a document is built by choosing a random set of multinomial probabilities for a set of possible "topics", and then repeatedly generating words by sampling from the topic mixture. The authors claim that a limitation of this model is the lack of sensitivity according to the number of topic. Nevertheless, researchers assert that this parameter can be obtained empirically. Thus, this work is similar to this thesis because document collections are simulated. Indeed, the words (i.e., terms) that compose a document are chosen from "topics" using several

probability distributions. Besides, the main difference is that Bayesian networks are not applied in this thesis.

Lillis et al. (2006), promoted the Probfuse algorithm, which merges results from several IR algorithms. It was compared with the well-known CombMNZ algorithm. Using Profuse, for each document of a result, a score is associated according of its relevance likelihood, which is used in ranking the documents at the end, in a fused result set. This likelihood is obtained considering both the performance of the underlying IR models and the number of training queries. Experiments on four document collections – Cranfield, LISA, NPL and Med – were carried out, showing best results for Probfuse. Like in this work, in this thesis the randomized algorithms use the result list of recovered documents assigning a relevance likelihood for relevant documents. However, in this thesis probabilities are not applied for fusing results from several IR algorithms producing different results for a submitted query. Probabilities are used to determine relevant documents in most similar queries to build results for new submitted queries.

More recently, a method to expand queries based on Bayesian networks was introduced by de Campos et al. (2013). Using a learning algorithm the method builds the network which represent some of the relationships among the terms that appear in a document collection. Thus, this network is used as a thesaurus (specific for that collection) with the purpose to provide new terms with a high probability of being related with the initial query. Experiments have been carried out on three standard test collections – Adi, Cranfield and Medlars. Despite the fact that final results are moderate, the authors pointed out that their approach is a good baseline for heterogeneous document collections. Unlike this research, this thesis does not rely on neither Bayesian Networks nor learning algorithm. However, adding new terms with some probabilities is an inspiring idea for future work.

As stated previously, probabilistic approaches in IR can be categorized in two types, learning techniques and optimization. The main drawback that present both categories is that these, can take considerable time. Some optimization techniques imply the use of genetic algorithms, where an inadequate election of the fitness function and the initial population can take a substantial execution time. Learning techniques are applied commonly after a data mining process, which can be the result of a difficult human task. Most probabilistic approaches in IR involve an exhaustive analysis of all possible options to look for the best answer to a query (typically, the best precision for each type of query). It can be seen as trying to find for each system a function that not only provide the relevant documents given a query but also put this list of relevant documents at the beginning of the result list. Therefore, trying to discover this function to assign relevant documents (i.e., a distribution function) is almost impossible.

By contrast, simple solutions in terms of representation and implementation can be proposed through randomized algorithms. Typically, randomized algorithms are used when the problem to solve implies high resources in time or when

algorithm input is non-deterministic (i.e., data of algorithm input have a probability distribution). It is important to mention that the main difference of our algorithms with respect to the existing approaches is that we assign a relevance probability (i.e., an assignment of the probability according to the document position in the result list is provided) to the relevant documents obtained from the most similar past query. The most important assumption used here, is that the relevant documents tend to appear at the top of the result list. The randomized algorithms presented in this thesis suppose that when the position of the document (in the list of documents) is at the top (close to the query), the document has higher probability to be relevant than a document at the bottom of the list. Furthermore, these randomized algorithms are simple to implement, they do not require learning time, are robust, effective and can be adapted to several domains.

5.3 Randomized Algorithms

In contrast to Bayesian Networks and Genetic Algorithms, Monte Carlo and Las Vegas algorithms are employed typically when the problem is hard to solve like NP problems (i.e., time is a relevant factor in order to supply an answer) or when algorithm input is non-deterministic. In the world, all processes are not deterministic, several processes rely on random. For example, simple processes such as flipping a coin are random. Another example of non-deterministic problems is a card game. In such a game, someone attempts to guess a card. After every round the probability to guess is increased. At the end, in the 51st round, both cards have accomplished their maximum probability (50 % percent for both). Therefore, it is always possible to find a wrong probability (except for the last card). Contrary to deterministic methods involved in subjects such as geometry and logic, which have been treated mathematically for several thousand years, the mathematical study of probabilities is relatively new ; the first known attempts to seriously formalize it came about in the 1600s. The study of random processes in computer science is more recent yet, but it has been involved from its creation.

Kleinberg and Tardos (2005) pointed out that random processes in the computation domain can be split in two different ways. The first, is to consider the world behavior randomly, where a traditional algorithm tackles randomly generated input. This approach is well-known as average-case analysis, due to the study about the behaviour of an algorithm on an average input rather than a worse-case input. A second way corresponds to the notion of randomized, it is to consider algorithms that behave randomly, that is the data provides the same worst-case input as always, however the role of randomization in this approach is purely internal to the algorithm.

Las Vegas algorithms provide an answer either true or false, and this

answer is always soundness. For example let consider an unsorted array, in which one wants to find a number. Here, the algorithm provides a random position (from 1 to N), and the searched number is compared with the number, which is at the random position. If both numbers are equal, the algorithm returns true. Otherwise, the number which is in the random position is swapped for the number at first position. Now, the algorithm can choose a random position (from 2 to N). Again the searched number is compared with the number, which is at the random position. If the searched number is different to the number at the random position, the number at the random position is swapped for the number at second position, and so on. Finally, the number can always be found because it is inside the array. On the other hand, if the searched number is not inside the array, the algorithm should respond false. In both cases (true or false), the answer is always soundness. Las Vegas algorithms were introduced by the mathematician Laszlo Babai in 1979, with the aim to tackle the graph isomorphism problem. The denomination of "Las Vegas" comes from a popular city in Nevada, which is internationally famous for gambling.

By contrast, Monte Carlo algorithms supply an answer, which can be unsoundness (i.e., when algorithm gives true, it could be false, or when algorithm gives false, it could be true). When the algorithm gives true or false, and one of these answers is soundness, it is called true-biased. Otherwise, both answers can be unsoundness, it is called two-sided errors. Using the previous example (i.e., given an unsorted array), if one wants to review only k elements of the array ($k < N$), in order to reduce the searching time and whether the number is inside of k elements, the algorithm should return true (it is soundness). On the contrary, if the searched number is in the other portion of elements ($N - k$), the algorithm will respond false, when the number is inside the array. This example corresponds to true-biased Monte Carlo algorithms, because when the algorithm returns true, it is soundness, but if the answer is false, it is not a soundness answer. In our particular case, the randomized algorithms presented in this thesis correspond to two-sided errors (i.e., both answers true or false can be unsoundness). It is because, we are not sure about judgments of users with respect to whether a document is either relevant or non-relevant regarding the query. Nevertheless, we can suppose that when the position of the document (in the list of documents) is at the top, the document has higher probability to be relevant than a document at the bottom of the list.

Finally, it is important to emphasize that it is not the aim of this chapter to provide a review of theoretical literature in this area. Further readings include books by Kleinberg and Tardos (2005); Cormen et al. (2001); Banachowski et al. (1991); Gonnet (1984).

5.4 Contribution to the reuse of past searches

5.4.1 Description of the algorithms

In this section, the main contribution of this thesis is exposed. Four randomized algorithms are presented and analysed. These algorithms are categorized as Monte Carlo, specifically two-side error. As stated previously, the main reason by which this algorithm type have been chosen, is because it is not possible to know the probabilistic distribution about judgments of users over the list of retrieved documents given a query. It is because relevance judgments about a document considering the query are issued traditionally by a group of experts without considering the ranking functions. Nevertheless, it is feasible to assume that documents that appear at the top of the result list have higher probability to be relevant than documents that appear at the bottom of the list.

This approach considers a storage process in which each computed query is stored with its documents and relevance judgments. Second, for the retrieval process, each new query is compared with the past queries in the system, and if there is a past query quite similar in the system then relevant documents of this past query are recovered according to our algorithms. In this way, relevant documents are used to respond to the new query. Thus, each algorithm works on the list of retrieved documents, which belongs to the most similar past query with respect to the new query.

In the following paragraph, we present a set of definitions and notations, which are applied on all the algorithms.

5.4.2 Definitions and notations

It is important to highlight that the definitions comprised between 1 and 13 are common to the four algorithms. Broadly speaking, each algorithm uses an additional array called $B[N]$, which has as purpose to support the obtaining of probability. This array contains 1 and 0, and its use can be seen as flip on coin with head "1" or the tail side "0", half of array contains 1, and the other half contains 0. Thus, the probability to have a head "1" in the second round rises to 75 %. Besides, $V_N(q)$ can be seen as the list of retrieved documents given a query q.

Definition 1. Let $DB = D \cup Q$ be an IR dataset, composed of a finite set of documents D, and a finite set of past queries Q

a) $\forall d_i \in D \wedge \forall q_j \in Q$, both d_i and q_j are unique.

By this definition, we have a collection of documents D, and a collection of past queries Q.

The point a) indicates that there are not two equal documents as well as not two equal queries.

Definition 2. Let $Q' = \{q'\}$ be a finite set of new queries, such as every query is unique.

Definition 3. Let $V_N(q)$ be a set of N retrieved documents given q.

Definition 4. Let $A(q)$ be the set of all the relevant documents retrieved for the query q, such as $A(q) \subseteq V_N(q)$.

Definition 5. Let $A'(q)$ be the set of all the non-relevant documents retrieved for the query q, such as $A'(q) \subseteq V_N(q)$.

Definition 6. Given a query q, such as $q \in Q$, $p(q) = (q, V_N(q))$ is defined as the profile of the query q, which the pair composed of the query and its retrieved documents.

Definition 7. Let $P = \bigcup p(q)$ be the set of all the query profiles stored in the system.

Definition 8. Given a query q', such as $q' \in Q'$. $R_p(q', P)$ corresponds to the set of retrieved documents from the most similar past query in P according to a similarity measure (e.g., cosine).

Definition 9. $\partial : R_p(q', P) \rightarrow A(q')$ is a function, which assigns the most relevant documents to a new query q', such as $q' \in Q'$ (see *Definition 6* and *Definition 7*).

Definition 10. $\|x\|$ denotes the integer part of a real number, such as $x \geq 0$.

Definition 11. $\lceil x \rceil$ denotes the upper integer of x.

Definition 12. Given a binary array $B[N]$, such as B has N elements, and $\frac{a}{N}$ is the proportion of values in B that are equal to 1 (true).

This array is the baseline to provide a level of general probability for all the documents. Nevertheless, the probability of each document according to its position in the list $V_N(q)$, is computed by all the algorithms.

The first algorithm

The first algorithm splits the list of retrieved documents from the most similar past query in NG groups, where each group contains the same quantity of documents. The first group is composed of the first documents appearing at the top of the list. The first group has the greatest probability to get a *hit* with respect to all other groups. The second group has higher probability to get a *hit* than the third group, and so on. Besides, each document of a given group has a different likelihood to have a *hit*. The first element of the group has a higher likelihood to have a *hit* than the last element of the group. On balance, each group has different probabilities to have a *hit*, and at the same

time, the documents of a group have different likelihoods. Thus, a document in the position i has higher probability than a document in the position *i+1*. The main reason that supports this design, is because generally the documents that appear at the top of lists are relevant meanwhile that the documents at the bottom of lists are not relevant.
In Figure 5.1, the algorithm 1 splits the list in NG groups ($NG = 4$), with the same quantity of documents ($2^{ne} = 8$). The probability for each document is determined by two factors. First, it depends on the group the document belongs to, and second to the relative position of the document in the group, for example, for the document d_1 (group 1), which is the same for the document d_{13} (group 2). Moreover, the number of iterations K (see *Definition 16*) with the purpose to find "1" inside the $B[N]$ is the same. Nevertheless, the probability in the first group is bounded by $\frac{2^4}{32}$. Finally, the probability for the document in the position i, considering the *group* (from 1 to NG), it belongs to, is given by $\sum_{l=1}^{K} \frac{1}{(2^{group})^l}$

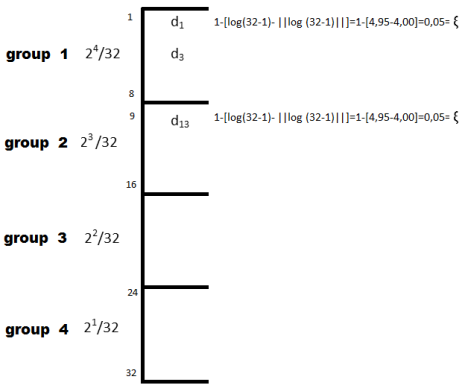

FIGURE 5.1: List of retrieved documents, which is split by the Algorithm 1

The following definitions are used for the first algorithm.

Definition 13. Let $N \simeq NG * 2^{ne}$ be the approximate number of documents, where NG is the number of groups containing documents of $V_N(q)$, and 2^{ne} corresponds to the number of elements by group.

Definition 14. Let i be the position of a document in $V_N(q)$, such as the first element ($i = 1$) represents the most similar document, then
$G_x(ne, i, NG) = min\{x \mid x \in \mathbb{N} \wedge x \in [1, NG] \wedge i \leq x * 2^{ne}\}$, corresponds to the assigned set for the element i.

Definition 15. Let $\varepsilon(ne, i, NG, N) = 1 - [log_2(2^{M(N)} - \langle i\ MOD\ (G_x(ne, i, NG) + 1)\rangle) - \|log_2(2^{M(N)} - \langle i\ MOD\ (G_x(ne, i, NG) + 1)\rangle\|]$ be the error assigned for the document at the position i in $V_N(q)$.

Definition 16. Let $K(ne, i, N, NG) = \lceil log_2\langle\frac{1}{\varepsilon(ne,i,NG,N)}\rangle\rceil$ be the number of iterations on $B[N]$, to assign the likelihood for the document at the position i in $V_N(q)$.

Thus, $\varphi : F(i) \rightarrow \{0, 1\}$, is the probability function used by the algorithm 1.

$$F(i) = \begin{cases} 1 & : \quad Pr_i(1) = \sum_{l=1}^{K(ne,i,N,NG)} \frac{2^{(M(N)-G_x(ne,i,NG))}}{(2^{M(N)})^l}, \\ 0 & : \quad Pr_i(0) = 1 - Pr_i(1) \end{cases}$$

where $Pr_i(1)$ is the likelihood of *hit* (1) for the element i, and $Pr_i(0)$ represents the probability of *miss* (0) for the element i.

As shown in Figure 5.1, it is possible to calculate the probability for the document d_1. Let consider the values as follows : $N = 30$, $NG = 4$, $ne = 3$, $i = 1$ and the value of $M(N) = 5$. Thus, the probability for the first element of $V_N(q)$ is computed as : $G(3, 1, 4, 30) = 1$ (the first document in the list, is in the first group), $\varepsilon(3, 1, 4, 30) = 0.005$, (see *Algorithm 1*). Therefore, $K(ne, i, N, NG) = 5$. As the document d_1 belongs to the first group, which has $\frac{1}{2}$, then the probability for the first document is $\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32}$, therefore $Pr_1(1) = 0.968$.

In a similar way, for elements of the second group, the search to find a "1" inside the array B, is limited between the range 1 to $\frac{N}{2}$ or $\frac{N}{2} + 1$ to N (see *Algorithm 1*, lines 29 and 32). Finally, the probability for the first element of second group d_{13} is $\frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256} + \frac{1}{1024}$, $Pr_9(1) = 0.333$.

Preliminary results are presented in figures 5.2, 5.3, 5.4 and in Table 5.1, exponential distribution have been used to select the terms from topics (these are terms that belong to a particular area, such as biology, mathematics to mention a few) and built the documents, Zeta distribution have been used to simulate the judgments of users. Similarly, other results are presented in figures from 5.5 to 5.10 and in Tables 5.2 and 5.3.

It is important to emphasize that the empirical results are analysed and discussed in detail in section 5.5.

Algorithm 1:

Data: $B[N]$ (*see Definition 12*) is a boolean array, ne is a number of elements in each group, $V_N(q)$ is the list of retrieved documents for the query q, q' is the most similar query with respect to q

Result: $A_{New}(q')$ is a list of relevant documents for the query q'

$A_{New}(q) \leftarrow \emptyset$;
$Index.Inf \leftarrow 0$;
$Index.Sup \leftarrow 0$;
for $i \leftarrow 1$ **to** N **do**
  $B[i] \leftarrow false$;
end
for $i \leftarrow 1$ **to** $\frac{N}{2}$ **do**
  $j \leftarrow random(1, ..., N)$;
  $B[j] \leftarrow true$;
end
$i \leftarrow 1$;
// b is the group number
for $b \leftarrow 1$ **to** NG **do**
  for $c \leftarrow 1, c < 2^{ne}$ **do**
    $u \leftarrow log_2(2^{M(N)} - c)$;
    $U \leftarrow \|u\|$;
    $E \leftarrow u - U$;
    $E \leftarrow 1 - E$;
    // This is the error, see *Definition 15*
    $K \leftarrow log_2(\frac{1}{E})$;
    $K \leftarrow \ulcorner K \urcorner$;
    // This is the iteration number, see *Definition 16*
    for $l \leftarrow 1$ **to** K **do**
      if $b = 1$ **then**
        $Index.Inf \leftarrow 1$;
        $Index.Sup \leftarrow N$;
      else
        $valor \leftarrow random(0, 1)$;
        if $valor = 0$ **then**
          $Index.Inf \leftarrow 1$;
          $Index.Sup \leftarrow \frac{N}{2^b}$;
        else
          $Index.Inf \leftarrow \frac{N}{2}$;
          $Index.Sup \leftarrow \frac{N}{2} + \frac{N}{2^b}$;
        end
        $indexF \leftarrow (index.Inf + (rand(1, .., index.Sup))$;
        if $B[indexF] = true$ **then**
          if $([idDoc = Position(i\ of\ V_N(q))]\ is\ in\ A_{Past}(q))$ **then**
          $A_{New}(q') \leftarrow A_{New}(q') \cup d_{idDoc}$;
          $i \leftarrow i + 1$;
          ;
          else $i \leftarrow i + 1$;
          ;
        else
        end
      end
    end
  end
end

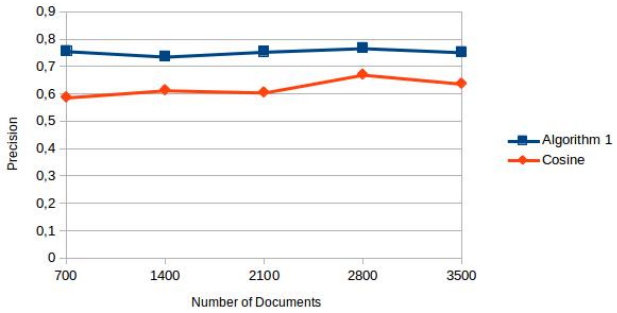

FIGURE 5.2: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 2 (for judgments)

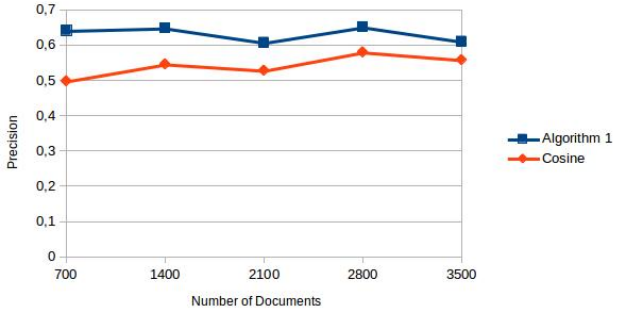

FIGURE 5.3: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 3 (for judgments)

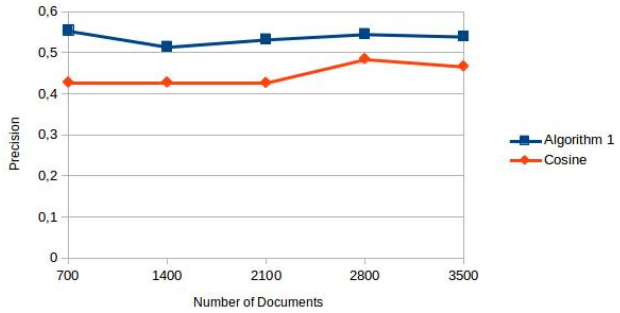

FIGURE 5.4: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.1: Evaluation results of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.0$

		Results	
Zeta Distribution	D Size	Algorithm 1	Cosine
$S = 2$	700	0.758	0.589
	1400	0.738	0.615
	2100	0.756	0.607
	2800	0.769	0.672
	3500	0.754	0.639
$S = 3$	700	0.642	0.499
	1400	0.649	0.547
	2100	0.608	0.529
	2800	0.652	0.581
	3500	0.611	0.559
$S = 4$	700	0.555	0.429
	1400	0.516	0.429
	2100	0.534	0.428
	2800	0.547	0.486
	3500	0.541	0.468

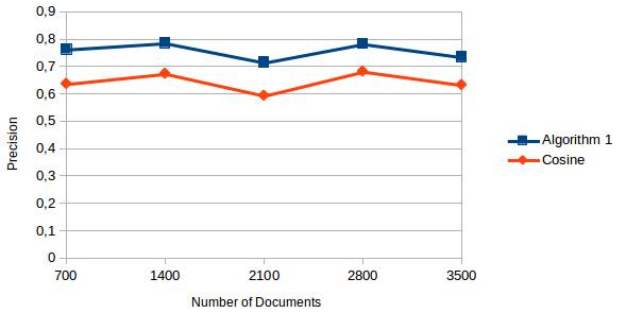

FIGURE 5.5: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 2 (for judgments)

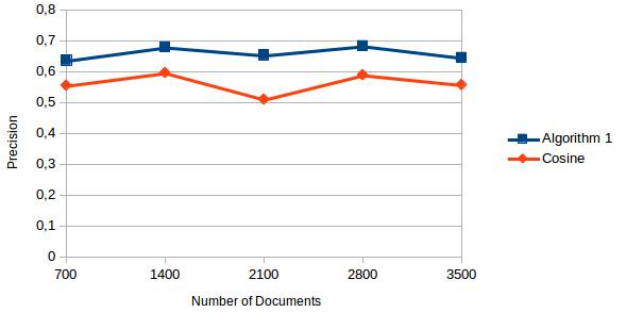

FIGURE 5.6: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 3 (for judgments)

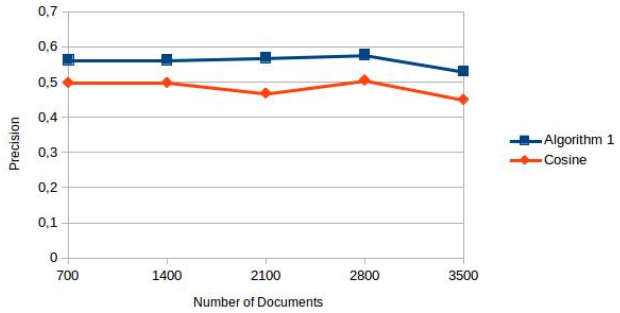

FIGURE 5.7: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.2: Evaluation results of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.5$

		Results	
Zeta Distribution	D Size	Algorithm 1	Cosine
$S = 2$	700	0.764	0.638
	1400	0.787	0.675
	2100	0.716	0.595
	2800	0.784	0.683
	3500	0.736	0.634
$S = 3$	700	0.637	0.556
	1400	0.680	0.597
	2100	0.654	0.511
	2800	0.684	0.590
	3500	0.646	0.558
$S = 4$	700	0.564	0.499
	1400	0.564	0.499
	2100	0.570	0.469
	2800	0.578	0.506
	3200	0.531	0.451

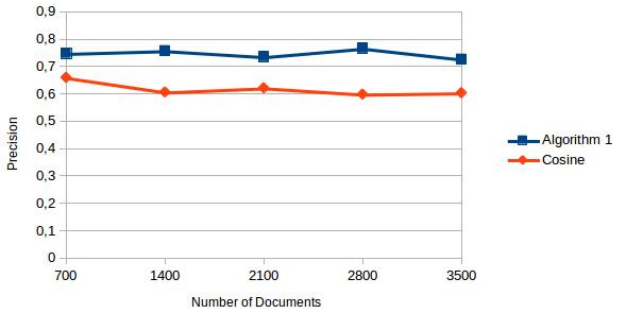

FIGURE 5.8: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 2 (for judgments)

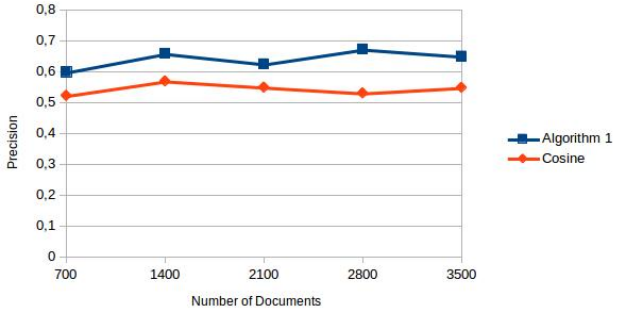

FIGURE 5.9: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 3 (for judgments)

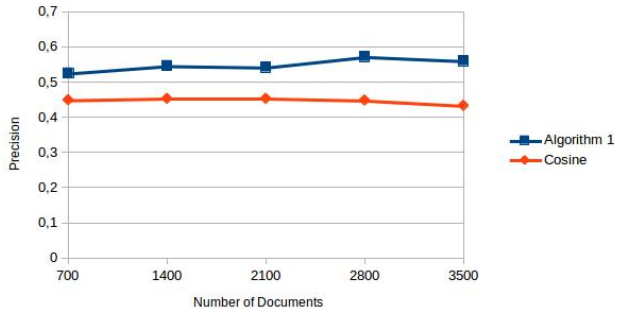

FIGURE 5.10: Evaluation of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.3: Evaluation results of the two approaches (i.e., *The first algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on Zipf distribution using $\lambda = 1.6$

		Results	
Zeta Distribution	D Size	Algorithm 1	Cosine
$S = 2$	700	0.748	0.660
	1400	0.758	0.607
	2100	0.736	0.622
	2800	0.767	0.599
	3500	0.727	0.604
$S = 3$	700	0.599	0.523
	1400	0.659	0.570
	2100	0.625	0.550
	2800	0.673	0.531
	3500	0.650	0.549
$S = 4$	700	0.526	0.450
	1400	0.547	0.454
	2100	0.543	0.454
	2800	0.573	0.449
	3500	0.561	0.434

The second algorithm

The second algorithm is inspired by the logistic distribution. Commonly, the denomination "logistic regression" is used because it corresponds to a general model of regression. At the same time, it is denominated logistic discrimination because employed to solve classification issues. When the estimation of probabilities involves two or more variables, it is called ranking. Usually, this method is applied when there are two possibilities of answers. It can be seen as "1 or 0", or "relevant or non-relevant". Broadly speaking, the closest documents to the query (documents are at the top of the list) have a higher probability to be "relevant". Therefore, the previous paragraph provides the main argument to support this design.

The following definitions are used for the second algorithm.

Definition 17. Let $logit(i, \alpha, \beta) = log\frac{p'_1}{1-(\alpha+\frac{\beta}{i})}$ be a bound used to calculate the number of iterations, according to the position i of the document in $V_N(q)$, where p'_1 corresponds to the initial probability, and $\alpha + \frac{\beta}{i}$ is a polynomial of degree 1.

Definition 18. Let $K'(i, \alpha, \beta, \gamma) = \lceil e^{logit(i,\alpha,\beta)} * \gamma \rceil$ corresponds to the iteration number on $B[N]$ for each $i > 1$, and γ is a real number. For $i = 1$, $K'(1, \alpha, \beta, \gamma) = \lceil log_2(\frac{1}{1-p'_1}) \rceil$.

Thus, $\varphi' : F'(i) \rightarrow \{0, 1\}$, is the probability function used by the algorithm 2.

$$F'(i) = \begin{cases} 1 & : \quad Pr'_i(1) = \sum_{l=1}^{K'(i,\alpha,\beta,\gamma)} \frac{1}{2^l}, \\ 0 & : \quad Pr'_i(0) = 1 - Pr'_i(1) \end{cases}$$

$Pr'_i(1)$ is the likelihood of *hit* (1) for the element i using the algorithm 2, meantime $Pr'_i(0)$ represents the probability of *miss* (0) for the element i.

Aiming to provide an example for Algorithm 2, the values are set as follows $\alpha = -4, \beta = 2.0, p'_1 = 0.92$ and $\lambda = 10$. It is important to highlight that half of elements in $B[N]$ has 1, and the rest of elements are 0. Therefore, for $i = 1$, $logit(1, -4, 2.0) = -1.181$. Thus, the probability for the first element is $Pr'_1(1) = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} = 0.9375$. In a similar way, the probability for the 9th element is calculated as follows, $i = 9$, $logit(9, -4, 2.0) = -1.647$ and $K'(9, -4, 2.0, 10) = 2$ (see the Algorithm 2). Whereby, the probability is given by $Pr'_1(9) = \frac{1}{2} + \frac{1}{4} = 0.75$.

Preliminary results are presented in figures 5.11, 5.12, 5.13 and in Table 5.4, exponential distribution have been used to built the documents, Zeta distribution have been used to simulate the judgments of users. In a similar way, other results are presented in figures from 5.14 to 5.19 and in Tables 5.5 and 5.6.

Algorithm 2:

Data: $B[N]$ (*see Definition 12*) is a boolean array, $V_N(q)$ is the list of retrieved documents for the query q, q' is the most similar query with respect to q

Result: $A_{New}(q')$ is a list of relevant documents for the query q'

$A_{New}(q) \leftarrow \emptyset$;
for $i \leftarrow 1$ **to** N **do**
  $B[i] \leftarrow false$;
end
for $i \leftarrow 1$ **to** $\frac{N}{2}$ **do**
  $j \leftarrow random(1, ..., N)$;
  $B[j] \leftarrow true$;
end
for $i \leftarrow 1$ **to** N **do**
  $pi' \leftarrow 1 - (\alpha + \beta)$;
  $pi \leftarrow e^{log(\frac{P1}{pi'})}$;
  $E \leftarrow pi * \gamma$;
  if $i = 1$ **then**
    $E \leftarrow 1 - P1$;
    $K \leftarrow \ulcorner log_2(\frac{1}{E}) \urcorner$;
  else
    $K \leftarrow \ulcorner E \urcorner$;
  end
  for $l \leftarrow 1$ **to** $l \leq K$ **do**
    $indexF \leftarrow random(1, ..., N)$;
    if $B[indexF] = true$ **then**
      if $([idDoc = Position(i\ of\ V_N(q))]\ is\ in\ A_{Past}(q))$ **then**
      $A_{New}(q') \leftarrow A_{New}(q') \cup d_{idDoc}$;
      $i \leftarrow i + 1$;
      ;
      else $i \leftarrow i + 1$;
      ;
    else
    end
  end
end
$return(A_{New}(q'))$;

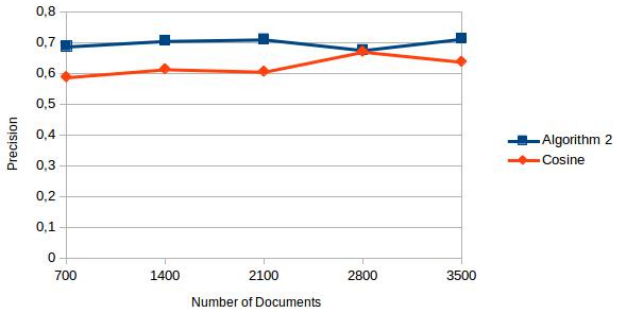

FIGURE 5.11: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 2 (for judgments)

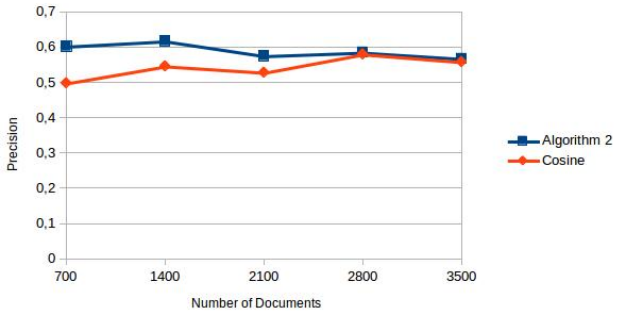

FIGURE 5.12: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 3 (for judgments)

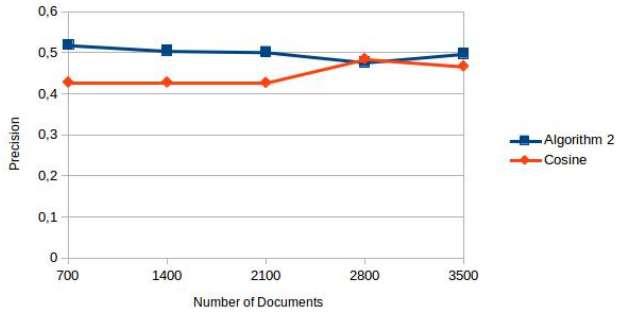

FIGURE 5.13: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.4: Evaluation results of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.0$

		Results	
Zeta Distribution	D Size	Algorithm 2	Cosine
$S = 2$	700	0.689	0.589
	1400	0.707	0.615
	2100	0.712	0.607
	2800	0.677	0.672
	3500	0.714	0.639
$S = 3$	700	0.603	0.499
	1400	0.618	0.547
	2100	0.576	0.529
	2800	0.586	0.581
	3500	0.568	0.559
$S = 4$	700	0.520	0.429
	1400	0.506	0.429
	2100	0.503	0.428
	2800	0.478	0.486
	3500	0.499	0.468

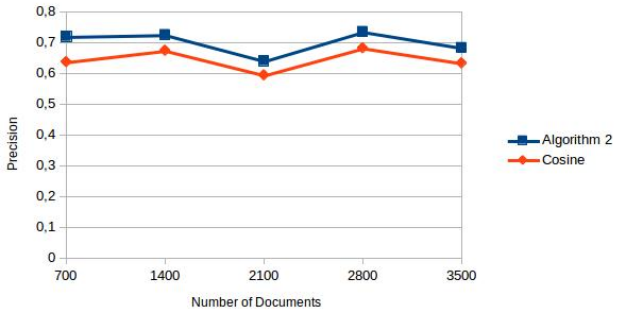

FIGURE 5.14: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 2 (for judgments)

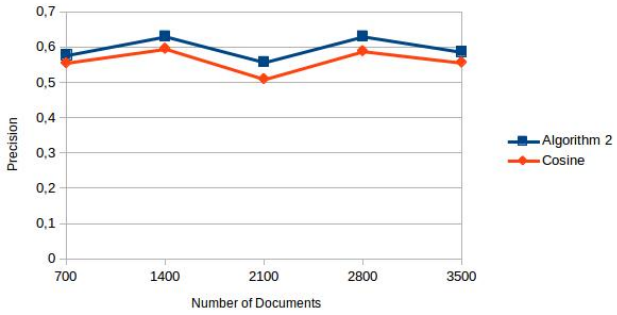

FIGURE 5.15: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 3 (for judgments)

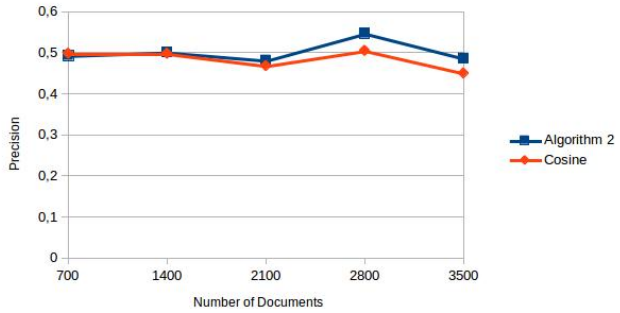

FIGURE 5.16: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.5: Evaluation results of the two approaches (i.e., *The second Algorithm Using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.5$

		Results	
Zeta Distribution	D Size	Algorithm 2 4	Cosine
$S = 2$	700	0.720	0.638
	1400	0.726	0.675
	2100	0.641	0.595
	2800	0.736	0.683
	3500	0.684	0.634
$S = 3$	700	0.579	0.556
	1400	0.632	0.597
	2100	0.559	0.511
	2800	0.632	0.590
	3500	0.588	0.558
$S = 4$	700	0.494	0.499
	1400	0.502	0.499
	2100	0.482	0.469
	2800	0.548	0.506
	3200	0.487	0.451

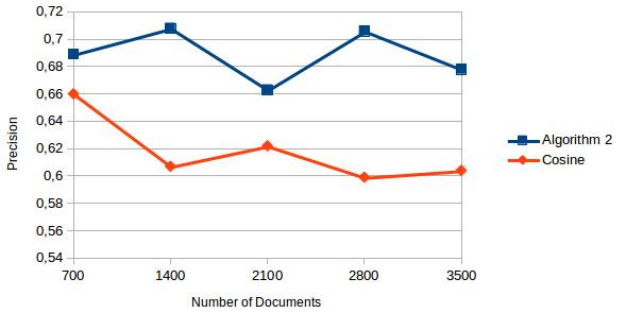

FIGURE 5.17: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 2 (for judgments)

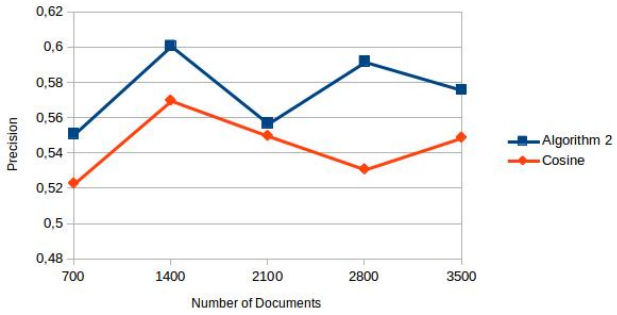

FIGURE 5.18: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 3 (for judgments)

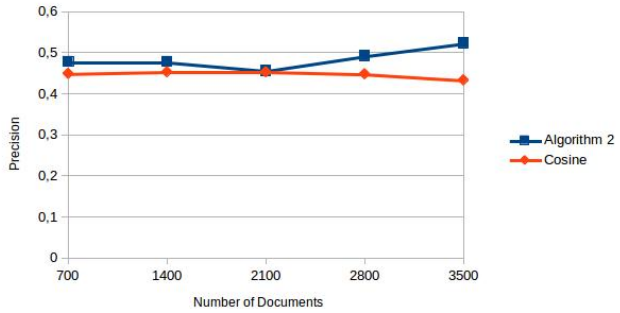

FIGURE 5.19: Evaluation of the two approaches (i.e., *The second algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.6: Evaluation results of the two approaches (i.e., *The second Algorithm Using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on Zipf distribution using $\lambda = 1.6$

		Results	
Zeta Distribution	D Size	Algorithm 2	Cosine
$S = 2$	700	0.689	0.660
	1400	0.708	0.607
	2100	0.663	0.622
	2800	0.706	0.599
	3500	0.678	0.604
$S = 3$	700	0.551	0.523
	1400	0.601	0.570
	2100	0.557	0.550
	2800	0.592	0.531
	3500	0.576	0.549
$S = 4$	700	0.479	0.450
	1400	0.479	0.454
	2100	0.457	0.454
	2800	0.493	0.449
	3500	0.524	0.434

The third algorithm

The third algorithm divides the list of retrieved documents in groups of power two, where each group contains different quantities of documents. For example, if the list of documents comprises 30 documents, the number of documents will be rounded up to the next number in power two, i.e., $n = 32$. Thus, the number of groups is 5 (because $32 < \{2^4 + 2^3 + 2^2 + 2^1 + 2^0\} > 30$ see *Definition 14*). Later on, groups of documents are defined as follows. The first group is composed by 2^4 documents. The second group involves 2^3 documents, the third group is composed of 2^2 documents, and so on, in such a way that the sum of documents does not outperform $n = 32$. The biggest group is composed of documents that appear in the first positions (between the position 1 and 16). The next biggest group comprises documents that appear from the position 17 to 24, and so on. Similarly to the first algorithm, the first group has the greatest probability to get a *hit* with respect to all other groups. The second group has higher probability to get a *hit* with than the third group, and so on. Besides, a document in the position i has higher probability than a document in the position *i+1*. To sum up, the likelihood of a document to be relevant is determined by two factors : the group it belongs to and its position in the group. The essential assumption behind this design, is that this is similar to the use of tree entropy (similar to compression algorithms, where the most used data appear close to the root, it means that these data has highest probabilities to be accessed), where the highest probabilities are given close to the root.

In Figure 5.20, the algorithm 3 splits the list of retrieval documents in $M(N)$ (see *Definition 20*) groups, thus $M(32) = 4$.

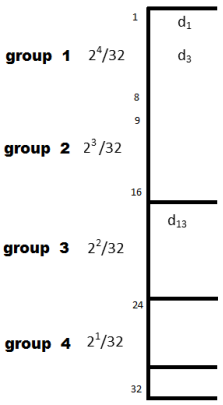

FIGURE 5.20: List of retrieved documents, which is split by the Algorithm 3

The following definitions are used for the third algorithm.

Definition 19. $M(N) = min\{m \mid m \in \mathbb{N} \wedge \sum_{k=0}^{m} 2^k \geq N \wedge N < 2^{m+1}\}$ be the upper bound set, which involves documents of $V_N(q)$(in power two)

Definition 20. Let i be the position of a document in $V_N(q)$, such as the first element ($i = 1$) represents the most similar document, then $f_x(i, N) = min\{x \mid x \in \mathbb{N} \wedge i \leq \sum_{k=1}^{x} \frac{2^{M(N)}}{2^k}\}$, corresponds to the number of sets assigned for the document i.

Definition 21. Let
$v(i, N) = (2^{M(N)-f_x(i,N)} - 1) - [\langle(\sum_{k=1}^{f_x(i,N)} 2^{M(N)-k}) - i\rangle mod(2^{M(N)-f_x(i,N)})]$ be the value assigned to i, from 0 to $2^{M(N)-f_x(i,N)}$.

Definition 22. $\Phi(i, N) = log_2(2^{M(N)-f_x(i,N)} - v(i, N)) - \|log_2(2^{M(N)-f_x(i,N)} - v(i, N))\|$ a decimal number, which is $[0, 1[$.

Definition 23. Let
$K(i, N) = \begin{cases} \lceil \Phi(i, N) \rceil * \|M(N) - f_x(i, N)\| & : \quad if \quad \Phi(i, N) \geq 0.5 \\ \lfloor \Phi(i, N) \rfloor * \|M(N) - f_x(i, N)\| & : \quad if \quad \Phi(i, N) < 0.5 \end{cases}$ be the number of iterations to look for a *hit* in the array B.

Thus, $\beta : F(i) \rightarrow \{0, 1\}$, is the *hit* and *miss* function.

$$F(i) = \begin{cases} 1 & : \quad Pr_i(1) = \sum_{l=1}^{K(i,N)} \frac{2^{\|M(N)-f_x(i,N)\|}}{(2^{M(N)})^l}, \\ 0 & : \quad Pr_i(0) = 1 - Pr_i(1) \end{cases}$$

where $Pr_i(1)$ is the probability of a *hit* (1), and $Pr_i(0)$ corresponds to the probability of a *miss* (0) for the element i.

The third algorithm works as follows. The list of retrieved documents is split in subsets of elements in power two. In our case, $V_N(q)$ has 30 documents, however it can be approximated to 32 documents. Thus, if we apply *Definition 20*, then $M(N) = 5$. Therefore, $V_N(q)$ is split in 5 subsets. In general terms, $Pr_i(1)$ for every subset is different. Specifically on $\frac{2^{\|M(N)-f_x(i,N)\|}}{(2^{M(N)})}$. Thus, the space of possible candidates for the first subset is $\frac{2^{\|5-1)\|}}{(2^5)} = \frac{1}{2}$, for the second subset is $\frac{2^{\|5-2)\|}}{(2^5)} = \frac{1}{4}$ and so on. To show how the probability decreases according to the subsets, two examples are provided, for the first subset and the third subset. The second element of the first subset is $i = 2$, thus applying the *Definition 20*, $f_x(2, 30) = 1$, therefore, $v(2, 30) = (2^{5-1} - 1) - [\langle(\sum_{k=1}^{1} 2^{5-1}) - 2\rangle) \; mod \; 2^{5-1}] = 1$ (see *Definition 22*). Applying *Definition 23*.
$\Phi(2, 30) = log_2(2^{5-1} - 1) - \|log_2(2^{5-1} - 1)\| = 3.906 - 3 = 0.906$.

In figures 5.21, 5.22, 5.23 and in Table 5.7, preliminary results are presented exponential distribution have been used to built the documents, Zeta distribution have been used to simulate the judgments of users. Similarly, other results are presented in figures from 5.24 to 5.29 and in Tables 5.8 and 5.9. As mentioned previously, in section 5.5 the final results are discussed and the algorithms are compared.

Algorithm 3:

Data: $B[N]$ (*see Definition 12*) is a boolean array, $V_N(q)$ is the list of retrieved documents for the query q, q' is the most similar query with respect to q

Result: $A_{New}(q')$ is a list of relevant documents for the query q'

```
1  A_New(q) ← ∅;
2  for i ← 0, sum ← 0 to sum < N + 1 do
3  |  sum ← sum + 2^i;
4  end
5  k ← i − 1;
6  for i ← 1 to N do
7  |  B[i] ← false;
8  end
9  for i ← 1 to N/2 do
10 |  j ← random(1, ..., N);
11 |  B[j] ← true;
12 end
13 l ← 1;
14 while k ≥ 0 AND l < N do
15 |  for i ← 0 to < 2^k do
   |  |  // This is the document position at the list
16 |  |  I ← 2^k − i  u ← log_2(I);
17 |  |  U ← ||u||;
18 |  |  u ← u − U ; // This is the decimal number, see Definition 22
19 |  |  if u − 0,5 ≥ 0 then
20 |  |  |  K ← ⌈log_2(I)⌉;
   |  |  |  // This is the number of iterations see Definition 23
21 |  |  else
22 |  |  |  K ← ⌊log_2(I)⌋;
23 |  |  end
24 |  |  for j ← 1 to j ≤ k * K do
25 |  |  |  if 2^k * 2 ≥ N then
26 |  |  |  |  index = N;
27 |  |  |  else
28 |  |  |  |  index = 2^k * 2 − 1;
29 |  |  |  end
30 |  |  |  if B[indexF] = true then
31 |  |  |  |  if  ([idDoc = Position(i of V_N(q))] is in A_Past(q)) then
   |  |  |  |  A_New(q') ← A_New(q') ∪ d_idDoc;
32 |  |  |  |  l ← l + 1;
33 |  |  |  |  ;
34 |  |  |  |  else  l ← l + 1;
35 |  |  |  |  ;
36 |  |  |  else
37 |  |  |  end
38 |  |  end
39 |  end
40 |  k ← k − 1;
41 end
42 return(A_New(q'));
```

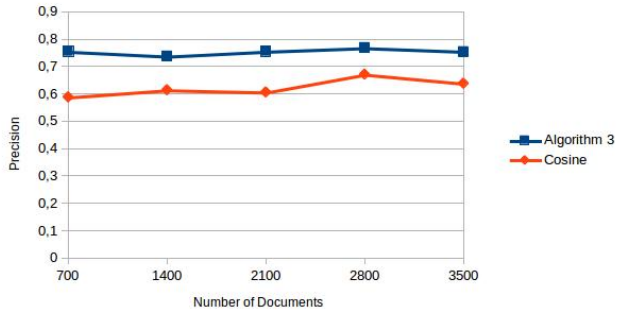

FIGURE 5.21: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 2 (for judgments)

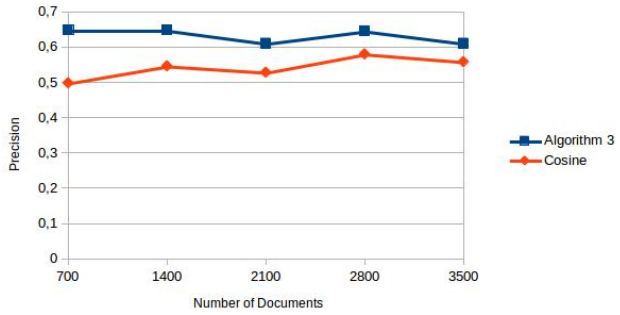

FIGURE 5.22: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 3 (for judgments)

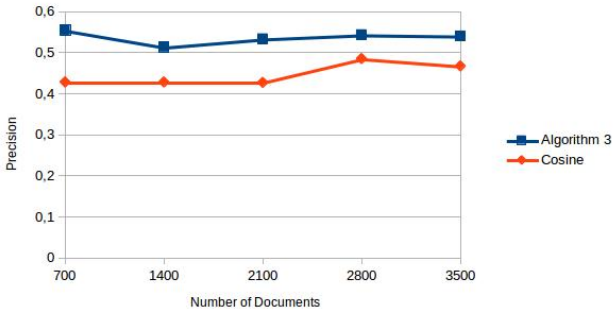

FIGURE 5.23: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.7: Evaluation results of the two approaches (i.e., *The third Algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.0$

		Results	
Zeta Distribution	D Size	Algorithm 3	Cosine
$S = 2$	700	0.756	0.589
	1400	0.738	0.615
	2100	0.756	0.607
	2800	0.769	0.672
	3500	0.754	0.639
$S = 3$	700	0.649	0.499
	1400	0.649	0.547
	2100	0.611	0.529
	2800	0.646	0.581
	3500	0.611	0.559
$S = 4$	700	0.555	0.429
	1400	0.514	0.429
	2100	0.534	0.428
	2800	0.544	0.486
	3500	0.541	0.468

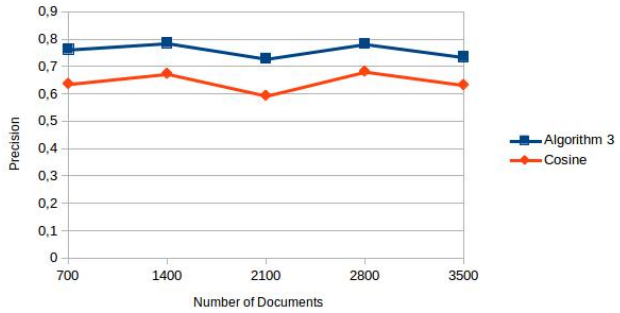

FIGURE 5.24: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 2 (for judgments)

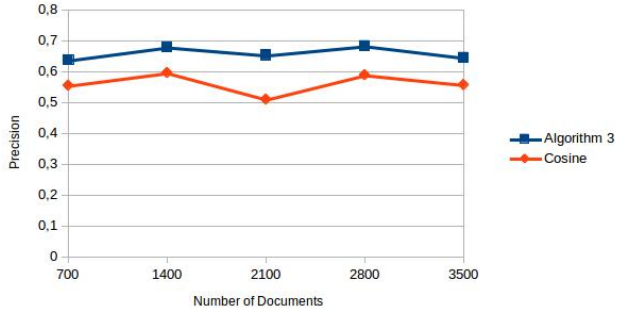

FIGURE 5.25: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 3 (for judgments)

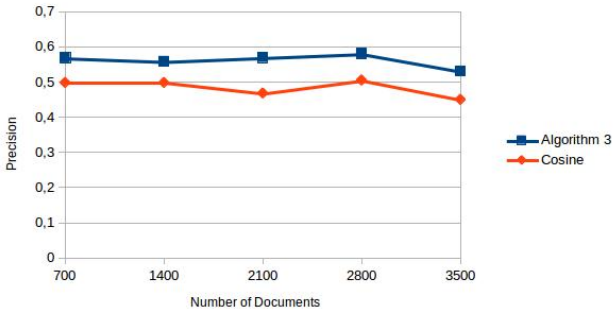

FIGURE 5.26: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.8: Evaluation results of the two approaches (i.e., *The third Algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.5$

		Results	
Zeta Distribution	D Size	Algorithm 3	Cosine
$S = 2$	700	0.764	0.638
	1400	0.787	0.675
	2100	0.730	0.595
	2800	0.784	0.683
	3500	0.736	0.634
$S = 3$	700	0.638	0.556
	1400	0.680	0.597
	2100	0.654	0.511
	2800	0.684	0.590
	3500	0.646	0.558
$S = 4$	700	0.569	0.499
	1400	0.559	0.499
	2100	0.570	0.469
	2800	0.581	0.506
	3200	0.531	0.451

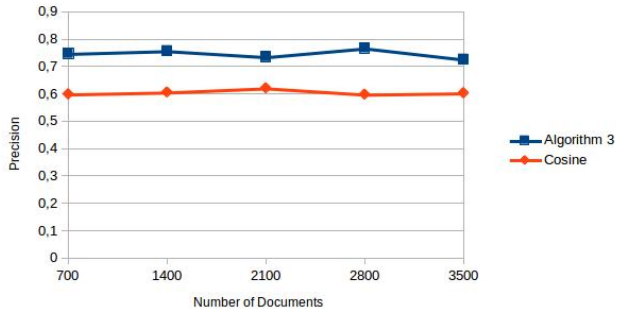

FIGURE 5.27: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 2 (for judgments)

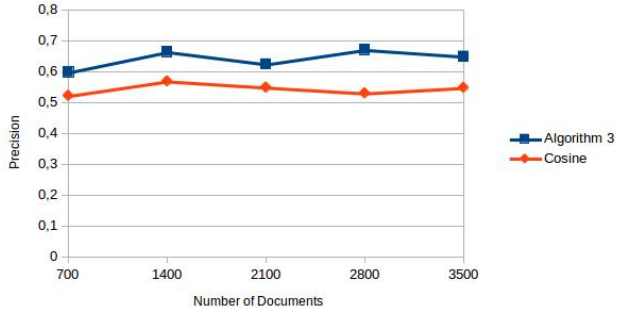

FIGURE 5.28: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 3 (for judgments)

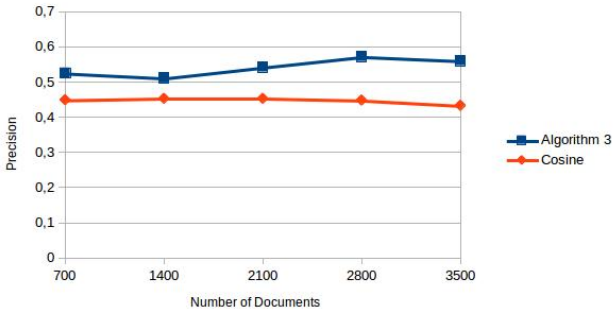

FIGURE 5.29: Evaluation of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.9: Evaluation results of the two approaches (i.e., *The third algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on Zipf distribution using $\lambda = 1.6$

		Results	
Zeta Distribution	D Size 2	Algorithm 3	Cosine
$S = 2$	700	0.748	0.660
	1400	0.758	0.607
	2100	0.736	0.622
	2800	0.768	0.599
	3500	0.727	0.604
$S = 3$	700	0.599	0.523
	1400	0.665	0.570
	2100	0.625	0.550
	2800	0.672	0.531
	3500	0.650	0.549
$S = 4$	700	0.526	0.450
	1400	0.512	0.454
	2100	0.543	0.454
	2800	0.573	0.449
	3500	0.561	0.434

The fourth algorithm

Finally, the fourth algorithm assigns probabilities for each document according to the position in the list. Thus, the first document, which is at the top of the list (for $V_N(q)$ (first position i=1)) has the highest probability to have a *hit* (1 or true). The next document with highest probability to have a *hit*, corresponds to the second document (the position i=2), and so on. Different to the previous design, this algorithm is naive, and takes into consideration only the list. Therefore, this assumption is based on that closer documents with respect to the query q, should be *relevant documents* (according to our approach).

The following definitions are used for the fourth algorithm.

Applying *Definition 24*, Let $K(2,30) = \lceil \Phi(2,30) \rceil * \|5-1\| = 1 * 4 = 4$.
Thus, $Pr_2(1) = \sum_{l=1}^{4} \frac{2^{\|5-1\|}}{(2^5)^l} = 0.757$. In the same way, $v(26,30) = (2^{5-3}-1) - [\langle (16+8+4) - 26 \rangle) \ mod \ 2^{5-3}] = 1$. Finally, $Pr_{26}(1) = \sum_{l=1}^{2} \frac{2^{\|5-3\|}}{(2^5)^l} = 0.128$

Definition 26. $K(i,N) = log_{\lceil \frac{b}{a} \rceil} \left(2^{\frac{N}{i}}\right)$ be the number of iterations to look for *hit* in the array B. Thus, $\beta : F(i) \rightarrow r$, is the function to guess the judgment of the user. $F(i) = \begin{cases} 1 & : \quad Pr_i(1) = \sum_{l=1}^{K(i,N)} \frac{a^l}{b^{l+1}}, \\ 0 & : \quad Pr_i(0) = 1 - Pr_i(1) \end{cases}$

and $Pr_i(1)$ is the probability of *hit* (1) for the document i, and $Pr_i(0)$ represents the probability of *miss* (0) for the document i. Therefore, the last document in $V_N(q)$ has less probability to have a *hit*. It is important to highlight that the previous designs are based on assumptions about the distribution of user judgments (relevance of documents in the list), because theoretically the distribution of user judgments is unknown. In other words, different collections, stopping, stemming processes and algorithms provided by IRS to recover documents, should provide different relevance distributions for the documents. Therefore, designs are general assumptions and not particular, but these are founded on the hypothesis that "closer documents with respect to the query q, should be *relevant documents*".

The probability for each document is determined by the number of iterations in order to look for an *1*, inside the array $B[N]$.

For instance, if $N = 20, a = 2, b = 3, i = 10$ (*the document in the position* i) and we give a natural number on $\lceil \frac{b}{a} \rceil = 2$. Thus $K(10,20) = 2$. $Pr_{10}(1) = \sum_{l=1}^{2} \frac{2^l}{3^{l+1}} = 0.370$. Initial results are exposed in figures 5.30, 5.31, 5.32 and in Table 5.10, exponential distribution have been used to built the documents, Zeta distribution have been used to simulate the judgments of users. In a similar way, other results are presented in figures from 5.33 to 5.38

and in Tables 5.11 and 5.12.

Algorithm 4:

Data: $B[N]$ (*see Definition 12*) is a boolean array, $A_{Past}(q)$ is a list of relevant documents for the query q, $V_N(q)$ is the list of retrieved documents for the query q, q' is the most similar query for q

Result: $A_{New}(q')$ is a list of relevant documents for the query q'

1 $A_{New}(q) \leftarrow \emptyset$;
2 **for** $i \leftarrow 1$ **to** N **do**
3 | $B[i] \leftarrow false$;
4 **end**
5 **for** $i \leftarrow 1$ **to** $\frac{N}{2}$ **do**
6 | $j \leftarrow random(1, ..., N)$;
7 | $B[j] \leftarrow true$;
8 **end**
9 $l \leftarrow 1$;
10 **for** $j \leftarrow 1$ **to** N **do**
11 | $flag \leftarrow false$;
12 | $k \leftarrow \lceil log_{\lceil \frac{b}{a} \rceil}\left(2^{\frac{N}{j}}\right)\rceil$;
| // This is the number of iterations see *Definition 24*
13 | **for** $i \leftarrow 1$ **to** k **do**
| | // From lines 13 to 18, the probability is obtained
14 | | $pos \leftarrow random(1, ..., N)$;
15 | | **if** $B[pos] = true$ **then**
16 | | | $flag \leftarrow true$;
17 | | **end**
18 | **end**
19 | **if** $flag = true$ **then**
20 | | **if** $([idDoc = Position(l\ of\ V_N(q))]\ is\ in\ A_{Past}(q))$ **then**
21 | | | $A_{New}(q') \leftarrow A_{New}(q') \cup d_{idDoc}$;
22 | | | $l \leftarrow l + 1$;
23 | | **end**
24 | **else**
25 | | $l \leftarrow l + 1$;
26 | **end**
27 **end**
28 $return(A_{New}(q'))$;

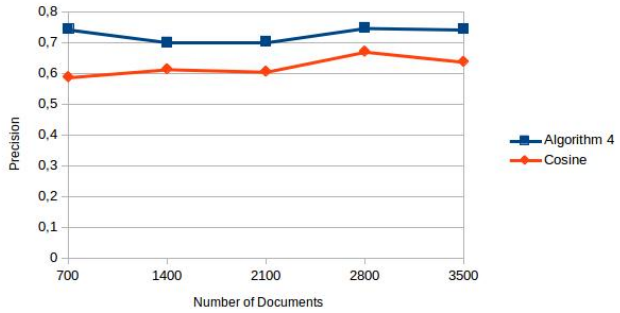

FIGURE 5.30: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 2 (for judgments)

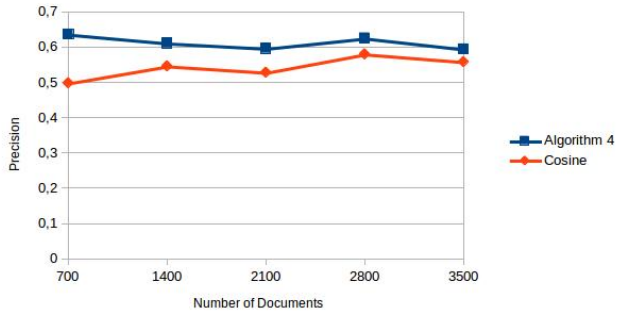

FIGURE 5.31: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 3 (for judgments)

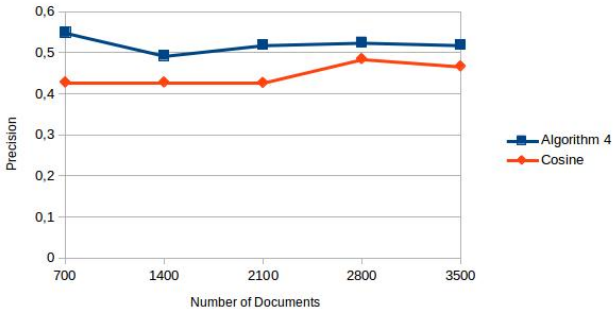

FIGURE 5.32: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.0 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.10: Evaluation results of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.0$

		Results	
Zeta Distribution	D Size	Algorithm 4	Cosine
$S = 2$	700	0.745	0.589
	1400	0.701	0.615
	2100	0.704	0.607
	2800	0.749	0.672
	3500	0.745	0.639
$S = 3$	700	0.637	0.499
	1400	0.612	0.547
	2100	0.597	0.529
	2800	0.626	0.581
	3500	0.595	0.559
$S = 4$	700	0.550	0.429
	1400	0.494	0.429
	2100	0.520	0.428
	2800	0.526	0.486
	3500	0.520	0.468

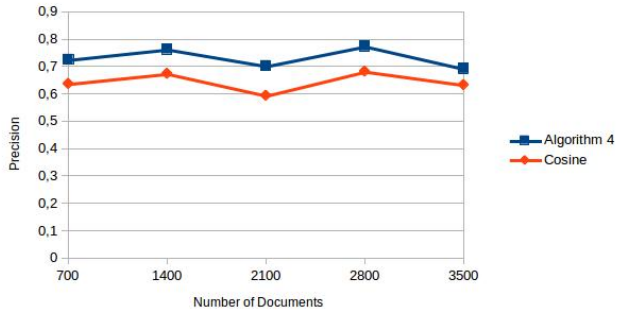

FIGURE 5.33: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 2 (for judgments)

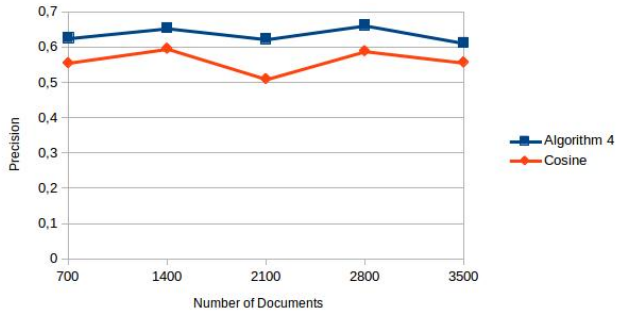

FIGURE 5.34: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 3 (for judgments)

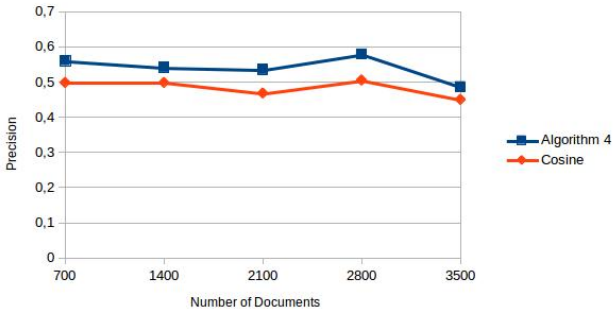

FIGURE 5.35: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Exponential Distribution 1.5 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.11: Evaluation results of the two approaches (i.e., *The fourth Algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.5$

		Results	
Zeta Distribution	D Size	Algorithm 4	Cosine
$S = 2$	700	0.726	0.638
	1400	0.764	0.675
	2100	0.703	0.595
	2800	0.775	0.683
	3500	0.693	0.634
$S = 3$	700	0.627	0.556
	1400	0.655	0.597
	2100	0.624	0.511
	2800	0.663	0.590
	3500	0.613	0.558
$S = 4$	700	0.561	0.499
	1400	0.542	0.499
	2100	0.536	0.469
	2800	0.580	0.506
	3200	0.487	0.451

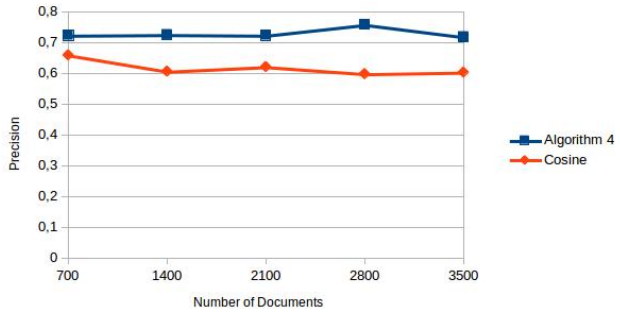

FIGURE 5.36: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 2 (for judgments)

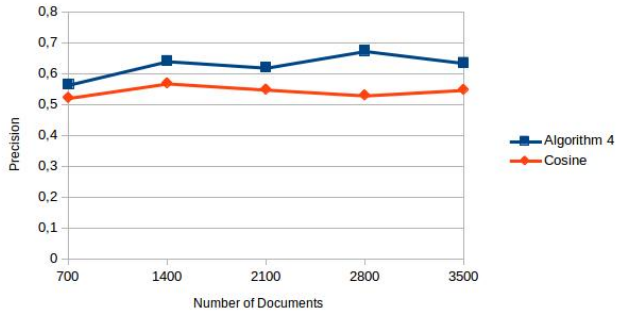

FIGURE 5.37: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 3 (for judgments)

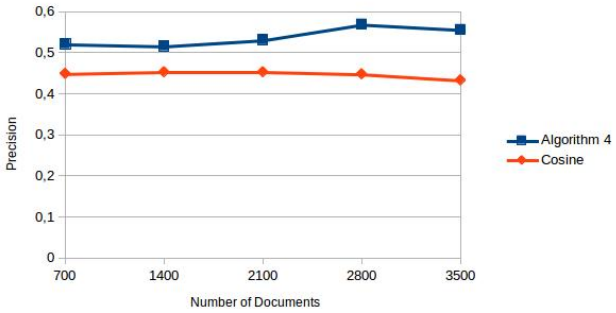

FIGURE 5.38: Evaluation of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 with a collection using Zipf Distribution 1.6 (for D) and Zeta Distribution 4 (for judgments)

TABLE 5.12: Evaluation results of the two approaches (i.e., *The fourth algorithm using Past Results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on Zipf distribution using $\lambda = 1.6$

		Results	
Zeta Distribution	D Size	Algorithm 4	Cosine
$S = 2$	700	0.724	0.660
	1400	0.726	0.607
	2100	0.724	0.622
	2800	0.759	0.599
	3500	0.719	0.604
$S = 3$	700	0.566	0.523
	1400	0.642	0.570
	2100	0.621	0.550
	2800	0.675	0.531
	3500	0.636	0.549
$S = 4$	700	0.522	0.450
	1400	0.517	0.454
	2100	0.532	0.454
	2800	0.570	0.449
	3500	0.557	0.434

5.5 Empirical Results

As stated in chapter four, specifically in section 4.3, which is related to the framework developed in this thesis, three sets compose the framework : documents, queries and relevance judgments per query (i.e., indications on relevant or non-relevant documents). In a first step, terms, documents, and queries are created. Two distributions are used, specifically exponential with values 1.5 and 1.0 in order to build documents. In the same way, Zipf distribution is used to represent the Heaps' law with the purpose to build the set of documents (Navarro et al., 2000; Silva de Moura et al., 2000). Zeta distribution is used to represent the Bradford's law, which provides relevant judgments assigned by users about document relevance for a specific query (Garfield, 1980). Both processes, elimination of stop-words and stemming were not carried out.

5.5.1 Experimental Environment

Experimental environment has been set up in the same way as in chapter four. Thus the length of a term $\mid t \mid$, was between 3 and 7. Uniform distribution was used to establish the length. The number of terms $\mid T \mid$ was 700 in each experiment. The number of terms for each document was between 15 and 30. Therefore, according to Heaps' law (Heaps, 1978), it was feasible to represent documents between 300 to 900 terms. The number of topics (i.e., a topic is subject or particular area, such as biology, computer sciences, chemistry among others) used in each experiment was 7, whereby each topic was formed by 100 terms. When a document was built, terms of other topics were chosen using either Exponential distribution or Zipf distribution. Whereby, most of the words, which composed a document were chosen from a specific topic. The number of documents used in each experiment corresponded to 700, 1400, 2100, 2800 and 3500.

On the other hand, terms for a query were between 3 and 8. Terms of a query were chosen from a particular document. Both terms and documents were chosen using uniform distribution to build the past queries. 15 composed the past queries. From the set of past queries, 15 new queries were built. Finally, the number of queries in each experiment was 30.

5.5.2 Experimental Results

In this section, three experiments for the four randomized algorithms are reported. Both, in first and second experiment, Exponential distribution have been used to build the collection of documents D. Like previous, in the third experiment, Zipf distribution has been applied to build D. Additionally, we have used the Student's Paired t-Test (paired samples) on each average P@10 (our

approach with respect to cosine) for each set of documents (700, 1400, 2100, 2800 and 3500). Results are shown as average P@10 over all the queries.

5.5.2.1 Experiment 1

a) Exponential distribution (with parameter $\theta = 1.0$) was used to build D. Zeta distribution (with parameter $S = 2$) was applied to determine the list of relevant documents for each query. The average results for P@10 are displayed in Table 5.13. Algorithm 1 had an average of 3.8 queries that were not improved applying our approach (with respect to the past query). An average of 25.8 queries led to result improvements with our approach. The highest p-value was lower than 1.0 E-6 applying the Paired t-test. Algorithm 2 presented an average of 10.6 queries that were not improved and an average of 19.4 queries were improved using this algorithm. The highest p-value for algorithm 2 was also lower than 1.0 E-6 applying the Paired t-test. The algorithm 3 had an average of 3.8 queries that were not improved applying our approach (with respect to the past query). An average of 26 queries led to result improvements with our approach. The highest p-value was lower than 4.0 E-5 applying the Paired t-test. For the algorithm 4, the average number of queries where our approach was improved, corresponds to 23.8. The highest p-value was 1.169 E-19.

b) Here, exponential distribution ($\theta = 1.0$) was applied to build D. Zeta distribution ($S = 3$) was used to determine the list of relevant documents for each query. Average results for P@10 are displayed in Table 5.13. Algorithm 1 had an average of 7.4 queries that were not improved with our approach. An average of 21 queries led to result improvements with our approach. The highest p-value was lower than 1.0 E-6 applying the Paired t-test. Algorithm 2 presented an average of 10.6 queries not improved and an average of 18.4 queries were improved using this algorithm. The highest p-value for algorithm 2 was 9.91 E-005 applying he Paired t-test. The algorithm 3 presented an average of 7.6 queries not improved and an average of 21.2 queries were improved using this algorithm. The highest p-value for algorithm 3 was 0.014 applying he Paired t-test. For the algorithm 4, The average number of queries, where our approach lost with respect to the past query was 9. The average number of queries where our approach was improved, corresponds to 19.2. The highest p-value was 1.129 E-09.

c) In this scenario, exponential distribution (with parameter $\theta = 1.0$) was used to build D. Zeta distribution (with parameter $S = 4$) was applied to determine the list of relevant documents for each query. In Table 5.13, the average results for P@10 can be seen. Algorithm 1 had an average of 5.8 queries that were not improved with our approach. Our approach improved, on average, 21.1 queries. The highest p-value was lower than 1.0E-6 applying the Paired t-test. The algorithm 2 led to an average of 10.6 queries not improved and an average of 17 queries were improved using this algorithm. The highest

p-value for the algorithm 2 was 1.71 E-005 applying the Paired t-test. The algorithm 3 had an average of 6.8 queries that were not improved applying our approach (with respect to the past query). An average of 20.4 queries led to result improvements with our approach. The highest p-value was lower than 0.0029 applying the Paired t-test. For the algorithm 4, The average number of queries where our approach did not improve regarding past queries, was 8. The average number of queries where our approach was improved, corresponds to 19.6. The highest p-value was 3.070 E-10.

TABLE 5.13: Evaluation results of the two approaches (i.e., *the 4 algorithms reusing past results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.0$

		Approaches				
Zeta Distribution	D Size	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Cosine
$S = 2$	700	0.758	0.689	0.756	0.745	0.589
	1400	0.738	0.707	0.738	0.701	0.615
	2100	0.756	0.712	0.756	0.704	0.607
	2800	0.769	0.677	0.769	0.749	0.672
	3500	0.754	0.714	0.754	0.745	0.639
$S = 3$	700	0.642	0.603	0.649	0.637	0.499
	1400	0.649	0.618	0.649	0.612	0.547
	2100	0.608	0.576	0.611	0.597	0.529
	2800	0.652	0.586	0.646	0.626	0.581
	3500	0.611	0.568	0.611	0.595	0.559
$S = 4$	700	0.555	0.520	0.555	0.550	0.429
	1400	0.516	0.506	0.514	0.494	0.429
	2100	0.534	0.503	0.534	0.520	0.428
	2800	0.547	0.478	0.544	0.526	0.486
	3500	0.541	0.499	0.541	0.520	0.468

5.5.2.2 Experiment 2

a) In this scenario, exponential distribution (with parameter $\theta = 1.5$) was applied to build the collection D. Zeta distribution (with parameter $S = 2$) was used to determine the list of relevant documents for each query. Average results for P@10 are displayed in Table 5.14. Algorithm 1 had an average of 4 queries that were not improved applying our approach while an average of 25.6 queries with result improvements. When calculating the Student's paired t-test, the highest p-value was lower than 1.0 E-6. The algorithm 2 presented an average of 10.2 queries without improvements and an average of 19.6 queries were improved using this algorithm. The highest p-value for the algorithm 2 was of 1.50 E-006 for the Paired t-test. The algorithm 3 had an average of 3.8 queries that were not improved applying our approach (with respect

to the past query). An average of 26.4 queries led to result improvements with our approach. The highest p-value was lower than 0.00019 applying the Paired t-test. For the algorithm 4, the average number of queries where our approach did not improve was 6.2 over 30 queries. The average number of queries improved corresponds to 23.4. The highest p-value for the t-test was 7.081 E-05.

b) Exponential distribution (with parameter $\theta = 1.5$) was used to build the collection D. Zeta distribution (with parameter $S = 3$) was used to determine the list of relevant documents for each query. In Table 5.14 shows the average results for P@10. The algorithm 1, presented an average of 5.4 queries that were not improved applying our approach. Our approach improved, on average, 24 queries. The highest p-value was lower than 1.0 E-06 for the Paired t-test. The algorithm 2 had an average of 11.4 queries not improved and an average of 18.6 queries were improved using this algorithm. The highest p-value for the algorithm 2 was 0.0045 applying the Paired t-test. The algorithm 3 had an average of 5.4 queries that were not improved applying our approach (with respect to the past query). An average of 24.2 queries led to result improvements with our approach. The highest p-value was lower than 0.004 applying the Paired t-test. The algorithm 4 had an average of 8.4 queries that were not improved applying our approach. The average number of queries, where our approach was better, corresponds to 21.2. The highest p-value for the t-test was 4.464 E-05.

c) In this scenario, exponential distribution (with parameter $\theta = 1.5$) was applied to build the collection D. Zeta distribution (with parameter $S = 4$) was used to determine the list of relevant documents for each query. Average results for P@10 are presented in Table 5.14. The algorithm 1 did not improve, on average, 8 queries. An average of 19.4 queries were improved. The highest p-value applying the Paired t-test was lower than 1.0 E-06. The algorithm 2 had an average of 10.2 queries that were not improved and an average of 17.8 improved queries using this algorithm. The highest p-value for the algorithm 2 was 0.028 applying the Paired t-test. The algorithm 3 had an average of 6.6 queries that were not improved applying our approach (with respect to the past query). An average of 21.2 queries led to result improvements with our approach. The highest p-value was lower than 0.0031 applying the Paired t-test. For the algorithm 4, The average number of queries, where our approach did not improve was 8. The average number of queries where our approach was improved, corresponds to 18.6. The highest p-value was 1.304 E-08.

TABLE 5.14: Evaluation results of the two approaches (i.e., *the 4 algorithms reusing past results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on exponential distribution using $\theta = 1.5$

		Approaches				
Zeta Distribution	D Size	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Cosine
$S = 2$	700	0.764	0.720	0.764	0.726	0.638
	1400	0.787	0.726	0.787	0.764	0.675
	2100	0.716	0.641	0.730	0.703	0.595
	2800	0.784	0.736	0.784	0.775	0.683
	3500	0.736	0.684	0.736	0.693	0.634
$S = 3$	700	0.637	0.579	0.638	0.627	0.556
	1400	0.680	0.632	0.680	0.655	0.597
	2100	0.654	0.559	0.654	0.624	0.511
	2800	0.684	0.632	0.684	0.663	0.590
	3500	0.646	0.588	0.646	0.613	0.558
$S = 4$	700	0.564	0.494	0.569	0.561	0.499
	1400	0.564	0.502	0.559	0.542	0.499
	2100	0.570	0.482	0.570	0.536	0.469
	2800	0.578	0.548	0.581	0.580	0.506
	3500	0.531	0.487	0.531	0.487	0.451

5.5.2.3 Experiment 3

a) In this scenario, Zipf distribution (with parameter $\lambda = 1.6$) was applied to build D. Zeta distribution (with parameter $S = 2$) was used to determine the list of relevant documents for each query. Average results for P@10 are shown in Table 5.15. The average number of queries, where our approach did not improve results using past queries, was 4. The average number of queries where our approach improved results was 24.8. The highest p-value was lower than 1.0 E-06 applying the Paired t-test. The algorithm 2 had an average of 9.8 queries that were not improved and an average of 19.6 queries were improved using this algorithm. The highest p-value for the algorithm 2 was also lower than 1.0 E-06 for the Paired t-test. The algorithm 3 had an average of 4.2 queries that were not improved applying our approach (with respect to the past query). An average of 24.8 queries led to result improvements with our approach. The highest p-value was lower than 0.00013 applying the Paired t-test. For the algorithm 4, The average number of queries, where our approach did not improve regarding past queries, was 6. The average number of queries where our approach was improved, corresponds to 22.6. The highest p-value was 2.205 E-18.

b) Here, Zipf distribution (with parameter $\lambda = 1.6$) was used to build D. Zeta distribution (with parameter $S = 3$) was applied to determine the

list of relevant documents for each query. The results for P@10 can be seen in Table 5.15. The average number of queries where our approach failed was 6.8. The average number of queries where our approach was better was 22.4. The highest p-value was lower than 1.0 E-06 applying the Paired t-test. The algorithm 2 had an average of 11.2 queries that were not improved and an average of 17.4 queries were improved using this algorithm. The highest p-value for the algorithm 2 was 0.0190 applying the Paired t-test. The algorithm 3 presented an average of 8.2 queries not improved and an average of 21 queries were improved using this algorithm. The highest p-value for algorithm 3 was 0.0017 applying he Paired t-test. For the algorithm 4, The average number of queries where our approach lost with respect to the past query, was 8. The average number of queries where our approach was better, corresponds to 20.8. The highest p-value was 5.779 E-13.

c) Zipf Distribution (with parameter $\lambda = 1.6$) was used to build D. Zeta distribution (with parameter $S = 4$) was applied to determine the list of relevant documents for each query. Average results for P@10 are shown in Table 5.15. The average number of queries, where our approach did not improve, corresponded to 3.6. The average number of queries where our approach was better was 25.2. The p-value was lower than 1.0 E-06 applying the Paired t-test. The algorithm 2 had an average of 11 queries that were not improved and an average of 17.8 improved queries using this algorithm. The highest p-value for the algorithm 2 was 0.00162 applying the Paired t-test. The algorithm 3 presented an average of 4.6 queries not improved and an average of 24.4 queries were improved using this algorithm. The highest p-value for algorithm 3 was 0.006 applying he Paired t-test. For the algorithm 4, The average number of queries, where our approach did not improve, corresponds to 3.8. The average number of queries where our approach was better, was 24.4. The highest p-value was 9.413 E-21.

It is important to point out that both, the number of queries and number of documents do not influence final results. It is because there is no relation between the increase number of queries and the number of documents on precision results. The reasons are the following :

— Each past query is obtained from a document and the new query is built from a unique past query. In other words, for each past query there is a unique new query. Moreover, the intersection between past queries is empty as well as the intersection between new queries (see *Definition 1. b)* and *Definition 2*). For example, if we want to build one hundred queries, first we build fifty old query, which are different and subsequently the others fifty new queries are built from the past queries. On balance, increasing the number of queries does not have impact on the final results.

— Analogously, increasing the number of documents does not alter the final results, because the relevance of the documents (judgments of users) is obtained by applying Zeta distribution on the list of retrieved documents, which

are obtained from the matching between the query and all the documents of the collection (see section 4.3.2).

In summary, both increases (the number of documents and the number of queries) will not provide a variation on final results.

TABLE 5.15: Evaluation results of the two approaches (i.e., *the 4 algorithms reusing past results* and *Cosine*) according to average P@10 (over 30 queries) with a collection D based on Zipf distribution using $\lambda = 1.6$

		Approaches				
Zeta Distribution	D Size	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Cosine
$S = 2$	700	0.748	0.689	0.748	0.724	0.660
	1400	0.758	0.708	0.758	0.726	0.607
	2100	0.736	0.663	0.736	0.724	0.622
	2800	0.767	0.706	0.768	0.759	0.599
	3500	0.727	0.678	0.727	0.719	0.604
$S = 3$	700	0.599	0.551	0.599	0.566	0.523
	1400	0.659	0.601	0.665	0.642	0.570
	2100	0.625	0.557	0.625	0.621	0.550
	2800	0.673	0.592	0.672	0.675	0.531
	3500	0.650	0.576	0.650	0.636	0.549
$S = 4$	700	0.526	0.479	0.526	0.522	0.450
	1400	0.547	0.479	0.512	0.517	0.454
	2100	0.543	0.457	0.543	0.532	0.454
	2800	0.573	0.493	0.573	0.570	0.449
	3500	0.561	0.524	0.561	0.557	0.434

5.6 Conclusions

As mentioned in Chapter 3, few approaches take advantage of previously submitted queries. Queries previously processed can be useful when a new similar query is submitted to an IRS. Similar past queries (with respect to a new query) can be recovered along with their relevant documents and these documents can be reused to provide a list of documents for the new query. The research reported in this chapter, had as purpose to analyze the feasibility of using randomized algorithms to select relevant documents, which have been retrieved from the most similar past query.

The approach proposed is very different from the probabilistic that can be found in the literature. Probabilistic approaches can be categorized in two branches, learning and optimization approaches, which are founded on bayesian networks and their variants, or genetic algorithms. It should be noted that approaches based on randomized algorithms have not been studied on IR implementations.

Four randomized algorithms were designed and evaluated on three global experimental scenarios. Two of them split the list of retrieved documents in groups, an algorithm is inspired by logistic distribution because it provides the ranking notion for two possible answers (relevant or non-relevant), and the last is a basic algorithm. The four algorithms works over the list of retrieved documents from the most similar past query. It is important to emphasize that the designs of the algorithms were different, because they suppose in different ways the relevance of some documents according to their position in the list. In other words, the relevance distribution depends on the judgments of users issued for a list of recovered documents, given a query. Thus, different IRSs should consider different distributions of relevance. Therefore, designs are based on an assumption, which is general for all cases. It is that the relevant documents tend to be at the top of the result list returned for the query.

The experimental scenarios were simulated, and several distributions were used to build documents and provide the judgments of users (see chapter 4). Empirical results showed better precision (P@10) of randomized algorithms (using results retrieved with cosine model) compared with the initial traditional retrieval (cosine). The experimental results were validated applying the Student's paired t-test in each experiment. The best results were provided by algorithm 1, which splits the list of retrieved documents in similar groups (with the same quantity of documents). The main advantages of these algorithms are : first, these algorithms are easy to implement, do not require time of learning as used in approaches relying on optimization techniques. Second, these algorithms can be implemented inside IRSs or search engines, or externally.

The approach presented in this chapter relies on the storage of past queries with their results and the search of similar past queries when a new submitted query. An improvement of the approach can reside in clustering past queries and find the most similar cluster for a new submitted query. The result associated to the query representing the cluster (i.e., centroid) could then be used to build the result for the new query. The next chapter introduces a first idea of clustering approach for this purpose.

Chapter 6

Clustering in IR

Résumé : Chapitre 6

Comme mentionné dans le chapitre 3, plusieurs approches tirent avantages des requêtes soumises précédemment. Néanmoins, la plupart d'entre elles exploitent les requêtes répétées dans le but d'améliorer leur performance en temps de traitement. Ces approches sont liées principalement à la gestion de cache. D'autres types d'approches visent à améliorer les résultats en étendant ou modifiant les requêtes.

Dans les chapitres 4 et 5, nous avons présenté des résultats sur l'utilisation de requêtes passées similaires. Bien que ces résultats soient préliminaires, ceux-ci éclairent sur la possibilité d'améliorer la précision. À partir des résultats empiriques, deux aspects importants sont à mentionner. Tout d'abord, la similarité entre les requêtes a été effectuée en utilisant la mesure cosinus. Deuxièmement, certains documents pertinents de la requête passée la plus similaire peuvent être utilisés pour répondre à une nouvelle requête. Par conséquent, les deux aspects doivent être considérés dans le but de fournir de bons résultats pour les approches reposant sur l'utilisation de requêtes passées. Un troisième aspect non négligeable correspond au stockage. Par conséquent, il est nécessaire de stocker et de recueillir les requêtes passées similaires avec leurs documents, et de comparer une nouvelle requête soumise à toutes les requêtes passées stockées pour en trouver de similaires. Une structure de données qui permet d'optimiser le stockage et la recherche de requêtes passées similaires correspond aux clusters. Le clustering repose généralement sur des mesures de similarité. Dans le cadre de cette thèse, qui porte sur la réutilisation des recherches précédentes, une nouvelle mesure de similarité pour le clustering est proposée, qui tient compte des requêtes et de leurs résultats. Cette mesure est appelée QDSM (Query-Document Similarity Measure). QDSM est basée sur l'utilisation des documents pertinents, qui sont associées aux requêtes. QDSM est calculée en utilisant la plus longue

séquence commune (LCS).

Deux grandes catégories de clustering sont facilement identifiables dans la littérature : le clustering statique et le clustering post-recherche. D'une part, le clustering statique est l'application traditionnelle de la méthode de clustering sur une collection de documents. D'autre part, le clustering post-recherche utilise de l'information liée à la requête dans le clustering de documents. La similarité entre les requêtes est liée à l'intersection entre les termes de la requête. Typiquement, les fonctions telles que Jaccard ou cosinus sont utilisées pour mesurer la similarité entre les requêtes. Toutefois, ces fonctions ne prennent pas en compte le contexte spécifique dans lequel la similarité des deux objets est jugée.

Ce chapitre vise à étudier une approche préliminaire sur une nouvelle mesure de similarité (QDSM) pour un clustering requête-document. Le chapitre est organisé comme suit. Dans la section 6.2, les principes du clustering en RI sont décrits. Dans la section 6.3, les travaux connexes sur les approches qui traitent de clustering en RI sont présentés. Dans la section 6.4, notre contribution est décrite. Dans les sections 6.5 et 6.6, les expérimentations et les résultats finaux sont décrits. Enfin, dans la section 6.7, les conclusions sont rapportées.

6.1 Introduction

As mentioned in chapter 3, several approaches obtain advantages from submitted queries previously. Nevertheless, most of them benefit from repeated queries with the purpose to improve their performance in run-time. Approaches that address the performance in run-time are related mainly with caching. Other types of approaches aiming to improve the results extending or modifying the queries.

In chapter 4 and chapter 5, we have presented some results about the use of similar past queries. Despite the fact that these results are incipient, these are interesting giving hints on the possibility to improve precision. From the empirical results, two aspects are important to mention. First, the similarity between queries was carried out using cosine. Second, some relevant documents from the most similar past query can be used to answer a new query. Therefore, both aspects should be considered with the aim to provide good results for the approaches relying on the use of past queries. A third aspect not negligible corresponds to the storage. Therefore, it is necessary to store and gather the similar past queries along with their documents, and compare a new submitted query to all the stored past queries to find similar ones. A data structure that enables optimizing the search of similar past queries corresponds to clusters. Clustering usually relies on similarity measures. In the domain of this thesis, dealing with the reuse of past searches, a new similarity measure for clustering is proposed, which considers the queries and their results. This measure is called Query-Document Similarity Measure (QDSM). QDSM is based on the use of relevant documents, which are associated to queries. Relevant document are known when the retrieval process has finished for a past query (i.e., relevant documents are unknown before to be submitted in an IRS). QDSM is calculated by using the longest common subsequence (LCS).

Two major categories of clustering are easily identifiable in the literature : static clustering and post-retrieval clustering. On one hand, static clustering is the traditional application of the cluster method on a document collection. On the other hand, post-retrieval clustering includes information from the query into the clustering of documents. Similarity between queries is related to the overlapping among query terms. Typically, similarity functions such as Jaccard or cosines are used to measure the similarity between queries. Nevertheless, these functions do not consider the specific context under which the similarity of two objects is judged.

This chapter aims at investigating a preliminary approach on a new similarity measure (QDSM) for query-document clustering. The chapter is organized as follows. In section 6.2, document clustering for IR is described. In section 6.3, related work about approaches which deal with clustering in IR are presented. In section 6.4, our contribution is described. In sections 6.5 and 6.6, the experimental environments and final results are provided. Finally, in section 6.7, conclusions are reported.

6.2 Document Clustering for IR

The clustering task has been carried out by humans (Willett, 1988) for a long time. Cluster analysis or clustering is a multivariate statistical technique that aims to collect object groups in such a way that objects in the same group are similar in a space, which is usually multi-dimensional. Groups of objects are composed in such a way that objects in the same cluster are similar to one another and dissimilar to objects in other clusters (Gordon, 1987). Cluster analysis techniques have been widely applied in different research fields such as medical sciences, social sciences, earth sciences and engineering sciences, among others (Anderberg, 1973). Indeed, several applications of cluster analysis can be found in many areas of research. Nowadays, this task has been fully automated thanks to advances in computer technology (Willett, 1988). Furthermore, the application of cluster analysis has been carried out in IR not only for term clustering, but also for document clustering. Document clustering is applied on the basis of shared terms between documents. Term clustering provides a group representation of terms that belong to documents or queries. Usually, term clustering is applied in query expansion, thesaurus linking, and automatic thesaurus construction.

Typically, document clustering has been applied statically to whole document collections before querying. By contrast, the resulting groups of documents obtained from post-retrieval clustering tend to be different for different queries. Two types of clustering are widely known in IR, partitioning and hierarchic.

Commonly in partitioning clustering, documents are represented by a vector in an n-dimension space, where n corresponds to the number of terms that compose the indexing vocabulary of the database. Thus, given a set of N documents, partitioning cluster builds a single organization of k mutually exclusive clusters, where k is either priori provided, or determined as part of clustering method. Usually, the computation requirements of this method are low, which are comprised between $O(N)$ and $O(NlogN)$ for the clustering of N documents (Willett, 1988). As a result, partitioning methods were very popular in a started early in IR (Salton, 1971). The basic idea behind this method is to choose some initial partition of documents, later on add objects incrementally to the clusters to obtain a better partition (Anderberg, 1973) (e.g., cluster membership, number of clusters, cluster size) to achieve an optimal solution (Salton and Wong, 1978; Willett, 1988). First experiments showed that the effectiveness of searches based on document partitions is significantly inferior to searches based on unclustered files (Salton, 1971).

Many hierarchic clustering applications exist in the IR literature, which have been carried out considering the use of single terms. On the contrary, recent approaches deal with document representations through phrasal units by employing different levels of linguistic analysis. This type of clustering has been widely accepted in the IR community Willett (1988), since it provides a sound theoretical basis. Commonly, each document D is represented as a vector

$D = \{d_1, d_2, ..., d_n\}$, where n is the number of terms that compose the indexing vocabulary of the document collection. All terms that compose the indexing vocabulary are used in the indexing representation (Rijsbergen, 1979). Before clustering the documents, several processes take place, such as stemming, stopping, and normalization. Subsequently, a relative weight about the importance of each term considering the whole document collection is obtained to increase effectiveness (Salton and Buckley, 1987). Once an appropriate representation is obtained for the set of documents to cluster, it is necessary to have a measure according to the similarity degree for all possible pairs of documents that belong to this set. To achieve this goal, a large number of measures that quantify the resemblance between objects can be applied to provide a categorization. Four main classes of measures are distinguished : association, dissimilarity, probabilistic, and correlation coefficients (Sneath and Sokal, 1973). Most of the literature deals with the association and dissimilarity, whereas the use of correlation coefficients and probabilities in document clustering is limited.

It is important to highlight that in the domain of this thesis, experimentations will be restricted to single-term indexing units. Thus, a hierarchic clustering is employed in this thesis. This type is appropriate for our issue and is widely accepted in the IR community (Willett, 1988), since it provides a sound theoretical basis. A contribution of this thesis is to change the static use of similarity, and to provide a new query-sensitive similarity measure.

6.3 Related Work

This section presents approaches of the literature dealing with query clustering which take into account the context in which the similarity is used.

A pioneering approach was proposed by Tombros et al. (2002) by using hierarchic query-specific clustering with the aim to enhance of effectiveness in the retrieval process. To this end, a set of experiments have been studied. The assumption is that the hierarchy should be adjusted to a specific query increasing the probability to put relevant documents to the query in close clusters. Specifically, two main aspects of this research have been considered. First, studying progressively the variation of the optimal cluster effectiveness achieved from the application of hierarchic clustering to a larger number of top-ranked documents returned from an IFS, and second comparing this effectiveness with the effectiveness of an IFS. In the experimental environment five document collections CACM, CISI, LISA, Medline and TREC (WSJ) have been considered. Furthermore, four hierarchic agglomerative methods : Group average, Ward, Complete Link and Single Link were compared, meanwhile seven different numbers of top-ranked documents were employed in the experiments. Final results indicated that there is not a statistically significant variation in query-specific clustering effectiveness for different values of top-ranked documents, and that query-specific clustering significantly outperforms static clustering for all expe-

rimental conditions. The main conclusion from these results is that they provide evidence for the application of hierarchic query-specific clustering to IR based on improved effectiveness. Inspired for this pioneering work on the QSSM measures, from the empirical results of this work, we have decided to use Average Link algorithm because this algorithm provided the best results. However, the approach presented in this thesis is incipient and more experiments should be carried out in a future work considering more document collections.

A next work by Tombros and van Rijsbergen (2004) proposed an axiomatic view where relevant documents tend to be highly similar to each other, therefore these should appear in the same cluster. The approach proposes the use of query-sensitive similarity measures (QSSM) whose purpose is to join a pair of documents, which have attributes that are expressed in the query. Query-sensitive measures can be defined through a function with two components. The first component corresponds to a conventional similarity between two documents, meantime the second component takes into consideration a common similarity for three objects : a pair of documents and the query. Three query-sensitive measures were tested M1, M2 and M3. Two measures (M1 and M3) used a different function to combine static and variable similarities (M1 used a product of the two sources while M3 used a linear combination). The third measure only took into account common terms between documents and in query terms (measure M2). Six document collections have been used in the experimental scenarios. To determine the separation degree between relevant and non-relevant documents, the N-Nearest Neighbour test have been applied, in particular the 5-NN test. Final results indicate that measures M1 and M3 are always significantly more effective than the cosine, and are not strongly dependent on query length. Otherwise, the measure M2 is sensitive to variations of query length, but despite this, it also brought significant improvements over the cosine in a large number of experimental conditions. The main conclusion from this research is that the use of query-sensitive measures for the calculation of inter-document relationships is highly effective. Similar to this work, in this thesis a new measure which takes into account the context is proposed. This work is an inspiration because good results were obtained taking into account the queries. Differing from Tombros and van Rijsbergen (2004), the queries are stored along with their documents, which were retrieved for the submitted query.

Similarly, Hasanzadeh and Keshavarzi (2009) pointed out that relevant documents tend to be more similar to each other than non-relevant documents, thus they tend to appear in the same clusters. Whereby, query-specific clustering was investigated by using a QSSM on an experimental environment. Broadly speaking, the system takes a query and a large document collection as input. In the first step, relevant documents are found and displayed as a ranked list according to their similarity with the query. Subsequently, some of the top documents in the list along with the query are clustered by using a QSSM. The TIPSTER document collection and the K-mean algorithm have been employed to carry out the experiments. The measure of similarity considers the cosine between two relevant documents and the cosine between the documents and the query. From

the conclusions, the authors claimed that query-specific clustering improves the effectiveness of ranked lists. Like in this work, in this thesis the queries are also stored along with their documents. Nevertheless, the main difference is that our measure considers both the number of relevant documents and the cosine between the queries, where LCS algorithm is applied to obtain the similarity (i.e., the query is stored with n most similar documents with respect to the query, the documents are relevant and non-relevant documents). Furthermore, the Average Link algorithm was used for clustering and tested in our experiment on the CACM collection.

With a different goal, Baeza-Yates et al. (2004) developed a method, which provides a list of related queries for a submitted query to a search engine. Related queries correspond to previously submitted queries. The method is grounded on a query clustering process, where groups of semantically similar queries are identified. Historical preferences of registered users in query logs are used to build the clusters. The clustering process is based on a term-weight vector representation of obtained queries considering the clicked URLs from the query. The method presents two advantages. First, it determines the related queries and second, it ranks the queries according to a relevance criterion. The queries are ranked considering two criteria : (a) the similarity between the queries which belong to the cluster and the input query, and (b) the support, which measures how much the answers of the query have attracted the attention of users. Finally, through the combination of the measures (a) and (b), it is feasible to define the interest of a recommended query. To test this approach, a set of experiments was carried out. Log files of a search engine were used. Empirical results showed improvements on average precision. Similar to this research, in our approach the previously submitted queries are stored in the cluster along with their documents. We also consider the related queries (i.e., past queries stored in the cluster), which are obtained using our similarity measure. Nevertheless, we do not consider user sessions nor collections based on log files.

An interesting classification of query clustering was proposed by Fu et al. (2004) who compared different query similarity measures. In this classification three categories : Content-based approaches, Feedback-based approaches and Results-based approaches have been proposed. Content-based approaches compare query term vectors. In simple words, common terms can be used to characterize the clusters of queries. Several similarity functions such as cosine similarity, Jaccard similarity, and Dice similarity were used in this category. The authors suggested that this method is not profitable in the context of search engines, due to the average length of query terms. In contrast, feedback-based approaches use users' selections on search results as similarity measure. Such approaches rely on clicked documents by user using log files. Thus, two queries are similar if they promote the selection of similar documents. Two disadvantages are mentioned, the first is given when the number of relevant documents is high, and the second occurs when there are few common documents. Results-based approaches calculate the similarity between queries by calculating the overlap on returned documents for the queries. Relevant documents should appear at the begin-

ning of the result lists. The main drawback of this kind of approach is that it consumes high execution time. In the experimental scenarios, URLs have been considered to obtain the empirical results. Final results showed good results when the three approaches were used at the same time. This work is interesting for us, because it supplies a classification of query clustering. From this classification, the approach exposed in this thesis is a mix between Content-based approaches and Results-based approaches, because we use both the term queries and their results. Specifically, we also calculate the overlap on returned documents. Furthermore, the author pointed out that "relevant documents should appear at the beginning of the result lists", which is one of main assumptions in this thesis.

In the context of query-oriented multi-document summarization, Wei et al. (2008) developed a cluster-sensitive graph model. Over this model, an algorithm called QoCsR is executed. Global and local information (intra and inter-document) are considered in this model. Five types of relations are used, which consider three objects : document, sentence and query. The relevance between sentence and query as well as document and query are considered. A cluster is defined as a document which has a collection of sentences. It considers the relations sentence-document, and most important the relation sentence-sentence (called intra-document). Using two clusters the relation sentence-sentence denominated inter-document is defined. Experimental evaluations using ROUGE and DUC 2005 datasets show improvements when using QoCsR. Similar to this work, in this thesis we want to represent the relations between the documents and queries. However, different to this approach, we do not present a cluster as a document with a collection of sentences. In addition, a graph model appears interesting as future work, specially when the number of clusters is considerable.

More recently, Saravanakumar and Moturi (2011) proposed a framework to identify and summarize the semantic of the user query. At the beginning, results are classified in potential groups, subsequently the user chooses a group to re-rank the final result. Whereby, this framework offers to the user a personalized search service. The proposed framework is composed of preprocessing module and clustering modules. In the preprocessing module, the query is given by the user where non-relevant words from the query are removed and the semantic is provided taking into account the context. Clustering modules are responsible to yield the semantic of this model. It is important to emphasize that the user's history is used to customize the information. User interests are stored in the user's profile to improve the search. Different to this research, in this thesis, we do not consider the user interactions with the clusters. Moreover, the session users are not covered in this thesis. However, this approach is relevant for us, because when a user is surfing on the web with the aim to cover his/her information needs, a query can be redefined as many times as necessary. Therefore, this research presents an interesting trend for future work.

A new interesting problem was introduced by Niederberger et al. (2012). They presented an approach concerned in clustering and visualization, with which it

is possible to learn at the same time that the dataset is increasing in size. A German document corpus have been used. This corpus comprises between 300 to 1000 documents with an average length of 54 terms. Experimental results show that this method outperforms standard clustering algorithms with respect to classification reliability in real time. At the same time, this approach provides an innovative visualization for the problem of dimensionality reduction. Despite the fact that this work is totally different with respect to this thesis, it provides a new interesting problem, which also was exposed by Fu et al. (2004). This problem is related with execution time, specially in real time context.

Moreno et al. (2013) suggested a new methodology, which adapts the K-means algorithm to a third-order similarity measure initially developed for Topic Segmentation. The adaptation of K-means algorithm allows labeling each cluster directly from its centroids. The evolution of the objective function on the adapted K-means is modeled to define automatically the "best" number of clusters. Two datasets have been used ODP-239 and MORESQUE. The performance of this approach have been compared to : PRC algorithms, OPTIMSRC, and the classical bisecting incremental K-means. A new measure denominated "b-cubed F" was used to evaluate not only cluster homogeneity but also completeness. Empirical results showed an improvement with respect to traditional approaches. This work is recent, and different to this research, which does not propose a new methodology, and does not use partition algorithms like K-means. Nevertheless, this approach is interesting not only from an efficiency point of view but also for the way to find the "best" number of clusters. As future work, we should consider other types of algorithms and datasets to corroborate our measure and its performance.

Different to the traditional metrics used in the IR clustering context, the query-sensitive similarity as metric has achieved a relevant position on IR clustering environment. Na (2013) suggested a probabilistic framework that defines query-sensitive similarity rooted in probabilistic co-relevance, where the similarity between two documents is proportional to the probability of co-relevance for a specific query. Two cases were considered to determine the co-relevance. First, the relevance of a document is independent from to the relevance of other documents. Finally, the relevance of a document is dependent of others. Aiming to prove this approach, several experimental scenarios have been executed using TREC collections and the nearest neighbour test. Final results showed that the proposed query-sensitive similarity measure performs better than term-based similarity. Like this research, our assumption is that a relevant document has a probability not inconsiderable to be relevant for another similar query. Nevertheless, we do not calculate that probability directly, instead, we use our measure.

6.4 Our Contribution

In this section, we describe our approach of clustering for reusing past queries and their results. This approach stores past queries along with a set of documents, in order to respond to new queries using relevant documents from the most similar query. The approach relies on a hierarchic clustering, and more precisely, on the group average link method. Hierarchic clustering is appropriate for our issue and is widely accepted in the IR community (Willett, 1988), since it provides a sound theoretical basis. Among the various approaches, the average link method is the most used hierarchic methods (Sneath and Sokal, 1973). Finally, a new similarity measure is defined in this query-document clustering context, which is denominated QDSM (query-document similarity measure).

6.4.1 Hierarchic Clustering Methods

Hierarchic clustering methods provide a classification in a tree-shaped structure, which is composed by objects (i.e., documents), where objects in the same cluster are strongly similar to each other. At the same time they are within larger clusters that contain less similar objects.

Let X be a set of documents, which will be clustered, $X = \{x_1, x_2, \ldots, x_N\}$, where each document x_i corresponds to a n-dimensional vector, and each dimension is an indexing term. A clustering of X in k sets can be defined as $R = \{C_1, C_2, ..., C_k\}$, such that the following conditions are satisfied :

— Each cluster C_i contains at least one document : $C_i \neq \emptyset, i = 1, ..., k$
— The union of all clusters is the set $X : \bigcup_{k_i=1}{}^{m}; Ci = X$
— Two clusters do not have documents in common : $C_i \cap C_j = \emptyset, i \neq j, i, j = 1, ..., k$

Let R_1 be a clustering composed by k clusters, which is nested in the clustering R_2, such that R_2 has $r < k$ clusters, if each cluster in R_1 is a subset of a cluster in R_2, and at least one cluster of R_1 is a proper subset of R_2 (Theodoridis and Koutroumbas, 1999). For example, the clustering $R_1 = \{\{x_1, x_3\}, \{x_4\}, \{x_2, x_5\}\}$ is nested in $R_2 = \{\{x_1, x_3, x_4\}, \{x_2, x_5\}\}$.
Otherwise, R_1 is not nested within $R3 = \{\{x_1, x_4\}, \{x_3\}, \{x_2, x_5\}\}$.

Hierarchic methods can be split into two categories, agglomerative and divisive. The agglomerative method provides a series of $(N - 1)$ merges, for a collection of N documents, where results of clustering are built from the bottom to the top of the structure. In the divisive method, a single initial clustering is split into smaller groups of documents consecutively (Rijsbergen, 1979). Typically, divisive methods supply as result monothetic classifications, where documents that belong to a specific cluster must contain certain terms with the

aim to obtain membership Sneath and Sokal (1973); Rijsbergen (1979); Gordon (1987). On the other hand, in polythetic clustering it is not necessary to have specific terms for membership in a cluster. Moreover, such structures are results of agglomerative methods. It should be noted that polythetic clusterings have been widely used in IR and that hierarchic agglomerative clustering methods (HACM) are still used in this field (Willett, 1988).

Agglomerative methods can be distinguished on two different theories, concepts of matrix theory or concepts of graph theory. Built on the premise that methods based on matrix theory are most used, in this thesis I will use this type of methods (Willett, 1988). Thus, the input to an HACM corresponds a similarity matrix $S(X)$ that contains the values for all interdocument associations.

Commonly, hierarchic agglomerative methods follow the next generic procedure (Murtagh, 1983) :

1. Determine all interdocument similarities.
2. Form a cluster from the two closest objects or clusters.
3. Redefine the similarities between the new cluster and all other objects or clusters, leaving all other similarities unchanged.
4. Repeat steps 2 and 3 until all objects are in one cluster.

Several agglomerative methods do not apply exactly the third step mentioned previously. At each step t of the clustering process, the size of the similarity matrix $S(X)$ (which initially is $N \times N$) is transformed into $(N-t) \times (N-t)$. Thus, the matrix in the step t, is derived from the matrix $S_{t-1}(X)$ by deleting the two rows and columns that correspond to the newly merged documents (or clusters), and by adding a new row and column that contain the new similarities between the newly formed cluster and all unaffected (from step t of the process) documents or clusters.

The representation obtained after applying a hierarchic clustering method corresponds to the form of a dendrogram (Jardine and Sibson, 1968) (see Figure 4.1). Usually, a dendrogram is represented as a tree, which has numeric levels associated to its branches. The numeric values represent the different levels of similarity by which clusters are formed. For each similarity level, a line perpendicular can be traced. Thus, each branch of the tree that is fragmented corresponds to a cluster. At the bottom level of similarity, all documents compose a single cluster.

Most hierarchic agglomerative algorithms run on the stored matrix approach, where the similarity matrix is built in secondary memory (Hartigan (1975)). Thus, a typical algorithm that works on N documents by using the stored matrix approach has storage requirements of $O(N^2)$(i.e., for the storage of the similarity matrix), meantime that time requirements accomplish the $O(N^3)$.

It is important to point out that despite the fact that the storage effi-

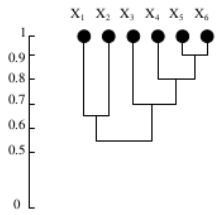

FIGURE 6.1: A similarity dendrogram

ciency of clustering methods is relevant, in this thesis, the storage efficiency is not a relevant factor. Although the efficiency of the various clustering methods is not of primary importance in this thesis, for reasons of completeness I will also refer to the efficiency of commonly used algorithms that implement the various methods. Readings that offer significant amount of detail on aspects of efficiency include Murtagh (1984); Willett (1988).

In the following paragraphs, I will present the group average link method, which is a hierarchic clustering method that have been used in this thesis.

6.5 Group average link

The similarity between two clusters in the group average link method corresponds to the mean of the similarities between all pairs of documents, such that one document of the pair is in one cluster and the other document in the other cluster.

As a result, to apply the group average link algorithm, clusters are formed on the basis of average similarities, and therefore it is not possible to infer something about the maximum or minimum similarities between documents in a cluster (Voorhees, 1985). As a consequence of vast range of comparative studies carried out by different researchers, Sneath and Sokal (1973) pointed out that the average link method is the most used hierarchic methods.

6.5.1 Query-document similarity measure

This section describes formally the new similarity measure proposed in the query-document clustering context, which is denominated QDSM (query-document similarity measure) and the clustering process.

Let q, d, q', D, Q, and Q' denote a past query, a document, a new query, a

corpus of documents, a finite set of past queries, and a finite set of new queries respectively.

Definition 1 *Let $V_N(q)$ be a set of N retrieved documents given q, using a measure as cosine between query q and documents.*

Definition 2 *Let $A(q)$ be the finite set of all relevant documents for the query q, such as $A(q) \subset V_N(q)$.*

Definition 3 *Let $A'(q)$ be the finite set all non-relevant documents for the query q,such as $A'(q) \subset V_N(q)$.*

Given a query q, such as $q \in Q$, then it is possible to denote $c(q) = q \cup V_N(q)$ as the set of all retrieved documents and the query q. Let $C = \bigcup c(q)$ be the set of all retrieved documents with their respective queries.

Definition 4 *Let $LCS(V_N(q), V_N(q')) \rightarrow \mathbb{N}$ be the function which is the result applying the Longest Common Subsequence algorithm between two lists of documents for the queries q and q'. It is applied considering just relevant documents.*

In Figure 6.2 a), relevant documents contain the value 1, in the format $d_{idDoc}(1)$, meanwhile irrelevant documents contain the value 0. The distance provided by $LCS(V_N(q), V_N(q'))$ corresponds to the match of relevant documents.

Definition 5 $QDSM(q, q') = \frac{sim(q,q') + LCS(V_N(q), V_N(q'))}{1 + Max(|A(q)|, |A(q')|}$ *corresponds to the Query-Document Similarity Measure, which is used for the query-document clustering.*

In Figure 6 b) $V_N(Q)$ corresponds to a centroid, thus every $V_N(q)$ corresponds to an object in the cluster. $QDSM(q, q')$ is the distance used to build the cluster. $sim(q, Q)$ corresponds to the distance between a new query and the centroid. Basically, query-document clustering is supported by two steps. First, the executed query is stored with its documents (see *Definition 1*). To build the query-document clusters using the stored queries, the QDSM distance is used (see *Definition 5*). In Figure 6.2 b), the centroid ($\overline{c_q}$) corresponds to $V_N(Q)$, meanwhile $V_N(Q')$, $V_N(Q'')$ and $V_N(Q''')$ are objects of the cluster. It is important to mention that every time that an object is added to some cluster, the centroid is updated. Finally, when the cluster is built and a new query q is provided, it is compared with every centroid. Thus, from the cluster with the most similar centroid (see *Figure 6.2, b*) is used to respond to the new query. Therefore, to built the list of documents for q using N-Nearest Neighbour, such as $N = 2$ (see *Figure 6.2, b)*), relevant documents from $V_N(Q')$ and $V_N(Q'')$ are used to answer the new query.

6.6 Experimental Environment

The experimental environment in this section, is composed basically of the well-known dataset "CACM". The main reason by which this dataset was used, is

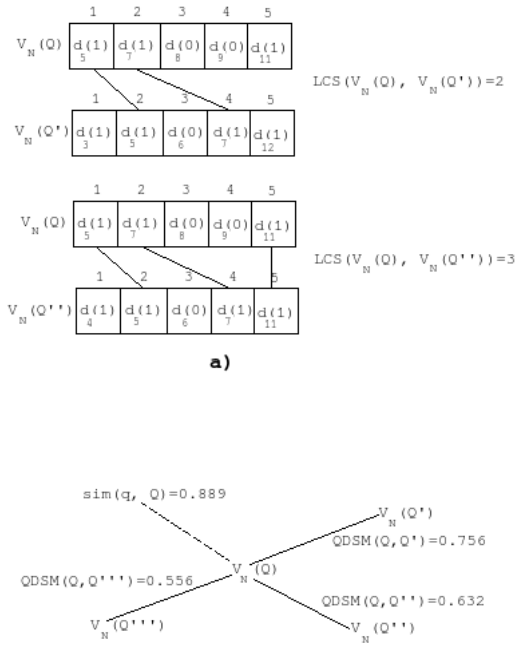

FIGURE 6.2: the distance used to build the cluster

because it is small and provides judgments of users. It is important to highlight that four datasets were evaluated "CISI", "LISA", "Medlards" and "CACM", but "CACM" has more queries than the previous. "CACM" contains 64 queries and 3204 documents. On the other hand, it is important to mention that datasets founded on log files have been omitted because the relevance of documents is complex to obtain. Several applications relying on log files provide their own definitions of precision. The main idea behind this contribution, is to have a general perspective with the aim to extend and analyze deeply once obtained the first experimental results. The clustering algorithm employed in this section is "Average Link". Average Link have been widely used in the IR domain. Furthermore, it provides good performance regarding effectiveness (Tombros and van Rijsbergen, 2004).

6.7 Experimental Results

In this section, an experiment has been carried out. The main idea was to build a cluster of queries along with their lists of documents taking into consideration the first ten documents. The number of terms used was 6,354 on the documents, where terms such as numbers 1,2, and so on were deleted, as well as the stop-words. Similarly, the number of terms used to build the cluster of queries was 491, the stop-words were eliminated. The stemming process was not carried out on the two clusters. Within the most similar pairs of queries using cosine, which have been obtained from the cluster of queries, correspond to the queries q_{19} and q_{63} with a similarity of 0.667. The first clusters have been considered because they are the most similar. Here, I show the first seven most similar clusters considering the cosine and QDSM distance. The final results are not always similar because as mentioned previously only were considered the first ten documents of lists of documents for each query. Besides, it is important to point out that the overlap among relevant documents for every pairs of queries is very small. Even in several cases the overlap was empty (see Table 6.1). Thus, this experiment is only a first approach, therefore, more experiments should be considered in order to obtain a more robust conclusion. Besides, it is not important to analyse the difference among the clusters with 3 or more queries, because almost all they tend to the cosine when the clusters are grouped.

From the Table 6.1, it is possible to see that the similarity is increased when the difference between $LCS(V(q), V(q'))$ and $max(|A(q)|, |A(q')|)$ is small, for example in (Q_{10}, Q_{19}) and (Q_{17}, Q_{13}). By contrast, if the difference between $LCS(V(q), V(q'))$ and $max(|A(q)|, |A(q')|)$ is greater, the similarity decreases (see (Q_{19}, Q_{63})). On the other hand, when the list does not have relevant documents and the overlap between the list is empty, the similarity is equal to the cosine (see (Q_{50}, Q_{51})).

In addition, it is important to mention that the centroid notion mentio-

ned previously was not used here, because "Average Link" is hierarchic clustering.

TABLE 6.1: The seven closest clusters (the similarities *Cosine* and *QDSM*) are compared. Overlap- relevant documents correspond to the same relevant documents for both queries, $LCS(V(q), V(q'))$ is the quantity of relevant documents for both queries taking into consideration the first ten documents and $max(|A(q)|, |A(q')|)$ is the list that contains most relevant documents.

	Approaches				
Pairs	Cosine	QDSM	Overlap-relevant documents	LCS(V(q), V(q'))	max(|A(q)|, |A(q')|)
(Q_{19}, Q_{63})	0.667	0.518	7	4	8
(Q_{50}, Q_{51})	0.544	0.544	0	0	0
(Q_{10}, Q_{19})	0.455	0.691	6	3	5
(Q_{28}, Q_{9})	0.446	0.089	0	0	5
(Q_{8}, Q_{9})	0.446	0.074	0	0	5
(Q_{17}, Q_{13})	0.424	0.774	7	5	7
(Q_{25}, Q_{52})	0.394	0.043	0	0	9

6.8 Conclusions

In this section, a review of the state-of-art of clustering in IR have been presented. The contribution of this section corresponds to the definition of a new QSSM measure. This measure is denominated "QDSM", which takes into consideration the relevant-recovered documents (i.e., documents from the list) for each query. Thus, two queries are similar considering not only the similarity between them, but also how many documents are similar in both lists. Final conclusions are difficult to obtain, because several experimental scenarios should be analysed. First of all, despite the fact that "CACM" contains 64 queries, the lists of documents for similar queries do not have an important quantity of relevant documents. Furthermore, in many cases the overlap between the lists of documents is empty. Therefore, it is necessary to look for other datasets, which have considerable overlap between the lists of documents. Another option is to increase the number of relevant documents, for example, using relevant terms from relevant documents. In any case, further experiments should be carried out.

Chapter 7

Conclusion

Résumé : conclusions

La littérature de RI est nourrie de différentes contributions telles que des approches d'indexation, des fonctions d'appariemment, des modèles formels et des approches de retour de pertinence. Cependant, peu d'approches visent à tirer un avantage des recherches effectuées dans le passé par d'autres utilisateurs. En outre, la plupart de ces approches sont liées à des moteurs de recherche dans le contexte du Web, basées sur des historiques de requêtes stockés dans des fichiers de logs. Néanmoins, la plupart de ces approches se concentrent sur les requêtes répétitives soumises par un même utilisateur. À notre connaissance, peu de travaux traitent de l'utilisation de requêtes passées similaires. Ainsi, cette recherche aborde les questions de l'exploitation des résultats de recherche passés avec l'objectif de construire les résultats retournés pour de nouvelles requêtes à l'aide des documents pertinents de requêtes passées similaires et en améliorer la précision. Pour atteindre cet objectif, cette thèse répond à trois problématiques :

— fournir un cadre pour simuler des collections dans le contexte de la réutilisation des recherches passées,
— fournir des algorithmes de type Monte Carlo pour sélectionner les documents de résultats de recherches passées suivant différentes stratégies,
— fournir une mesure de similarité (QDSM) pour une approche de clustering requête-documents fondée sur le stockage des requêtes passées.

1) Un cadre de simulation de collections de documents, requêtes et jugements des utilisateurs.

Dans cette thèse, un cadre pour simuler des collections de documents, requêtes et jugements des utilisateurs a été défini. Ce cadre fournit un environnement idéal pour évaluer les approches fondées sur les résultats de recherche passés.

Plusieurs scénarios expérimentaux peuvent être construits en utilisant ce cadre. En d'autres termes, il est possible de construire non seulement un nombre différent de documents, mais aussi un nombre différent de requêtes sous différentes distributions de probabilité. Les documents peuvent être construits en utilisant une distribution uniforme, la distribution exponentielle, et la distribution Zipf. Les jugements des utilisateurs sont eux construits en utilisant la distribution Zeta, qui représente une loi de Bradford. Par conséquent, l'interaction humaine n'est plus nécessaire pour déterminer la pertinence des documents pour une requête, ce qui implique une réduction de coût et de temps. De plus, la suppression des mots-vides, la racinisation et l'adaptation (i.e., modifier les caractéristiques de certains systèmes afin de l'adapter aux besoins expérimentaux) ne sont pas nécessaires. Ainsi, il est possible d'évaluer non seulement les approches fondées sur les résultats de recherches passées, mais aussi d'autres approches. Plusieurs scénarios expérimentaux ont été simulés sous différentes distributions de probabilité. L'efficacité de notre approche a été évaluée et comparée à une méthode de RI traditionnelle (cosinus). Le test pairé de Student a été appliqué pour confirmer les résultats expérimentaux. Les expérimentations présentent les meilleurs résultats pour notre approche par rapport à la recherche traditionnelle, pour les dix premiers documents récupérés (P@10). La prochaine étape naturelle à cette recherche est d'obtenir de nouvelles distributions pour la construction de documents, requêtes et jugements de l'utilisateur et d'ajouter ensuite de nouvelles fonctionnalités au cadre de simulation.

2) Un ensemble d'algorithmes de type Monte Carlo pour la réutilisation de résultats de recherches passées.

Dans cette thèse quatre algorithmes probabilistes ont été évalués au moyen de plusieurs scénarios expérimentaux qui ont été fournis par notre cadre de simulation. Ces algorithmes visent à réutiliser les documents pertinents extraits de la requête passée la plus similaire. Ces algorithmes ont comme avantages d'être faciles à mettre en œuvre et de ne pas nécessiter d'apprentissage. De plus, ces algorithmes peuvent être implémentés de manière interne à un SRI ou de manière externe. Le premier algorithme divise la liste de documents récupérés en groupes, où chaque groupe contient le même nombre de documents. Le second algorithme est inspiré par la distribution logistique. Le troisième algorithme divise la liste de documents récupérés en groupes de puissance deux, où chaque groupe contient un nombre différent de documents. Enfin, le quatrième algorithme assigne des probabilités pour chaque document en fonction de sa position dans la liste de documents récupérés. Ces algorithmes ont été comparés avec une méthode de RI traditionnelle (cosinus). Les résultats expérimentaux ont été validés en appliquant le test pairé de Student pour chaque expérience. Toutes les expériences présentent de meilleurs résultats pour nos algorithmes par rapport à la recherche traditionnelle. Ces expériences ont considéré les dix premiers documents récupérés (P@10). Il est important de souligner que les meilleurs résultats ont été fournis par le premier algorithme.

Enfin, dans les expériences menées, le nombre de requêtes et le nombre de documents n'ont pas d'influence sur les résultats finaux. Comme recherche future, il sera intéressant d'appliquer ces algorithmes avec des données réelles. En outre, la performance en temps n'a pas été étudiée, et devrait être comparée avec d'autres approches probabilistes telles que les algorithmes génétiques.

3) Une nouvelle mesure de similarité (QDSM) pour stocker des groupes de requêtes passées similaires avec leurs documents pertinents.

La principale contribution dans cette section était la définition d'une nouvelle mesure de similarité pour un clustering de requêtes avec leurs documents, baptisée QDSM (Query Document Similarity Measure). La méthode de clustering comprend deux étapes. Tout d'abord, les requêtes exécutées sont stockées avec leurs documents pertinents (et non pertinents). Pour construire les groupes de requêtes-documents à partir des requêtes stockées, la distance QDSM est utilisée. Cette mesure prend en compte les documents pertinents associés à leurs requêtes. L'algorithme de plus longue sous-séquence commune est appliqué entre deux listes de documents (correspondant à deux requêtes). Documents et requêtes ont été regroupés en utilisant le lien moyen. La collection Cranfield a été utilisée pour de premières expérimentations car elle fournit des jugements des utilisateurs. Bien que cet ensemble de données soit de taille limitée, cette étude permet de donner un éclairage sur les performances de la méthode proposée.

The literature of IR is crammed with different contributions such as indexing approaches, matching functions, formal models and relevance feedback approaches. However, few approaches gain advantage from searches performed in the past. Furthermore, most of these approaches are in the domain of web search engines, which are based on historical queries stored in log files. Nevertheless, most of these approaches concern repetitive queries. To the best of our knowledge, few works deal with the use of similar past queries. Thus, this research addressed the issues of exploiting past search results to improve precision for new queries using relevant documents from similar past queries. Achieving this goal, implied to cover three objectives :

— To provide a framework to simulate collections in the domain of past search results.
— To propose a set of Monte Carlo algorithms to improve precision.
— To give a similarity measure (QSSM) on query-document clustering, in order to store past queries in clusters.

In the following paragraphs, I list the contributions of this thesis. I first present the overall contribution that I believe has been achieved, when the work of this thesis is taken as a whole. I then list in more details some of the individual contributions.

1) A framework to simulate document collections, queries and judgments of users.

In chapter 3, an extensive bibliographic review have been presented about the approaches which deal with the use of past search results. Two categories of approaches based on past queries (past results) are identifiable -approaches rooted in user session and without user session. Approaches based on user sessions are related personalization and customization, however these approaches use either matching learning techniques or other techniques, which involve high resources of time. To sum up, the most relevant conclusions related to past queries based on user sessions, in particular query adaptation and adaptive retrieval require that users provide ad-hoc relevance judgments or keywords for the tasks of classifications.

A problem pointed out is the lack of suitable collections to evaluate systems based on past results. Most approaches are rooted in the use of historical queries on the Web, most of which are focused on repetitive queries. Therefore, these collections are not suitable to evaluate approaches based on past queries because the relevance judgments are not provided. A solution for the previous problem is to build suitable environments to analyze the advantages of approaches relying on past search results using simulation.

As a result of previous analysis, in this thesis a framework to simulate document collections, queries and judgments of users have been built. This

framework provides an ideal environment to evaluate approaches based on past search results. In section 4.3, the framework is described. Several experimental scenarios can be built by using this framework. In other words, it is possible to build not only different numbers of documents but also different numbers of queries under different probability distributions. Documents can be built by using uniform distribution, exponential distribution, and Zipf distribution. Judgments of users are built using Zeta distribution, which represents the Bradford's law. Therefore, human interaction is unnecessary to provide the relevance of documents given a query, which implies a reduction of cost and time. Moreover, stemming, stopping and tunning (to change characteristics of some particular system in order to adapt to the experimental needs) processes are not necessary. Thus, not only approaches founded on past search results can be evaluated, but also other approaches such as randomized algorithms. Hence, this framework serves as baseline to provide preliminary results, and simultaneously supplies a window for future researches.

In section 4.5, several experimental scenarios were simulated under different probability distributions. Aiming to build the documents, exponential distributions with parameters 1.0 and 1.5, was instantiated. Additionally, Zipf distribution was used to build the documents with parameter 1.6. The main reason was to choose terms from different topics (different fields, i.e, computer science, marketing and medicine among others), which compose a document. Otherwise, three parameters for Zeta distributions were used in the experiments ($S = 2, S = 3$ and $S = 4$), in order to cover a wide range of applications on these distributions. The effectiveness of our approach have been evaluated and compared with traditional retrieval (cosine). Experiments showed homogeneous effectiveness of the two tested approaches according to average P@10 over all the queries. Furthermore, performances differ from one simulated collection to another supporting the idea that the different collections could be used as different scenarios.

2) A set of Monte Carlo algorithms to improve precision.

A bibliographic review about the state-of-the-art of probabilistic approaches is presented in Chapter 5. On balance, the major part of probabilistic algorithms in IR can be categorized in two classes, learning techniques and optimization. Approaches based on learning techniques such as neural networks, genetic algorithms and support vector machines imply a high cost in learning time as well as diverse convergence times when the used datasets are heterogeneous.

In this thesis, four randomized algorithms have been evaluated. These algorithms aim at reusing relevant documents retrieved from the most similar past query. Two advantages are provided for these algorithms : first, these algorithms are easy to implement, and do not require time of learning as used in approaches relying on optimization techniques. Second, these algorithms can be implemented inside IRSs or search engines, or externally. In section 5.4.1, the

algorithms are described. Roughly speaking, the first algorithm splits the list of retrieved documents in groups, where each group contains the same quantity of documents. The second algorithm is inspired by the logistic distribution. The third algorithm divides the list of retrieved documents in groups of power two, where each group contains different numbers of documents. Finally, the fourth algorithm assigns probabilities for each document according to the position in the list of retrieved documents.

In section 5.5, the four randomized algorithms were evaluated using several experimental scenarios, which were provided by our framework. Similar to the experimental environments described in section 4.5, exponential distributions with parameters 1.0 and 1.5, are used to build the documents. In addition, Zipf distribution was used to build the documents with parameter 1.6. On the other hand, three parameters for Zeta distribution were used in the experiments ($S = 2, S = 3$ and $S = 4$), aiming to provide the user judgments. Our algorithms were compared with traditional retrieval (cosine). The experimental results were validated applying the Student's paired t-test in each experiment. All experiments present better results of our algorithms than the traditional retrieval. These experiments have considered the top ten retrieved documents (P@10). It is important to highlight that the best results were issued by Algorithm 1.

3) A new QSSM measure to store similar past queries along with their relevant documents.

A wide range of approaches deals with the use of clustering in IR. Several approaches have been presented in chapter 6. Clustering in IR have been used to improve efficiency and effectiveness of IRS. Overall, two major categories of clustering are easily identifiable : static clustering and post-retrieval clustering. On one hand, static clustering is the traditional application of the cluster method on a document collection. On the other hand, post-retrieval clustering includes information from the query into the clustering of documents. The main contribution in this section was a new definition of a similarity measure for query-document clustering, which is denominated query document similarity measure (QDSM). Basically, query-document clustering is supported by two steps. First, the executed query is stored with its documents (relevant and non-relevant). To build the query-document clusters using the stored queries, the QDSM distance is used. This measure takes in consideration the relevant documents associated with their queries. Th Longest Common Subsequence algorithm is applied between two lists of documents (for two queries) to obtain the (QDSM). Documents and queries were clustered using the average link. The main reason by which this algorithm was used, is because it provides good results, which are highly accepted by the IR community (Tombros et al., 2002). In section 6.8, one experiment was carried out. "CACM" dataset was used because it contains judgments of users. Despite the fact that this dataset is little, it was chosen because it has 64 queries. Final results are not enough

to obtain a conclusion about this measure. First of all, because the overlap among the lists of documents for similar queries is small. Even in several cases the overlap between them was empty. Furthermore, the quantity of relevant documents is small considering the quantity of documents. It is important to point out that this is only a first approach, therefore, many experiments should be carried out with the purpose to obtain conclusions on this measure.

7.1 Directions for Future Research

The research presented in this thesis addressed the issues of exploiting past search results, with the aim to improve precision for new similar queries. A framework to simulate collections was built. Four randomized algorithms were proposed and evaluated. Finally, a basic approach for a new QSSM measure was developed and evaluated, however, some issues remain open.

1) Extending the framework to incorporate new functionalities.

The current state of the design and implementation of our framework allows building several experimental scenarios. However, many other general and particular characteristics of IRSs can be incorporated and simulated. Moreover, some particular characteristics of search engines could be simulated too. Therefore, the natural next step in this research is to obtain new empirical distributions about building documents, queries, and judgments of user, to subsequently add new functionalities inside the framework.

2) A set of Monte Carlo algorithms to improve precision.

Nowadays, few randomized algorithms are used in the IR domain. In this particular case, randomized algorithms have been used to improve precision, however, these algorithms were not evaluated on real datasets. Furthermore, performance in time was not studied. In summary, several results should be evaluated on a traditional ad-hoc collection, besides the efficiency in time should be compared with other probabilistic approaches such as genetic algorithms.

3) Expanding evaluation of experimental scenarios for the new QSSM measure.

The IR literature is crammed with approaches that deal with the use of clustering. In this thesis, a basic approach for a new QSSM measure have been evaluated. Nevertheless, the results reported in this thesis are preliminary and therefore, these should be extended to other datasets, which contain a considerable overlap between the lists of documents for two similar queries. It should consider a considerable number of relevant documents for both lists

of documents. Furthermore, several trends could be studied, for instance, the performance in real time and web documents among others. However, it is important to point out that this is not easy, because most of the current datasets must be adapted in the domain of past search results.

References

Alexandrov, V., Dimov, T., Karaivanova, A., and Tan, C. (2003). Parallel monte carlo algorithms for information retrieval. *Mathematics and Computers in Simulation (MATCOM)*, 62(3) :289–295.

Alonso, O. and Mizzaro, S. (2012). Using crowdsourcing for trec relevance assessment. *Inf. Process. Manage.*, 48(6) :1053–1066.

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. Academic Press.

Azzopardi, L., Järvelin, K., Kamps, J., and Smucker, M. D. (2011). Report on the sigir 2010 workshop on the simulation of interaction. *SIGIR Forum*, 44(2) :35–47.

Baeza-Yates, R., Hurtado, C., and Mendoza, M. (2004). Query recommendation using query logs in search engines. In *Proceedings of the 2004 International Conference on Current Trends in Database Technology*, EDBT'04, pages 588–596, Berlin, Heidelberg. Springer-Verlag.

Baeza-Yates, R. A. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Baeza-Yates, R. A. and Saint-Jean, F. (2003). A three level search engine index based in query log distribution. In Nascimento, M. A., de Moura, E. S., and Oliveira, A. L., editors, *SPIRE*, volume 2857 of *Lecture Notes in Computer Science*, pages 56–65. Springer.

Banachowski, L., Kreczmar, A., and Rytter, W. (1991). *Analysis of Algorithms and Data and Structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Barry, C. L. (1994). User-defined relevance criteria : an exploratory study. *Journal of the American Society for Information Science*, 45 :149–159.

Bedi, P. and Chawla, S. (2010). Agent based information retrieval system using information scent. *Journal of Artifical Intelligence, Asian Network for Science Information*, 4(3) :220–238.

Beitzel, S. M., Jensen, E. C., Chowdhury, A., Grossman, D., and Frieder, O. (2004). Hourly analysis of a very large topically categorized web query log.

In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 321–328, New York, NY, USA. ACM.

Bharat, K. and Henzinger, M. R. (1998). Improved algorithms for topic distillation in a hyperlinked environment. In *In SIGIR Conference on Research and Development in Information Retrieval*.

Bigot, A., Chrisment, C., Dkaki, T., Hubert, G., and Mothe, J. (2011). Fusing different information retrieval systems according to query-topics : a study based on correlation in information retrieval systems and trec topics. *Inf. Retr.*, 14(6) :617–648.

Blei, D., Ng, A., and Jordan (2003). *Hierarchical Bayesian models for applications in information retrieval*, pages 25–44. Oxford University Press,.

Borlund, P. and Ingwersen, P. (1997). The development of a method for the evaluation of interactive information retrieval systems. *Journal of Documentation*, 53 :225–250.

Broder, A. (2002). A taxonomy of web search. *SIGIR Forum*, 36(2) :3–10.

Brown, E. W., Callan, J. P., Croft, W. B., Moss, J. E. B., Eliot, J., and Moss, B. (1994). Supporting full-text information retrieval with a persistent object store. In *In 4th Intl. Conf. on Extending Database Technology*, pages 365–378.

Cemgil, A. T. and Kappen, B. (2011). Monte carlo methods for tempo tracking and rhythm quantization. *CoRR*, abs/1106.4863.

Cetintas, S., Si, L., and Yuan, H. (2011). Using past queries for resource selection in distributed information retrieval. Technical Report 1743, Department of Computer Science, Purdue University.

Chen, Y.-S. and Leimkuhler, F. F. (1986). A relationship between Lotka's Law, Bradford's Law, and Zipf's Law. *Journal of The American Society for Information Science*, 37 :307–314.

Chowdhury, G. (2010). *Introduction to Modern Information Retrieval, Third Edition*. Facet Publishing, 3rd edition.

Cooper, M. D. (1973). A simulation model of an information retrieval system. *Information Storage and Retrieval*, 9(1) :13 – 32.

Cormen, T. H., Stein, C., Rivest, R. L., and Leiserson, C. E. (2001). *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition.

Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y. (2003). Query expansion by mining user logs. *IEEE Trans. on Knowl. and Data Eng.*, 15(4) :829–839.

de Campos, L. M., Fernández-Luna, J. M., and Huete, J. F. (2013). Query expansion in information retrieval systems using a bayesian network-based thesaurus. *CoRR*, abs/1301.7364.

Dou, Z., Song, R., and Wen, J.-R. (2007). A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 581–590, New York, NY, USA. ACM.

Fagni, T., Perego, R., and Silvestri, F. (2004). A highly scalable parallel caching system for web search engine results. In *Euro-Par 2004 Parallel Processing*, pages 347–354.

Fagni, T., Perego, R., Silvestri, F., and Orlando, S. (2006). Boosting the performance of web search engines : Caching and prefetching query results by exploiting historical usage data. *ACM Trans. Inf. Syst.*, 24(1) :51–78.

Fitzpatrick, L. and Dent, M. (1997). Automatic feedback using past queries : Social searching ? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 306–313, New York, NY, USA. ACM.

Fonseca, B. M., Golgher, P. B., de Moura, E. S., and Ziviani, N. (2003). Using association rules to discover search engines related queries. In *Proceedings of the First Conference on Latin American Web Congress*, LA-WEB '03, pages 66–, Washington, DC, USA. IEEE Computer Society.

Fu, L., Dion, D. H., and Schubert, S. S. (2004). The effect of similarity measures on the quality of query clusters. *Journal of Information Science*, 30(5) :396–407.

Fuhr, N. and Lalmas, M. (2007). Advances in xml retrieval : The inex initiative. In *Proceedings of the 2006 International Workshop on Research Issues in Digital Libraries*, IWRIDL '06, pages 16 :1–16 :6, New York, NY, USA. ACM.

Fuhr, N., Lalmas, M., Malik, S., and Szlávik, Z. (2005). *Advances in XML Information Retrieval : Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Dagstuhl Castle, ... 2004 (Lecture Notes in Computer Science)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Garfield, E. (1980). Bradford's Law and Related Statistical Patterns. *Essays of an Information Scientist*, 4(19) :476–483.

Gonnet, G. (1984). *Handbook of Algorithms and Data Structures*. International Computer Science Series. Addison-Wesley Publishing Company.

Gordon, A. D. (1987). A Review of Hierarchical Classification. *Journal of the Royal Statistical Society. Series A (General)*, 150(2) :119–137.

Gray, P. and Watson, H. J. (1998). Present and future directions in data warehousing. *SIGMIS Database*, 29(3) :83–90.

Gutiérrez-Soto, C. (2014a). Simulation, randomized and clustering algorithms for information retrieval. In *Proceedings of the Doctoral Consortium (RR*

2014), The 8th International Conference On Web Reasoning And Rule Systems, Athens, Greece.

Gutiérrez-Soto, C. (2014b). Taking advantage from past search results through a probabilistic algorithm. In *The 3rd Spanish Conference on Information Retrieval (CERI 2014), A Coruña, Spain*, pages 1–9.

Gutiérrez-Soto, C. and Hubert, G. (2013). Evaluating the interest of revamping past search results. In Decker, H., Lhotská, L., Link, S., Basl, J., and Tjoa, A., editors, *Database and Expert Systems Applications*, volume 8056 of *Lecture Notes in Computer Science*, pages 73–80. Springer Berlin Heidelberg.

Gutiérrez-Soto, C. and Hubert, G. (2014). Probabilistic reuse of past search results. In *Database and Expert Systems Applications - 25th International Conference, DEXA 2014, Munich, Germany, September 1-4, 2014. Proceedings, Part I*, pages 265–274.

Gutiérrez-Soto, C. and Hubert, G. (2014). Randomized algorithm for information retrieval using past search results. In *Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on*, pages 1–9.

Gutiérrez-Soto, C. and Hubert, G. (2015). On the reuse of past searchesin information retrieval :study of two probabilistic algorithms. *International Journal of Information System Modeling and Design*, 6(2) :71–90.

Harman, D. (1993). Overview of the first trec conference. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 36–47, New York, NY, USA. ACM.

Hartigan, J. A. (1975). *Clustering Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 99th edition.

Hasanzadeh, S. and Keshavarzi, A. (2009). Application of query sensitive similarity measure in ir systems. In *Proceedings of the 2009 Third Asia International Conference on Modelling and Simulation*, AMS '09, pages 73–78, Washington, DC, USA. IEEE Computer Society.

Heaps, H. S. (1978). *Information Retrieval : Computational and Theoretical Aspects*. Academic Press, Inc., Orlando, FL, USA.

Hiemstra, D. (1998). A Linguistically Motivated Probabilistic Model of Information Retrieval. In *ECDL '98 : Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 569–584, London, UK. Springer-Verlag.

Hust, A. (2004). Introducing Query Expansion Methods for Collaborative Information Retrieval. In Dengel, A., Junker, M., and Weisbecker, A., editors, *Reading and Learning - Adaptive Content Recognition*, volume 2956 of *Lecture Notes in Computer Science*, pages 252–280, Berlin, Heidelberg, New York. Springer-Verlag.

Huurnink, B., Hofmann, K., De Rijke, M., and Bron, M. (2010). Validating query simulators : an experiment using commercial searches and purchases. In *Proceedings of the 2010 international conference on Multilingual and multimodal information access evaluation : cross-language evaluation forum*, CLEF'10, pages 40–51, Berlin, Heidelberg. Springer-Verlag.

Indrawan, M., Ghazfan, D., and Srinivasan, B. (1996). Using bayesian networks as retrieval engines. In *TREC*.

Jansen, B. J., Spink, A., and Saracevic, T. (2000). Real life, real users, and real needs : A study and analysis of user queries on the web. *Inf. Process. Manage.*, 36(2) :207–227.

Jardine, N. and Sibson, R. (1968). The construction of hierarchic and non-hierarchic classifications. *Computer Journal*, (11) :184.

Joachims, T. (1997). A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 143–151, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28 :11–21.

Jónsson, B. T., Franklin, M. J., and Srivastava, D. (1998). Interaction of query evaluation and buffer management for information retrieval. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, pages 118–129, New York, NY, USA. ACM.

Kearns, M. J. (1989). *The Computational Complexity of Machine Learning*. PhD thesis, Harvard University, USA, Cambridge, MA, USA. UMI Order No : GAX89-26128.

Keen, E. M. (1992). Presenting results of experimental retrieval comparisons. *Inf. Process. Manage.*, 28(4) :491–502.

Kelly, D. (2009). Methods for Evaluating Interactive Information Retrieval Systems with Users. *Foundations and Trends in Information Retrieval*, 3 :1–224.

Kleinberg, J. and Tardos, E. (2005). *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5) :604–632.

Lempel, R. and Moran, S. (2003). Predictive caching and prefetching of query results in search engines. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 19–28, New York, NY, USA. ACM.

Lewis, D. D. and Jones, K. S. (1996). Natural language processing for information retrieval. *Commun. ACM*, 39(1) :92–101.

Lillis, D., Toolan, F., Mur, A., Peng, L., Collier, R., and Dunnion, J. (2006). Probability-based fusion of information retrieval result sets. *Artif. Intell. Rev.*, 25(1-2) :179–191.

Liu, X. and Croft, W. B. (2002). Passage retrieval based on language models. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, pages 375–382, New York, NY, USA. ACM.

Luo, Q., Naughton, J. F., Krishnamurthy, R., Cao, P., and Li, Y. (2000). Active query caching for database web servers. In *WebDB*, pages 29–34. Springer.

Markatos, E. (2001). A linguistically motivated probabilistic model of information retrieval. *Comput. Commun.*, 24(2) :137–143.

Moreno, J. G., Dias, G., and Cleuziou, G. (2013). Post-retrieval clustering using third-order similarity measures. In *ACL (2)*, pages 153–158. The Association for Computer Linguistics.

Murtagh, F. (1983). A survey of recent advances in hierarchical clustering algorithms. *Computer Journal*, 26(4) :354–359.

Murtagh, F. (1984). Complexities of hierarchic clustering algorithms : state of the art. *Computational Statistics Quarterly*, (1) :101–113.

Na, S.-H. (2013). Probabilistic co-relevance for query-sensitive similarity measurement in information retrieval. *Inf. Process. Manage.*, 49(2) :558–575.

Navarro, G., De Moura, E. S., Neubert, M., Ziviani, N., and Baeza-Yates, R. (2000). Adding compression to block addressing inverted indexes. *Inf. Retr.*, 3(1) :49–77.

Niederberger, T., Stoop, N., Christen, M., and Ott, T. (2012). Hebbian Principal Component Clustering for Information Retrieval on a Crowdsourcing Platform. *Nonlinear Dynamics of Electronic Systems, Proceedings of NDES 2012*, pages 1–4.

Nopiah, Z. M., Khairir, M. I., Abdullah, S., Baharin, M. N., and Arifin, A. (2010). Time complexity analysis of the genetic algorithm clustering method. In *Proceedings of the 9th WSEAS International Conference on Signal Processing, Robotics and Automation*, ISPRA'10, pages 171–176, Stevens Point, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS).

Ouksel, A. (2002). Mining the world wide web : An information search approach by george chang, marcus j. healey (editor), james a. m. mchugh, jason t. l. wang. *SIGMOD Rec.*, 31(2) :69–70.

Poosala, V. (1997). Zipf's law. Technical Report 900 839 0750, Bell Laboratories.

Puolamäki, K., Salojärvi, J., Savia, E., Simola, J., and Kaski, S. (2005). Combining eye movements and collaborative filtering for proactive information re-

trieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 146–153, New York, NY, USA. ACM.

Raghavan, V. V. and Sever, H. (1995). On the reuse of past optimal queries. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 344–350, New York, NY, USA. ACM.

Reid, J. (2000). A task-oriented non-interactive evaluation methodology for. *Information Retrieval Systems, Information Retrieval*, 2 :115–129.

Rijsbergen, C. J. V. (1979). *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition.

Robertson, S. E., van Rijsbergen, C. J., and Porter, M. F. (1981). Probabilistic models of indexing and searching. In *Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval*, SIGIR '80, pages 35–56, Kent, UK, UK. Butterworth & Co.

Roitman, H., Hummel, S., and Kurland, O. (2014). Using the cross-entropy method to re-rank search results. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 839–842, New York, NY, USA. ACM.

Salton, G. (1971). *The SMART Retrieval System and ;Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Salton, G., Allan, J., and Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 49–58, New York, NY, USA. ACM.

Salton, G. and Buckley, C. (1987). Term weighting approaches in automatic text retrieval. Technical report, Cornell University, Ithaca, NY, USA.

Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill computer science series. McGraw-Hill International.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Salton, G. and Wong, A. (1978). Generation and search of clustered files. *ACM Trans. Database Syst.*, 3(4) :321–346.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11) :613–620.

Sanderson, M. (2010). Test Collection Based Evaluation of Information Retrieval Systems. *Foundations and Trends in Information Retrieval*, 4(4) :247–375.

Saraiva, P. C., Silva de Moura, E., Ziviani, N., Meira, W., Fonseca, R., and Riberio-Neto, B. (2001). Rank-preserving two-level caching for scalable search

engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 51–58, New York, NY, USA. ACM.

Saravanakumar, K. and Moturi, M. (2011). Article : Semantic based personalized framework for information retrieval. *International Journal of Computer Applications*, 20(4) :14–17. Full text available.

Schaer, P. (2013). Applied informetrics for digital libraries : an overview of foundations, problems and current approaches. *Historical Social Research*, 38(3) :267–281.

Schamber, L., Eisenberg, M., and Nilan, M. S. (1990). A re-examination of relevance : Toward a dynamic, situational definition. *Inf. Process. Manage.*, 26(6) :755–776.

Shen, X., Tan, B., and Zhai, C. (2005). Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 43–50, New York, NY, USA. ACM.

Silva de Moura, E., Navarro, G., Ziviani, N., and Baeza-Yates, R. (2000). Fast and flexible word searching on compressed text. *ACM Trans. Inf. Syst.*, 18(2) :113–139.

Sneath, P. and Sokal, R. (1973). *Numerical Taxonomy : The Principles and Practice of Numerical Classification*. A Series of books in biology. W. H. Freeman.

Song, S.-K. and Myaeng, S. H. (2012). A novel term weighting scheme based on discrimination power obtained from past retrieval results. *Inf. Process. Manage.*, 48(5) :919–930.

Sparck Jones, K. and Willett, P., editors (1997a). *Readings in Information Retrieval*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Sparck Jones, K. and Willett, P., editors (1997b). *Readings in Information Retrieval*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Steichen, B., Ashman, H., and Wade, V. (2012). A comparative survey of personalised information retrieval and adaptive hypermedia techniques. *Inf. Process. Manage.*, 48(4) :698–724.

Tague, J. M. and Nelson, M. J. (1981). Simulation of user judgments in bibliographic retrieval systems. In *Proceedings of the 4th annual international ACM SIGIR conference on Information storage and retrieval : theoretical issues in information retrieval*, SIGIR '81, pages 66–71, New York, NY, USA. ACM.

Teevan, J., Adar, E., Jones, R., and Potts, M. A. S. (2007). Information re-retrieval : repeat queries in yahoo's logs. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 151–158, New York, NY, USA. ACM.

Theodoridis, S. and Koutroumbas, K. (1999). *Pattern Recognition*. Academic Press, San Diego, CA, USA.

Tombros, A. and van Rijsbergen, C. J. (2004). Query-sensitive similarity measures for information retrieval. *Knowl. Inf. Syst.*, 6(5) :617–642.

Tombros, A., Villa, R., and Rijsbergen, C. J. V. (2002). The effectiveness of query-specific hierarchic clustering. In *in information retrieval. Information Processing and Management*, pages 559–582.

Van Rijsbergen, C. J. (1986). A new theoretical framework for information retrieval. *SIGIR Forum*, 21(1-2) :23–29.

Voorhees, E. M. (1985). *The effectiveness and efficiency of agglomerative hierarchical clustering in document retrieval*. PhD thesis, Cornell University.

Voorhees, E. M. and Harman, D. K. (2005). *TREC : Experiment and Evaluation in Information Retrieval*. MIT Press, Cambridge, MA, USA.

Wei, F., Li, W., Lu, Q., and He, Y. (2008). A cluster-sensitive graph model for query-oriented multi-document summarization. In Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., and White, R. W., editors, *ECIR*, volume 4956 of *Lecture Notes in Computer Science*, pages 446–453. Springer.

White, R. W., Ruthven, I., Jose, J. M., and Van Rijsbergen, C. J. (2005). Evaluating implicit feedback models using searcher simulations. *ACM Trans. Inf. Syst.*, 23(3) :325–361.

Willett, P. (1988). Recent trends in hierarchic document clustering : A critical review. *Inf. Process. Manage.*, 24(5) :577–597.

Xie, Y. and O'Hallaron, D. (2002). Locality in search engine queries and its implications for caching. In *In IEEE Infocom 2002*, pages 1238–1247.

Yue, Y. and Joachims, T. (2009). Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1201–1208, New York, NY, USA. ACM.

Zerchaninova, I. L. (2008). Bradford's law of scattering for climate-friendly technologies and metainformational effect of time. Technical report, Institute for Time Nature Explorations.

Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort*. Addison-Wesley (Reading MA).

Résumé

La recherche d'informations (RI) concerne l'obtention d'éléments (habituellement documents) de nature non structurée (habituellement du texte) qui satisfait un besoin d'information, dans de grandes collections de documents. Un système de recherche d'information (SRI) a pour objectif de représenter et de stocker de grandes quantités d'informations, pour faciliter et accélérer l'identification des informations pertinentes estimées pour une requête de l'utilisateur. Les deux processus principaux mis d'un SRI sont l'indexation et l'appariement. Le processus d'indexation vise à représenter les représentations des documents, de manière efficace non seulement pour le stockage, mais aussi pour l'accès. Le processus d'appariement vise à estimer si un document est pertinent pour une requête exprimée par un utilisateur. Cette mise en correspondance est généralement représentée par un score. Lorsque le processus est appliqué, un ensemble de documents est retourné à l'utilisateur sous forme de liste classée par score décroissant. Bien que les systèmes de RI aient émergé dans les années 1940, des améliorations sont vraiment apparues dès la fin des années 1950. Les améliorations en RI les plus importantes sont liées à l'évaluation des SRI. La communauté RI a bénéficié notamment de collections d'évaluation, notamment au travers de l'initiative TREC, qui organise chaque année un atelier. Ces ateliers ont offert aux chercheurs la possibilité de mesurer l'efficacité de leur système et de comparer les approches.

De nombreuses approches en RI traitant de l'indexation, de fonctions d'appariement, de modèles formels, et de retour de pertinence ont été proposées. Cependant, peu d'approches tirent avantage des recherches effectuées précédemment par d'autres utilisateurs. Les recherches passées constituent pourtant une source d'information utile pour les nouveaux utilisateurs (nouvelles requêtes). Par exemple, un utilisateur intéressé par un nouveau sujet pourrait bénéficier des recherches antérieures menées par les utilisateurs précédents intéressés par le même sujet. En raison de l'absence de collections ad-hoc de RI, à ce jour il y a un faible intérêt de la communauté RI autour de l'utilisation des recherches passées. En effet, la plupart des collections de RI existantes sont composées de requêtes indépendantes. Ces collections ne sont pas appropriées pour évaluer les approches fondées sur les requêtes passées parce qu'elles ne comportent pas de requêtes similaires ou qu'elles ne fournissent pas de jugements de pertinence.

Par conséquent, il n'est pas facile d'évaluer ce type d'approches. En outre, l'élaboration de ces collections est difficile en raison du coût et du temps élevés nécessaires. Une alternative consiste à simuler les collections. Par ailleurs, les documents pertinents de requêtes passées similaires peuvent être utilisées pour répondre à une nouvelle requête. De nombreuses contributions ont été proposées portant sur l'utilisation de techniques probabilistes pour améliorer les résultats de recherche. Des solutions simples à mettre en oeuvre pour la réutilisation de résultats de recherches peuvent être proposées au travers d'algorithmes probabilistes. De plus, ce principe peut également bénéficier d'un clustering des recherches antérieures selon leurs similarités. Ainsi, dans cette thèse un cadre pour simuler des collections pour des approches basées sur les résultats de recherche passées est mis en oeuvre et évalué. Quatre algorithmes probabilistes pour la réutilisation des résultats de recherches passées sont ensuite proposés et évalués. Enfin, une nouvelle mesure dans un contexte de clustering est proposée.