Deanship of Graduate Studies

Al-Quds University

New IR & Ranking Algorithm for Top-K Keyword Search on Relational Databases
'Smart Search'

Ala' Juma' Sulaiman Eghrayeb

M.Sc. Thesis

Jerusalem – Palestine

1435 AH / 2014 AD

New IR & Ranking Algorithm for Top-K Keyword Search on Relational Databases
'Smart Search'

Prepared By:
Ala' Juma' Sulaiman Eghrayeb

B.Sc Computer Science- Al-najah University- Palestine

Supervisor: Dr. Badie Sartawi

A thesis Submitted in Partial fulfillment of requirements for the Master's Degree of Computer Science / Department of Computer Science / Faculty of Graduate Studies – Al-Quds University.

Jerusalem – Palestine

1435 AH / 2014 AD

Al Quds University
Dean of Graduate Studies
Master of Computer Science

Thesis Approval

New IR & Ranking Algorithm for Top-K Keyword Search on Relational Databases ‘Smart Search’

Prepared By: Ala’ Juma’ Suliman Eghrayeb

Registration No: (20810084)

Supervisor: Dr. Badie Sartawi

Master thesis submitted and accepted 12/11/2014

The names and signatures of the examining committee members are as follows:

1-Head of Committee: Dr. Badie Sartawi Signature ……

2-Internal Examiner: Dr. Rashid Jayousi Signature ……

3-External Examiner: Dr. Mahmoud Al-Saheb Signature ……

Jerusalem – Palestine

1435 AH / 2014 AD

Dedication

I dedicate this thesis

To my wife and children for their support all the past years.

To my mother for her standing and love.

To my doctors at Al-Quds University for their supervision and support.

To my friends and colleagues.

To my country Palestine, May we be free with our own state.

Ala' Juma' Sulaiman Eghrayeb

Declaration

I certify that this thesis submitted for the degree of Master of Computer Science in Al Quds University, is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of it) has not been submitted for a higher degree to any other university or institution.

Signed …….

Ala' Juma' Sulaiman Eghrayeb

Date: 12/ 11 /2014

Acknowledgements

First, My thanks for God for his many graces and giving me the health and ability to finish my master degree and for this thesis.

I gracefully thank **Al Quds university** for with the great teachers and facilities that enabled me to accomplish my research.

I would also like to extend thanks to Dr. Badie Sartawi for his continuous support and guidance. His efforts and encouragement where the main reason for me to finish this work.

My endless thanks go to my **wife** for her support and standing beside me throughout all the years that past and to come.

Last, many thanks go to my **mother** for here strong encouragement and prayers, for her many sacrifices.

Abbreviations

Abbreviations	Full word
SSearch	'Smart Search', web interface developed in this thesis for the purpose of testing.
CN	Candidate Network.
DBLP	Digital Bibliography Library Publications.
SQL	Structured Query Language.
IR	Information Retrieval.
DBMS	Database Management System.
JTT	Joint Tuples Tree.
B-tree	Binary Tree.
STree	Steiner Tree.
MPWST	Minimum Path Weight Steiner Tree.
SG	Schema Graph.
CSTree	Compact Steiner Tree.
IMDB	Internet Movie Database.
TF-IDF	Term Frequency–Inverse Document Frequency.
XML	Extensible Markup Language.
RDBMS	Relational Database Management System.
TREC	Text REtrival Conference

New IR & Ranking Algorithm for Top-K Keyword Search on Relational Databases, 'Smart Search'

Prepared By: Ala' Juma' Sulaiman Eghrayeb

Al-Quds University MSc. Program in Computer Science

Supervisor: Dr. Badie Sartawi

Abstract

Database management systems are as old as computers, and the continuous research and development in databases is huge and an interest of many database venders and researchers, as many researchers work in solving and developing new modules and frameworks for more efficient and effective information retrieval based on free form search by users with no knowledge of the structure of the database. Our work as an extension to previous works, introduces new algorithms and components to existing databases to enable the user to search for keywords with high performance and effective top-k results.

Work intervention aims at introducing new table structure for indexing of keywords, which would help algorithms to understand the semantics of keywords and generate only the correct CN's (Candidate Networks) for fast retrieval of information with ranking of results according to user's history, semantics of keywords, distance between keywords and match of keywords. In which a three modules where developed for this purpose.

We implemented our three proposed modules and created the necessary tables, with the development of a web search interface called 'Smart Search' to test our work with different users.

The interface records all user interaction with our ‘Smart Search’ for analyses, as the analyses of results shows improvements in performance and effective results returned to the user.

We conducted hundreds of randomly generated search terms with different sizes and multiple users; all results recorded and analyzed by the system were based on different factors and parameters. We also compared our results with previous work done by other researchers on the DBLP database which we used in our research.

Our final result analysis shows the importance of introducing new components to the database for top-k keywords search and the performance of our proposed system with high effective results.

تطوير خورزميات للحصول على أفضل ترتيب في استرجاع المعلومات من قواعد البيانات العلائقية وإضافة طرق جديدة لإسترجاع البيانات، "البحث الذكي".

برنامج ماجستير علم الحاسوب – جامعة القدس.

إعداد: علاء جمعه سليمان اغريب.

إشراف: د. بديع سرطاوي.

ملخص

نظم إدارة قواعد البيانات قديمة مثل أجهزة الكمبيوتر، والبحث والتطوير المستمر في قواعد بيانات ضخم وهنالك اهتمام من العديد من مطوري قواعد البيانات والباحثين، كما يعمل العديد من الباحثين في حل وتطوير وحدات جديدة وأطر لاسترجاع المعلومات بطرق أكثر كفاءة وفعالية على أساس نموذج البحث الغير مقيد من قبل المستخدمين الذين ليس لديهم معرفة في بنية قاعدة البيانات. ويأتي عملنا امتدادا للأعمال السابقة، ويدخل الخوارزميات ومكونات جديدة لقواعد البيانات الموجودة لتمكين المستخدم من البحث عن الكلمات المفتاحية (Keyword search) مع الأداء العالي ونتائج فعالة في الحصول على أعلى ترتيب للبيانات (Top–K).

ويهدف هذا العمل إلى تقديم بنية جديدة لفهرسة الكلمات المفتاحية (Index Keywords Table)، والتي من شأنها أن تساعد الخوارزميات المقدمة في هذا البحث لفهم معاني الكلمات المفتاحية المدخلة من قبل المستخدم وتوليد فقط الشبكات المرشحة (CN's) الصحيحة لاسترجاع سريع للمعلومات مع ترتيب النتائج وفقا لأوزان

مختلفة مثل تاريخ البحث للمستخدم، ترتيب الكمات المفتاحية في النتائج والبعد بين الكلمات المفتاحية في النتائج بالنسبة لما قام المستخدم بأدخاله.

قمنا بأقتراح ثلاث مكونات جديدة (Modules) وتنفيذها من خلال هذه الاطروحة، مع تطوير واجهة البحث على شبكة الإنترنت تسمى "البحث الذكي" لاختبار عملنا مع المستخدمين. وتتضمن واجهة البحث مكونات تسجل تفاعل المستخدمين وتجميع تلك التفاعلات للتحليل والمقارنة، وتحليلات النتائج تظهر تحسينات في أداء استرجاع البينات والنتائج ذات صلة ودقة أعلى.

أجرينا مئات عمليات البحث بأستخدام جمل بحث تم أنشائها بشكل عشوائي من مختلف الأحجام، بالاضافة الى الاستعانة بعدد من المستخدمين لهذه الغاية. واستندت جميع النتائج المسجلة وتحليلها بواسطة واجهة البحث على عوامل ومعايير مختلفة .وقمنا بالنهاية بعمل مقارنة لنتائجنا مع الاعمال السابقة التي قام بها باحثون آخرون على نفس قاعدة البيانات (DBLP) الشهيرة التي استخدمناها في أطروحتنا.

وتظهر نتائجنا النهائية مدى أهمية أدخال بنية جديدة لفهرسة الكلمات المفتاحية الى قواعد البيانات العلائقية، وبناء خوارزميات استنادا الى تلك الفهرسة للبحث بأستخدام كلمات مفتاحية فقط والحصول على نتائج أفضل ودقة أعلى، أضافة الى التحسن في وقت البحث.

Table of Contents

List of Figures

List of Tables

List of Appendices

CHAPTER ONE: INTRODUCTION

1.1 Introduction

The world of information is now homogenous and heterogeneous, structured and semi structured databases of large number of tuples in rational representation, and millions of users interact with databases every day to retrieve information, most of them are not familiar with the databases or the structure of the data, not even the relations between tables, which is a main challenge for computer researcher to present new frameworks to overcome the limitations of current databases information retrieval. Some of the main providers of databases management systems start to invoke some aspects of IR (Information Retrieval) and ranking methods, but the main effort nowadays are the introduction of new modules, algorithms and ranking styles that are efficient and effective [2,5,7,10,15,16].

Most major commercial DBMS allow a full text search engine to be invoked while processing SQL (that is extended by specialized predicates). However, such engines cannot by themselves identify matching rows that result from joining multiple stored tables' on-the-fly[1], the existing DBMS technologies have not been designed with supporting keyword search in mind; and the keyword search methods recently proposed by the database community are also largely independent of the underlying DBMS[5], so it is just a search for keywords in text, without the capabilities of joining tuples or relations or the ranking of results, and existing IR-style ranking strategies cannot be applied directly in databases[2], which rises the need for a module not part of the DBMS and can work on any database after identifying the relations and schema of the database.

Database vendors also have evaluated their full-text search capabilities by participating in Text REtrival Conference (TREC) competitions [18], by comparing their integrated text search paradigm to Lucene, which is a powerful, free, open source IR library written entirely in Java. It is suitable for nearly any application that requires full-text search and its popularity is increasing because of simplicity, high performance, maturity and scalability [18].

For naïve users the process of retrieving related information to their search query sets are with no concern to them, as they only need to know how to write the words that they know related to information stored in databases within relations and tuples under different relations and connections, which raise the research for frameworks and algorithms with simple interface to interact with users and connect to different databases to retrieve information represented in different ways to the users in the most efficient and effective way.

By the beginning of this century, many researchers presented algorithms and systems to satisfy the effectiveness and efficiency factors for retrieving information from relational databases, and in this thesis we will show some of the most important researches in the domain of keyword search over relational databases with respect to performance and effectiveness.

Effectiveness and efficiency are the major factors for any keyword search framework, as researchers during the past decade has largely focused on performance [4], which will be the main focus of our work with introducing a new structure of index keywords table and a new ranking algorithm, which would have a great effect on the IR style and ranking of results.

Ranking is a major part of IR due to the fuzzy nature of keyword queries, results ranking is vital for retrieval effectiveness. [2].

In this thesis, we introduce new candidate network generator algorithm based on the new introduced index keywords table structure and information table for a given databases, also we develop our own ranking function according to different factors. The work include extensive experiments with real users over large database with millions of tuples, also the recording of test results with analyses to compare our work with others.

1.1.1 Relational Database

The concept of relational database was developed in 1969 by Edgar Codd and defined the databases as a structured collection of information that relates to a special domain, subject or purpose.

Any given relational database is represented in containers called tables with relations between tables through fields inside the tables, and each table contains rows that represent a tuple and a column that represents a field of information.

In any given table, we can find different information types contained in one tuple as a row with unique identifier for each row (tuple), and the column represents the same type of information with different input data for different tuples. For example, the Digital Bibliography Library Publications (DBLP2) database contains different tables, and each table contains different tuples, and each tuple represents information related to one unique identifier, like the author table which contains an id for the author (the primary key) and the author name.

To be able to link different information tuples for different table, relations were created between tables with identifiers in each table, called primary key in the source table and foreign key in the destination table, and databases now can identify different types of relations. For example, each student in a university has a relation to courses as the student's information stored in one table and the courses stored in another table.

With time, the relational databases were developed by vendors and researchers to meet the needs of representing information in a structured way, with an easy way to retrieve information or manage huge information in one container.

The progress in developing databases and databases management system was and still going in a fast track, as the need of users and various groups of users grow bigger and bigger, which leads to improvement of the database management systems in performance and effectiveness, with the

generating of various types of tools and algorithms for retrieving information from any given database. This leads to the existence of Structured Query language (SQL) as a main language for managing databases and information retrieval (IR).

1.1.2 Structured Query Language (SQL)

The universal language is for talking to any database regardless of its structure or content, it's a communicating way to retrieve, update, add, and manage records in databases with predefined functions and procedures.

The use of SQL language needs expert users and full knowledge of the database structure, as the SQL statements needs information from the schema of the database to be able to build correct syntax of SQL statements.

The SQL language was accepted as a standard International Organization for Standards (ISO) in 1987, and was developed by various vendors and open source community to become a part of any database management system.

Some of the Most Important SQL Commands:

- SELECT - extracts data from a database
- UPDATE - updates data in a database
- DELETE - deletes data from a database
- INSERT INTO - inserts new data into a database
- CREATE DATABASE - creates a new database
- ALTER DATABASE - modifies a database
- CREATE TABLE - creates a new table
- ALTER TABLE - modifies a table

- DROP TABLE - deletes a table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

For more information and examples about the SQL language, go to
http://www.w3schools.com/sql/default.asp

1.1.3 DBLP2 database Structure and Records

For the purpose of testing the new introduced components and algorithms in Top-K keyword search over relational databases, a well-known database DBLP adapted to be used in all stages of the work done, the database as an open source database is available for researchers through the website (http://dblp.uni-trier.de/db/) in XML format, which can be converted to any known DBMS to work with.

In our thesis, we converted the XML data into MySQL database for more powerful control and management, as the new version of MySQL (Ver. 5.5) provides the user with powerful SQL statements and indexing of fields, also introduced new commands in searching through SQL statements with high performance or retrieving data and the ability to maintain databases with millions of records.

Researchers in [7, 20, 21, 10, 11, 17, 19] and many others used the DBLP database for implementing algorithms and testing their work, the only difference with other researchers is the size of the database used and DBMS engine.

The database used in our research is the research database (DBLP), which is an open source database for researcher and has the following information:

Table 1.1.3.1: DPLB2 database information.

Table Name	Number of records	Representation Symbol
books	1,208	B
journals	547	J
msthesises	7	M
papers	728,509	PA
persons	456,764	PE
phdthesises	85	PH
proceedings	7,482	PR
raw_links	2,507,780	RA
wwws	5,510	W
9 tables	3,707,892	

The raw_links table represents the relation table that stores all relations between other tables and type of relation.

Example: Person X: is presented in Raw_links table by his ID, and linked to different other ID's which represents papers, book, proceedings, journals … etc.

The structure and data sample for each table in the DBLP database found in Appendix (D). With indication to the **searchable fields** in each table, which contains all the information in the field of computer science research papers, researchers, journals, proceedings, journals … etc, which is dated early 1960 until the year 2012, and still growing.

1.1.4 DBLP2 database Schema and CN's.

The following table shows all possible CN's stored in the information table about the database DBLP2, which is derived from the schema of the database.

Table 1.1.4.1: All CN's represents the DBLP2 database.

No.	CN	Relation
1-	CN1	B
2-	CN2	J
3-	CN3	M
4-	CN4	PA
5-	CN5	PE
6-	CN6	PH
7-	CN7	PR
8-	CN8	W
9-	CN9	B →PA
10-	CN10	J → PR
11-	CN11	M →J
12-	CN12	PA →PR
13-	CN13	PE →PA
14-	CN14	PH →PR
15-	CN15	PR →B
16-	CN16	W →PR
17-	CN17	PE → J
18-	CN18	PE →PR
19-	CN19	PE →B
20-	CN20	PE →M
21-	CN21	PE →PH
22-	CN22	PA → J
23-	CN23	B →PR
24-	CN24	PE →W
25-	CN25	PR →PA
26-	CN26	PE →PA →PR
27-	CN27	PE →PA →J
28-	CN28	PE →B →PA
29-	CN29	PE →M →J
30-	CN30	PE →W →PR
31-	CN31	PE →PH →PR
32-	CN32	PA →J →PR
33-	CN33	PA →PR →B
34-	CN34	PR →B →PA

1.1.5 Database Management System (MySQL)

MySQL is an open source relational database management system (RDBMS) based on Structured Query Language (SQL).

MySQL runs on virtually all platforms, including Linux, UNIX, and Windows. It can be used in a wide range of applications, but most often MySQL is associated with web-based applications and online publishing through web development languages like PHP, and it is an important component of an open source community as well as the internet.

Nowadays, the MySQL RDBMS is considered one of the important databases as the huge scale of development, and any developer or programmer can find different models of the MySQL to fulfill his needs.

The main reasons for choosing MySQL to manage the DBLP database are:

1. The powerful management system for databases and performance regarding to huge databases.
2. The indexing used in MySQL for huge databases and fast retrieval of information.
3. The New future of SQL language and the capabilities of using indexes in searching.
4. The easiness of generating complex SQL statements and connecting to many web developing applications.

1.2 History of Keyword search

Keyword search where developed along with the development of the internet, as more and more information where stored in the world wide web, and the need to search the huge information scattered over the internet increases, which increases the development of keywords search over the internet for information stored in pages and documents with no structure.

By the end of the last century and beginning of current century, the main focus of keywords search was over the internet, which led to the development of the major search engines in the world, which provided the regular users with powerful search interface with retrieval of huge data related to the user search term from various documents and web pages.

As the databases grow largely in the last decade, the need for efficient and effective search frameworks over relational database grows and the research community and database vendors started to develop search algorithms to enable the users to search the databases especially over the web, as databases considered a main base of information resource on the web and large portion of information stored in such relational databases.

DBXplorer [1], DISCOVER [8] and BANKS [21] were the first to introduce the concept of keyword search over relational databases with Top-K answers in 2002, and many others follow to present or develop existing work done in the field of Top-K keyword search over relational databases.

The work on Top-K keyword search over relational databases can be divided into two categories:

1. Efficiency: which is related to developing algorithms that generate candidate networks of joint tuples or joint tuples tree or graphs based on the keywords entered by the user and the information provided from the database structure, schema or any introduced indexes, with the elimination process of unrelated or unsatisfying certain conditions of CN's or joint tuples to achieve a higher performance.

2. Effectiveness: the process of ranking and ordering the results or CN's or joint tuples or graphs to present the user with the most related results to achieve a higher effective algorithms.

1.3 Problems and Objectives

The problem that we address in this thesis is the efficiency and effectiveness of current methods for Keyword search approaches over relational databases, which have some limitations regarding:

1- Most relevant results do not have the highest rank [1].
2- Keyword search does not take into account the domain of data [1,2,9].
3- The trees generated (Joint-Tuples-Tree (JTT)) have a large size regarding the size of the data itself [2].
4- The ranking used is a general ranking style (IR-style) [7].
5- History of search and hits are not taken into consideration in ranking methods [10].

The research problem can be summarized by two main parts that researchers and previous work focus on:

First part: The efficiency (performance of keyword search over relational database is optimal and the effect of database size, number of keywords, and Top-K are minor).

Second part: The effectiveness (all the Top-K results are relevant to the user search terms).

Objective

The main objectives of this thesis are to achieve the following:

1. Help millions of users to easily search for keywords in relational database in an effective and efficient way.
2. Present users with the best and most relevant choices to their search.
3. Help exploring huge databases and publishing knowledge.

1.4 Motivation

Our motivation for this thesis and all the work done in the domain of Top-K keyword search over relational database can be summarized as follows:

1. The user community has limited knowledge of the structure of database and current information retrieval methods like Structured query language.
2. Databases nowadays are one of the main sources of information and knowledge.
3. The number of limitations on current research about Top-K keyword search over relational databases.
4. No easy interface for exploring and retrieving information in a free form keywords search over relational databases that present the user with the best choices.

1.5 Contribution

The contribution of our work in this thesis will include the following:

1. Introducing new index keywords table structure and information table to any given database for the purpose of generating only the CN's with high weights and eliminating the low weight CN's to reduce the number of props to the database, and increase the performance of the proposed Top-K keyword search.

2. Improving the existing Ranking style algorithm by involving new factors in the ranking process to present the user with more related results and effective search, which is measured by calculating the percentage of related results to the overall results, and showing the Top-K results that are most wanted by the user. The ranking style would include the semantics, domain and user history as weight factors in the ranking process.

3. Introducing Candidate Network generator algorithm based on the index keywords table, and information table which minimizes the database props by minimizing number of CN's and increase the performance.

1.6 Thesis Structure

This thesis includes several chapters to illustrate the work and research about the Top-K keyword search over relational databases, the thesis structure is as follows:

- Chapter One: introduction to Top-K keyword search over relational databases, which shows the history of keyword search over relational databases, the relational databases concept, the DBLP database structure and tuples used in this thesis for testing, the problems that this thesis adresses the objective and the contribution to the filed of Top-Keyword search over relational databases.

- Chapter Two: the literature review ,which introduces the work done by other researchers in the filed of keyword search over relational databases in specific aspects like graph presentation of databases, the candidate network genration, difrrent sytlyes of retriving information from databases, improving the performance of the search, improving the qulaity of the returend results with different ranking styles and ways to index the database.

- Chapter Three: introduces the new structure of indexing keywords with the information table, also introduces the algorithms used in CN generation based on the index keywords and information table and the elements used for ranking the results.

- Chapter Four: shows the build of a web interface for the perpose of testing the modified and new components of Top-k keyword search over the relational database DBLP. The interface also records the users interaction and stores all data to the database for future analysis.

- Chapter Five: shows the data gathered through the testing phase, the analysis and comparing of results, main findings, and conclusions and future work.

1.7 Methodolgy

To be able to compare the new modified and introduced components to the Top-K keyword search over relational databases, the following phases would be followed:

- Phase one: downloads and configures the DBLP database, as the DBLP is the most database used by researchers to test and validate different algorithms in the field of keywords search over relational databases.

- Phase two: Runs the database through a database management system (MySQL), as it is an open source, and have a good performance in running databases with millions of tuples, which also include functions for matching strings and indexing strings with very fast retrieval of tuples.

- Phase three: Build the structure of the index keywords table and the information table, with the implementation of the offline index keywords generator to store all keywords in the database in the index keywords table and related information like relations in the information table.

- Phase four: build the associated tables for the data gathering and record the users' interaction with the web interface to record different results for analysis and comparing results with the same work done by researchers on the same database.

- Phase five: constructs the CN generator algorithm, the elimination algorithms, the ranking function, the supported algorithms for recording the users' interaction and storing all related information in the database for future analyses by using the PHP language.

- Phase Six: builds the ‘Smart Search’ (web interface) to enable users to test the algorithms and the proposed keyword index table with the recording of time and return results, also the users choices for each search term to determine the quality of the results.

- Phase seven: preparing the testing environment which includes the random generation of more than 400 search terms from different tables and relations, and the chosen of five users to use the search terms through the web interface and choose the correct answers with a background process of storing all information related to each search term for analysis.

- Phase eight: for the purpose of generating different statistics and charts, all gathered data were converted to Excel sheets and analyzed with the generation of different charts, also compared the outcome for each search term size and the overall search terms with other researchers in the same field as listed in chapter six and appendix (E).

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

2.2 Graph Representation of database

2.3 Candidate Network Generation

2.4 Information Retrieval Algorithms in relational Databases

2.5 Effectiveness in keyword search over relational Databases

2.6 Efficiency in keyword search over relational Databases

2.7 Indexing Relational Databases

2.8 Ranking Styles

2.9 Summary

2.1 Introduction

Researchers relied in the last decade on the importance of keyword search over the world wide web, through documents and different types of information representations like homogeneous and heterogeneous data structures, especially for structured data with relations like databases, as the databases became bigger and bigger and the retrieving of information from large databases needs the knowledge of special languages like Structured Query Language (SQL), which is not design for regular users and need to learn a new language to be able to retrieve information from databases. For many reasons, the regular user do not also need to know the full schema of the database, which leads to the concept of keyword search over relational databases, which enables any user to search the full database and retrieve all related information stored in different relations with just inputting keywords in a free form search.

Relational databases nowadays are considered the backbone of major web sites, and millions of regular users interact with information stored in databases through web interface with specific search forms to retrieve information.

In this chapter, we review the research and work done in developing new algorithms and frameworks for information retrieval from relational databases using keywords and styles for ranking the results. We also try to identify the main aspects of research intervention and relate it to the work done in this thesis, and our work would cover the main aspects of IR like performance, effectiveness, data representations, CN's generations and joint tuples trees.

2.2 Graph Representation of database

The BANKS [21] models the database as a directed graph, with each node containing a tuple from the database and the direct edge between nodes represents the relation between primary-key to foreign-key. For any given keyword query with more than one keyword , the BANKS system finds the sets of nodes matching the search term entered by the user, then the BANKS system creates all the sub-graphs that covers the keywords in the search term entered by the user.

The BANKS system relays on the AND semantics, which means that all keywords must be contain in the sub-graphs created by the system, which would generate sub-graphs containing nodes with no matching keywords as they are considered a connected nodes to nodes that contain keywords from the search term entered by the user.

Summary

With the existing of large database sizes we must consider the following main challenges of representing the database as a graph:

1- The time needed to represent a given database with all its relations and tuples as a whole graph.
2- The complexity of updating the graph with new inserted records or updated records in the database.
3- The memory space limitation for loading the whole database as a graph in the memory, or the complexity of loading certain blocks of the graph and swapping other blocks as needed.
4- The size of the database and the complexity of relations between records, which leads to large graph representation of database and high occurrence of same tuples with many relations.
5- Graph representation is not realistic for online databases with many updates and transactions.

2.3 Candidate Network Generation

In effective Top-K keyword search [2], a Candidate Network (CN) is a tree of tuple sets R^{Q} or ^{RF} with the restriction that every leaf node must be a query tuple set. Every edge (R_i,R_j) in a CN corresponds to an edge (R_i,R_j) in the schema graph (SG). A CN can be easily transformed into its equivalent SQL statement and executed through the DBMS. The size of a CN is the number of its tuple sets.

By this definition, other researchers depend on the generation of CN's and introduce algorithms for generating CN's that satisfy the users entered keywords, CN's vary in size and number according to the search terms and algorithms used to build the necessary CN's. The degree of the CN is also an important factor that affects the number of CN's for a certain search term, as the CN represents the schema of the database; the number of edges in the CN represents the degree of the CN.

For example:
Author ^{K1} degree zero.
Author ^{K1} $\rightarrow$ paper ^{K2} degree one.

Research in generating CN's can be classified as follows:

- Performance (time to generate all the CN's for a given search term with all joint tuples retrieved).
- Effectiveness in generating CN's, which represents the generation of only the related CN's that would produce the Top-K results.

In SPARK [10], they proposed the three rules for generating CN's that produces the best results as follows:

Rule 1: Prune duplicate CNs.
Rule 2: Prune non-minimal CNs i.e., CNs with free tuple sets as leaf nodes.

Rule 3: Prune CNs of type: $R^Q \rightarrow S^* \rightarrow R^Q$. The rationale is that every resulting JTT would contain the same tuple from R^Q for two times.

All the three rules were adopted by many researchers, and also adopted by this thesis for generating the CN's, as the three rules lead in improving efficiency and effectiveness of Top-k keyword search over relational databases.

DISCOVER [8] proposed the breadth-first algorithm that is both sound and complete. It can enumerate all the CN's of size no more than a specified number without violating any pruning rules.

The following figure in Top-K keyword search [2], shows the CN generation according to a given keywords and how the CN's are translated to the corresponding SQL statements for retrieving results as tuples, which is shown in Top-K keyword search [2].

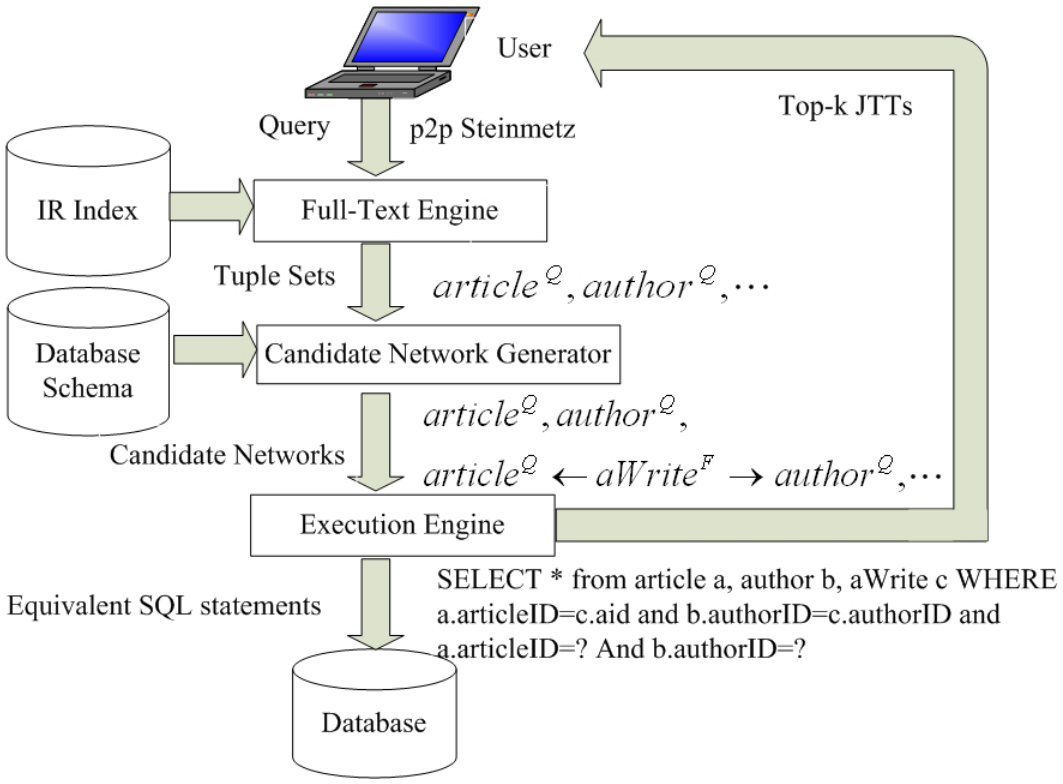


Figure 2.3: model of Top-K keyword search [2]

Figure 2.3 shows a framework for a free form search over relational databases with CN's generation and ranking style to present the user with the Top-K results.

Summary

In SPARK [10] they presented the three rules for improving the performance of generating the CN's and the effectiveness of results generated by the CN's, which we can see an improvement in their work, without the building of any additional index tables and relies on the schema of the database.

In our work, the three rules are implemented in generating the CN's, but not directly based on the schema of the database, but on the new structure of an index keywords table with the information table, which we would try to improve the performance by reducing the number of CN's and reducing the number of props to the database.

2.4 Information Retrieval Algorithms in relational Databases

The informational retrieval styles (IR) in databases differ in some aspects with the conventional information retrieval styles for known search engines and algorithms, as the structure of the database and relations between records forces the information retrieval to take into consideration the databases schema and relations that control the management and retrieval of information.

Conventional information retrieval main focus is in finding information in separate various types of documents that satisfy the users search term, and the information retrieved are mostly found in one location with no relation to other locations or documents, but in databases the job of IR is to find the information related to the user search term which can be found across different relations and the keywords from the search term are represented in different tables with existing relation between them, which is the job of the IR algorithm to build the results according to the schema of the database and present the user with the best choices based on the database schema.

DISCOVER [8] has proposed a breadth-first CN enumeration algorithm. The algorithm is essentially enumerating all sub graphs of size k that does not violate any of the three pruning rules, which represents the whole database as a graph and found the sub graphs.

In Progressive Keyword Search [5], they proposed the Minimum Path Weight Steiner Tree (MPWST) by presenting the database as a graph and by finding the MPWST, the result is a Steiner Tree (STree) and by using polynomial time complexity algorithms to produce a Compact Steiner Tree (CSTree), then each CS Tree is scored using a ranking function, and the results of this approach are shown in the following figures:

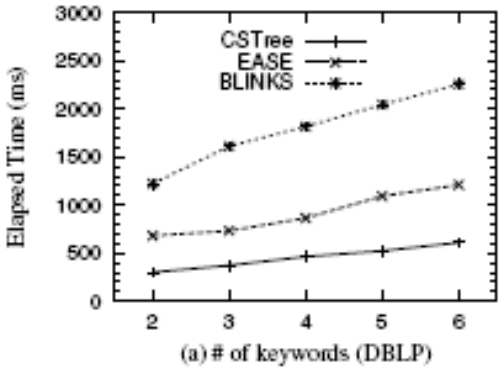


Figure 2.4.1: performance in Progressive Keyword Search [5].

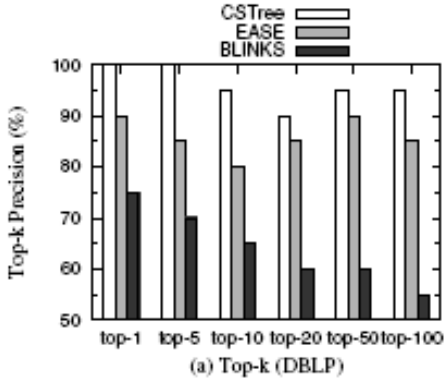


Figure 2.4.2: Top-K precision in Progressive Keyword Search [5].

By figure 2.4.1 and figure 2.4.2 we can see the effect of the CSTree on performance and precision of results compared to other algorithms like BLINKS and EASE.

Other researchers and previous work adopted the CN's representation of information to answer a certain search term in information retrieval styles like in [2,7,10,16,17], and their work address the following problems and issues in information retrieval:

- The problem of generating the minimum number of CN's with the best results contained within.
- They introduce new and modified supporting components like indexing keywords table, for more efficient and effective information retrieval styles.
- The problem of pruning unrelated CN's and with minimum relevancy scores.
- The efficiency and effectiveness of the information retrieval styles as the databases grows and the size becomes bigger.

2.5 Effectiveness in keyword search over relational Databases

Effectiveness as defined in term definitions is the accurate of the returned results according to the users and the percentage of correct results among the Top-K results, The main challenge beside the efficiency of the keyword search over relational databases is to have an effective keyword search, which can be achieved by using ranking algorithms to decide on the Top-K results based on different factors.

In finding Top-K answers [19], they proposed a single keyword-based structure-aware index called SKSA, which is similar to inverted index and contains the keywords and preserves the tuple units that directly or indirectly contain the keyword, and the score of each keyword in the tuple unit which they based their scoring for each keyword. However, the size of this index can be very large and the paper presents a strategy of pruning many irrelevant tuples based on a threshold factor, they test their work on the DBLP and IMDB databases and compare their work with other methods, as shown in the following figures which shows an improvement in efficiency and effectiveness.

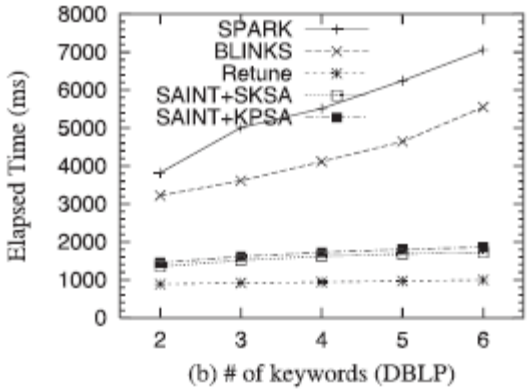


Figure 2.5.1: performance comparison for different IR-styles in Finding Top-K answers [19].

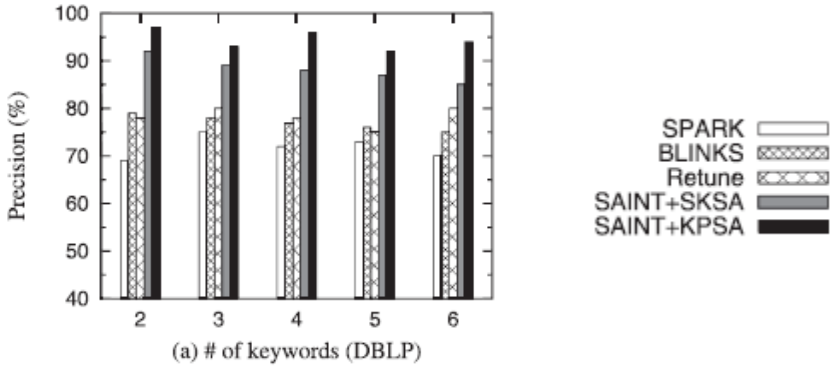


Figure 2.5.2: precision comparison for different IR-styles in Finding Top-K answers [19].

In effective Top-K keyword [2], the main contribution is the introduction of new ranking strategy based on the query semantics, which is presented by the relations between keywords entered by the user, and score each result based on relations and distance between relations of keywords.

The query semantics where incorporated in SPARK [7] ranking function with a precision factor (p), and the following table shows the relevancy at Top-20 by using different precision factor and the incorporation of semantics into SPARK [7].

Table 2.5.1: relevancy at Top-20 in SPARK [7] with different precision factor.

[5](p = 2)	[5](M)(p = 1:5)	[5](M)(p = 2)	[5](S)(p = 1:5)	[5](SM)(p = 1:5)	[3]	[3](S)
4	17	18	12	19	0	5

SPARK [7] proposed to represents the joint tuple tree as a virtual document which is scored by three scoring functions and the final score are represented by:

$score(T,Q) = score_a(T,Q) + score_b(T,Q) + score_c(T,Q)$ as

$score_a(T,Q)$ represents a function for the weight of the virtual document containing the keywords.

$score_b(T,Q)$ represents a function for the distance of a virtual document to an ideal position represented by P as a weight factor.

$score_c(T,Q)$ represents a function to calculate the size of the normalization factor of the JTT or CN.

And to be able to calculate the Ranking for each tuple efficiently, SPARCK [7] proposed the used of two algorithms: 1- The Skyline Sweeping Algorithm. 2- Block Pipeline Algorithm which they aim to speed up the ranking calculations and stop the execution when Top-K results found.

The following table shows the effectiveness of their work compared to other works and done on the DBLP database.

Table 2.5.2: effectiveness over the DBLP database on Top-20 results in SPARK [7].

	[15]	[22]	P = 1.0	P = 1.4	P = 2.0
#Rel	2	2	16	16	18
R-Rank	≤ 0.243	≤ 0.333	0.926	0.935	1

By Table 2.5.2 we can see the effectiveness of algorithms used in SPARK [7] comparing with previous work and the top relevancy percentage for p = 2.0 is 0.90 that the algorithms achieved.

Summary

The good practice of ranking styles would be used in this thesis, mainly the introduction of semantics of keywords and the distance between keywords and tuples, which are adopted in this thesis for more effectiveness of results.

We can see Cleary that SPARK [7] achieves at most 93% relevancy of returned results for Top-20 using the DBLP database.

2.6 Efficieny in keyword search over relational Databases

In the research paper “efficient schema based keyword search in relational databases [22], stated that there are ways to reduce the CN generated, and in this paper the researchers follow one way and let other ways for future direction, as they clearly stated that there can be work done previously to the generation of CN's, like semantics of building indexing or any way that would help in reducing or choosing the most related CN's, which would lead to reduce in propping the database, leading to an increase in performance; that is "efficiency".

In this paper, Digital Bibliography and Library Project (DBLP) dataset and the Internet Movie Database (IMDB) where used for testing large number of queries containing keywords (2-6), and the results where compared to two known keyword search modules (DISCOVERY [8], SPARK [7]).

The researchers shows that their ATCNGen algorithm for generating and evaluating CN's is better regarding the execution time (performance) and CN's size, for the quires with several keywords.

The following table’s shows the datasets used in efficient schema [22] for testing algorithms and comparing results.

Table 2.6.1: statistics of DBLP dataset used in efficient schema [22].

Relation Schema	#Tuples
Person(Pid,Name)	174,709
InProceeding(Iid,Title,Pages,Rid)	212,273
Proceeding(Rid,Title,Uid,Sid,…)	3,007
Publisher(Uid,Name)	86
Series(Sid,Title)	24
RelationPersonInProceeding(Pid,IPid)	491,777

Table 2.6.2: statistics of IMDB dataset used in efficient schema [22].

Relation Schema	#Tuples
Actors(Aid,Name)	817,718
Directors(Did,Name)	86,880
Movies(Mid,Name,Year,Rank)	388,269
Movies-Directors(Mid,Did)	406,967
Movies-Genres(Mid,Genre)	417,784
Roles(Aid,Mid,Role)	3,432,630

In efficient schema [22], the researchers stated the same research problem in my work; also the work and solutions introduced in their work are in the same component as my proposed work (generation and evaluation of CN's). They used algorithms to generate and evaluate CN's in the execution time of a keyword search, my work is to introduce new stage previous to the first stage, which would only introduce the related CN's with enough knowledge of some factors about each CN's for evaluation of each CN to eliminate unrelated and with low weight CN’s, which result in reducing the number of props to the database, which affects the time of executing all the associated SQL statements with each CN.

Summary

1- The main focus of efficient schema [22] is on the performance of keyword search over relational databases, in the field of generation and evaluation of CN's.
2- There is an approximation of the results returned to the user-based on his query.
3- The algorithms developed and used achieve high performance in efficient schema [22] compared to other approaches.
4- There is no ranking style used after executing the CN's, as the paper only evaluates the CN's, and no ranking style used for the returned results.
5- The logic of keyword search depends on two stages : a- generating all results through JTT , b- evaluate the JTT's to get the Top-K keyword search results , these two stages need time for each stage and we can donate stage a as S1 , and stage b as S2
   So the total time (Ts) = S1 + S2. And we can achieve performance by working on the two stages or one stage, but we cannot just work on one stage and not include the second stage in time calculation.

2.7 Indexing Relational Databases

In Indexing Relational Databases for Efficient Keyword Search [20], the paper focuses on building an index table that contains all searchable fields in all tables, for efficient retrieval of information by keyword search and ranking of results. The researchers used simple ranking style with two parameters to reduce database propping and increase performance (efficiency), the index -as the author stated- have a manageable size, and builds in offline mode with reasonable building time.

The index table developed in Efficient Keyword Search [20] contains two fields: keyword: combined searchable fields for one raw, MapId: points to the table name and row number of the fields in the original table.

And the search algorithm would use this table to search for keywords entered by the user, and then an algorithm for scanning relations between return results, all return tuples evaluated using a ranking equation:

$$\text{Score}(k) = \frac{\textbf{count f(k)}}{\textbf{n(k)}}$$

Where n(k) = number of keywords in a user query,

Count f(k) = user keyword match keyword occurrence in m(U,V),

Score(k) = score of each relevant record.

The ranking equation is very simple and direct, and no need to prop the database for more information, which increases performance of the search.

The database used in this paper is the DBLP Dataset with the following Characteristics:

Table 2.7.1: DBLP database tuples and tables used in indexing [20].

Table name	Number of records
Inproceeding	208,086
Series	24
Publisher	81
Person	162,907
Proceeding	2,749
RelationPersonInProceeding	491,777
Total	1,357,401

The only tests in this paper are for efficiency with comparison with existing IR algorithms for keyword search over relational databases, and the results show that their method is better.

Our work will introduce new index table structure for storing all keywords in the database, but the size of our index keywords table would be much smaller, and our index would contain information related to keywords that would be used in indirect way to reduce the number of CN's, which would have a huge reduce in the size of JTT, also the information stored in the index keywords

table would be directly used in the ranking and eliminating of tuples that do not satisfy the minimum ranking weight.

Summary

1- In indexing relational database [20], the work was done to examine the performance of introducing new index keywords table.
2- The index table used in indexing relational database [20] contains information that is not searchable like ISBN numbers and page ranges.
3- Through looking at the index table in [20], we can see that the table contains all the text fields from the original fields in different tables, which impacts on the size of the index table.
4- There is no testing in indexing relational database [20] for the proposed ranking equation or any comparing of results.

2.8 Ranking Styles

During the last decade, the ranking styles developed from simple ranking equations to complex ranking styles, in DBXplorer [1], the ranking style was based on the SQL match function, which just simply matches the search term to fields in the database and presents the users with highest match score with no other factors to consider in ranking the results or just presenting the user with the Top-K results.

In SPARK [7], if we are looking for “Steinmetz p2p”, the data in table 2.8.1 shows that most of the results are not useful to the user, as no papers written by “Ralf Steinmetz" with “p2p" in the title are returned, as the ranking style used in SPARK [7] do not give higher weights to CN’s with depth more than one like the CN: authors → papers.

Table 2.8.1: Top ten results for query “Steinmetz p2p" by SPARK [7].

JTT	Score	CN
E.Steinmetz	3.40	Author
Uli Steinmetz	3.37	Author
2 P2P or Not 2 P2P?	3.35	Article
2 P2P or Not 2 P2P?	3.35	Article
Ralf Steinmetz	3.34	Author
Arnd Steinmetz	3.34	Author
Rita Steinmetz	3.34	Author
Aase Steinmetz	3.34	Author
Oliver Steinmetz	3.28	Author
Ulrich Steinmetz	3.28	Author

In effective top-K keyword search [2], the problem of the ranking style found in SPARK [7] was solved with the introduction of semantics of keywords to the evaluation equation to find the relevancy of each CN to the keywords enter by the users, and the final results show improvement in the quality of the returned results.

In effective Top-K keyword search [2] for a keyword w, we can easily find all relations R1;R2; ….;Rt that have tuples containing w, and for each $R_i(1 < i < t)$ the researchers used a TF-IDF weighting formula to compute its relevance to keyword w.

The TF-IDF weighting formula is commonly used for ranking documents.

2.9 Summary

In this section we will summarize the work done by other researchers on Top-K keyword search over relational databases, with the concentration on the main domains that researchers address which they are:

- The performance (efficiency) of Top-K keyword search algorithms and modules introduced by researchers and measured by execution time.

- The effectiveness of modified or new introduced algorithms and components over the Top-K keyword search on relational databases, which is measured by the relevancy of returned results to the user according to the inputted keywords.

In table 2.9.1 we can see the work done by other researchers and which components introduced or algorithms enhanced for the purpose of improving the IR-style over relational databases.

Table 2.9.1: Summary table of previous work with comparison to work done in this thesis.

No.	Research paper	Work done / description	Parameters / algorithms / components, would be enhanced.	Category (Effectiveness / Efficiency)
1-	DBXplorer [1]	Introduce a web free form search on relational database that enables the user to search for any given keywords without the knowledge of data structure, with generating joint trees and creating hash tables to speed up the search, also they mentioned the ranking of the results, with the use of match function.	This research did not include any index for keywords, the ranking function just compare of number of keywords in the returned results, the search need to prop the database for each CN.	Our work would introduce an index for keywords with an information table that would help the CN generator to generate only the required CN. The ranking function in our work will include four parameters for each returned result to provide the user with the best choices according to his search

				term.
2-	Discover [8]	Researchers main work in this paper, the search over databases and generating optimized CN's, with introducing algorithms for prune CN's, they implemented the Greedy algorithm for retrieving results, which provides near-optimal plan execution time cost.	The researchers main work in building an efficient algorithm for pruning CN's, with greedy algorithm to retrieve the results and present the results to the users with no ranking style used.	The CN generator and pruning algorithm used based on the master index of the database with evaluation of all possible CN's in the database, which lead to prop the database by the number of CN's Our work only prop the database for the CN's that would have tuples containing search keywords, and prune more CN's.
3-	Efficient IR-style keyword search over relational databases [16]	The introduction of new IR-style document-relevance ranking strategies to the problem of processing free-form keyword queries over RDBMSs , there are huge work done on one large database , many ranking algorithms , that just retrieve the most relevant results which Leeds to more efficient top-keyword search. There a good part for analyses, they used the And + OR semantics.	The researchers used the basic way to rank the CN retrieved for keyword search, and the main work on the efficient ways to retrieve the information, also the effectiveness of the IR is important and need to be studied and suggest new effective algorithms for IR. Used algorithms like sparse, pipeline and hybrid for retrieving results to search terms entered by users.	This paper main contribution in developing efficient algorithms for retrieving results from databases without the introduction of new components, and new concentration on ranking of results, all analyses were done on the efficient factor.
4-	SPARK: Top-k keyword query in relational databases [7]	The researchers in this paper study the effectiveness and efficiency of ranking and IR on top-k keyword search, they proposed new Ranking style, also the development of new IR algorithm, that by extended experiments on two different large databases show a significant improvement in IR and Ranking efficiency and effectiveness. They did some optimization on the SQL generation and fetching of results. This paper considered a base	The researchers on this paper conduct and proposed new functions and algorithms to improve the efficiency and effectiveness of keyword search, all the work concentrated on the query optimization and ranking style, the generation of CN, more work need to be done on generating CN's and how to optimize the search size and time before the stage of IR.	The elimination of CN's can be improved by introducing new components to the database like indexing of keywords, to help the CN generator to generate only the required CN's and prune all CN's that do not meet the minimum requirements.

		for comparison of our work		
5-	SPARK: A Keyword Search Engine on Relational Databases [10]	In this paper, the researcher lunched the engine of spark (search , prop and ranking framework for top-k keyword search , it is a continuous of their previous work , they just implement it in a full web interface page with simple and power search , and compare it with other search frameworks on the same databases.	As it is a continuous work to the previous work done, so no new ideas where presented here, just a full implantation to their work.	No analyses for using the web interface keywords search over relational databases.
6-	Effective Top-k Keyword Search [2].	Researchers propose a novel IR ranking strategy for Top-K keyword search on relational database, with real experiments on large databases with regards to relations between tuples, the main work in this paper is the retrieval effectiveness. Their work shows the database schema and the framework for Top-K keyword search, with the mathematical equations for ranking each tuple.	The work just concentrate on IR ranking method, which is a part of the Keyword search with no mention of the efficiency, and the work, can be developed by taking the efficiency factor into the framework of research. Their work shows that the precision of returned results for different Top-K can reach up to 90% by using a ranking style similar to a document ranking styles.	The ranking function introduced in this thesis would achieve a higher precision compared to this paper, also with a main work done on enhancing the performance of the Top-K keyword search.
7-	Progressive Keyword Search in Relational Databases [5]	The researchers introduce a structure-aware-index with CSTrees for information retrieval from databases based on keyword search with new ranking strategy based on documents ranking styles that assign weights to nodes.	The results shows a high performance of the system comparing to other approaches, with a high precision of results (effectiveness) that exceeds 90% at all times. All experiments done using the DBLP and IMOVE databases	Our work represents the results as CN's, while their work construct Compact Steiner Trees to represents the relations of returned results, our ranking style depends on four factors and their ranking style depends on known document ranking style (TF-IDF).
8-	Efficient Continuous Top-k Keyword Search in Relational Databases [9]	The researchers propose a new method for evaluating the Top-K answers to the user on continues updated database, and continuous keyword search over the database, as the main problem of re-evaluating the	Their main work on how to maintain a high precision of results for continues updated databases with high performance without the need to regenerate all CN's for a given search terms.	The researchers did not show how to store the Top-K results for search queries, and for large number of quires, how to it affect the performance of re-ranking results.

		ranking on any keyword if the database is updated which is time consuming and not efficient. Their approach is storing the state of a query evaluation process, and in most cases they do not need to re-evaluate the scores, just insert the new tuples.		
9-	SPARK2: Top-k Keyword Query in Relational Databases	It is a continuous work of previous research, that introduces new ranking algorithm, which is an improvement on the previous algorithm, also presenting new three algorithms for minimizing database props, many testing were done using large databases (DBLP, IMDB). Many charts presented that satisfy the efficiency and effectiveness factors for a Top-K keyword search on relational databases.	They did not investigate the keyword search over online databases, or with continuous keyword search , also the CN generation stage and the keyword index table not mentioned or studied , their work focus or IR and ranking , which they did a good work , with many testing and analyzing of results , also comparing their results with others. Their work can be used and build upon, as it is a good framework for Top-K keyword search.	

CHAPTER THREE: INDEXING KEYWORDS WITH CN GENERATOR AND NEW RANKING STYLE

3.1 Introduction

In this chapter we will introduce the structure of the information table, as well as the information stored in the information table, which considered the first components added to the DBLP database, that aims in helping the Smart Search in presenting the users with the results according to the entered search terms.

We show the importance of the information table, and the relations that it contain which lead in generating all the CN's for any given search term, as the information table contains the direct and indirect relations between tables and tuples in tables, based on keywords found in the index keywords table.

The second component added to the DBLP database is the index keyword table, which we introduce a new structure of indexing the keywords in the database, the new structure only stores keywords with no repetition for the same filed and discards the stop words, also we show how to build the index keywords using two different algorithms, the online algorithm and the offline algorithm.

The main algorithm used for building the index keywords table is the offline algorithm. We also show how to update the index keyword table for online databases, with real examples to show the importance of our introduced structure and effect on CN generator algorithms which is designed for generating CN's based on the index keywords table and the information table.

Also in this chapter we show the CN generator and eliminator as a main contribution in this thesis, which is developed based on the structure of the index keywords table and only generates CN's which would contain results according to the users search term, the CN generator and eliminator with experiments and examples that shows the importance of the new CN generator algorithm.

After executing all CN's, the returned Joint tuples of results are evaluated according to a comprehensive ranking function to present the user with the Top-K results, which evaluates each

result according to four different weights: weight of the match function, the semantics of keywords, the distance between keywords and the search history. Which we have explained with examples in this chapter.

3.2 Information Table structure and Contents

A table containing the required information about the database structure in a away differs from the database schema presented by database vendors, which contains information to help the CN's generator to identify all relations and searchable fields within relations in different tables to build the necessary SQL statements, and only the required SQL statements related to keywords entered by the user.

The information table shown in table 3.2.1 represents the overall structure of the database DBLP, which is designed specifically for keyword search, as regular schema do not express the full paths and relations between searchable fields in the database, as the schema of any database represents the relations between tables regarding fields called primary keys to foreign keys, which we can conclude that they are not searchable fields, and may not include the full path to the targeted table or fields inside a table, which contains an important factor or information to obtain to fully search the database for certain keywords and to have a good results presented to the user with high relative to his request.

The information table does not only state the relations between tables, but also lists the field names that are searchable for each table, and the indirect relation between different tables, which will see the importance of this information in the CN's generations, as the SQL constructor will have a clear view and would be an easy way for the algorithm for building all the corresponding SQL statements for the generated CN's.

The size of the information table will be very small, as the information table only states the relations between tables, so the number of records in this table is limited.

In our work, the number of tuples in the information table is 34 tuples which corresponds to all possible relations between tables in the database and represents all possible CN's that can be generated for any given search terms.

Importance of information table

In this part, we will illustrate the need and importance of building an information table as part of the database, to be used as a reference for any keyword search, and its impact on the performance generated by the SQL constructor.

1- All information regarding the database is in one small table with clear fields that reflect the relations and information about all searchable fields that the database can have.

2- Number of SQL statements issued to retrieve information about the database, and all fields for each table are limited to one SQL statement, by which we can retrieve all the necessary information we need to construct our SQL statements and related fields in just one SQL statement.

3- All searchable fields in the database are found in the information table, which will form the needed information for the index keyword builder, as the index keyword builder needs to know all searchable fields to be indexed, with relations and information regarding fields containing keywords.

4- The indirect relations between tables and fields are defined by the information table. For example, we can see that the person name in the persons table that has many relations to other tables through the intermediate table "raw_links", and we can also see the fields in the target tables that are important to our keyword search, the indirect relation between persons like the author of the same paper, or audit the same journal or web site, which will have impact on ranking of results.

Table 3.2.1: Information table added for Top-K keyword search.

Id	K_id	K_filed	K_tname	K_Pkey	F_key	F_tname	direct	C_tname	C_filed	relation	F_filed
1	1	Name	Persons	Id	Id	papers	no	Raw_links	From To	author	title
2	1	name	Persons	Id	Id	books	no	Raw_links	From To	author	Title publisher series
3	1	name	Persons	Id	Id	journals	no	Raw_links	From To	Editor of	Title publisher
4	1	name	Persons	Id	Id	phdthesises	no	Raw_links	From To	author	Title publisher series school year
5	1	name	Persons	Id	Id	msthesises	no	Raw_links	From To	author	Title school year
6	1	name	Persons	Id	Id	proceedings	no	Raw_links	From To	author	Title publisher series conference year
7	1	name	Persons	Id	Id	wws	no	Raw_links	From To	Editor of	Title url
8	2	title	Papers	Id	Id	proceedings	no	Raw_links	To From	in-proceedings	Title publisher series conference year
9	2	title	Papers	Id	Id	journals	no	Raw_links	To From	in-journal	Title publisher
10	2	title	Papers	Id	Id	persons	no	Raw_links	To From		Title

Information table fields:

Id: Serial number to present the primary key of the table.

K_id: Keyword id, which presents a foreign key associate with a keyword id in the keyword index table, to be able to identify the information related to each keyword, especially the occurrence of the keyword in more than one table or field.

K_field: This is the field name that the keyword was found in and used for constructing SQL statements.

K_tname: Presents the keyword table name, used for constructing SQL statements.

K_Pkey: Presents the field of the primary key in the table.

F_Key: Presents the field of the foreign key in the corresponding table.

F_tname: Presents the name of the table containing the foreign key.

Direct: States if the two tables are direct connected to each other, or if there is an intermediate table.

C_tname: Intermediate table name, if any.

C_field: The two field names in the intermediate table correspond to the primary key and foreign key.

Relation: Presents the relation type between the source table and the targeted table.

F_field: Presents the fields in the targeted table, which may contain searchable keywords.

3.3 Structure and importance of index keywords table

The work done on top-K keywords search over relational databases has three modules, which we implemented. The three modules that contain our intervention on the selected database for research named DBLP2, in this section the work focused on designing and implementing the first module, which is designed for creating the index keywords table and the information table. The creation and importance of the information table where illustrated in section 3.2.

The first module considered the base and main work introduced in top-K keyword search over relational databases, and the importance of adding an index keywords table that contains all keywords for all searchable fields in the database.

The index keywords table is a table that contains keywords from searchable fields in the database after removing all the stop words as listed in appendix (B). All keywords are related to the information table to identify the existence of keywords and the location of keywords within different tables in the database, and in table 3.3.1 we can see the different fields of the index keywords tables with illustration to each filed.

Table 3.3.1: The index keywords table structure and sample of keywords stored.

id	Keyword	order_co	oc_co	col_name	table_name	key_hits	pr_key	t_name1	fr_key1	count_key1	Flag1	Flag2	Info_t
1	**Stephan**	**1**	**4**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**10**			**1**
2	**Vollmer**	**2**	**2**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**5**			**1**
3	**Kurt**	**3**	**5**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**7**			**2**
4	**Rita**	**1**	**6**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**8**			**2**
5	**Ley**	**3**	**7**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**4**			**2**
6	**Tolga**	**2**	**4**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**9**			**1**
7	**Yurek**	**1**	**3**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**12**			**3**
8	**Fielding**	**4**	**1**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**4**			**4**
9	**Berners**	**2**	**7**	**name**	**Persons**	**0**	**id**	**raw_links**	**From**	**7**			**5**

Id: Serial number auto increment.

Keyword: A clean nonstop keyword for the process of scanning tables and searchable fields.

Order_co: The order of the keyword in the text extracted from.

Oc_co: The number of times the keyword occurs in the same column in the same table.
Col_name: The column name where the keyword exists.
Table_name: The table name where the keyword exists.
Key_hits: Number of keyword searches.
Pr_key: The column name that presents the primary key for this keyword.
T_name1: Table name that contains the foreign key corresponds to the primary key.
Fr_key1: Foreign key column name in the table name above.
Count_key1: Number of tuples connected to the keyword according to the relation identify above.
Flag1: Used for pre-calculations for ranking of keywords.
Flag2 : Used for pre-calculations for ranking of keywords.
Info_t: Primary key related to foreign key in the information table to identify the number of CN's, also the SQL statements needed to satisfy the keyword search.

The existing of the index keywords table is important as the schema of the database in any database that we need to use is free form keyword search, the index keywords table is the first table to be searched for all keywords to retrieve all information related to keywords especially the semantics of keywords and the existence of each keyword, all of this can be done in one prop to the database by identifying all related candidate networks, and just the required CN's.

The building of index keywords table can be achieved by different ways:

1. Builds the index keywords table during the search (online mode), which leads to adding all searched keywords by users to the keywords table, and with time the keyword table will have almost all keywords in the database.
2. Builds the index keywords table in an offline mode, which can be done on an image of the database, and the results, can be integrated with the online database.

id	keyword	order_co	oc_co	table_name	col_name	key_hits	pr_key
66664	mml	0	102	papers	title	0	id
66663	dstl	1	1	papers	title	0	id
66662	summation	0	288	papers	title	0	id
66661	ageing	0	17	papers	title	0	id
66660	compressing	0	414	papers	title	0	id
66659	chainmail	0	5	papers	title	0	id
66658	styles	0	703	papers	title	0	id
66657	exegesis	0	10	papers	title	0	id
66656	dfd	1	30	papers	title	0	id
66655	auto	0	1174	papers	title	0	id

Figure 3.3.1: Snapshot of index keywords table.

The process of generating the index keywords table used in this thesis is the offline building of index keywords table, which does not affect the performance of the database, especially for online databases with frequent update and transactions, and the whole process of generating the index keywords table may take a lot of time for large databases, depending on the number of searchable fields and fields lengths, and also on the list of stop words, which makes it a challenge to design a good algorithm with high performance. The other solution is to build the keyword table during the search, which leads to add all searched keywords by users to the keywords table, and with time the keyword table will have almost all keywords in the database. This way can be called parallel index keywords generator, which can be run in the background and may not affect the performance of the database and our keyword search system. This can be done by using smart index builder that runs only if database is idle and stops if there are actions on the database.

Main challenges for creating index keywords table:

1- Numbers of tables that contain searchable fields are seven tables with six fields.
2- Total number of records for searchable fields = 1.5 million filed denoted as T_{rf}.
3- Approximate number of stop words that I have = 1 K denoted as S_t.
4- Average number of words in each field = 4 words denoted as A_{wf}.
5- By Simple math , expected number of executions for the index builder is:

a. $(Trf * Awf)^{5t} \approx (6000000)^{1000}$ times.

b. By using B-Tree to represent the stop words list we reduce the number of runs in a significant way.

c. Furthermore, as the index does not allow duplicate keywords for the same table and filed name, so for each keyword, before storing it in the index keywords table, we need to check for its existing with same information, which will have more runs.

d. By running simple algorithm to find keywords in the field name in table person, for the first 10 K records, the algorithm returned about 14 K unique keywords in about 100 seconds, which can be considered a lot of time for online databases.

e. For the n'th keyword, we need to examine the existence of it by using two ways :

- We can store each keyword in the index table, and then we need to query the table for each new keyword for existence.
- We can store the keywords in B-Tree for fast search and retrieval, and we used this way for temporary store of keywords to enable the algorithm to check for duplicates, and after we finish the whole table we store the B-Tree in the index keywords table.

3.3.1 Offline building index keywords table

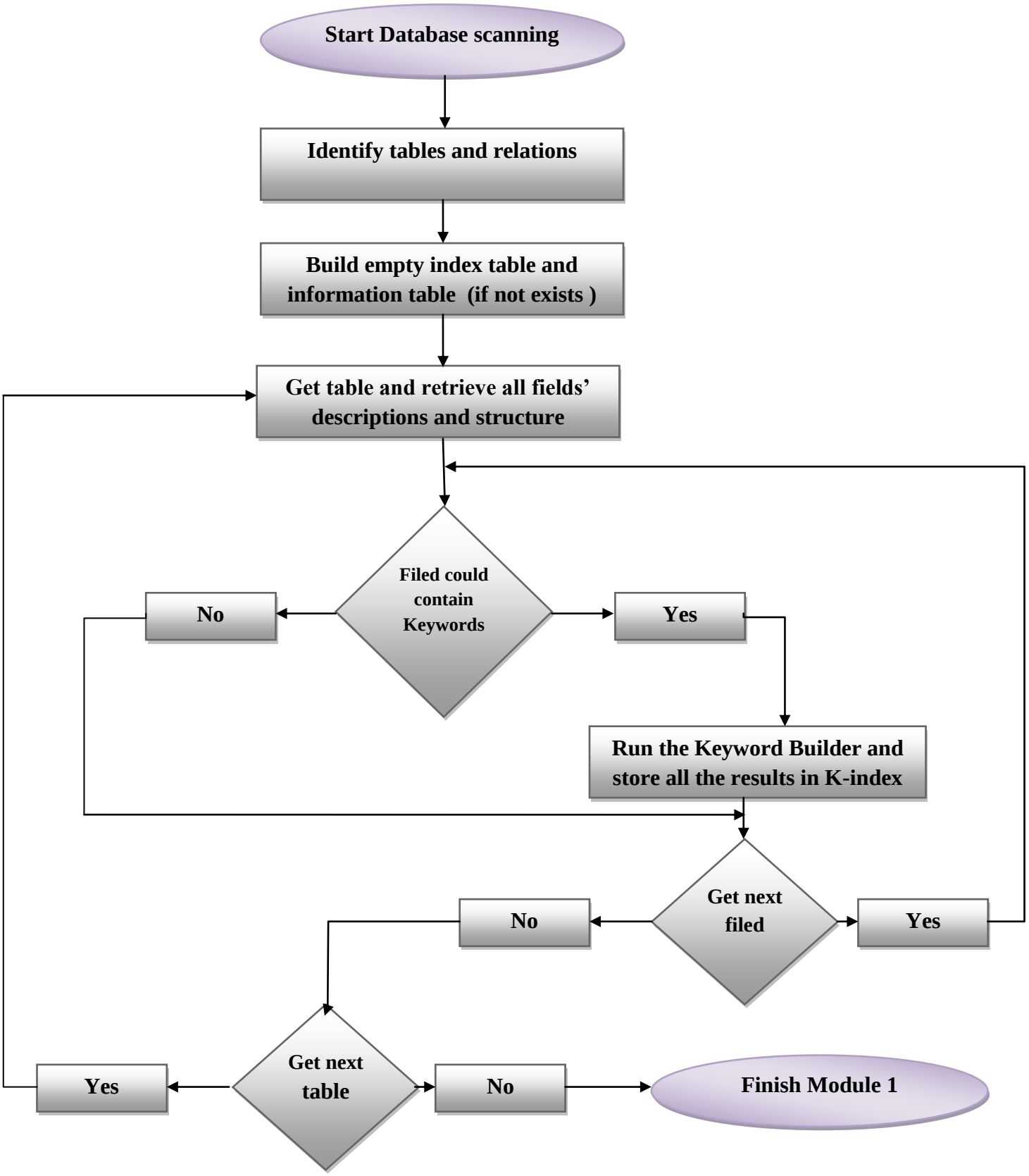
Our work proposed three modules to form our ‘Smart Search’ engine through a Smart Search web interface. The first modules is to maintain information about the database used in ‘Smart Search’ related to the structure of the database and the keywords found in the database that are searchable by users, and we need to run this module for one time to generate an index keywords table and an information table.

In this thesis, we used the offline building of the index keywords table and the following module 1, and algorithm 4.2.1 shows the generation of the information table and the keywords table, which can be run in an offline mode.

The run of this algorithm does not affect the performance of the database, as it can be run in an offline mode, and the average time to process (100,000) tuples and store all the keywords found in those tuples is about (180) seconds, By running the algorithm on the backup copy of the database.

If we denoted the time for updating the index keyword table with the update of any table in the database as N, then N is a fixed execution time which would be the time to insert or update or delete a record from a given table depending on the DBMS used, and would have a very small effect on the database performance.

Module 1: Building information tables and index keywords for a database (offline mode)



Definitions:

D: Represents the database.

S: Represents the schema of the database.

T: Represents the tables in the database.

K: Represents the keywords index table.

F: Represents the fields in the tables.

R: Represents the information table.

ST: Represents the stop keywords.

KF: Represents the keywords in the fields.

Algorithm 3.3.1: Building information tables and index keywords for a database

---

Algorithm IndexBuilderDB

Input: Well defined database with relations, stop keywords file.
Output: Index keywords table.

```
1  Begin
2     If ( D exists and not empty ) then
3     scan schema of D
4     Load ST to B-Tree B
5       For each T in D do
6          For each F in T do
7             For each   KF in F do
8                If KF in  B then
9                  Do nothing
10             Else // it's a keywords
11                   If KF exists in K
12                        Increase occurrence counter
13                   Else
14                      Store KF in K with related data
15               End for
16            End for
17      End for
18
19 else
20     print out " database error or empty "
21 End
```

Example for generating the index keywords table with keywords

Table 3.3.2: Example of running the index keyword table generator.

No.	Input	Process / algorithms	Output
1.	DBLP database.	Database exists / has tables with relations.	Yes.
2.	Stop words file.	Load the Stop words into B-Tree.	B-Tree B.
3.	Tables of DBLP.	Get the first Table.	Table: Research Papers with about 800,000 tuples.
4.	Research Paper Table.	Get first Filed of table.	Filed: Title.
5.	Filed: Title from table Research Papers.	Run Tokenize function for each tuple in Title.	List of words
6.	Title: "Artificial Agent Societies Structure and Its Implications for Autonomous"	Run Tokenize function for each tuple in Title.	[Artificial] [Agent] [Societies] [Structure] [and] [Its] [Implications] [for] [Autonomous]
7.	[Artificial] [Agent] [Societies] [Structure] [and] [Its] [Implications] [for] [Autonomous]	Compare with B-Tree B and drop Stop words.	[Artificial][Agent] [Societies][Structure] [Implications][Autonomous]
8.	[Artificial][Agent] [Societies][Structure] [Implications][Autonomous]	Store in B-Tree KB with no repetition of keywords.	B-Tree KB with keywords and counts for each keyword for the first filed of the first table.
9.	Rest of tuples in the Title field.	Repeat steps (6,7,8).	B-Tree KB with complete keywords for the field Title in the Research Paper table.
10.	B-Tree KB	Store B-Tree KB in the index keywords table.	K: index keywords table generated and contain keywords for field Title in the Research Paper Table.
11.	DBLP database.	Repeat steps from 3 to 10.	K: index keywords table generated and contain keywords for the whole DBLP database.

3.3.2 Online building index keywords table

By this, we mean the running of the index keywords builder while the database is online and operational, which introduces new challenges to the index keywords builder as follows:

- The time to run the index builder through the whole database.
- The performance of the database while running the index builder.
- The updated and new records in the database while the index keywords builder running.
- The new tables introduced to the database while running.
- The follow up of errors of the index keywords builder while the database is running.

Those challenges need to be more investigated in future work, as in this thesis we used the offline mode of running the index keywords builder.

We can use module 1 and algorithm 3.3.1 in online mode for generating the index keywords table, but we need to add extra conditions and functions to be able to overcome the challenges of building the index keywords table in online mode.

3.3.3 Updating index keywords table

Another alternative way of building the index keywords table is by updating the index keywords table through the search of keywords, which means that the index keywords table are updated as users search for information from the database and each search term entered by users and found in the database through different records band CN's.

By using the update index, keywords table algorithms have advantages and disadvantages as follows:

Advantages:

- The index keywords table would have a small size and number of records, which leads to more performance for the keyword search.
- The index keywords search only contains keywords that users search for, which means less CN's generated and high performance.
- As the index keywords table updated with each search term entered by users, the performance of the database would not be affected, not like using the online mode for creating the index keywords table.
- The offline mode for creating the index keywords table needs complex operations and needs to have the whole database in offline mode to be able to create the index keywords table.

Disadvantages:

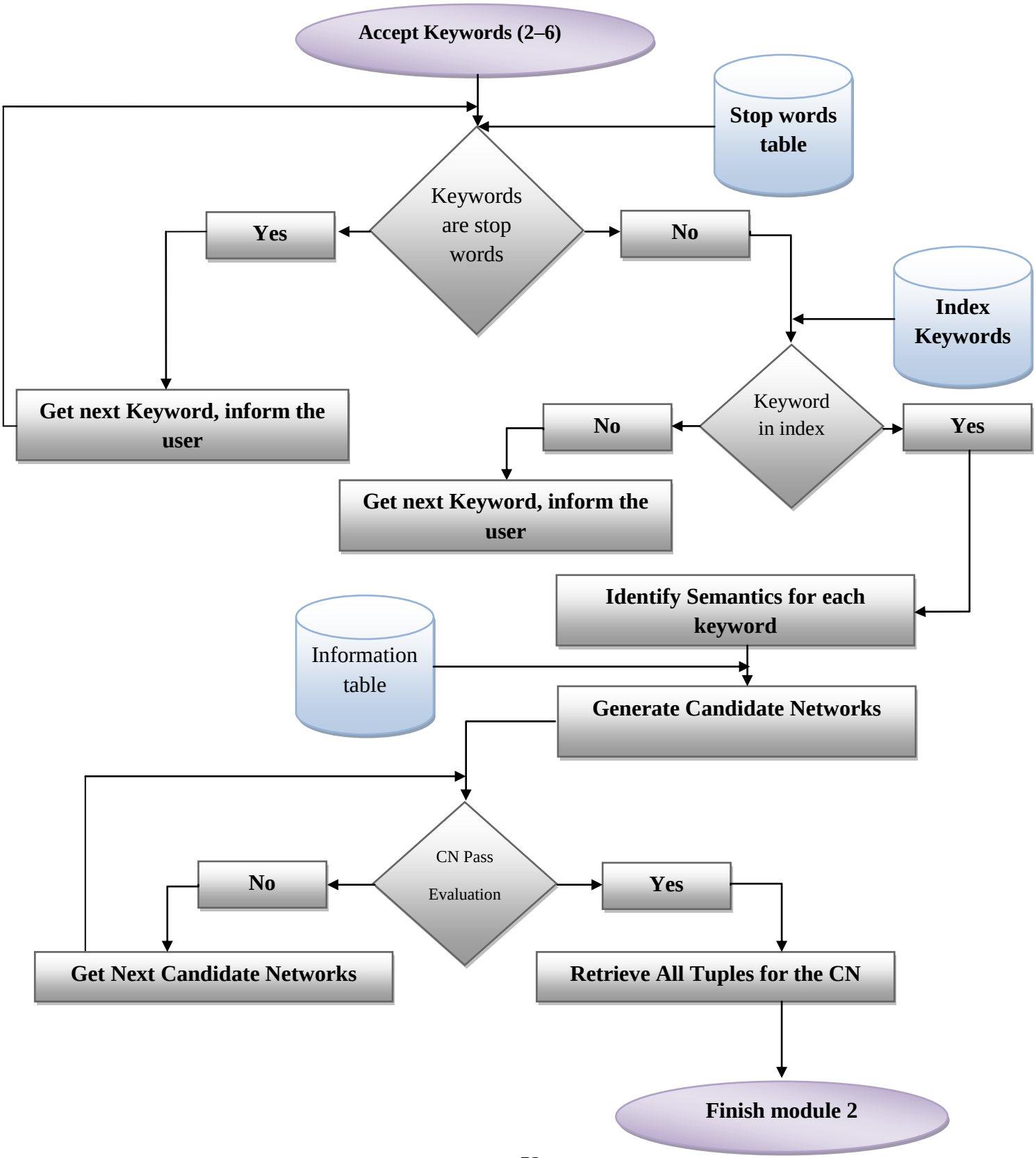
- For new keywords not found in the index keywords table, the performance of the search would be low , as the CN generator needs to check for all CN's in the database.
- For databases with high update rate and search, the updating of the index keywords table would affect the performance of the database access and update.
- The index keywords table may not represent the whole database and information retrieval and generating of results may be affected.

3.4 Candidate Network Generator (CN)

Based upon the creation of the index keywords table and the information table, the CN generator can generate the CN's that only contain keywords in each node of CN and the weight of each CN would be greater that a weight factor, which would be explained in the next section.

The following module (module 2) shows how the CN generator works based on the index keywords table and information table.

Module 2: Semantics of Keywords and Generation of Candidate Networks.



3.4.1 Candidate Network Generator Algorithm

Algorithm 3.4.1 shows the CN generator work and pruning process of networks that do not meet the conditions proposed by our search.

Let Q represents the search term entered by the user, Q_i represents tokens generated from Q , to generate all possible CN's, we must eliminate all stop words and query the reset of keywords in the index keywords table to have information regarding each keyword, after that the algorithm combines all keywords with the same information (table name, field name) as a sub search term, and according to all sub search terms and from the information table we can determine all the CN's that could have tuples corresponding to our query.

To determine the final CN's, our algorithm uses two phase check , which means that the algorithm eliminate CN's in two steps depending on the next conditions:

Condition 1: All CN's contain at least one keyword.

Condition 2: CN's are ordered according to number of keywords contained, and least number of keywords in any CN would be eliminated by a weight factor.

Example: Search term entered by user “john network parallel algorithm”.

Keywords entered by the user = 4.

Table 3.4.1.1 Example of CN generated for four keywords entered by user.

CN	Number of keywords
CN1	4
CN2	4
CN3	3
CN4	2
CN5	2
CN6	1
CN7	1

In table 3.4.1.1, we can see that we have 7 CN's, which are generated according to the index keywords table and the information table. And in table 3.4.2.1 we can see the location of each keyword, that are combined together by the CN generator algorithm based on the information table to formulate all possible CN's and the number of expected number of keywords in each CN.

Definitions:
Q: query term entered by the user.
Ki: keywords extracted from Q.
Qi: represents the keywords to search for.
Gk: set of keywords in same table and filed.
CN: represents CN's generated.
Wt: weight factor for pruning CN with least keywords.

Algorithm 2: **Candidates Networks Generator**

Algorithm CN_Generator
Input: set of keywords entered by the user, Database
Output: CN's represents the query search entered by the user.

```
1  Begin
2  Load ST in B-Tree
3  Load information table in  array info_T
4  Generate_keywords(Q,Ki)
5  For each k in Ki  do
6       IF ( k in ST )
7            // discard the keywords  / stop keyword
8       Else
9            Store k in Qi
10     End For
11  Construct_Sql(Qi,Sql_string,K)
12  //get all keywords from index keywords table
13  Retrive_data(Sql_string , K , keys_info)
14  For each k in Keys_info do
15       group_k(k, Keys_info,G_K)
16  End for
17  For each Gk_i in G_K do
18       For each Gk_j in G_K do
19            Genrate_CN(Gk_i , Gk_j , CN )
20       End for
21  End for
22  For each CN_i in CN do
23       IF (CN_i < Wt )
24            Prune CN_i
25            count_pruned ++
26       Else
27            Store CN_i
28            count_CN ++
27            Genrate_Sql(CN_i , CN_Sql )
28            Execute( CN_Sql , Results )
29  End algorithm
```

3.4.2 Evaluating and Pruning CN's

The limitation factor expresses the lower bound that the CN generator can prun CN's less that it, as these CN's contain results, but the results would not appear in the Top-K.

For example:
The search term entered by the user is " john network parallel algorithm "

The index keywords table returns the following results:

Table 3.4.2.1: Results returned from the index keyword based on the user search term.

Keyword	Occurrence	Table name	Field name
John	**>3000**	persons	Name
Network	>4000	Papers	Title
Network	>300	Proceedings	Title
Network	>200	Journals	Title
Network	>10	Master thesis	Title
Network	>20	Books	Title
Parallel	>200	Papers	Title
Parallel	>100	Proceedings	Title
Parallel	>50	Journals	Title
Algorithm	>1000	Papers	Title
Algorithm	>150	Proceedings	Title

In the CN "Persons" according to table 3.4.1.1 it only contain one keyword "john", and the number of keywords are 4, so the weight of this CN is ¼ = 25%, so this CN would be pruned, and many others also, and just leaves the CN with bigger weights, but the CN "Persons To Papers" contains all keywords, so the weight of this CN would be 100%.

In SPARK [7], the Prune process needs to construct the corresponding SQL statement for each CN, then prop the data base for each CN, and according to the return results, the CN's

with no results returned are prune, and only CN with each node in the network contains at least on keyword, for our work the prune of CN's are done before the prop of the database, as we already know from the index keywords table each keyword, where it is found in the database, and according to that we generate only the CN's containing keywords in each node.

Assume in SPARK [7], N donates number of CN's in the database, T_p: time to prop the database for each CN, T_t: total time of propping the database for each CN, M: number of CN's after pruning.

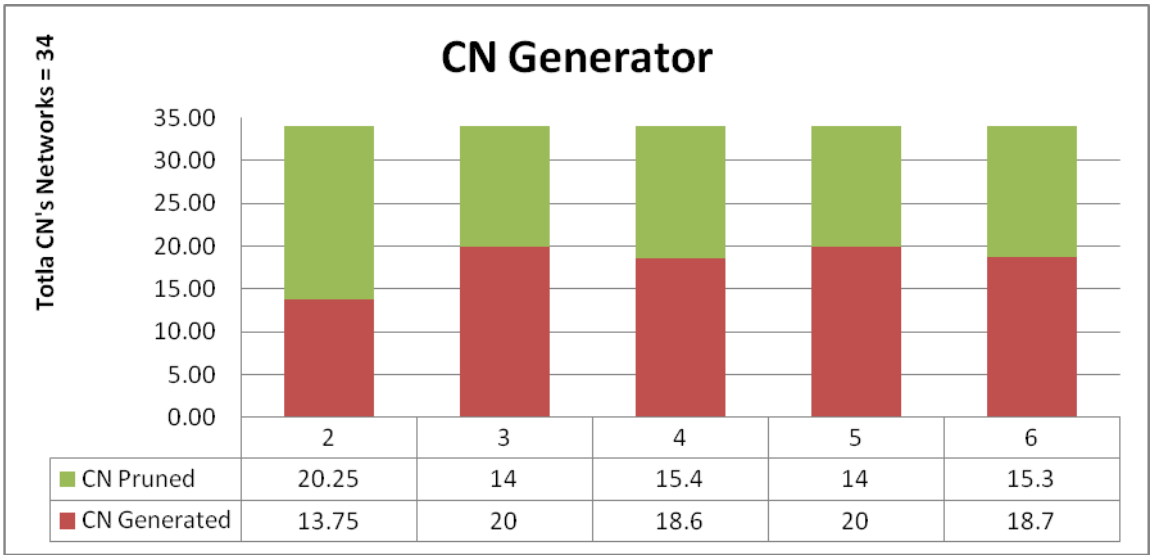
In SPARK [7]: Time are fixed are $T_t = T_p * N$ for each query search.

In our work: Time vary according to the query term entered by the user: $T_t = T_p * (N - M)$, which at worst cases when no CN pruned, M = 0, gives the time as SPARK [7].

The following table shows the results of running all the search terms as in appendix A, which shows the pruning of CN's for each search term size.

Table 3.4.2.2: CN's pruning.

Number of keywords in the search term	CN's generated	CN's Pruned	Total CN's in database
2	13.75	20.25	34
3	20	14	34
4	18.6	15.4	34
5	20	14	34
6	18.7	15.3	34
Average	18.21	15.79	34
Percentage	53.56%	46.44%	100%



	2	3	4	5	6
CN Pruned	20.25	14	15.4	14	15.3
CN Generated	13.75	20	18.6	20	18.7

Figure 3.4.2.1: CN generator results.

From table 3.4.2.2 and figure 3.4.2.1, we can see that our CN generator algorithm eliminate 46% of overall CN's, which leads to decrease in database propping about 46%, and increase in the performance of IR by implementing the CN generator based on the index keywords table introduced in our work.

In SPARK2 [17], the CN pruning eliminate just 40% of CN's after propping the data base, in our work we do not need to prop the database to eliminate CN's, in the second stage of pruning the CN's in our example needs to prop the database for the remaining of CN's and also prune in average for all search term sizes about 20% by removing the CN's contains the least score for all keywords entered by the user.

The Average pruning vary according to the size of search terms (number of keywords), and we can found that when the number of keywords entered by the user increases; the number of CN's increases and the number of pruned CN's decreases in general.

3.5 Ranking Style

Our new ranking function ranks each returned tuples based on the following factors:

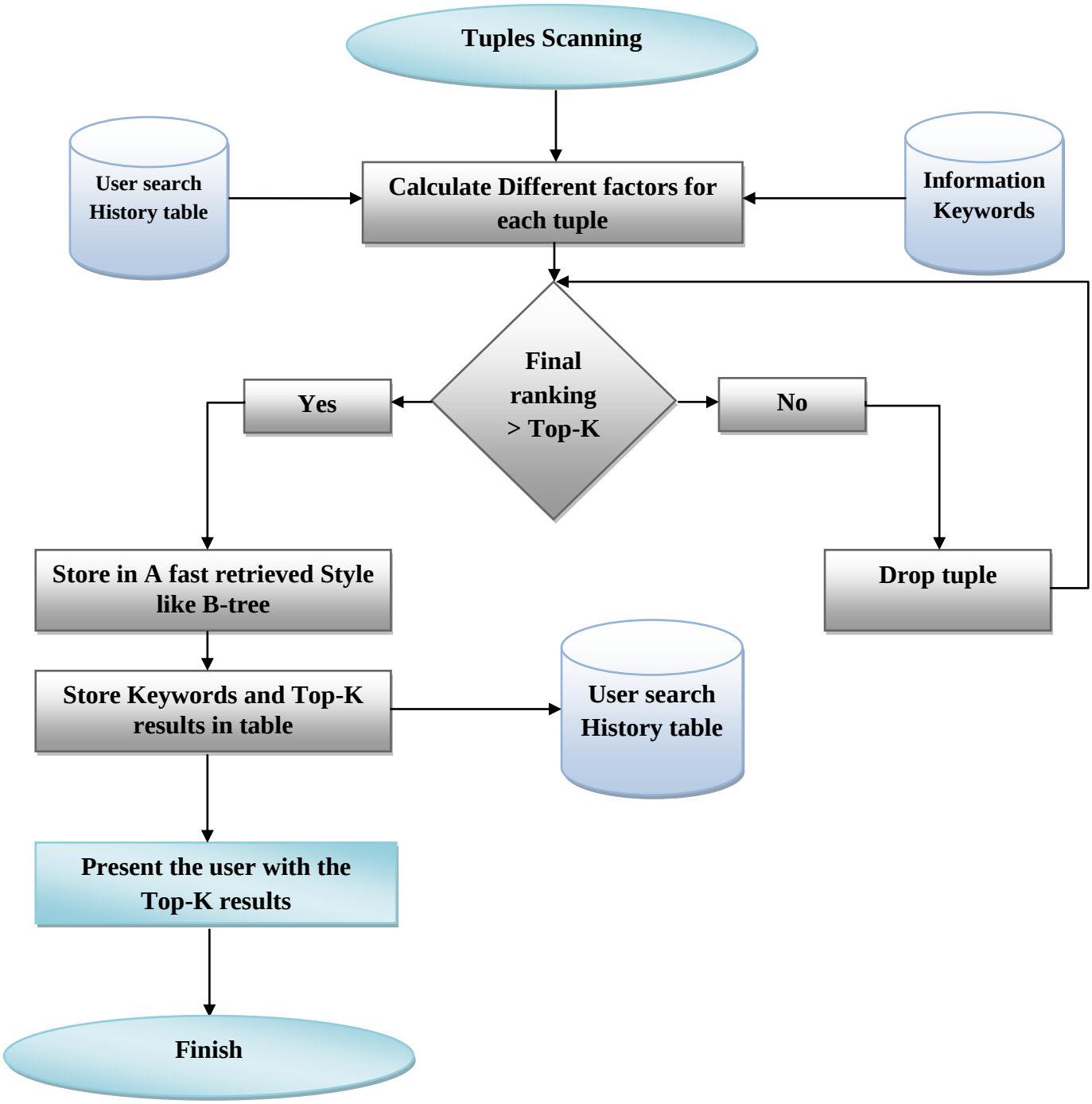
- Weight of match function provided in MySQL as a base score for each tuples.
- Semantics of keywords entered by the user, the order of keywords against tuples retrieved.
- The distance between the keywords (number of words between the keywords) in retrieved tuples, in the same tuple, relation with one degree, relation with degree two, and so on.
- The search history, users clicks of returned results.
- Number of keywords in each tuples against number of keywords entered by the user.

We evaluate each factor and give a weight to the final rank for each tuples, and the following algorithm shows how the algorithm works and the ranking of results.

Our ranking formula is similar to [16,9,12], which is based on the factors mentioned earlier, and the following code represents the final score for each result retrieved to be presented to the user.

The following module (module 3) shows the work of our ranking style, which gives the final score for each returned result and rearrange the results to represent the user with the top-K results.

Module 3: Ranking style for retrieved tuples.



Definitions:

Cn_i: Represents a CN from the Total CN's generated to specific Query entered by the user.

Lr: List of results, ordered by rank and limited by Top-K.

TLr: Temp list to store the data before ranking.

T: Tuple retrieved for specific CN.

Score: Score by the MySQL engine for each result.

Weight_order: Evaluates each tuples with the order of keywords entered by the user.

History_order: Retrieves the user clicks for same result and convert to a weight for ranking.

Count_keywors: Represents number of keywords in T_i match keywords entered.

Algorithm 3: Ranking Function

Input : Results for CN_Generator
Output : Top-K results
1 Begin
2 For each Cn_i in CN do
3 //execute Sql for the CN and store data in
4 //List with ranking each result
5 Retrive_data(Cn_i, TLr)
6 For each T_i in TLr do
7 rank_1 = score(T_i) / number_keywords
8 rank_1 = rank_1 + weight_order(T_i)
9 rank_1 = rank_1 +history_order(T_i)
10 rank_1 = rank_1* ( )
11 // store result in list and rearrange
12 // list according to rank and top-k
13 Posh(T_i , rank_1 , Lr, Top-K)
14 End for
15 End for
16 Show Lr to user
17 Listen to user interaction
18 Store user actions.
19 End

3.5.1 Scoring of matched results

By using the match function provided by the MySQL and PHP, the returned score for matched results is considered the base for our ranking style, as the score of match results returned for each tuples in each CN, and to be able to clarify the match score. The following examples show the score of different match results to certain keywords search terms entered by users.

Example: The search term entered by the user " hall software engineering ", The final CN's after the elimination of other CN's according to weight factor are shown in table 3.5.1.

Table 3.5.1: Example of scoring on matched results according to keywords.

CN	Keywords found	Score CN	Final score
Persons	Hall	1/3	0.33
Papers	software engineering	2/3	0.66
Persons → Papers	hall software engineering	(1/1 + 2/2) /2 = 3/3	1
Journals	software engineering	2/3	0.66
Books	software engineering	2/3	0.66
Proceedings	software engineering	2/3	0.66
Papers → Proceedings	software engineering software engineering	((2/3) + (2/3))/ 2	0.66
Papers → Journals	software engineering software engineering	((2/3) + (2/3))/ 2	0.66
Proceedings → Journals	software engineering software engineering	((2/3) + (2/3))/ 2	0.66

By table 3.5.1 we can see the expected score for each CN for the given keywords, which means if we have a record in CN (Persons → Papers) contain "hall" in persons table and related to "software engineering" in table papers, then the final score for the match score is 1.

3.5.2 Scoring of order Semantics

The function of scoring the order of semantics for the results generated from the CN's, returns a weight factor to be added to the ranking function, which is shown as follows:

Example: The search term entered by the user "analysis software engineering ", The scoring of the order semantics are showing in table 3.5.2.

Table 3.5.2: Scoring of order semantics of results returned to the user.

Result	Score result	Final score
Research on Software Engineering Requirement Analysis Method Based on Five-Key Elements Arrange.	2/3	0.66
Advances in engineering and software analysis and their relations to pattern recognition and image processing.	0/3	0
Software Engineering with Formal Methods: The Development of a Storm Surge Barrier Control System Revisiting Seven Myths of Formal Methods	2/3	0.66
Using Formal Analysis in software engineering Techniques in Business Process Redesign	3/3	1
Using Formal Analysis in engineering software Techniques in Business Process Redesign	1/3	0.33

By table 3.5.2 we can see that the function returns 1 for the same order of keywords entered by the user and found in the same order in the returned results, as users order important for the user and want to see the same keywords in the text with the same order, if the order of keywords are different in the returned result, the function calculates the percentage of order of keywords to give a final weight for each result.

3.5.3 Scoring of users history

The user's interaction with returned results would have an effect on the ranking of results and Top-K results, as results with high number of clicks by users for the same search term would have a high score in this section .The following table 3.5.3 shows the scoring function of the user history:

Table 3.5.3: Top-5 results returned for the search term "Business Process Redesign DATA BASE".

Result	order	Final score	Hits
[From Result] Using Formal Analysis Techniques in Business Process Redesign. [To Result] Business Process Management	1	1.555555556	251
[From Result] An MCDM based interactive support system with application to business process redesign. [To Result] Business Process Re-Engineering	2	1.555555556	327
[From Result] Using Formal Analysis Techniques in Business Process Redesign. [To Result] Business Process Management~ Models~ Techniques~ and Empirical Studies	3	1.555555556	105
[From Result] An MCDM based interactive support system with application to business process redesign. [To Result] Business Process Re-Engineering: Information Systems Opportunities and Challenges~ Proceedings of the IFIP TC8 Open Conference on Business Re-engineering: Information Systems Opportunities and Challenges~ Queensland Gold Cost~ Australia~ 8-11 May~ 1994	4	1.555555556	70
[From Result] Business Process Redesign and Information Architecture: Exploring the Relationships. [To Result] DATA BASE	5	1.555555556	131

By table 3.5.3, we notice that results 1 and 2 have the same score, but we can see that number of hits for the second result is bigger than hits for the first result, and if any new search for the same search term, then the second result would be ranked and become the first result.

The effect of the user history is limited, as it only can move any result one order forward, and cannot move the result number 5 to be result number one.

3.5.4 Scoring of keywords distance

This function weights the distance between keywords in any given results, as long as the result scores in the order of semantics is 1, which means that all keywords are arranged in the same arrangement as the user input them.

The calling of this function is just for the highest ranked results by the previous functions which are limited to a small number of results to be evaluated by this function. Table 3.5.4 illustrates the output of this function.

Table 3.5.4: Distance evaluation of results for the search term "analysis software engineering".

Result	Score of number of words	Final Score
Analysis Research on Software Engineering Requirement Analysis Method Based on Five-Key Elements Arrange.	Words between keyword 1 and 2 + words between keyword 2 and 3 = 2.	1- (2/8)
Analysis in pattern recognition and relation to software development and engineering.	Words between keyword 1 and 2 + words between keyword 2 and 3 = 6 + 2 = 8.	1- (8/8)
Analysis of Software Engineering with Formal Methods:	Words between keyword 1 and 2 + words between keyword 2 and 3 = 1.	1- (1/8)
Using concept "Analysis software engineering".	Words between keyword 1 and 2 + words between keyword 2 and 3 = 0.	1

Here the function calculates the weight by two functions:

1. If the score of number of words = 0, then the final score of keywords distance for the result = 1.
2. If the score of number of words > 0, then the final score = 1- (score of number of words / maximum score of number of words).

CHAPTER FOUR: SMART SEARCH IMPLEMENTATION & TESTING

4.1 Introduction

In this chapter we introduce the Smart Search interface, which a web based interface page that enables users to interact with our search for Top-K keywords search, the interface is designed to be used through the web by using PHP code to build the different algorithms and components of the search, and to be used over different platforms to serve multiple users and publish the search over the web.

Also we show the different components of the smart search, which were designed to record the interaction of users with the interface for the purpose of setting up a testing environment to collect all the necessary data to be able to analyze and compare with other researchers.

For the purpose of testing our algorithms introduced with the new components, we show in this chapter the steps of designing the testing environment, which include the users used for testing the Smart Search, the search terms generated randomly, the specifications of hardware and software used for testing and data gathered by the system for further analyses.

We elaborated how the Smart Search Work by using a scenario with an input search term and the final results with all the stages and algorithms used to generate the final results for the user.

4.2 Building the 'Smart Search' interface

For the purpose of testing the new introduced components, CN generator and ranking style, we developed a web interface page and published online, to enable different users to interact with the smart search and monitor and record all users' interactions.

The following components and software were used in developing and running the smart search interface:

- Laptop computer with quad core CPU of 2.1 GHZ and 4 G.B of Ram.
- An operating system of windows 7 professional.
- Internet information server 7 as web server.
- Php compiler and libraries version 5.
- MySQL 5.5 as database management system to manage the DBLP database.
- Web development software's for PHP, JavaScript and html web pages.
- Network infrastructure and at least 5 computers connected to the network.

We developed the web interface using html for the interface and PHP code for the functionality and implementing the three modules for creating the index keywords table, CN generator and ranking style. The interface also includes functionality for recording users' interactions and different time intervals to be analyzed and compared with others.

4.3 Components of 'Smart Search'

In our research, we developed online web based interface to search with Top-K keywords search on DBLP2 using our proposed algorithms for IR and Ranking style.

The smart search interface provides the user with the following functionality and options:

1- Interactive search box: Free form search box that accepts words with the condition that number of keywords is between 2 and 6, with the ability to connect to different searchable fields in the database with the auto complete option, to help the user specify the search term for more accurate search.

2- Search type: By default, the search type would be – fast search - which lets the search engine check for the search history and retrieve the results from the search results table without the need of CN generation or ranking styles. The other type is

the slow search, which forces the search to perform the full search algorithms to retrieve the results.

3- The database option: Lets the user specify the database that he wants to search.

4- The auto complete option: Enables the user to choose the field to link to the search box for listing data while the user is typing his search term, which helps the users in writing, especially users who are weak in English.

5- About page: Information about the 'Smart Search' web page, algorithms used and development.

6- Help page: Illustrates how to use the 'Smart Search' interface.

7- User history statistics page: For statistics about the search terms entered by users related to efficiency and effectiveness.

8- Top-K statistics page: Shows the users' interaction with the search and users options.

9- Documentation page: Lists all files related to the research paper and master thesis.

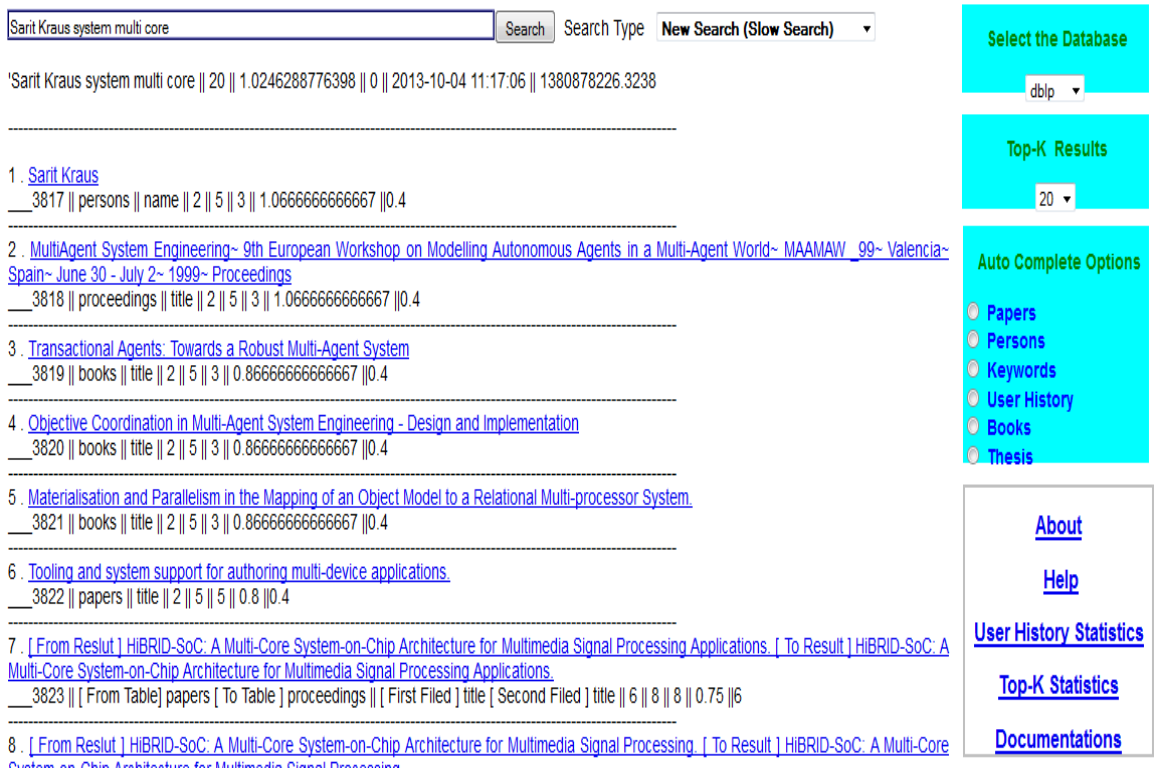


Figure 4.3.1: Snapshot of Smart Search interface.

4.4 Working Flow of 'Smart Search'

In this section we explain the functionality of our Smart Search web interface by using real examples and show the output at each stage, and the final output according to the search term entered by the user.

4.4.1 Scenario Example:

User enters the query search term "**Yikui Zhang Markov networks**".

The following steps show the process of 'Smart Search' at each stage, the input, the process and the output:

Table 4.4.1: Example search and the 'Smart Search' interaction with the user.

Action / Algorithm	Input	Output
The tokenize algorithm	Yikui Zhang Markov networks	Array of 4 words [Yikui , Zhang , Markov , networks]
The elimination of stop words	[Yikui , Zhang , Markov , networks]	No stop words all words are keywords.
The verifier of number of keywords	Length of array	Yes within range 2-6
The prop to the index keywords table	[Yikui , Zhang , Markov , networks]	Yikui: in the persons table, in field name. Zhang: in the persons table, in field name. Markov: in the paper table, in field title. Networks: in the paper table, in field title.
The construct of search terms according to the results of index keywords table	[Yikui , Zhang , Markov , networks]	1: Yikui + Zhang → persons table. 2: Markov + Networks →papers table.
The load of information table	---	Information table stored in array.
The construct of CN's according to the information table	1: Yikui + Zhang → persons table. 2: Markov + Networks →papers table.	CN1: persons table. CN2: Papers table. CN3: Persons → Papers.
The elimination of CN's according to the factor weight	Factor weight = 30%	CN1: pass. CN2: pass. CN3: pass.
The construct of SQL statements for each CN	CN1, CN2, CN3	Three select statements with conditions that they contain the keywords.

The prop of database for each SQL statement and retrieve the highest scores	Three SQL statements correspond to each CN.	Hundreds and thousands of tuples that satisfy the SQL statements.
The ranking of each result	All tuples.	All tuples with ranking score for each tuple.
The sorting of results according to the ranking	All tuples.	All tuples ordered according to the highest ranking score.
The output of results.	All tuples ordered according to the highest ranking score.	Only the Top-K results shown to the user.
The recording of user interaction with results	Clicks, navigations, time intervals.	Stored data in tables.

4.5 Setting up the testing environment

To be able to test our 'Smart Search', we set up a testing environment with the following specifications:

- One computer to hold the database and the web interface for 'Smart Search'.
- Four computers connected through a local network with the main computer that runs the 'Smart Search'.
- Four users to run the search terms as shown in table 4.5.1.
- Random generation of search terms by developing a special function to randomly generate search terms with different lengths from various table in the database.
- Random distribution of search terms across the users.

For the purpose of comparing our results with others , we show the testing environment used in SPARK[7], SPARK[17] and Indexing[20] compared with our testing environment, and the following table shows the different parameters used for preparing the testing environment.

Table 4.5.1: Testing environment parameters and components used.

Parameter	Our work	SPARK[7]	SPARK[17]	Indexing[20]
Web Server	Yes	No	No	No
Computers	Five computers, one as web server and four as clients.	One computer	One computer	One computer
Computer specifications	Intel Core I3 with CPU 2.1 GHz, Memory 2 GB.	Intel 1.8 GHz with 512 MB Memory.	Intel 1.8 G.H with 512 MB Memory.	Intel Dual Core 2.0 GHz with 1 GB Memory
Software	IIS web server for publishing the system, PHP code for functionality, MYSQL for DBMS	JDK (Java) connected directly through JDBC to the database.	JDK (Java) connected directly through JDBC to the database.	Java programming for implementation of algorithms.
Network	Yes	No	No	No
Direct connect to database	No	Yes	Yes	Yes

By table 4.5.1, we can see that our computer specifications are higher, but we build our system for web publishing and all testing were done through clients connected through a local area network to a web server that hosts the database and the web pages, which would increase the time of execution.

The preparing of the experiments focuses on two main issues:

1- The accuracy of our proposed algorithms in IR and ranking style , which would be determined by real users working online on the Smart Search to evaluate the returned results, and the system records all actions done by users to be analyzed.

2- The performance of our proposed search algorithms with the new introduced tables to the database, which would be evaluated by comparing search times with other researchers with the same environment.

Length of search terms entered by users (2 - 6):

After looking into famous search engines statistics about search terms entered by users, the research done in 2011 titled " *An analysis of web proxy logs with query distribution pattern approach for search engines* ", we can Cleary find that most of the search terms sizes are between 2 – 6 keywords, with a percentage more than 80%.

The 'Smart Search' does not allow the use of one keyword for search or more than 7 keywords, and here are the reasons for using just 2 – 6 keywords:

- One keyword can generate huge number of results and tuples, which cannot be ranked by the top-k keyword search ranking style.
- The search can only return top-k results, not like conventional search engines, which return all results in paging system, but our work just returns the k results, which cannot be more than 20 results.
- The search in databases relay on CN's and tuples according to relations between tuples and fields, which is not found in web search engines that do not build the results according to relations, in relational database we need at least two keywords from different tables to build CN's and returned tuples.
- As the users using the free form search on relational databases, they do not know the database structure of the existence of information, as they are regular users with limited knowledge of search styles, which lead to a query search term that is shorter than six keywords.
- Many research papers and statistics about the big search engines show that more than 80% of search terms contain between 2 – 6 keywords.

Table 4.5.2: Search term size and numbers.

Number of keywords in the search term	Number of search terms
2	86
3	26
4	77
5	126
6	76
Total	391

Table 4.5.2 shows the number of search terms for each length and the full search terms lists are found in appendix A.

Each user was given a number of search terms from different sizes to work on and evaluate, and the following table shows users and search terms.

Table 4.5.3: Distribution of users and search terms.

users	Number of search terms
User1	78
User2	78
User3	78
User4	78
User5	79
Total	391

Table 4.5.3 shows number of search terms tested for each user with different sizes.

CHAPTER FIVE: RESULTS ANALYSES & CONCLUSION

5.1 Introduction

5.2 Gathered Data

5.3 Efficiency

5.4 Effectiveness

5.5 Conclusion

5.6 Future work

5.1 Introduction

This chapter is dedicated for analyses of gathered data and comparing our results with others that work on Top-K keyword search over relational databases. The gathered data generated by the testing phase contains thousands of tuples stored in the database that records all users input and the system output, with indication to the performance of each introduced components and algorithm, like the execution time for each search term regarding the access of the index keywords table, the generation of CN's, the elimination of CN's, the retrieving of results for all the remaining CN's and finally the ranking of results to present the user of the best choices.

The process of gathered data concentrated on data gathered related to efficiency and effectiveness, to be able to compare our results with SPARK [7], indexing [20], that worked on the same database and generated results of different algorithms related to the performance of their systems, which they measured based on execution time.

The other comparison part done is the effectiveness of the introduced components and algorithms for retrieving results, which represents the precision of the returned results according to the user, that we measured based on the user choices and done manually by each user for each search term entered.

Our data presented in tables and figures in this chapter where compared to SPARK2 [17], to measure the precision of the returned results.

5.2 Gathered Data

For the purpose of analyzing and comparing our results through the online web interface ‘Smart Search’, different data were stored in tables generated for this purpose, which are updated for each search term entered and user clicks, and the following shows the tables and illustrate the need for each one in our work.

Search Sets

A table containing all search terms entered by users with information for testing purposes, to be able to analyze our work and algorithms, and can be used for future improvements to our work.

The table was also used in fast retrieval of results. If we used the same search term which can be seen in table 5.3.1 and speed up performance to about 70 times faster, we then use the search sets table and the search results table.

search_term	top_k_time	over_all_time	fast_time	keyword_time	cn_count	keys_count
'Polynomial Restrictions || Principles and Pract...	0.147235155	0.2906091213	0.118480920	0.00122690200	12	4
'Algebraic System Specification and Development -...	0.739551067	0.9074008464	0.108247041	0.00140786170	20	5
'Algebraic System Specification and Development -...	0.743494987	0.8745729923	0.078464984	0.00209593772	20	5
'Polynomial Restrictions || Principles and Pract...	0.438837051	0.4419939517	0.052531003	0.05069899559	12	4
'A Unified Approach to Interior Point Algorithms	0.645830154	0.6514549255	0.044208049	0.00105285644	17	5
'International Journal of Geographical Informatio...	0.919618844	0.9272630214	0.041142940	0.00181794166	26	5
'Foundations of Databases.	0.447949171	0.4512090682	0.039222955	0.00086784362	16	2

Figure 5.2.1**:** Snapshot of search_sets table

In figure 5.2.1 we can see the search sets (search terms entered by users) and the different fields recorder for each search term like the overall execution time, the fast time execution, the time for retrieving information from the index keywords table and the number of CN's generated for each search term.

Search Results

A table containing the search results related to a search term entered by users, also the related information for each result like the final rank for each result, the order of the result, the relation the result taken from, the CN that generated the result and all the results are used in the analyzing phase and for future improvements of the CN generator and ranking style used.

result	order1	hit	table_name	col_name	score	rank1
Concepts~ Design~ and Performance Analysis of a Pa...	1	1	books	title	4	2
[From Reslut] Multiprocessor and Distributed...	2	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] conferen...	4	1.57142
Intentional learning in an intentional world: new ...	3	0	papers	title	3	1.5
Orthonormal integer block transforms for lossless ...	4	0	papers	title	3	1.5
[From Reslut] Timed compiled-code simulation...	5	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] title1	4	1.5
Concepts~ Design~ and Performance Analysis of a Pa...	1	1	books	title	4	2.2
[From Reslut] Multiprocessor and Distributed...	2	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] conferen...	4	1.57142
Intentional learning in an intentional world: new ...	3	0	papers	title	3	1.5
Orthonormal integer block transforms for lossless ...	4	0	papers	title	3	1.5
[From Reslut] Timed compiled-code simulation...	5	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] title1	4	1.5

Figure 5.2.2: Snapshot of search_results table.

In figure 5.2.2 we can see the returned results and the rank of each result for search terms entered by the user, also the system store the order of each result and the users hits.

User History Table

The existence of user search history is important to the user, as most popular search engines like Google present the user with search history with number of expected results, which help the user in typing the keywords and auto complete the search terms for the user, as an important supporting tool for users with some weakness in writing some English words in the domain of computer research fields.

With this table the user will not be worried about how to spell the keywords, as the search engine helps him with the auto complete tool. Also, the search history of set of keywords and users interaction with results would be an important factor in ranking the results for the user, as the user history table keeps track of hits number for top-K results, which makes it an intelligent system that based the ranking style according to learning mechanism through different user's inputs and hits.

5.3 Efficiency

Efficiency in our work is expressed by recording time execution for different stages, and calculate the time for each search term with different lengths, also as we conducted several hundreds of search terms as shown in appendix A, and each search term or query set where executed for different top-K as follows:

- Top-5
- Top-15
- Top-20
- Top-30
- Top-50

The need to execute our search terms for different top-K is to be able to compare our results with other researchers, and also to be able to analyze the performance and effectiveness of our work according to different top-K results.

In our approach of measuring the performance of our Top-K keyword search with the new introduced components and algorithms, we record three different time execution for each search operations:

1. Time to get keywords from our index keywords table as a new introduced component in our Top-K keyword search over relational databases.
2. Time for the full search and displaying the results for the user.
3. Time of cashed results retrieved to the user according to the search history.

The following tables shows the three different execution times which we will use to compare our outcomes with SPARK [7], and see the performance of our SSearch according to the performance of the BP, SS, Sparse, and GP algorithms in SPARK [7] on the same database, but the total tuples in DBLP used in SPARK [7] are (881,867) comparing to our DBLP tuples (3,707,892).

Table 5.3.1: Size of search term Vs. average time in normal search (full search).

Number of keywords in the search term	Average time normal search (ms)
2	**393**
3	**513**
4	**571**
5	**623**
6	**698**
Average	**560**

In table 5.3.1 we can see clear that our system achieves an average execution time for all search terms with different sizes = 560 ms, and the biggest time is when number of keywords in the search term = 6, which is normal as the number of CN's increases when we have more keywords.

The following figures show the execution time (performance) for 20 query sets chosen from the gathered data as an output from our testing phase.

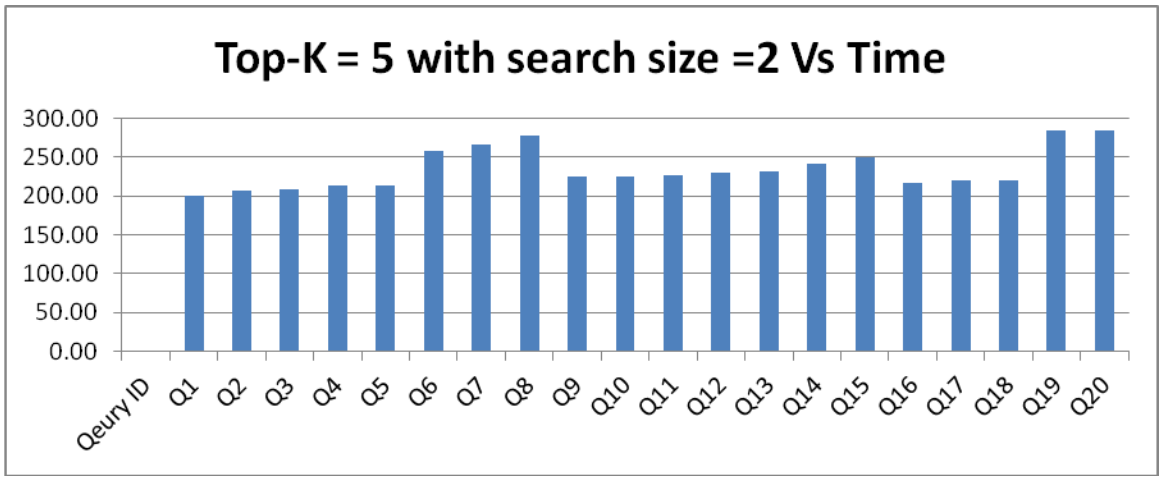


Figure 5.3.1: 20 search terms with size = 2 Vs. time.

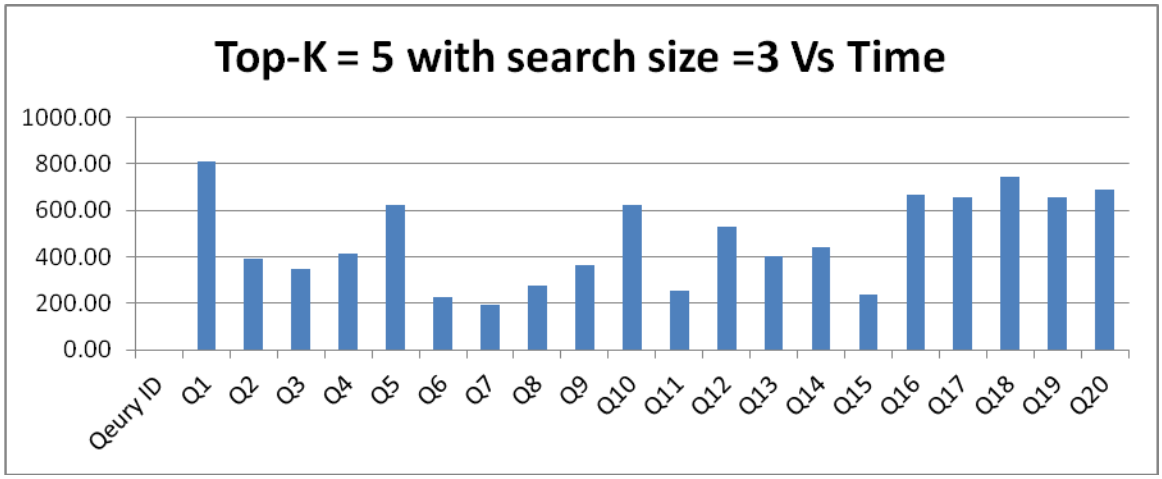


Figure 5.3.2: 20 search terms with size = 3 Vs. time.

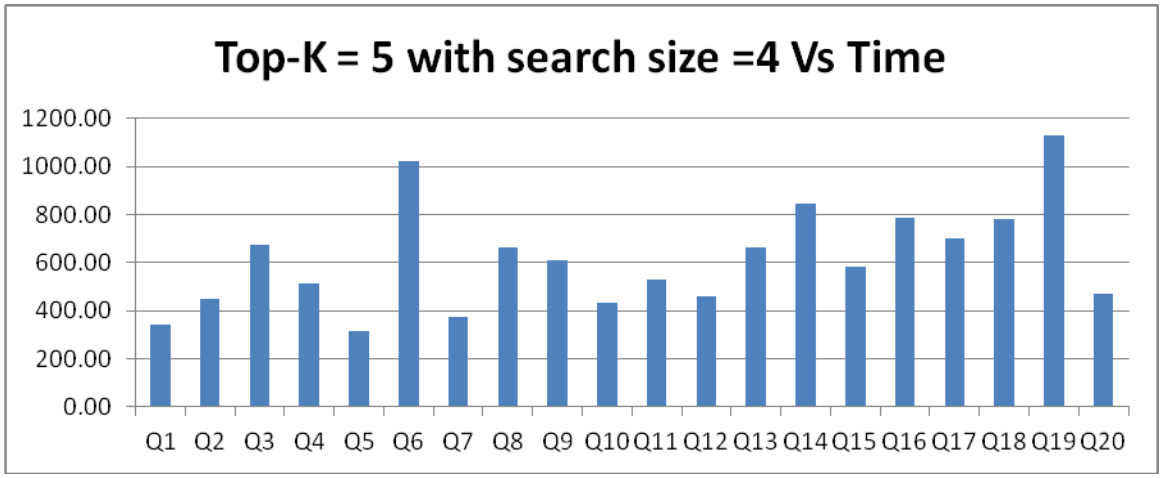


Figure 5.3.3: 20 search terms with size = 4 Vs. time.

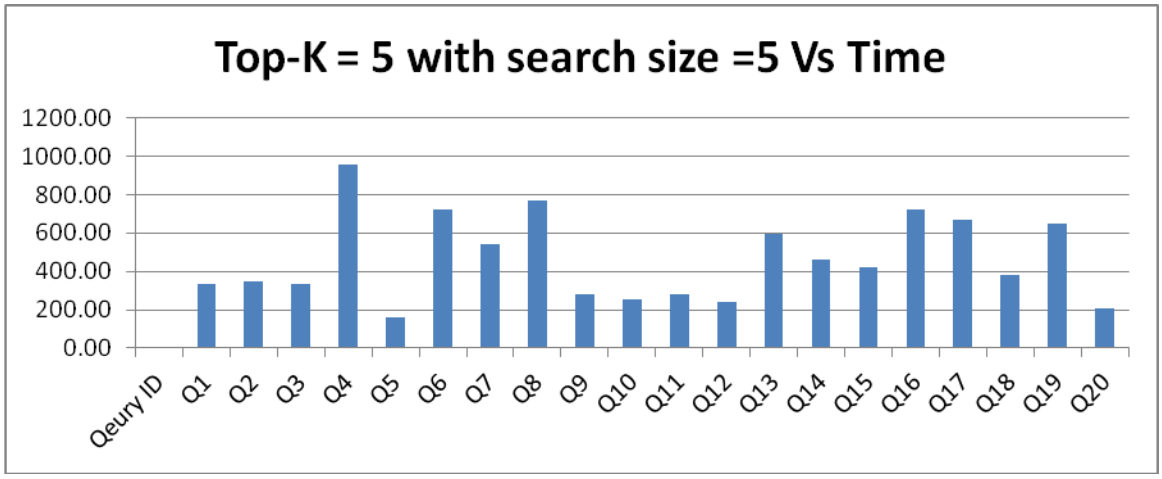


Figure 5.3.4 : 20 search terms with size = 5 Vs. time

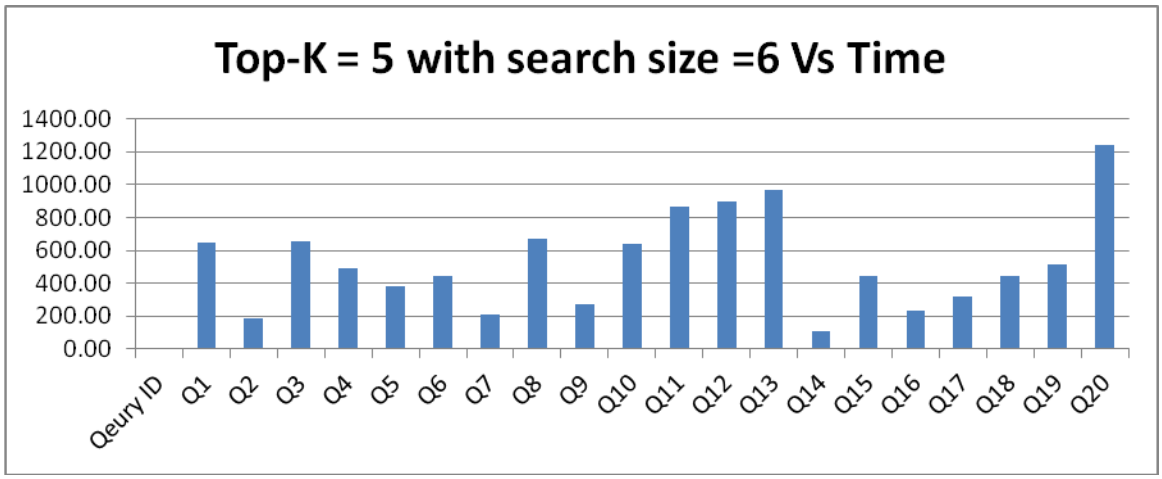
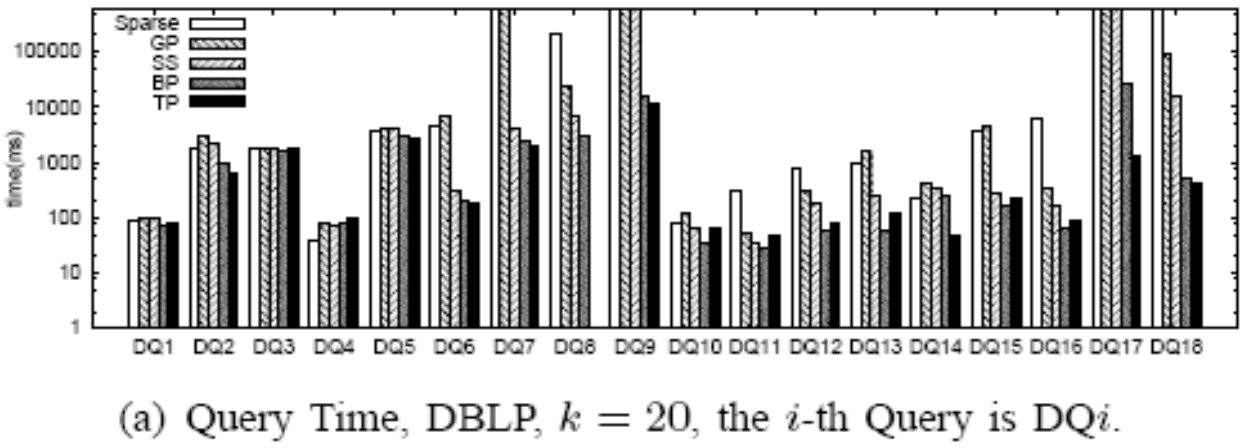


Figure 5.3.5: 20 search terms with size = 6 Vs. time.



(a) Query Time, DBLP, $k = 20$, the i-th Query is DQi.

Figure 5.3.6: SPARK2 [17] with 20 search terms Vs. time.

Through figures 5.3.1 to 5.3.5, we can see the time needed to retrieve the results with different search term sizes, and in figure 5.3.6 we see the time for retrieving the results for the same database used, but it is important to notice that in SPARK2 [17], the size of the database is about 800,000 records, and in our work about 3.7 million records, which is four times bigger, but if we look at the times, we can clearly see that our work achieves faster time and better performance than SPARK2 [17] and SPARK [7].

The following table shows the relation between time and pruned CN's.

Table 5.3.2: Time of search Vs. CN's pruned.

Number of keywords in the search term	CN's Pruned	Search time (normal mode) (ms)
2	20.25	393
3	14	513
4	15.4	571
5	14	623
6	15.3	698

In table 5.3.2, we can see the relation between CN's pruned and time to retrieved the results, the more CN's pruned, the more speed up to our IR.

Table 5.3.3: Size of search term Vs. average time in fast search (cashed search).

Number of keywords in the search term	**Average time cashed search (ms)**
2	**9**
3	**8**
4	**9**
5	**9**
6	**8**
Average	**8**

In table 5.3.3 the system would achieve about 70 times faster in search, if we retrieve the results from the cashed table, which would be considered the fastest retrieval algorithm of

results and only depends on the database engine time, as it only props the database once to retrieve the results.

Table 5.3.4: Size of search term Vs. average time propping the index keywords table.

Number of keywords in the search term	Average time index keywords (s)
2	16
3	16
4	11
5	6
6	7
Average	11

In table 5.3.4, we can clearly see that the impact of our new introduced index keywords table would not exceed 1.9% of the total time to retrieve the results**,** but in table 5.3.2 we can see the importance of removing CN's through the index keywords table, and the time saved in pruning CN's is much more than the time of propping the index keywords table, as each CN pruned needs to prop the database one time , and we have pruned multiple CN's, which lead to eliminating multiple props to the database, as with our method we need only one prop to the index keywords needs one prop.

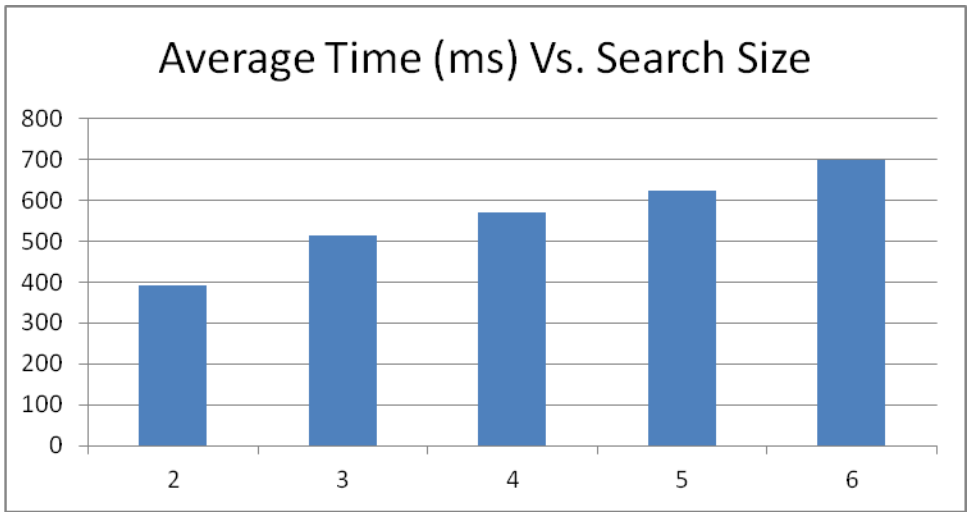


Figure 5.3.7: Performance of our SSearch Vs. Top-K.

By Figure 5.3.7 we can see the average time execution for different search term sizes and the smallest time is when number of keywords are 2.

Table 5.3.5: Performance of our SSearch Vs. other methods [20].

	IR Algorithms performance (execution ms)			
# keywords	**ITREKS**	**Saint**	**Indexing [20]**	**Our Method**
2	**1000**	**700**	**250**	**393**
3	**1400**	**900**	**360**	**513**
4	**1700**	**1000**	**440**	**571**
5	**1800**	**1300**	**500**	**623**
Database : DBLP , Our Data size 4 times bigger				

In table 5.3.5, we can see a comparison of our work with other algorithms used in Indexing[20] regarding the performance measured in ms, our work achieves a better time than ITREKS and Saint, and less time than Indexing[20] which is related to the size of the database and the testing environment. Which we used an online system through a web server and local area network to test our system, but in Indexing[20] the testing where done using one machine and direct access to the data through Java application.

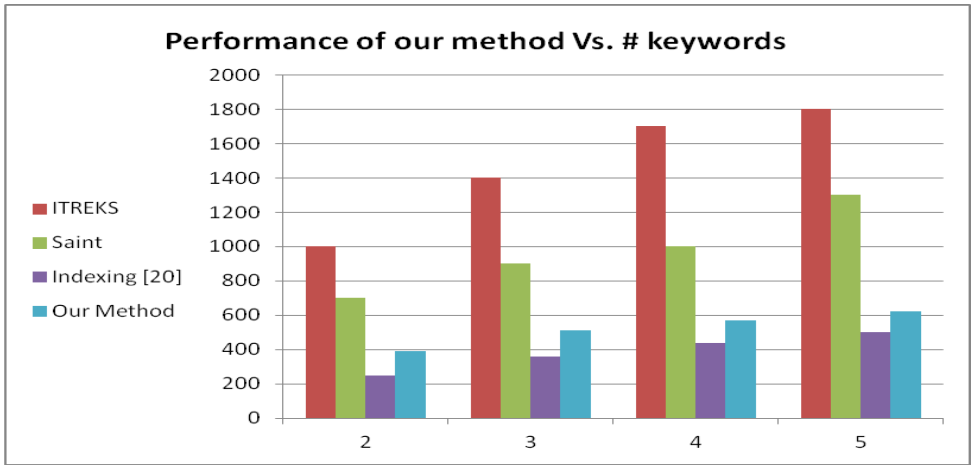


Figure 5.3.8: Performance of our SSearch Vs. other methods [20].

In this section, we have illustrated different aspects of efficiency to our new introduced components in the domain of top-K keyword search over relational databases, and our main work was comparing our work with previous work and other researchers, and our main conclusion that we improve the performance of top-K keyword search over relational databases by introducing the index keywords table with related information stored in it and other components, to help the search engine in generating the necessary CN's and only the CN's that contain results.

5.4 Effectiveness

To measure the quality of the returned results and evaluate our CN generator and ranking function, we used different users to evaluate the returned results by chosen the relevant result for different top-K, using a recording system with a voting for results returned.

We used the query sets in appendix A, with users in table 5.5.2 for K = 5, and K = 15, and we can see a sample of the output data with users hits in appendix D.

Each user checks for the correct result according to the search term and the order of the result, and our SSearch records all users' interaction with the returned results and record them in the database for analyses, and the following tables shows the outcome of our testing with real users.

The following tables and figures summarize the effectiveness of our work and percentage of correct answers through thousands of recorded hits by users.

Table 5.4.1: K= 5 with search term size = 2.

Rank (order results)	# of clicks	Percentage %
1	64	74.42%
2	5	5.81%
3	3	3.49%
4	7	8.14%
5	1	1.16%
Not found	6	6.98%
Total	86	100.00%

Table 5.4.2: K= 5 with search term size = 3.

Rank (order results)	# of clicks	Percentage %
1	18	69.23%
2	1	3.85%
3	4	15.38%
4	1	3.85%
5	0	0.00%
Not found	2	7.69%
Total	26	100.00%

Table 5.4.3: K= 5 with search term size = 4.

Rank (order results)	# of clicks	Percentage %
1	65	84.42%
2	4	5.19%
3	2	2.60%
4	3	3.90%
5	0	0.00%
Not found	3	3.90%
Total	77	100.00%

Table 5.4.4: K= 5 with search term size = 5.

Rank (order results)	# of clicks	Percentage %
1	102	80.95%
2	3	2.38%
3	9	7.14%
4	2	1.59%
5	3	2.38%
Not found	7	5.56%
Total	126	100.00%

Table 5.4.5: K= 5 with search term size = 6.

Rank (order results)	# of clicks	Percentage %
1	65	85.53%
2	1	1.32%
3	4	5.26%
4	1	1.32%
5	0	0.00%
Not found	5	6.58%
Total	76	100.00%

Table 5.4.6: K= 15 with search all term sizes.

Rank (order results)	# of clicks	Percentage %
1	243	73.64%
2	17	5.15%
3	27	8.18%
4	10	3.03%
5	4	1.21%
6	2	0.61%
7	1	0.30%
8	2	0.61%
9	1	0.30%
10	1	0.30%
11	2	0.61%
12	0	0.00%
13	0	0.00%

14	**1**	**0.30%**
15	**0**	**0.00%**
Not found	**19**	**5.76%**
Total	**330**	**100.00%**

Through tables 5.4.1 – 5.4.6, we can summarize our results by:

1. All search terms with different sizes achieves a precision more than 92%.
2. The percentage is more than 96% for search terms with 4 keywords.
3. At worst case the percentage of unrelated results for Top-K is about 8%.
4. The percentage of finding the result at Top-1 in average is more than 75%.

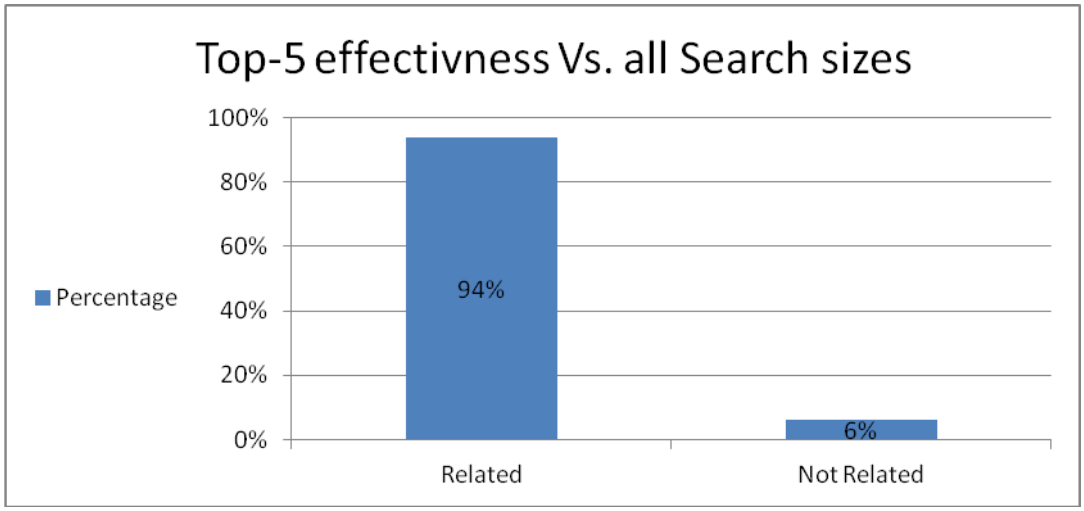


Figure 5.4.1: Top-5 effectiveness with all search sizes.

In figure 5.4.1, we can see that in average our system achieves 94% precision for all search term sizes in Top-5 and the percentage increases for Top-15 and Top-50.

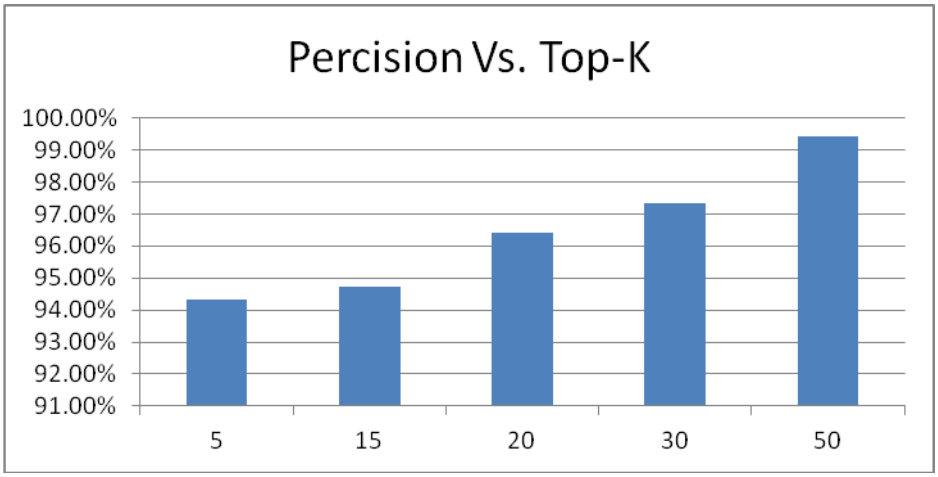


Figure 5.4.2: Effectiveness of our work Vs. Top-K.

Table 5.4.7: SPARK2 [17] Effectiveness of search based on Top-20.

	[15]	[22]	P = 1.0	P = 1.4	P = 2.0
#Rel	2	2	16	16	18
R-Rank	≤ 0.243	≤ 0.333	0.926	0.935	1

By table 5.4.7, we can see that SPARCK2[17] achieves at most for Top-20 93%, and in our system at Top-20 we achieve 96%.

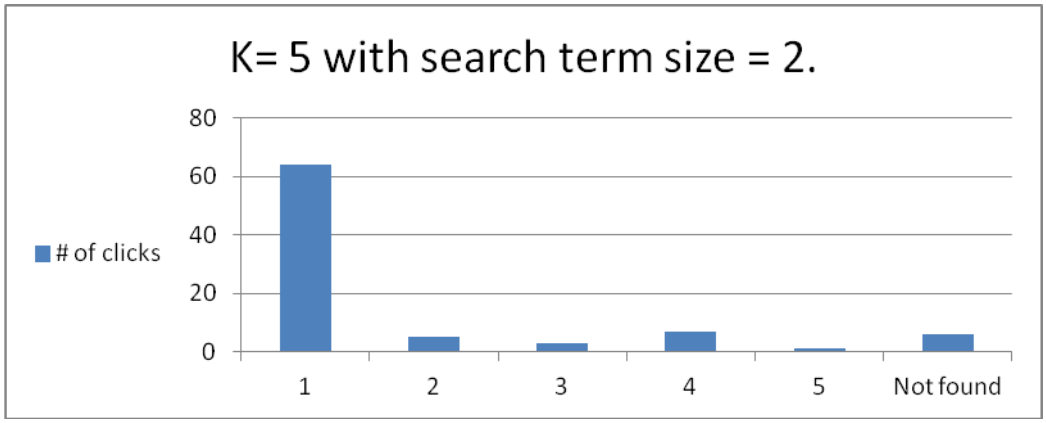


Figure 5.4.3: Effectiveness Top-5 with search Size = 2.

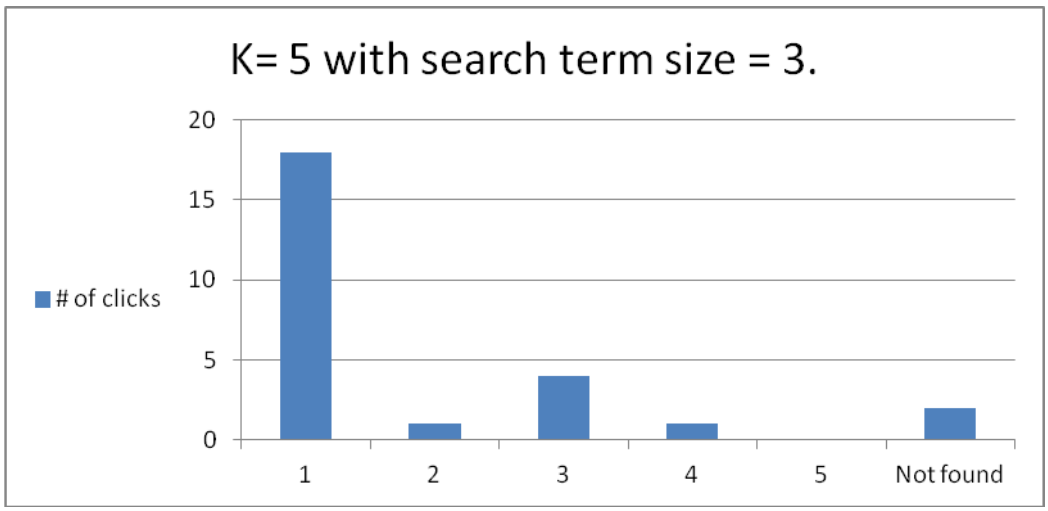


Figure 5.4.4: Effectiveness Top-5 with search Size = 3.

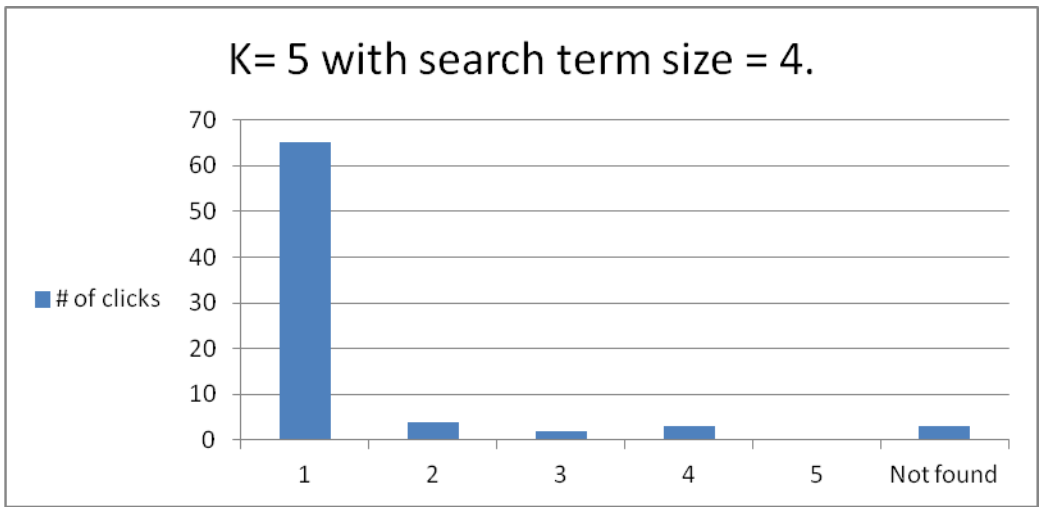


Figure 5.4.5 Effectiveness Top-5 with search Size = 4.

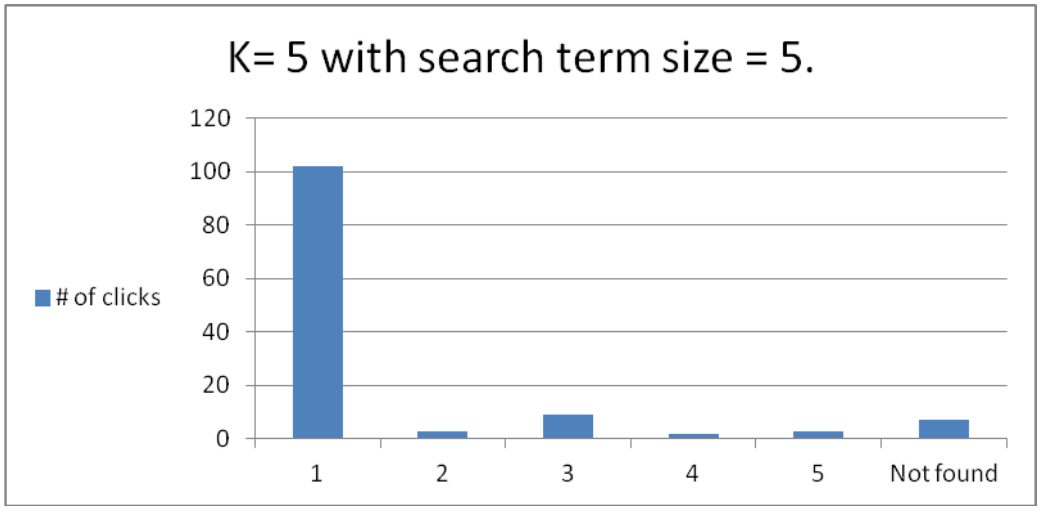


Figure 5.4.6: Effectiveness Top-5 with search Size = 5.

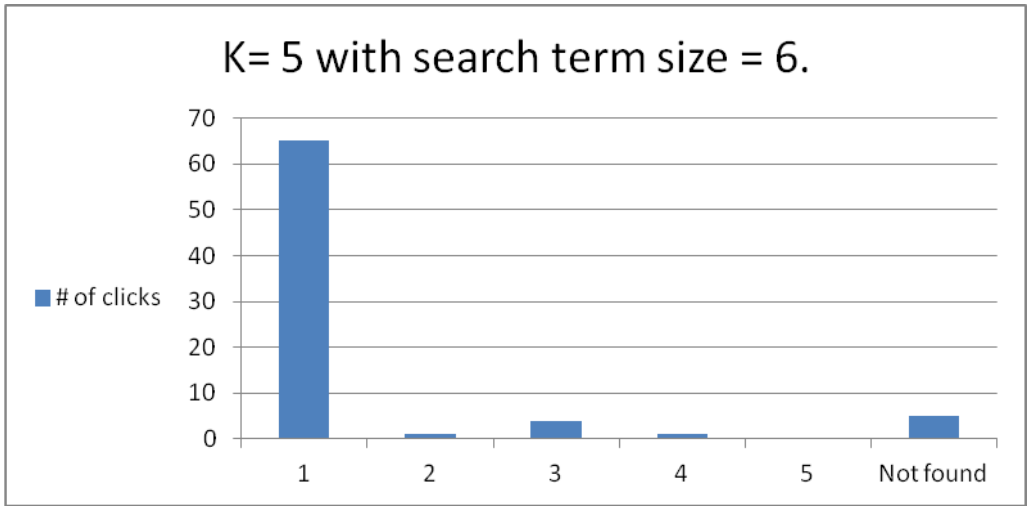


Figure 5.4.7: Effectiveness Top-5 with search Size = 6.

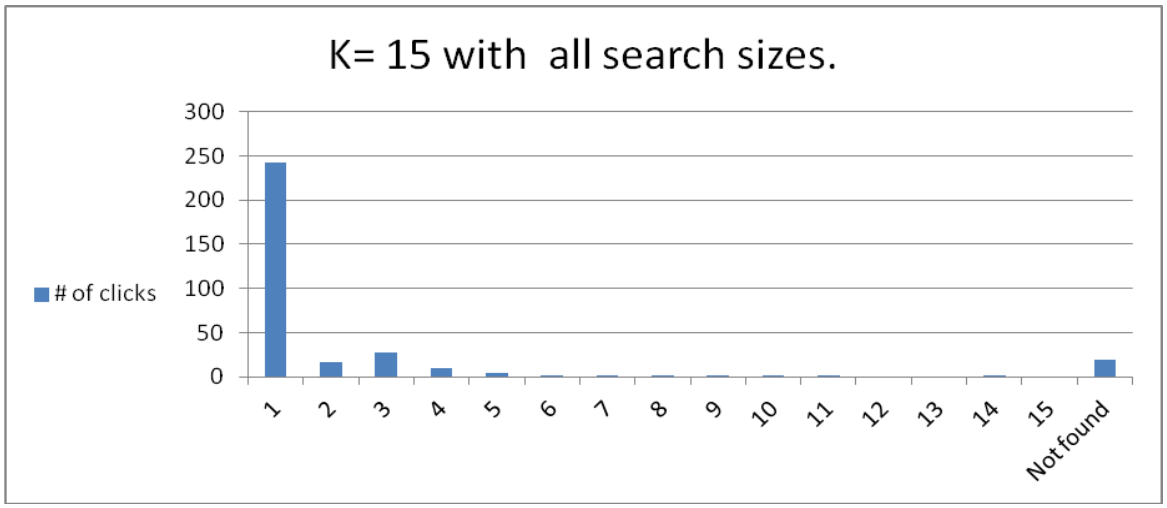


Figure 5.4.8: Effectiveness Top-15 with all search sizes.

Through figures 5.4.1 – 5.4.8, we can summarize our algorithms effectiveness as follows:

- All queries with different sizes achieve and return the required results in the Top-5 with a percentage more than 90%.
- 78.03% the result where found at Top-1.
- Less than 6% the result where not found in the Top-5.
- In SPARK2 [17], we can see that the average effectiveness for Top-20 is 93%, and in Figure 5.3.2 we can see that our algorithm achieves 96.5% accuracy and related results in Top-20 .

5.5 Conclusion

In this thesis, our work main focus was in improving the top-K keyword search over relational databases through the introducing of new components to the database, like the index keywords table and the information table as main part of our work, also the user search history and results history as an aided components for fast search and fast retrieval of results that have been searched by others.

By introducing the index keywords table, the conventional CN generator used by previous researchers would not satisfy the existing of the new component, which leads to the developing of new CN generator based on the index keywords table and the information, and we can clearly see the effect of our index keywords table to the CN generator and the eliminating process of CN's that would not generate results, or have results that do not meet the lower bound of CN's weights.

As the user history was not taken in consideration by previous work, our ranking style incorporates different factors to evaluate each results, and the user history was calculated and included in our work, also our ranking style include the semantics of keywords, order and distance of keywords to reach the best results and present the user with relative results.

To show that our approach generate results with more precision and in less time, we compared our results with other researchers worked on the same database structure with a bigger number or tuples, and their main work concentrate in performance and effectiveness, and we found that our proposed algorithms for CN generation and ranking style with the new index keywords table structured produced more efficient Top-K keywords search and effective results.

In this thesis, we studied the information retrieval over relational databases through free form Top-k keywords search by generating Candidate networks for Joint tuples trees and different ranking styles. We introduced new components to the database and new algorithm of CN generator and ranking style.

We introduced index keywords table and search history of users, we also developed new algorithm for CN generator based on the index keywords table, which by intensive testing of our smart search we found that our approach of CN generation and pruning were effectively and speed up the performance of our smart search by pruning 47% of CN's, and only prune unrelated CN's.

Our work include the introducing of ranking algorithm, which we evaluated with real users and hundreds of search terms, which at all times give us a precision of more than 94% with K =5, and more than 98% with K = 20.

Our approach, comparing to other approaches, performs more and the precision of our system outperforms other approaches.

We used in our testing the famous DBPL database with about 3.7 million records.

5.6 Future work

The work on the index keywords table can be more investigated and developed to have a suitable time in building the index keywords table for online databases, and to be updated with each updated record in the database.

In this thesis, the creation of the index keywords table was done using the offline mode and the algorithm used for that was the B-tree search algorithm, but the load of records and the store of keywords need to develop an efficient algorithms like using parallel algorithms for searching multiple table and finding keywords, also the online mode of creating the index keywords table can be a future direction to develop smart and efficient algorithms to be able to create and update the index keywords table with minimum effect on the performance of the database.

We recommend future work in the representation of the index keywords table in a style of fast retrieval of keywords to speed up the performance of our smart search, also the incorporation of search history in ranking style can be done in alternative ways.

As the CN generator achieves good results in performance and effectiveness, we can develop a CN generator algorithm to speed up the search of keywords and maintain a high percentage of related results to the user.

References

1. Agrawal, S., Chaudhuri, S., and Das, G. (2002): DBXplorer: A System for Keyword-Based Search over Relational Databases, 18th International Conference on Data Engineering (ICDE'02).

2. Xu1, Y., Ishikawa, Y., and Guan, J. (2009): Effective Top-k Keyword Search in Relational Databases Considering Query, APweb and WIAM.

3. Hristidis, V., Hu, Y., and Ipeirotis, P. G. (2010): Ranked Queries over Sources with Boolean Query Interfaces without Ranking Support, ICDE.

4. Coffman, J., and Weaver, A. C. (2010): A Framework for Evaluating Database Keyword Search Strategies, CIKM'10.

5. Guoliang, Zhou, X., Feng, J., and Wang, J. (2009): Progressive Keyword Search in Relational Databases, IEEE.

6. Tran, T., Wang, H., Rudolph, S., and Cimiano, P. (2009): Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (RDF) Data, IEEE.

7. Luo, Y., Lin, X., Wang, W., and Zhou, X. (2007): SPARK: Top-k keyword query in relational databases. In *SIGMOD*.

8. Hristidis, V., and Papakonstantinou, Y. (2002): Discover: Keyword search in relational databases. In *VLDB*, pp. 670–681.

9. Xu1, Y., Ishikawa, Y., and Guan, J. (2010): Efficient Continuous Top-k Keyword Search in Relational Databases, WAIM, pp. 755-767.

10. Luo, Y., Wang, W., and Lin, X. (2008): SPARK: A Keyword Search Engine on Relational Databases, ACM and SIGMOD.

11. Hulgeri, A., Bhalotia, G., Nakhe, C., and Chakrabarti, S. (2002): Keyword Search in Databases, IEEE.

12. Li, G., Feng, J., and Zhou, L. (2008): Retune: Retrieving and Materializing Tuple Units for Effective Keyword Search over Relational Databases, ER 2008, LNCS.

13. Garcia-Alvarado, C., and Ordonez, C. (2010): Keyword Search Across Databases and Documents, KEYS'10.

14. Agrawal, S., Chaudhuri, S., Das, G., and Gionis, A. (2003): Automated Ranking of Database Query Results, 2003 CIDR Conference.

15. Chen, Y., Wang, W., Liu, Z., and Lin, X. (2009): Keyword Search on Structured and Semi-Structured Data, SIGMOD'09.

16. Hristidis, V., Gravano, L., and Papakonstantinou, Y. (2003): Efficient IR-style keyword search over relational databases. In: VLDB, pp. 850-861.

17. Luo, Y., Wang, W., Lin, X., Zhou, X., Wang, J., and Li, K. (2011): SPARK2: Top-k Keyword Query in Relational Databases. In: TKDE Special Issue: Keyword Search on Structured Data.

18. Arslan, A., and Yilmazel, O. (2010): Quality Benchmarking Relational Databases and Lucene in the TREC4 Adhoc Task Environment, IEEE.

19. Feng, J., Li, G., and Wang, J. (2011): Finding Top-k Answers in Keyword Search over Relational Databases Using Tuple Units, IEEE.

20. Khine, P. T., Win, H. P., and Tun, K. N. (2011): Indexing Relational Databases for Efficient Keyword Search. International Journal of Scientific & Engineering Research.

21. Aditya, B., Bhalotia, G., Chakrabarti, S., Hulgeri, A., Nakhe, C., and Sudarshan, P. (2002): BANKS: Browsing and Keyword Searching in Relational Databases, VLDB.

22. Thein, M. M., and Thwin, M. M. (2012): Efficient Schema Based Keyword Search In Relational Databases. International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol.2, No.6.

List of Appendixes

Appendix A: List of query sets used.

Table A.1: query sets with size = 2.

No.	Number of keywords = 2	Table / tables
1	Design with Applications	Books
2	Language Processing in	Books
3	Introduction to Algorithms	Books
4	Foundations of Databases.	Books
5	Database Design	Books
6	The of Computer Programming	Books
7	Database System	Books
8	Sorting and Searching	books
9	Database Management	books
10	Principles and Systems	books
11	Games a Simulation	journals
12	Systems Review	journals
13	I. J. Robotic Res.	journals
14	Telecommunication Systems	journals
15	Model. Interact.	journals
16	Informatica Didactica	journals
17	Hybrid Intell.	journals
18	ACM. Model.	journals
19	Electronic Commerce	journals
20	Distributed Systems	journals
21	Adaptive Optimization	phdthesises
22	Random Sampling	phdthesises
23	Metric and Layered	phdthesises
24	Performance Analysis	phdthesises
25	Augmenting Databases	phdthesises
26	Randomized Algorithms	phdthesises
27	On the Foundations of Query	phdthesises
28	Computer Instruction	phdthesises
29	in Integrating Actice	phdthesises
30	The Optimization of Queries	phdthesises
31	Mobile Agents	proceedings
32	Autonomous Agents	proceedings
33	Graph-Theoretic	proceedings
34	Database Theory~	proceedings
35	Multiagent Systems	proceedings
36	Organising Systems	proceedings
37	Logic Programming	proceedings
38	Software Engineering	proceedings

39	Graph Concepts	proceedings
40	Conference on Multimedia	proceedings
41	Sign Language	papers
42	Mathematical Model	papers
43	Manipulate Graphical.	papers
44	Languages Recognition.	papers
45	Signing Gestures.	papers
46	Human Gestural	papers
47	Product Design.	papers
48	Natural Movements	papers
49	Measurement Framework	papers
50	Imitation Learning	papers
51	Hiroyoshi Hatazawa	persons
52	Akira Asato	persons
53	Yuka Takizawa	persons
54	Kazushi Hayashi	persons
55	Takeo Uehara	persons
56	Tamio Mano	persons
57	Helen Pull	persons
58	Satoshi Menju	persons
59	Taro Kawagishi	persons
60	Yutaka Ogawa	persons
61	Nathan Hierarchies	persons - papers
62	Detlefs Garbage	persons - papers
63	Paris Functional	persons - papers
64	Chandra Queries.	persons - papers
65	Giuseppe Fragmented .	persons - papers
66	Oded Dynamic	persons - papers
67	Minker Numerical	persons - papers
68	Joachim Relational.	persons - papers
69	Zaniolo Optimization.	persons - papers
70	Maier Object.	persons - papers
71	Marsan Quality	persons- proceedings
72	Andrea Service	persons- proceedings
73	Wojciech Rough	persons- proceedings
74	Forsyth Shape	persons- proceedings
75	Polkowski Trends	persons- proceedings
76	JosephGrouping	persons- proceedings
77	Giuseppe Networks~	persons- proceedings
78	Marco Quality	persons- proceedings
79	Michela Multiservice	persons- proceedings
80	Deprettere Embedded	persons- proceedings
81	power Automation	papers- proceedings
82	RoughTransactions on	papers- proceedings
83	memory Design	papers- proceedings

84	Approximation Rough	papers- proceedings
85	Energy Design	papers- proceedings
86	Algorithms Transactions on	papers- proceedings
87	Automatic Automation	papers- proceedings
88	Knowledge Sets III	papers- proceedings
89	integrated Conference~	papers- proceedings
90	Fuzzy Transactions	papers- proceedings
91	Compiler generalization	books - papers
92	Concurrency Replicated	books - papers
93	Databases. Inclusion	books - papers
94	Active Database	books - papers
95	Systems: Triggers	books - papers
96	Symbolic -Linear	books - papers
97	Java~ Mathematical .	books - papers
98	Algorithmen Boolean	books - papers
99	Databases. Path	books - papers
100	Knowledge Management	books - papers

Table A.2: query sets with size = 3.

	keywords generated from all tables	
No.	Number of keywords = 3	Table
1	Object - Oriented Reengineering	books
2	Introduction to Knowledge Base	books
3	Logics for Databases and Information	books
4	Modern Database : The Object	books
5	Mining the World Web:	books
6	Ahnlichkeitssuche in Multimedia - - Retrieval	books
7	Client Data Caching:	books
8	Indexing Techniques for Database	books
9	The Design and Implementation of a Log	books
10	Software Prototyping in Data and	books
11	Man - Machine Studies	journals
12	of Logic Language and Information	journals
13	Applied Non - Classical Logics	journals
14	Notre Dame of Formal	journals
15	Journal of Computational Linguistics	journals
16	Environmental Modelling and Software	journals
17	Chemical Information and Sciences	journals
18	of Information Technology and Decision	journals
19	Educational Resources in Computing	journals
20	IEEE Annals the of Computing	journals
21	Metric and Layered Temporal	phdthesises
22	Performance Analysis of Data	phdthesises

23	Databases with Generalized Transitive	phdthesises
24	Randomized for Query Optimization.	phdthesises
25	On the Foundations of Query Evaluation in	phdthesises
26	Automatic Design of Instruction	phdthesises
27	Issues in Integrating Rules Into	phdthesises
28	The Optimization of Queries in Databases.	phdthesises
29	Descriptive Name Services	phdthesises
30	Describing Database - Formalization	phdthesises
31	International on Rapid Prototyping	proceedings
32	Signal and Image Processing	proceedings
33	Security in Information Systems	proceedings
34	Shape Modeling and Applications	proceedings
35	Reasoning with Uncertainty in Robotics	proceedings
36	Theoretical Computer Science	proceedings
37	Solid Modeling and Applications	proceedings
38	Solid and Physical Modeling	proceedings
39	Advances in Spatial Databases	proceedings
40	Temporal Aspects in Information	proceedings
41	Boolean Query Reformulation.	papers
42	Interactive Systems in TREC.	papers
43	creating authority files.	papers
44	Information Design for a Network.	papers
45	Key Inducers of Links	papers
46	Measuring the Impact of Information	papers
47	The web as a classroom resource	papers
48	A clarification of the policy on printing	papers
49	relevance of spoken documents.	papers
50	Descriptive Assessment of Information	papers
51	Daniel Breaking the Walls	persons - papers
52	Perez Acceleration of Monte	persons - papers
53	Hujun Bao Nonlinear	persons - papers
54	Huang An Enhanced Framework	persons - papers
55	Kok Wong Hopping Figures.	persons - papers
56	Kadi Handling Dynamic	persons - papers
57	Jan Kautz Efficient	persons - papers
58	Anath Fischer Adaptive	persons - papers
59	Philippe Efficient Rendering	persons - papers
60	Frank Local Subsurface	persons - papers
61	Jingling Advances in Computer	persons- proceedings
62	Willard Accurate Scientific	persons- proceedings
63	David Parallel Processing	persons- proceedings
64	Roland neue Probleme	persons- proceedings
65	Roland Parallel Processing	persons- proceedings
66	Roland Algebraic and Coalgebraic	persons- proceedings
67	Harmelen Validation and Verification	persons- proceedings

68	Gorlatch Parallel Processing	persons- proceedings
69	Arabnia Foundations of Computer	persons- proceedings
70	Fitch on Symbolic and Algebraic	persons- proceedings
71	Peppercoin Micropayments. || Cryptography	papers- proceedings
72	Efficient Maximal. || Financial	papers- proceedings
73	Analysis of Document. || Digital	papers- proceedings
74	Inclusion in Online | Publishing	papers- proceedings
75	A Component Architecture || Electronic	papers- proceedings
76	Combining the Power || Principles	papers- proceedings
77	Oriented Model || Electronic	papers- proceedings
78	Spotting Singular || Document	papers- proceedings
79	Semantic Data || Approach	papers- proceedings
80	A Transaction Architecture for a. ||- Relationship	papers- proceedings
81	Concurrency Control|| Deadlock .	books - papers
82	Crafting a Compiler || Generation	books - papers
83	Crafting a Compiler || Constructing	books - papers
84	Recovery in Database || Performance.	books - papers
85	Control and Recovery . || Operating	books - papers
86	Fuzzy Sets . || Advances	papers - journals
87	Computer Communication || Advances	papers - journals
88	Analogical Reasoning || Artif	papers - journals
89	A Universal Model Computers	papers - journals
90	Numerical Methods || Advances	papers - journals

Table A.3: query sets with size = 4.

No.	keywords generated from all tables	
	Number of keywords = 4	Table
1	Concepts Design and Performance Analysis	books
2	Concurrent Reactive Plans Anticipation and	books
3	Ray Shooting Depth Orders	books
4	A Comparative Study of Large Data	books
5	High - Dimensional Indexing: Transformational	books
6	Vivid Logic : Knowledge - Based	books
7	Tractable Reasoning in Artificial Intelligence	books
8	Learning - Based Robot Vision	books
9	Mapping Scientific Frontiers: Visualization	books
10	Automatic Generation of Computer Animation :	books
11	IEEE Trans. Circuits Video	journals
12	J. Artificial Societies and Social Simulation	journals
13	Universal Access in the Information Society	journals
14	Wireless Communications and Mobile Computing	journals
15	Mobile Computing and Communications Review	journals
16	IEEE Transactions on Image Processing	journals

17	SIGARCH Computer Architecture News	journals
18	Hall and IBM Research Report	journals
19	IBM Research Report California	journals
20	Electronic Colloquium on Computational Complexity	journals
21	Line Processing in Large - Scale	phdthesises
22	Programming Data Structures in Logic.	phdthesises
23	Optimizing Bottom - Query Evaluation	phdthesises
24	Contribution to a Methodology for the Implementation of Logic	phdthesises
25	Abstract Interpretation for the Compile -Time	phdthesises
26	Support for Software Fault Tolerance	phdthesises
27	Design of Concurrency Control Protocols	phdthesises
28	Implementation of the SELF Compiler ~ an Optimizing	phdthesises
29	The Temporal Deductive Database ChronoLog	phdthesises
30	Compilet for Object - Oriented Programming	phdthesises
31	World Conference on Artificial Intelligence	proceedings
32	Trends in Congress of the Italian Association	proceedings
33	Applied Image Pattern Recognition	proceedings
34	Artificial Intelligence Planning Systems	proceedings
35	Advanced Information Processing Techniques	proceedings
36	Image Data Fusion workshop	proceedings
37	Integrating Symbolic Mathematical Computation	proceedings
38	LAN and MAN Management ~ Proceedings	proceedings
39	Brazilian Symposium on Neural Networks	proceedings
40	Computer Assisted Radiology and Surgery	proceedings
41	A taxonomy for distributed real time	papers
42	OOPSLA panel on concurrent programming.	papers
43	The role of methods and case in development.	papers
44	Schema updates for database abstract	papers
45	Building large distributed software	papers
46	expected impact of Ada9X on a computer science	papers
47	Three discussions on object-oriented typing.	papers
48	Software architecture panel object	papers
49	Verification and testing in an CS2 course.	papers
50	Medical image registration mutual	papers
51	Dinesh Pai Motion Perturbation	persons - papers
52	Li Chen Nonlinear View Interpolation.	persons - papers
53	Yeong Shin Template rendering	persons - papers
54	Qunsheng Peng Nonlinear Interpolation.	persons - papers
55	Norishige Simulation of Peeling Surface	persons - papers
56	Daniel Thalmann Live Participant	persons - papers
57	Jacob Barhak Adaptive Reconstruction	persons - papers
58	Kenji A Discrete Spring Model for	persons - papers
59	Daniel Compression of Indoor Video	persons - papers
60	Philipp Slusallek Bounded Clustering	persons - papers
61	Pedro Meseguer on the Validation and Verification Spain	persons- proceedings

62	Marie Christine Verification of Knowledge	persons- proceedings
63	Ewa Orlowska Theory and Applications	persons- proceedings
64	John L. Pfaltz of Graph Transformations	persons- proceedings
65	Frans Coenen Validation of Knowledge	persons- proceedings
66	Munindar P. Singh the Human and Artificial	persons- proceedings
67	Omer F. Rana Infrastructure for Agents	persons- proceedings
68	Thomas Wagner Multi Agent Systems	persons- proceedings
69	Manfred Nagl Industrial Relevance	persons- proceedings
70	Andy Schurr of Graph Transformations	persons- proceedings
71	Combinatorial OPtimization. || Constraint Programming	papers- proceedings
72	Symmetry Breaking. || Principles and Practice of	papers- proceedings
73	Polynomial Restrictions || Principles and Practice	papers- proceedings
74	A Composistional Theory || Constrained Systems	papers- proceedings
75	Eye interaction || Human Factors	papers- proceedings
76	breadth first || Factors in Computing	papers- proceedings
77	A Syntactic Schema . || ACM Annual	papers- proceedings
78	Database Management . || Annual Conference	papers- proceedings
79	Modeling Computer || Annual Conference	papers- proceedings
80	Blocking Probability || Networking ReunionIsland	papers- proceedings
81	Triggers and Rules || Multilevel Secure	books - papers
82	Engineering a Compiler || Hot Optimization	books - papers
83	Database Processing. || A Transactional Model	books - papers
84	Active Database . || Object - Oriented	books - papers
85	Rules For Advanced . || Incremental Production	books - papers
86	Analysis by Usability. || Comput Lang	papers - journals
87	An Algorithm for Generating || Comput Lang	papers - journals
88	Organizational Connectivity || DATA BASE	papers - journals
89	The Evolution of Information || DATA BASE	papers - journals
90	An Intermediate Language || Comput Lang	papers - journals

Table A.4: query sets with size = 5.

No.	keywords generated from all tables	
	Number of keywords = 5	Table / tables
1	Compositionality ~ Concurrency and Partial Correctness - Proof	books
2	Input Output Massively Parallel Computing -	books
3	Security and Privacy - Design and Use of Privacy-Enhancing	books
4	A Unified Approach to Interior Point Algorithms	books
5	Extensions of the UNITY Methodology - Compositionality ~ Fairness	books
6	NEWCAT : Parsing Natural Language Associative	books
7	Finite Representations of CCS and TCSP Programs	books
8	Consolidated Ada Reference Manual. Language	books
9	Network Calculus: A Theory of Deterministic Queuing	books

10	Algebraic System Specification and Development - A Survey	books
11	Journal of Research and Practice in Information Technology	journals
12	International Journal of Geographical Information Systems	journals
13	International Journal of Geographical Information Science	journals
14	IEEE Journal on Selected Areas in Communications	journals
15	CVGIP : Graphical Model and Image Processing	journals
16	Computer Vision ~ Graphics ~ and Image Processing	journals
17	International Journal of Internet and Enterprise Management	journals
18	Journal of Chemical Information and Computer Sciences	journals
19	ACM Journal of Educational Resources in Computing	journals
20	IEEE / ACM Comput. Biology Bioinform.	journals
21	Storage Management Technique for Object - Oriented	phdthesises
22	Towards Middleware Support for Mobile and Cellular	phdthesises
23	Power of Ordered Binary Decision Diagrams	phdthesises
24	Controlled Queueing Systems with Heterogeneous Servers	phdthesises
25	Glue : A Deductive Database Programming Language	phdthesises
26	Optimization and Execution Techniques for Queries With Expensive	phdthesises
27	File System Performance and Transaction Support.	phdthesises
28	High - level Methods for - Sequential Verification	phdthesises
29	Parsing and Quering XML Documents in SML.	phdthesises
30	nonrenewal and state dependent input traffic.	phdthesises
31	Quality of Service IWQoS Germany June	proceedings
32	Self - Adaptive Software~ Second International	proceedings
33	Case - Based Reasoning Research and Development	proceedings
34	Computational n Biology Informatics and Mathematics JOBIM	proceedings
35	Job Scheduling Strategies for Parallel Processing	proceedings
36	Formal Methods and Software Engineering Methods	proceedings
37	Formalization of Programming Concepts Colloquium Peniscola	proceedings
38	Image Analysis and Processing International Conference	proceedings
39	Applying Formal Methods : Testing Performance	proceedings
40	Formal Techniques for Networked and Distributed FORTE	proceedings
41	Modified least loaded routing in virtual	papers
42	Data Integration in a Personal Communication Network.	papers
43	Mean packet delay analysis for the selectivedelays.	papers
44	MMPP models for multimedia traffic.	papers
45	Mobile Agents and Java Mobile Agents	papers
46	Performance modeling of distributed automatic call	papers
47	Dynamic connection admission mechanisms for the Networking	papers
48	Performance Analysis of a Mobile Communication Network	papers
49	A discrete time limited vacation model	papers
50	Analysis of quality of service in a wide area	papers
51	John Darlington Optimisation of component applications	persons - papers
52	Jose Fortes A Scalable SNMP Distributed	persons - papers
53	Peter Steenkiste Network based multicomputers	persons - papers
54	Rajeev Thakur Scheduling Regular and Communication	persons - papers

55	Ken Kennedy Experience with interprocedural analysis	persons - papers
56	Paul Havlak analysis of array side effects.	persons - papers
57	Danny Kopec A taxonomy of concepts for chess	persons - papers
58	Alex Ropelewski of videoconferencing in the supercomputing environment.	persons - papers
59	George C. Polyzos analysis of characteristics of scientific	persons - papers
60	Ridgway Scott Limitations in Parallel Dynamics.	persons - papers
61	Martin Riedmiller Robot Soccer World	persons- proceedings
62	Andrzej Skowron Directions in Data Mining	persons- proceedings
63	Ning Zhong and Granular Soft Computing	persons- proceedings
64	Geoff Coulson Distributed Systems Platforms	persons- proceedings
65	Joseph S. Sventek Middleware Distributed Systems	persons- proceedings
66	Paddy Nixon Middleware for Pervasive and Computing	persons- proceedings
67	Stephen Hanson Machine Learning Theory	persons- proceedings
68	Ronald L. Rivest From Theory to Applications - Cooperative	persons- proceedings
69	Anja Feldmann and Protocols for Computer Communication	persons- proceedings
70	Martina Zitterbart Applications Technologies Architectures	persons- proceedings
71	Different Languages || Advances in Intelligent Data	papers- proceedings
72	Weighted Median Filtering. || Image Analysis	papers- proceedings
73	Tumor Segmentation || Image Analysis and Recognition	papers- proceedings
74	Histogram of Feasible Mappings. || Image Analysis	papers- proceedings
75	Change in Surveillance System. || Image Analysis	papers- proceedings
76	Efficient Translation Registration. |e Analysis and Recognition	papers- proceedings
77	Performance Testing of Web || Information and Databases	papers- proceedings
78	Data Relational Tables. || Information and Databases	papers- proceedings
79	Filtering Target Documents || Databases Tokyo	papers- proceedings
80	dimensional Scheduling Scheme|| Information e IASTED	papers- proceedings
81	Advanced Database Processing . || Composite Specification	books - papers
82	Engineering a Compiler || Design and Implementation of a Diagnostic	books - papers
83	Active Database Triggers|| A DOOD RANCH	books - papers
84	Engineering a Compiler || Improving Register Allocation for Subscripted Variables.	books - papers
85	Active Triggers and Rules || A Predicate Algorithm	books - papers
86	Business Process Redesign || DATA BASE	papers - journals
87	Towards a Convention on Protection. || Computer Networks	papers - journals
88	An intelligent usage ATM || Computer Networks	papers - journals
89	Admission Control and Routing || Computer and ISDN	papers - journals
90	Advances in Testing Approaches. || Computer Networks	papers - journals

Table A.5: query sets with size = 6.

No.	keywords generated from all tables	
	Number of keywords = 6	Table / Tables
1	Information Visualization in Data Mining and Knowledge Discovery	books
2	Computer Systems That Learn: Classification and Prediction Methods	books
3	Managing Gigabytes : Compressing and Indexing Documents and Images	books

4	Unified Modeling Language: Systems Analysis Development	books
5	Understanding SQL and Java A Guide to JDBC Technologies	books
6	Java in a Nutshell - A Desktop Quick Reference for Programmers	books
7	Iterative Software Engineering for Multiagent Systems: The MASSIVE	books
8	Artificial Agent Societies Structure and Its Implications for Autonomous .	books
9	Modular Compiler Verification - A Refinement - Algebraic Approach	books
10	Solution of Large and Sparse Systems of Linear Algebraic	books
11	Buffer: A Storage Management Technique for - Oriented Databases	phdthesises
12	Data Conversion in Machine Independent Code Generators	phdthesises
13	Modeling and Evaluation of Database Concurrency Control Algorithms	phdthesises
14	Subgoal Order for Query Optimization in Logic Databases.	phdthesises
15	Swizzling Techniques for Object - Oriented Database Systems	phdthesises
16	Metric and Layered Temporal Logic for Time Granularity	phdthesises
17	Self - Describing Database Systems - Formalization and Realization	phdthesises
18	Issues in Integrating Actice Rules Into Database Systems	phdthesises
19	Ordered Binary Decision Diagrams by Integrating Parity Nodes	phdthesises
20	High - Methods for OBDD - based Sequential Verification	phdthesises
21	International Conference on Image Analysis and Processing ICIAP	proceedings
22	Workshop on Future Trends of Distributed Computing learning	proceedings
23	Information and Communication Security International Conference ICICS	proceedings
24	Computer Assisted Learning International Conference Proceedings	proceedings
25	Computer Assisted Learning International Conference Wolfville	proceedings
26	Research Directions in Data and Applications Security XVIII	proceedings
27	Data and Applications Security Status and Prospects Annual	proceedings
28	Automata Languages and Programming International Colloquium ICALP	proceedings
29	Audio and Video Based Biometric Person Authentication	proceedings
30	Object Orientation to Formal Methods Essays in Memory	proceedings
31	Global optimization of univariate functionsalgorithms and computational	papers
32	A globally and quadratically convergent affine scaling method.	papers
33	A duality approach to minimax results for quasi functions	papers
34	Solving combinatorial optimization problems Karmarkar_s algorithm.	papers
35	Local convergence of Newton methods for differentiable equations.	papers
36	splitting method and the proximal point algorithm for maximal	papers
37	Nonlinear multiple objective optimization: An algorithm and theory.	papers
38	A parallel branch and bound for solving large asymmetric	papers
39	Tight integral duality gap in the Chinese Postman	papers
40	Generalized convexity on affine subspaces with an application to potential	papers
41	Andrew Chien Performance Messaging on Workstations : Messages	persons - papers
42	Lubomir Bic Partitioning declarative programs into communicating	persons - papers
43	Margaret Simmons The Performance Realities of Massively Parallel	persons - papers
44	Philip McKinley Multicast Virtual Topologies for Communication .	persons - papers
45	Francine Berman A decoupled scheduling approach GrADS	persons - papers
46	Hans Berliner A taxonomy of concepts for evaluating strength.	persons - papers
47	Karen Devine A massively parallel adaptive finite	persons - papers
48	Michael Berry of concurrent programs on the Cedar multiprocessor.	persons - papers

49	Charles Hansen data parallel polygon rendering.	persons - papers
50	Kyle Gallivan The Synergetic Effect of Compiler Architecture	persons - papers
51	Andrew S. Tanenbaum Systems Support for Worldwide Applications	persons- proceedings
52	Chris Jones SIGUCCS Conference on User Services	persons- proceedings
53	Ralf Steinmetz Multimedia : Advanced Teleservices and High	persons- proceedings
54	Alberto Prieto Artificial Neural Networks International	persons- proceedings
55	Lefteris M. Kirousis Structure Information and Communication Complexity	persons- proceedings
56	Luisa Gargano Structural Information a Communication Complexity	persons- proceedings
57	Mario Piattini Ingenieria del Software y Bases	persons- proceedings
58	Steffen Staab Workshop on Ontology Learning conjunction	persons- proceedings
59	Subbarao Kambhampati Workshop on Information Integration on the Web	persons- proceedings
60	Won Kim The Human Society and the Internet Economic	persons- proceedings
61	Degrees of Unsolvability in Abstract Complexity || Current Research	papers- proceedings
62	Triangulation Technique a Continuous Pattern. || Graphics and Robotic	papers- proceedings
63	Standardized Notation and Execution Architecture. || Model Based	papers- proceedings
64	Practical Aspects of Specialization Algol . || Partial Evaluation	papers- proceedings
65	Markov Localization Robot Navigation || Sensor Intelligent	papers- proceedings
66	Technology Transfer Teaching and Training. || Experimental Software	papers- proceedings
67	Computing Simplicial Homology Based Efficient || Algebra Geometry	papers- proceedings
68	Grammars for Mutliresolution || Graph Transformations in Computer Science	papers- proceedings
69	Specifying Algorithm Visualizations || Software Visualization Seminar	papers- proceedings
70	Fixed Parameter Tractability and Completeness. || Complexity Theory	papers- proceedings
71	Active Database and Rules For Advanced || Implementing Large	books - papers
72	Triggers and Rules For Advanced Database Processing. || Integrating	books - papers
73	Engineering a Compiler || A Fast and Usually Linear Algorithm	books - papers
74	Advanced Database Processing. || Efficient Monitoring of Assertions	books - papers
75	Engineering a Compiler || A Unified Approach to Global Program Optimization.	books - papers
76	Developer Reponsiveness on Perceptions of Usefulness || DATA BASE	papers - journals
77	Extension of the Distributed Control || Computer Networks and ISDN	papers - journals
78	Delay Analysis Channel Access Protocol. || Computer Networks	papers - journals
79	multiprocessor cache coherent mechanism || Parallel Computing	papers - journals
80	maximum matchings of bipartite graphs || Discrete Mathematics	papers - journals

Appendix B: List of Stop Words.

Table B.1: list of stop words.

No.	Word	No.	Word	No.	Word	No.	Word	No.	Word
1	a	36	admitted	71	allow	106	anywhere	141	awful
2	aah	37	adopted	72	allowed	107	arch	142	awfully
3	aaron	38	advice	73	almost	108	are	143	awhile
4	abandon	39	advise	74	alone	109	area	144	awkward
5	abandoned	40	affair	75	along	110	aren't	145	aww
6	abe	41	affairs	76	already	111	argue	146	aye
7	ability	42	After	77	alright	112	arguing	147	b
8	able	43	afternoon	78	also	113	arm	148	ba
9	aboard	44	afterwards	79	altar	114	armed	149	babe
10	about	45	again	80	although	115	arms	150	back
11	above	46	Age	81	always	116	army	151	bang
12	absolute	47	Ages	82	am	117	arnold	152	bat
13	absolutely	48	Ago	83	amp	118	art	153	bath
14	absurd	49	agree	84	amy	119	as	154	be
15	abuse	50	agreed	85	an	120	asa	155	became
16	accuse	51	ah	86	and	121	ash	156	because
17	accused	52	aha	87	andie	122	ashamed	157	become
18	accusing	53	ahead	88	anniversary	123	aside	158	becomes
19	ace	54	ahem	89	announce	124	ask	159	becoming
20	acid	55	ahh	90	announcement	125	asked	160	bed
21	act	56	ahhh	91	annoying	126	asking	161	been
22	acted	57	aid	92	anonymous	127	asks	162	before
23	acting	58	aidan	93	another	128	asleep	163	beg
24	actor	59	aids	94	any	129	ass	164	began
25	actress	60	ain't	95	anya	130	assault	165	begged
26	acts	61	air	96	anybody	131	asshole	166	begging
27	actual	62	airport	97	anybody's	132	at	167	begin
28	actually	63	aisle	98	anyhow	133	ate	168	beginning
29	ad	64	aitoro	99	anymore	134	aunt	169	begins
30	add	65	al	100	anyone	135	aw	170	begun
31	added	66	al's	101	anyone's	136	awake	171	being
32	addition	67	alarm	102	anything	137	award	172	below
33	address	68	alike	103	anytime	138	aware	173	beside
34	admire	69	alive	104	anyway	139	away	174	besides
35	admit	70	all	105	anyways	140	awesome	175	best

No.	Word	No.	Word	No.	Word	No.	Word	No.	Word
176	Bet	218	Comes	260	de	302	earned	344	exists
177	between	219	Comin	261	dear	303	ears	345	exit
178	Big	220	Coming	262	del	304	earth	346	expect
179	Bigger	221	Con	263	dick	305	ease	347	f
180	biggest	222	Cool	264	did	306	easier	348	fan
181	Bike	223	Cop	265	didn't	307	easily	349	far
182	Bitter	224	Copy	266	die	308	east	350	fat
183	Blew	225	Cord	267	dig	309	easy	351	fax
184	Bo	226	Cost	268	dish	310	eat	352	few
185	Boo	227	Costs	269	dna	311	eats	353	file
186	bought	228	costume	270	do	312	ed	354	filed
187	Bow	229	cottage	271	doc	313	ego	355	files
188	Bowl	230	cotton	272	document	314	eh	356	fill
189	bowling	231	couch	273	does	315	el	357	filled
190	Bro	232	cough	274	doesn't	316	else	358	filling
191	But	233	could	275	dog	317	else's	359	film
192	By	234	could've	276	dogs	318	em	360	filthy
193	Bye	235	couldn't	277	doin	319	empty	361	final
194	C	236	count	278	doing	320	end	362	finally
195	c'mon	237	cow	279	doll	321	ended	363	financial
196	Cab	238	cozy	280	don	322	ending	364	find
197	Came	239	cry	281	don't	323	ends	365	finding
198	Can	240	crying	282	done	324	entire	366	finds
199	can't	241	crystal	283	down	325	entirely	367	fine
200	cancel	242	cue	284	dr	326	entitled	368	first
201	canceled	243	culture	285	dru	327	er	369	fit
202	Cap	244	cup	286	dry	328	even	370	five
203	Cd	245	cure	287	du	329	ever	371	flu
204	Chose	246	cut	288	duck	330	every	372	for
205	chosen	247	cute	289	dude	331	everybody	373	from
206	Clock	248	cuts	290	due	332	everybody's	374	fuck
207	Close	249	cutting	291	duh	333	everyday	375	fucked
208	Closed	250	cuz	292	duke	334	everyone	376	fuckin
209	Closer	251	d	293	dull	335	everyone's	377	fucking
210	closest	252	d'you	294	e	336	everything	378	g
211	Closet	253	da	295	each	337	everything's	379	gain
212	closing	254	dad	296	eager	338	everywhere	380	gay
213	Co	255	dad's	297	ear	339	ew	381	get
214	Com	256	daddy	298	earl	340	ex	382	gets
215	Coma	257	daddy's	299	earlier	341	exist	383	gettin
216	Comb	258	day	300	early	342	existed	384	getting
217	Come	259	days	301	earn	343	existence	385	gia

No.	Word	No.	Word	No.	Word	No.	Word	No.	Word
386	gig	428	gym	470	how'll	512	lot	554	mud
387	giles	429	h	471	how's	513	low	555	mum
388	give	430	ha	472	huh	514	lt	556	must
389	given	431	had	473	I	515	m	557	must've
390	gives	432	hadn't	474	I'd	516	ma	558	mustn't
391	giving	433	hah	475	I'll	517	ma'am	559	my
392	glad	434	hal	476	I'm	518	mama	560	myself
393	glass	435	has	477	I've	519	max	561	n
394	glasses	436	hasn't	478	I've	520	may	562	nah
395	glen	437	hat	479	ian	521	maybe	563	nail
396	go	438	have	480	id	522	me	564	nails
397	goa'uld	439	haven't	481	if	523	mean	565	nbsp
398	goal	440	having	482	in	524	meaning	566	near
399	goat	441	he	483	inch	525	means	567	nearly
400	god	442	hed	484	inn	526	meant	568	need
401	god's	443	he'll	485	inside	527	meantime	569	net
402	goddamn	444	he's	486	into	528	meanwhile	570	next
403	gods	445	heh	487	is	529	mess	571	nice
404	goes	446	hello	488	it	530	met	572	night
405	goin	447	her	489	it'd	531	mia	573	night's
406	going	448	here	490	it'll	532	mid	574	no
407	gone	449	here's	491	it's	533	middle	575	noah
408	goodbye	450	hers	492	item	534	midnight	576	noble
409	goodness	451	herself	493	its	535	might	577	nobody
410	goodnight	452	hey	494	itself	536	might've	578	nobody's
411	goods	453	hi	495	ivy	537	mightn't	579	non
412	goose	454	hid	496	j	538	mighty	580	none
413	got	455	him	497	k	539	mix	581	nope
414	gotta	456	himself	498	kit	540	mixed	582	nor
415	gotten	457	hip	499	know	541	mm	583	not
416	gown	458	his	500	knowing	542	mmm	584	note
417	gt	459	hm	501	l	543	mob	585	notes
418	gum	460	hmm	502	la	544	mom	586	now
419	gun	461	hmmm	503	lap	545	mom's	587	nowhere
420	guns	462	ho	504	las	546	mommy	588	number
421	gus	463	hon	505	le	547	mommy's	589	nut
422	gut	464	hoo	506	led	548	mon	590	o
423	guts	465	hop	507	left	549	motherfucker	591	o'clock
424	guy	466	hour	508	let	550	Mr	592	o'neill
425	guy's	467	hours	509	let's	551	mrs	593	odd
426	guys	468	how	510	lets	552	ms	594	odds
427	gwen	469	how'd	511	lo	553	much	595	of

No.	Word	No.	Word	No.	Word	No.	Word	No.	Word
596	off	638	oz	680	sexy	722	the	764	uhh
597	often	639	p	681	she	723	thee	765	uhm
598	oh	640	pa	682	she'd	724	their	766	um
599	ohh	641	pace	683	she'll	725	them	767	umm
600	ohhh	642	pacey	684	she's	726	theme	768	un
601	oil	643	pal	685	shh	727	themselves	769	unless
602	ok	644	pan	686	shhh	728	then	770	unlike
603	okay	645	papa	687	should	729	There	771	until
604	ol	646	part	688	should've	730	there'll	772	up
605	old	647	pass	689	si	731	there's	773	upon
606	older	648	past	690	sin	732	therefore	774	upper
607	oldest	649	pat	691	since	733	Theresa	775	ups
608	on	650	per	692	sir	734	theresa	776	us
609	once	651	post	693	sis	735	these	777	use
610	one	652	posted	694	so	736	they	778	used
611	one's	653	pot	695	some	737	they'd	779	uses
612	ones	654	pro	696	somebody	738	they'll	780	using
613	only	655	pussy	697	somebody's	739	they'er	781	usual
614	onto	656	put	698	someday	740	they've	782	usually
615	ooh	657	puts	699	somehow	741	this	783	v
616	oooh	658	putting	700	someone	742	this'll	784	val
617	oops	659	q	701	someone's	743	those	785	van
618	or	660	quit	702	someplace	744	thou	786	very
619	order	661	quite	703	somethin	745	though	787	w
620	other	662	quote	704	something	746	thought	788	was
621	other's	663	r	705	something's	747	thy	789	wasn't
622	others	664	raw	706	sometime	748	tie	790	we
623	otherwise	665	ray	707	sometimes	749	til	791	we'd
624	ouch	666	re	708	somewhere	750	tis	792	we'll
625	our	667	rest	709	soon	751	to	793	we'er
626	ours	668	reva	710	sooner	752	today	794	we've
627	ourselves	669	s	711	st	753	today's	795	went
628	out	670	say	712	sub	754	too	796	were
629	outside	671	saying	713	such	755	took	797	weren't
630	outta	672	says	714	t	756	tool	798	wes
631	over	673	sd	715	ta	757	top	799	wh
632	ow	674	sec	716	th	758	tv	800	wha
633	owe	675	see	717	than	759	twas	801	what
634	owes	676	send	718	that	760	two	802	what
635	own	677	set	719	that'd	761	u	803	what
636	owns	678	sex	720	that'll	762	ugh	804	what
637	oxygen	679	sexual	721	that's	763	uh	805	whatever

No.	Word	No.	Word	No.	Word	No.	Word
806	whatever	826	whoever	846	y	866	yours
807	whatsoever	827	whole	847	y'all	867	yourself
808	when	828	whom	848	y'know	868	yourselves
809	when'd	829	whoo	849	ya	869	yup
810	when'll	830	whore	850	yay	870	zach
811	when's	831	whose	851	ye	871	zack
812	whenever	832	why	852	yeah		
813	where	833	why'd	853	yell		
814	where'd	834	why'll	854	yelling		
815	where'll	835	why's	855	yep		
816	where's	836	will	856	yes		
817	wherever	837	will's	857	yet		
818	whew	838	won	858	yo		
819	which	839	won't	859	york		
820	While	840	woo	860	you		
821	Who	841	would	861	you'd		
822	who'd	842	would've	862	you'll		
823	who'll	843	wouldn't	863	you're		
824	who's	844	wow	864	you’ve		
825	whoa	845	x	865	your		

Appendix C: Sample List of index keywords table.

Table C.1: sample of the most frequent keywords in index keywords table.

keyword	order_co	oc_co	table_name	col_name
Based	2	224628	papers	title
systems	1	148916	papers	title
system	2	133534	papers	title
Data	4	116907	papers	title
analysis	2	105391	papers	title
model	1	95819	papers	title
design	1	92884	papers	title
networks	1	89937	papers	title
approach	0	82908	papers	title
information	1	81166	papers	title
algorithm	0	78801	papers	title
Time	0	76416	papers	title
learning	3	68588	papers	title
software	0	68251	papers	title
parallel	1	67081	papers	title
performance	1	65871	papers	title
distributed	3	64577	papers	title
image	5	60031	papers	title
control	1	56875	papers	title
Web	0	55234	papers	title
Logic	1	53514	papers	title
management	0	51055	papers	title

Table C.2: Sample of the least frequent keywords in index keywords table.

keyword	order_co	oc_co	table_name	col_name
dstl	1	1	papers	Title
quickshunt	0	1	papers	Title
Samhita	0	1	papers	Title
Efdm	0	1	papers	Title

Bedsocs	0	1	papers	Title
Multiphath	0	1	papers	Title
Hyphens	0	1	papers	Title
Uniquest	0	1	papers	Title
Oufdm	1	1	papers	Title
autodescriptivity	0	1	papers	Title
Symontos	0	1	papers	Title
Frankel	1	1	papers	title
Triadic	0	1	papers	title
Syntropy	0	1	papers	title
Datafair	1	1	papers	title
Ufl	0	1	papers	title
Comar	0	1	papers	title
Moler	2	1	papers	title
Slpl	0	1	papers	title
Balakirsky	1	1	papers	title
treesvladimir	0	1	papers	title
Sampo	1	1	papers	title
Campey	2	1	papers	title
Stragies	0	1	papers	Title
Edinburg	0	1	papers	Title
Logbooks	0	1	papers	Title
Cmsr	0	1	papers	Title
Ipses	0	1	papers	Title

Order_co: order of keyword inside the text filed.

Oc_co: occurrence of the keyword across a certain table and filed.

Appendix D: DBLP tables and structures with samples of information.

Table D.1: Books table structure.

Id	Isbn	publisher	Series	title	conference	volume	year
1194152	3-540-40799-5	Springer	Natural Computing Series	Modelling in Molecular Biology			2004
1193584	1-4020-7247-3	Kluwer	Multimedia Systems and Applications	Multimedia Mining: A Highway to Intelligent Multimedia Documents		22	2002
1194162	0-387-24568-5	Springer	Multiagent Systems~ Artificial Societies~ and Simulated Organizations	Multi-Agent Programming: Languages~ Platforms and Applications		15	2005
1193829	3-540-41904-7	Springer	Monographs in Theoretical Computer Science. An EATCS Series	Incomplete Information: Structure~ Inference~ Complexity			2002
1193844	3-540-13641-X	Springer	Monographs in Theoretical Computer Science. An EATCS Series	Data Structures and Algorithms 2: Graph Algorithms and NP-Completeness		2	1984
1193845	3-540-13642-8	Springer	Monographs in Theoretical Computer Science. An EATCS Series	Data Structures and Algorithms 3		3	1984
1193858	3-540-13302-X	Springer	Monographs in Theoretical Computer Science. An EATCS Series	Data Structures and Algorithms 1: Sorting and Searching		1	1984

Table D.2: Journals table structure.

Id	Publisher	title
1185812	IBM Germany Science Center~ Institute for Knowledge Based Systems	IWBS Report
1185813	IBM Deutschland GmbH	LILOG-Report
1185814	IBM Deutschland GmbH	LILOG-Memo
1185815	IBM Corporation~ White Plains~ NY	IBM Publication
1185816	IBM Cambridge Scientific Center	IBM Research Report

Table D.3: Msthesises table structure.

Id	school	title	year
1200022	Diplomarbeit~ Universitat Trier~ FB IV~ Informatik	Portierung des DBLP-Systems auf ein relationales Datenbanksystem und Evaluation der Performance.	2006
1200023	Univ. of Wisconsin-Madison	PRPL: A Database Workload Specification Language~ v1.3.	1992
1200024	Diplomarbeit~ Universitat Trier~ FB VI~ Physische Geographie	Der Einfluss kleiner naturnaher Retentionsmasnahmen in der Flache auf den Hochwasserabfluss - Kleinruckhaltebecken -.	2006
1200025	University of California at Santa Barbara~ Department of Computer Science	Efficient View Maintenance at Data Warehouses.	1997
1200026	Universitat Karlsruhe~ Institut fur Logik~ Komplexitat und Deduktionssysteme	Entwurf und Implementierung eines ubersetzenden Systems fur das intuitionistische logische Programmieren auf der Warren Abstract Machine.	1991
1200027	University of California at Berkeley	A Prolog Compiler for the PLM.	1984

Table D.4: Papers table structure.

Id	title
1	Fuzzy Sets and Their Applications to Artificial Intelligence.
2	Computer Design and Description Languages.
3	The Structure of Design Processes.
4	Developments in Firmware Engineering.
5	Programmed Control of Asynchronous Program Interrupts.
6	Computer Supported Cooperative Work and Groupware.
7	Operating Systems Enhancements for Distributed Shared Memory.
8	Software Reliability.
9	Binary Arithmetic.

Table D.5: Persons table structure.

Id	name
728511	Stephan Vollmer
728512	Kurt P. Brown
728513	Rita Ley
728514	Tolga Yurek
728515	Roy T. Fielding
728516	Tim Berners-Lee
728517	Henrik Frystyk Nielsen
728518	Jerome Simeon

Table D.6: Phdthesises table structure.

Id	Isbn	Publisher	school	series	title	volume	year
1200029			Universitat Kaiserslautern		Implementierungskonzepte fur Non-Standard-Datenbanksysteme.		1989
1200030			Department of Computer Science~ Stanford University		Query Optimization in Deductive and Relational Databases		1991
1200031			Univ. of Wisconsin-Madison		Representing and Querying Complex Information in the Coral Deductive Database System.		1993
1200032			University of Aarhus	BRICS Dissertation Series	Efficient External-Memory Data Structures and Applications.		1996

1200033			Univ. of Wisconsin-Madison		Code Generation Techniques.		1992
1200034			Uni Trier~ Informatik		Optimierung der Anfrageauswertung in einem Deduktiven Datenbanksystem.		1992
1200035			Stanford University~ Department of Computer Science		Adaptive Optimization in a Database Programming Language		1992
1200036		LBL Technical Report	University of California at Berkeley		Random Sampling from Databases		1993

Table D.7: Proceedings table structure.

Id	conference	isbn	publisher	series	title	volume	Year
1185822	Text Understanding in LILOG	3-540-54594-8	Springer	Lecture Notes in Computer Science	Text Understanding in LILOG~ Integrating Computational Linguistics and Artificial Intelligence~ Final Report on the IBM Germany LILOG-Project	546	1991
1185823	Advanced Database Systems	3-540-57507-3	Springer	Lecture Notes in Computer Science	Advanced Database Systems	759	1993
1185824	Constraint Programming	3-540-59155-9	Springer	Lecture Notes in Computer Science	Constraint Programming: Basics and Trends~ Chatillon Spring School~ Chatillon-sur-Seine~ France~ May 16 - 20~ 1994~ Selected Papers	910	1995
1185827	J. Data Semantics V	3-540-31426-1	Springer	Lecture Notes in Computer Science	Journal on Data Semantics V	NULL	2006
1185828	J. Data Semantics IV	3-540-31001-0	Springer	Lecture Notes in Computer Science	Journal on Data Semantics IV	3730	2005
1185829	J. Data Semantics III	3-540-26225-3	Springer	Lecture Notes in Computer Science	Journal on Data Semantics III	3534	2005
1185830	J. Data Semantics I	3-540-20407-5	Springer	Lecture Notes in Computer Science	Journal on Data Semantics I	2800	2003
1185831	J. Data Semantics II	3-540-24208-2	Springer	Lecture Notes in Computer Science	Journal on Data Semantics II	3360	2005
1185832	T. Rough Sets	3-540-29830-4	Springer	Lecture Notes in Computer Science	Transactions on Rough Sets IV	3700	2005
1185833	T. Rough Sets	3-540-25998-8	Springer	Lecture Notes in Computer Science	Transactions on Rough Sets III	3400	2005

Table D.8: Raw_links table structure.

LinkId	From	To	link_type	month	pages	in-volume	in-year
81	15	1185275	in-journal	NULL	173-245	33	NULL
82	728562	16	author-of	NULL	NULL	NULL	NULL
83	16	1185275	in-journal	NULL	321-356	45	NULL
84	728563	17	author-of	NULL	NULL	NULL	NULL
85	728564	17	author-of	NULL	NULL	NULL	NULL
86	17	1185275	in-journal	NULL	163-216	22	NULL
87	728565	18	author-of	NULL	NULL	NULL	NULL
88	728566	18	author-of	NULL	NULL	NULL	NULL
89	728567	18	author-of	NULL	NULL	NULL	NULL
90	728568	18	author-of	NULL	NULL	NULL	NULL
91	728569	18	author-of	NULL	NULL	NULL	NULL
92	728570	18	author-of	NULL	NULL	NULL	NULL
93	728571	18	author-of	NULL	NULL	NULL	NULL
94	728572	18	author-of	NULL	NULL	NULL	NULL
95	18	1185275	in-journal	NULL	255-324	35	NULL
96	728573	19	author-of	NULL	NULL	NULL	NULL

Table D.9: Wwws table structure.

Id	title	year	url
1194512	Self-organizing Map	NULL	http://www.cis.hut.fi/nnrc/nnrc-programs.html
1194513	Beowulf Project at CESDIS	NULL	http://beowulf.gsfc.nasa.gov/beowulf.html
1194514	The MPEG Home Page	NULL	http://drogo.cselt.stet.it/mpeg/
1194515	Ardent Software	NULL	http://www.ardentsoftware.fr
1194516	WfMC Standards: The Workflow Reference Model~ Version 1.1.	1995	http://www.aiim.org/wfmc/mainframe.htm
1194517	KAFFE	NULL	http://www.kaffe.org/
1194518	W3C: Extensible Stylesheet Language (XSL)	NULL	http://www.w3.org/Style/XSL/
1194519	Internet Draft: Hypertext Transfer Protocol - HTTP/1.1	NULL	http://www.w3.org/Protocols/HTTP/1.1/spec.html
1194520	W3C: Document Oject Model (DOM)	NULL	http://www.w3.org/DOM/
1194521	W3C - The World Wide Web Consortium	NULL	http://www.w3.org/
1194522	Request for Comments: 1738~	1994	http://www.w3.org/Addressing/rfc1738.txt

	Uniform Resource Locators (URL)		
1194523	W3C: The w3c Query Language Workshop~ December 1998~ Boston~ MA~ USA	1998	http://www.w3.org/TandS/QL/QL98/
1194524	XQuery: A Query Language for XML	2001	http://www.w3.org/TR/xquery
1194525	The XML Query Algebra	2001	http://www.w3.org/TR/2000/WD-query-algebra-20001204/
1194526	W3C: Extensible Markup Language (XML) 1.0	NULL	http://www.w3.org/TR/REC-xml
1194527	XML Query Use Cases	2001	http://www.w3.org/TR/xmlquery-use-cases
1194528	XML Query Requirements	2001	http://www.w3.org/TR/xmlquery-req
1194529	XML Query Data Model	2001	http://www.w3.org/TR/query-datamodel
1194530	Xml-ql: A Query Language for XML	NULL	http://www.w3.org/TR/NOTE-xml-ql/
1194532	The Future of Classic Data Administration: Objects + Databases + CASE	1998	http://www.mitre.org/support/swee/rosenthal.html
1194533	The SGML/XML Web Page	NULL	http://www.oasis-open.org/cover/xml.html

Appendix E: Sample List of output data.

Table E.1: sample of search terms entered and recorded results (time in seconds).

search_term	Over all_time	top_k	top_k_time	fast_time	keyword_ time	Cn count	One table_cn	Multi table_cn	Keys count	burnedc n
'Giuseppe Fragmented .	0.0223598480	5	0.0133280754	0.0095911026	0.0005109310	2	1	1	2	32
'Akira Asato	0.0238518715	5	0.0141639709	0.0082800388	0.0008611679	2	1	1	2	32
'Takeo Uehara	0.0267181396	5	0.0170240402	0.0108728409	0.0008411407	2	1	1	2	32
'Spotting Singular || Document	0.0254459381	5	0.0180811882	0.0076320171	0.0006411076	1	1	0	2	33
'Helen Pull	0.0273480415	5	0.0199820995	0.0083951950	0.0004811287	3	2	1	2	31
'Tight integral duality gap in the Chinese Postman	0.0269808769	15	0.0215330124	0.0078289509	0.0007929802	4	1	3	6	30
'Tight integral duality gap in the Chinese Postman	0.0342199802	5	0.0243079662	0.0078849792	0.0011370182	4	1	3	6	30
'Ray Shooting Depth Orders	0.0362558365	15	0.0281629562	0.0064890385	0.0006070137	7	2	5	3	27
'Parsing and Quering XML Documents in SML.	0.0453989506	15	0.0399458408	0.0050067902	0.0008180141	7	2	5	5	27
'Daniel Thalmann Live Participant	0.0445179939	5	0.0405781269	0.0068378448	0.0005750656	7	1	6	4	27
'Daniel Thalmann Live Participant	0.0625240803	15	0.0408720970	0.0076601505	0.0007719994	7	1	6	4	27
'Wojciech Rough	0.0555670261	5	0.0491590500	0.0082340240	0.0005259514	8	1	7	2	26
'Qunsheng Peng Nonlinear Interpolation.	0.0546269417	15	0.0503790379	0.0066509247	0.0006790161	7	1	6	4	27
'Qunsheng Peng Nonlinear Interpolation.	0.0599691868	5	0.0527830124	0.0064439774	0.0009989738	7	1	6	4	27
'Parsing and Quering XML Documents in SML.	0.0655968189	15	0.0564439297	0.0057790279	0.0013201237	7	2	5	5	27
'Danny Kopec A taxonomy of concepts for chess	0.0680849552	15	0.0581459999	0.0084240437	0.0006489754	12	2	10	5	22
'Danny Kopec A taxonomy of concepts for chess	0.0737738609	5	0.0635800362	0.0091738701	0.0013248920	12	2	10	5	22
'Peppercoin Micropayments. || Cryptography	0.0691988468	15	0.0639090538	0.0049369335	0.0007700920	11	1	10	3	23
'Peppercoin Micropayments. || Cryptography	0.0737071037	5	0.0664191246	0.0076398849	0.0023751259	11	1	10	3	23
'Manfred Nagl Industrial Relevance	0.0717129707	15	0.0673468113	0.0100150108	0.0007600784	14	1	13	4	20
'Parsing and Quering XML Documents in	0.0746510029	5	0.0689418316	0.0048179626	0.0007948875	7	2	5	5	27

SML.										
'Informatica Didactica	0.0788559914	5	0.0722560883	0.0077049732	0.0376551151	4	1	3	2	30
'Danny Kopec A taxonomy of concepts for chess	0.0834310055	15	0.0783801079	0.0084660053	0.0008490086	12	2	10	5	22
'Manfred Nagl Industrial Relevance	0.0857129097	5	0.0788300037	0.0046970844	0.0011861324	14	1	13	4	20
'Concurrent Reactive Plans Anticipation and	0.0830359459	15	0.0788450241	0.0055360794	0.0008180141	11	2	9	4	23
'Danny Kopec A taxonomy of concepts for chess	0.0889620781	5	0.0809719563	0.0095841885	0.0007109642	12	2	10	5	22
'Yeong Shin Template rendering	0.0916500092	5	0.0832979679	0.0082361698	0.0012230873	11	2	9	4	23
'Joachim Relational.	0.0936300755	5	0.0839498043	0.0075850487	0.0011401176	15	1	14	2	19
'Concurrent Reactive Plans Anticipation and	0.0934460163	5	0.0869150162	0.0076799393	0.0011491776	11	2	9	4	23
'Yeong Shin Template rendering	0.0972809792	15	0.0905179977	0.0045669079	0.0011160374	11	2	9	4	23
'Li Chen Nonlinear View Interpolation.	0.1009678841	15	0.0937700272	0.0070991516	0.0006830692	9	1	8	5	25
'Blocking Probability || Networking ReunionIsland	0.1532571316	15	0.0954380035	0.0041890144	0.0008869171	9	1	8	4	25
'Li Chen Nonlinear View Interpolation.	0.1047308445	5	0.0965149403	0.0083858967	0.0012280941	9	1	8	5	25
'Blocking Probability || Networking ReunionIsland	0.1051259041	5	0.0981860161	0.0110421181	0.0006291866	9	1	8	4	25
'Java in a Nutshell - A Desktop Quick Reference for Programmers	0.1060829163	5	0.1007058620	0.0082840919	0.0011119843	16	2	14	6	18
'Java in a Nutshell - A Desktop Quick Reference for Programmers	0.1158061028	15	0.1070890427	0.0093770027	0.0016119480	16	2	14	6	18
'Marco Quality	0.1233379841	5	0.1138100624	0.0078420639	0.0011758804	16	1	15	2	18
'Pedro Meseguer on the Validation and Verification	0.1291930676	15	0.1204569340	0.0072858334	0.0011589527	15	1	14	4	19
'Pedro Meseguer on the Validation and Verification	0.1291050911	5	0.1206369400	0.0042099953	0.0014650822	15	1	14	4	19
'Spotting Singular || Document	0.1283972263	15	0.1250700951	0.0081248283	0.0548858643	1	1	0	2	33
'Lubomir Bic Partitioning declarative programs into communicating	0.1328501701	15	0.1257789135	0.0056490898	0.0012030602	13	1	12	6	21
'Lubomir Bic Partitioning declarative programs into communicating	0.1417410374	5	0.1282989979	0.0092060566	0.0015349388	13	1	12	6	21
'Roland neue Probleme	0.1458168030	5	0.1380898952	0.0089399815	0.0010139942	15	2	13	3	19
'Andy Schurr of Graph Transformations	0.1429328918	5	0.1385381222	0.0082769394	0.0009391308	16	1	15	4	18
'Andy Schurr of Graph Transformations	0.1443140507	15	0.1393589973	0.0072579384	0.0008249283	16	1	15	4	18

Over all_time: The time from executing a keyword query search until all results displayed for the user.

Top_k: Represents the top K answers to a certain query search term determined by the user.

Top_k_time: Time to execute just the chosen CN's and ranking their results with displaying the top-K results.

Keyword_time: Time to retrieve all keywords entered by user from the index keywords table.

Cn_count: Number of possible CN's found containing entered keywords by the user.

One_table_cn: Number of CN's generated for single table meets the criteria.

Multi_table_cn: Number of CN's generated through multiple tables with relations.

Keys count: Number of keywords valid in the search term.

Burned cn: Number of pruned CN's that does not meet the criteria.

Table E.2: Sample of results returned with user hits.

search_term	result	order	hit	From CN	Fields in CN	Final Rank
'Automatic Generation of Computer Animation :	Automatic Generation of Computer Animation: Using AI for Movie Animation	1	1	books	title	2.5
'Automatic Generation of Computer Animation :	[From Reslut] Automatic surface generation in computer aided design. [To Result] The Visual Computer	2	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.571428571
'Automatic Generation of Computer Animation :	[From Reslut] Automatic Cel Painting in Computer-assisted Cartoon Production using Similarity Recognition. [To Result] Journal of Visualization and Computer Animation	3	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.571428571
'Automatic Generation of Computer Animation :	[From Reslut] 3D Measurement Technologies for Computer Animation. [To Result] Computer Animation 1996 (CA 1996)~ 3-4 June 1996~ Geneva~ Switzerland	4	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] title1	1.5
'Automatic Generation of Computer Animation :	[From Reslut] Employing Approximate 3D Models to Enrich Traditional Computer Assisted Animation. [To Result] IEEE Computer Society	5	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] publisher	1.35
'IEEE Trans. Circuits Video	IEEE Trans. Circuits Syst. Video Techn.	1	1	journals	title	2.2
'IEEE Trans. Circuits Video	[From Reslut] Complexity of optimized H.26L video decoder implementation. [To Result] IEEE Trans. Circuits Syst. Video Techn.	2	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.625
'IEEE Trans. Circuits Video	[From Reslut] A unified architecture for real-time video-coding systems. [To Result] IEEE Trans. Circuits Syst. Video Techn.	3	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.625

'IEEE Trans. Circuits Video	[From Reslut] Color quantization of compressed video sequences. [To Result] IEEE Trans. Circuits Syst. Video Techn.	4	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.625
'IEEE Trans. Circuits Video	[From Reslut] Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. [To Result] IEEE Trans. Circuits Syst. Video Techn.	5	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.625
'Wireless Communications and Mobile Computing	[From Reslut] The future evolution of wireless mobile communications. [To Result] Wireless Communications and Mobile Computing	1	1	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	2.625
'Wireless Communications and Mobile Computing	[From Reslut] Special issue: mobility management in wireless and mobile networks. [To Result] Wireless Communications and Mobile Computing	2	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	2.25
'Wireless Communications and Mobile Computing	[From Reslut] The use of the simulated annealing algorithm for channel allocation in mobile computing. [To Result] Wireless Communications and Mobile Computing	3	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	2.25
'Wireless Communications and Mobile Computing	[From Reslut] Special Issue: Ultrawideband for Wireless Communications. [To Result] Wireless Communications and Mobile Computing	4	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	2.25
'Wireless Communications and Mobile Computing	[From Reslut] Logarithmic expected packet delivery delay in mobile ad hoc wireless networks. [To Result] Wireless Communications and Mobile Computing	5	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	2

' Hall and IBM Research Report	In: R. Rustin (ed.): Database Systems: 65-98~ Prentice Hall and IBM Research Report RJ 987~ San Jose~ California	1	1	journals	title	2
' Hall and IBM Research Report	IBM Research Report~ San Jose~ California	2	0	journals	title	1.416666667
' Hall and IBM Research Report	IBM Research Report	3	0	journals	title	1.416666667
' Hall and IBM Research Report	[From Reslut] Lawrence O. Hall [To Result] Report of research activities in fuzzy AI and medicine at USF CSE.	4	0	[From Table] persons [To Table] papers	[First Filed] name [Second Filed] title	1.35
' Hall and IBM Research Report	[From Reslut] Index Path Length Evaluation for the Research Storage System of System R. [To Result] IBM Research Report	5	0	[From Table] papers [To Table] journals	[First Filed] title [Second Filed] title1	1.25
' Hall and IBM Research Report	[From Reslut] Grounding Classification Research in Real World Problems: Report of the Third Panel of Presentations and Associated Discussion. [To Result] ASIS SIG/CR Classification Research Workshop	6	0	[From Table] papers [To Table] proceedings	[First Filed] title [Second Filed] conference	1.1
' Hall and IBM Research Report	[From Reslut] Wendy Hall [To Result] The Application of a Hypermedia Research System in Industry.	7	0	[From Table] persons [To Table] papers	[First Filed] name [Second Filed] title	0.9

Appendix F: Term Definitions

- Relational database: A database composed of a set of relations R1;R2; _ _ _ ;Rn. A Schema Graph (SG) is a directed graph with the relations as its nodes and the foreign key to primary key relationships of the relations as its edges.[2]

- Ranking: Method of rearranging the information retrieved by the user according to factors and element vary from one method to another. [2]

- IR-style: information retrieval method [16]

- JTT: A Joint-Tuple-Tree (JTT) is a joining tree of different tuples. Each node is a tuple in the database, and each pair of adjacent tuples in T is connected via a foreign key to primary key relationship, A JTT is an answer to a keyword query if it contains more than one keyword of the query and each of its leaf tuples must contain at least one keyword [2]

- CN: A Candidate Network (CN) is a tree of tuple sets RQ or RF with the restriction that every leaf node must be a query tuple set [2]

- Top-*k* : Describe a framework for generating an approximate top-*k* answer, with some probabilistic guarantees [2]

- Keyword : Word used by users for search

- Search Term: A set of keywords and non-keywords entered by the user for search.

- Stop word: A word that is known to search engines and dropped from the search query.

- Primary Key: A unique identifier, often an integer that labels a certain row in a table of a relational database.

- Foreign Key: A field in a relational table that matches the primary key column of another table.

- Searchable field: A filed in a database that contains information like text and strings can be searched.

- Database Schema: The definition of a database. It defines the structure and content in each data element within the structure.

- Master Index: Is the database that contains all the information needed to retrieve the user request answers in efficient way.

- Tuple: Represents a database record which is a row of data in a database table consisting of a single value from each column in a database table.

- Effectiveness: Represents the accurate of the returned results and percentage of related results according to the user in the search of keywords over relational databases.

- Efficiency: In keywords search over relational databases the term efficiency represents the performance of the search, time and resources needed to accomplish a search process.

- Ranking Score: It is an indicator which measures how well a particular answer is relevant to the user query. Answers with high ranking scores are more relevant than the answers with low ranking scores.

- Query: A set of structured statements used for information retrieval from any given database, which is built through special language called SQL.

- SQL: Structured query language considered type of programming language and developed for talking to databases by retrieving information or updating or deleting and many other different functions can be found in any SQL.

- Prop: An operation describes the compiling and executing of SQL queries to certain database with results returned.

- Prune: The elimination process of CN's in the top-K keyword search domain according to certain factors and weights.