

**DEVELOPMENT OF SMART WEB CRAWLER BY  
APPLYING BREADTH-FIRST ALGORITHM  
AND VECTOR SPACE MODEL**

**UNDERGRADUATE THESIS**

**CREATED BY :**

**IRVAN ARIYANTO**

**NIM. 13650104**



**JURUSAN TEKNIK INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM  
MALANG  
2017**

**DEVELOPMENT OF SMART WEB CRAWLER BY  
APPLYING BREADTH-FIRST ALGORITHM  
AND VECTOR SPACE MODEL**

**UNDERGRADUATE THESIS**

**Diajukan kepada :  
Fakultas Sains dan Teknologi  
Universitas Islam Negeri (UIN)  
Maulana Malik Ibrahim Malang  
Untuk Memenuhi Salah Satu Persyaratan Dalam  
Memperoleh Gelar Sarjana Komputer (S.Kom)**

**CREATED BY :  
IRVAN ARIYANTO  
NIM.13650104**

**JURUSAN TEKNIK INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM  
MALANG  
2017**

**LEMBAR PERSETUJUAN**

**DEVELOPMENT OF SMART WEB CRAWLER BY  
APPLYING BREADTH-FIRST ALGORITHM  
AND VECTOR SPACA MODEL**

**UNDERGRADUATE THESIS**

**Created by :**

**Irvan Ariyanto**

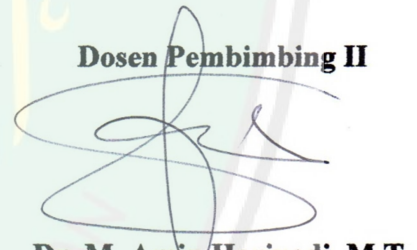
**NIM.13650104**

**Telah disetujui oleh :**

**Dosen Pembimbing I**

  
**Dr. Cahyo Crysdiان**  
**NIP.19740424 200901 1 008**

**Dosen Pembimbing II**

  
**Dr. M. Amin Hariyadi, M.T.**  
**NIP. 19670118 200501 1 001**

**Tanggal, 6 Juli 2017**

**Mengetahui,  
Ketua Jurusan Teknik Informatika**

  
**Dr. Cahyo Crysdiان**  
**NIP. 19740424 200901 1 008**

**LEMBAR PENGESAHAN**

**DEVELOPMENT OF SMART WEB CRAWLER BY  
APPLYING BREADTH-FIRST ALGORITHM  
AND VECTOR SPACE MODEL**

**UNDERGRADUATE THESIS**

Created by :

Irvan Ariyanto  
NIM. 13650104

Telah Dipertahankan di Depan Dewan Penguji Skripsi  
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan  
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)  
Tanggal: 6 Juli 2017

Penguji Utama	:	Fatchurrochman, M.Kom NIP. 19700731 200501 1 002	(  )
Ketua Penguji	:	A'la Syauqi, M.Kom NIP. 19771201 200801 1 007	(  )
Sekretaris Penguji	:	Dr. Cahyo Crysdiان NIP. 19740424 200901 1 008	(  )
Anggota Penguji	:	Dr. M. Amin Hariyadi, M.T. NIP. 19670118 200501 1 001	(  )

Mengesahkan,

Ketua Jurusan Teknik Informatika  
Fakultas Sains dan Teknologi  
Universitas Islam Negeri Maulana Malik Ibrahim Malang



Dr. Cahyo Crysdiان  
NIP. 19740424 200901 1 008

## PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : IRVAN ARIYANTO

NIM : 13650104

Fakultas/ Jurusan : Sains dan Teknologi / Teknik Informatika

Judul Skripsi : **DEVELOPMENT OF SMART WEB CRAWLER BY APPLYING BREADTH-FIRST ALGORITHM AND VECTOR SPACE MODEL**

Menyatakan dengan sebenar-benarnya bahwa hasil penelitian saya ini tidak terdapat unsur-unsur penjiplakan karya penelitian atau karya ilmiah yang pernah dilakukan atau dibuat oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata hasil penelitian ini terbukti terdapat unsur-unsur penjiplakan, maka saya bersedia untuk mempertanggungjawabkan, serta diproses sesuai peraturan yang berlaku.

Malang, 6 Juli 2017  
Yang membuat pernyataan



Irvan Ariyanto  
NIM. 13650104

## Motto

**“Jika kamu bersungguh-sungguh, kesungguhan itu untuk kebaikanmu sendiri.”**



## Halaman Persembahan

- Skripsi ini aku persembahkan untuk Bapak, Ibu, dan Kakakku Tercinta yang Selalu Memberikan Dukungan dan Nasehatnya KEPADAKU-



## KATA PENGANTAR

### السَّلَامُ عَلَيْكُمْ وَرَحْمَةُ اللَّهِ وَبَرَكَاتُهُ

Puji syukur penulis panjatkan kehadirat Allah SWT, karena berkat rahmat dan hidayah-Nya, penulis dapat menyelesaikan skripsi dengan judul “Development of Smart Web Crawler by Applying Breadth-First Algorithm and Vector Space Model” ini dengan baik dan lancar, dimana skripsi ini disusun untuk memenuhi syarat untuk memperoleh gelar Sarjana Komputer (S.Kom) dari Jurusan Teknik Informatika di Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Dalam proses penyelesaian skripsi ini, penulis memperoleh banyak bantuan dari berbagai pihak. Baik berupa bimbingan, dorongan, petunjuk, kritik, saran, serta data-data baik secara tertulis maupun lisan.

Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Bapak Dr. Cahyo Crysdiyan selaku dosen pembimbing I serta Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang yang telah banyak memberikan waktu untuk membimbing, memotivasi dan mengarahkan selama proses pengerjaan skripsi ini.
2. Bapak M. Amin Hariyadi, M.T. selaku dosen pembimbing II yang selalu membimbing, memberi masukan dan solusi dalam penyusunan laporan skripsi ini.



3. Seluruh dosen dan staf Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang yang telah memberikan bimbingan dan ilmu sehingga menjadi bekal untuk penulis dalam menyelesaikan studi dan skripsi tepat pada waktunya.
4. Bapak dan Ibu tercinta serta keluarga besarku yang selalu memberikan doa, motivasi, pengalaman berharga, serta membiayai penulis dalam menuntut ilmu dan menyelesaikan skripsi ini.
5. Teman-teman Jurusan Teknik Informatika, khususnya teman-teman angkatan 2013, yang telah memberi motivasi, informasi, dan masukannya pada penulis sehingga dapat menyelesaikan skripsi ini dengan baik.

Semoga apa yang telah diberikan mereka kepada penulis, akan mendapat imbalan dari Allah SWT. Akhir kata, semua kritik dan saran atas skripsi ini akan penulis terima dengan senang hati, dan akan menjadi bahan pertimbangan bagi penulis selanjutnya untuk menyempurnakan skripsi ini.

وَلَسَّلَامٌ عَلَيْكُمْ وَرَحْمَةُ اللَّهِ وَبَرَكَاتُهُ

Malang, 6 Juli 2017

Penulis

## TABLE OF CONTENTS

HALAMAN JUDUL .....	i
LEMBAR PERSETUJUAN .....	ii
LEMBAR PENGESAHAN .....	iii
PERNYATAAN KEASLIAN TULISAN .....	iv
TABLE OF CONTENTS.....	ix
LIST OF TABLES.....	xi
LIST OF FIGURES .....	xii
ABSTRAK.....	xiii
ABSTRACT.....	xiv
ملخص .....	xv
CHAPTER I.....	1
INTRODUCTION .....	1
1.1. Research Background.....	1
1.2. Research Question.....	3
1.3. Research Objectives .....	3
1.4. Research Scope .....	3
CHAPTER II.....	4
LITERATURE REVIEW .....	4
2.1. Web Crawler .....	4
2.2. Information Retrieval .....	9
CHAPTER III .....	15
RESEARCH METODOLOGY .....	15
3.1. System Design.....	15
3.1.1. Process Design.....	15
3.1.2. Interface Design.....	31
3.2. Experiment.....	33
CHAPTER IV .....	34
IMPLEMENTATION AND TESTING .....	34
4.1. Implementation.....	34
4.1.1. Web Crawler .....	34
4.1.2. Process preprocessing documents.....	38

4.1.3.	Pembobotan TF.IDF .....	43
4.1.4.	Thesaurus .....	49
4.1.5.	Cosine Similarity .....	50
4.1.6.	Design and Implementation GUI .....	53
4.2.	Testing Phase.....	56
4.3.	Testing Result and Analysis .....	56
4.3.1.	Crawler Testing Result .....	56
4.3.2.	Information Retrieval Experiment Result.....	58
4.4.	Integration of Research with Al-Quran.....	68
CHAPTER V	.....	71
CONCLUSION AND SUGGESTION	.....	71
5.1.	Conclusion.....	71
5.2.	Suggestion .....	71
REFERENSI	.....	72

**LIST OF TABLES**

Table 3.1 Format list of web crawler .....	18
Table 3.2 BF Crawling queue .....	19
Table 4.1 Result of Document Frequency.....	48
Table 4.2 Result of Inverse Document Frequency.....	49
Table 4.3 Result of TF.IDF .....	50
Table 4.4 Result of Weight Document & Long Vec.....	50
Table 4.5 Result of Cosine Similarity.....	52
Table 4.6 Result of Sorting Cosine Similarity .....	53
Table 4.7 Crawling testing result .....	57
Table 4.8 Supervised without query expansion .....	59
Table 4.9 Form testing without query expansion.....	60
Table 4.10 Result testing without query expansion.....	61
Table 4.11 Calculation testing without query expansion.....	62
Table 4.12 Supervised with query expansion .....	63
Table 4.13 Form with query expansion .....	64
Table 4.14 Result with query expansion.....	65
Table 4.15 Calculation with query expansion.....	66
Table 4.16 Result of IR Syatem.....	67

## LIST OF FIGURES

Figure 2.1 Map of literature .....	14
Figure 3.1 System Design .....	16
Figure 3.2 BF Crawling Process Sequence .....	19
Figure 3.3 Architecture of Crawling Mechanism .....	20
Figure 3.4 Flowchart Tokenizing .....	21
Figure 3.5 Flowchart Stopword Removal .....	22
Figure 3.6 Flowchart Thesaurus .....	24
Figure 3.7 Vector term weighting .....	24
Figure 3.8 Flowchart TF .....	25
Figure 3.9 Flowchart IDF .....	26
Figure 3.10 Flowchart TFIDF .....	28
Figure 3.11 Flowchart Get Long Vector Space Model .....	30
Figure 3.12 Flowchart Cosine Similarity .....	31
Figure 3.13 Form Training .....	32
Figure 3.14 Form Testing 2 .....	32
Figure 4.1 Source Code Web Crawler .....	35
Figure 4.2 Source Code Tokenizing .....	39
Figure 4.3 Source Code Stopword removal .....	41
Figure 4.4 Source Code Stemming .....	42
Figure 4.5 Source Code TF (Term Frequency) .....	44
Figure 4.6 Source Code IDF (Inverse Document Frequency) .....	46
Figure 4.7 Source Code TF.IDF .....	46
Figure 4.8 Source Code Thesurus .....	50
Figure 4.9 Source Code get Long Vector .....	51
Figure 4.10 Source Code Cosine Similarity .....	51
Figure 4.11 GUI Web Crawler .....	53
Figure 4.12 GUI Information Retrieval .....	55

## ABSTRAK

Irvan Ariyanto, **Development of Smart Web Crawler by Applying Breadth-First Algorithm and Vector Space Model**

Pembimbing I : Dr. Cahyo Crysdiان

Pembimbing II: M. Amin Hariyadi, M.T.

**Kata Kunci** : *Web crawler, Information Retrieval, Breadth-First Algorithm, Vector Space Model*

*Information retrieval system* merupakan sistem yang digunakan untuk menemukan informasi yang relevan dengan kebutuhan dari penggunanya.. Proses *retrieval* adalah proses untuk menghitung kemiripan *query* terhadap dokumen, perhitungan kemiripan menggunakan konsep *vector space model* dengan mencari nilai *cosine similarity*. *Query* yang dilakukan proses retrieval secara langsung membarikan hasil yang kurang bagus, sehingga peneliti melakukan proses ekspansi pada *query*. Ekspansi *query* dilakukan dengan mencari keterkaitan kata dalam *query* berdasarkan thesaurus. Hasil penelitian menunjukkan dengan melakukan ekspansi pada *query* dapat meningkatkan presisi sebesar 7% dan akurasi sebesar 0.6%.

## ABSTRACT

Irvan Ariyanto, **Development of Smart Web Crawler by Applying Breadth-First Algorithm and Vector Space Model**

Supervisor I : Dr. Cahyo Crysdiان

Supervisor II: M. Amin Hariyadi, M.T.

**Keywords** : *Web crawler, Information Retrieval, Breadth-First Algorithm, Vector Space Model*

Information retrieval system is a system used to find information relevant to the needs of its users. Retrieval process is a process to calculate the resemblance of query to the document, the calculation of similarity using the concept of vector space model by looking for cosine similarity value. Queries performed by the retrieval process directly result in less good, so the researchers make the process of expansion in the query. Query expansions are done by searching for a word connection in a query based on a thesaurus. The results showed that by expanding the query can increase the precision by 7% and the accuracy of 0.6%.

## ملخص

إيرفان أريانتو، تطوير ويب الذكية الزاحف بواسطة تطبيق اتساع والعشرين خوارزمية وناقلات الفضاء

نموذج

مشرف ١: كاهي كريسيدين

مشرف ٢: محمد أمين هريدي

كلمات البحث: زاحف الويب، استرجاع المعلومات، خوارزمية اتساع - الأولى، نموذج متجه الفضاء

انفورماتيون ريتريفال سيستيم هو النظام الذي يستخدم للعثور على المعلومات التي هي ذات الصلة لاحتياج مستخدمها. عملية ريتريفال هي عملية لحساب التشابه قويري إلى مستند، وحساب تشابه باستخدام مفهوم فيجتور سيفسي موديل من خلال إيجاد قيمة سيميلاريطي كوسين. قويري الذي يتم ريتريفال المباشرة يعطي نتائج أقل جيدة، بحيث الباحث بإجراء عملية التوسع في قويري. ويتم التوسع قويري عن طريق البحث الكلمات في قويري بواسطة تيسوروس. أظهرت النتائج من خلال توسيع قويري يزيد دقة ٧٪ ودقة ٦,٠.



# CHAPTER I

## INTRODUCTION

### 1.1. Research Background

Based on data from Netcraft, the number of websites reaches more than 1 billion hostname, and active website reaches more than 100 million in August 2016 across the world. That number has increased greatly since August 1995. when the number of websites only reached more than 10 thousands hostname (Netcraft,2016).

Currently there are overwhelming number of website. Imagine if someone have to go manually by search one by one from each website. It become to do some work consuming a lot of time. The future it is necessary to create a machine that capable of analyse pages of websites around the world, and establishing a system of information retrieval to simplify the search of information require by the user.

Browsing and retrieving of web pages through the building a web crawler. Web crawler is a program that automatically browsing the web. As an important part of the search engines, it is designed to download the web page for the search engines. General web crawler begins with one or several URLs of the start page, gets the start page's URL list while crawling the web page, the web crawler keep extracting new URLs from the current page, and putting them into the queue of URLs waiting to be fetched until meeting the system's stop condition (Bai, Xiong, Zhao, & He, 2014). Website page that has been collected will then be extracted to be taken article. All collected articles assigned weights use TF-IDF method.

A desired keyword the user will look for synonyms and the linkage word (thesaurus) to add a keyword. Because execute input query directly, causing low quality search result. Once the keywords are formed is weighted for the keyword, and then calculated the similarity of value weighting articles and keywords using the Vector Space Model, the results of similarity can be obtained in accordance with the article's ranking keywords.

Based on (Rasyidi & Romadhony, 2013) research on the application of thesaurus for query expansion in Indonesian Hadith Retrieval System. From the results, the accuracy for Precision increased by 34% between searches with query expansion using a thesaurus and without query expansion using thesaurus on Hadith Retrieval System.

يَا أَيُّهَا الَّذِينَ آمَنُوا إِنْ جَاءَكُمْ فَاسِقٌ بِنَبَأٍ فَتَبَيَّنُوا أَنْ تُصِيبُوا قَوْمًا بِجَهَالَةٍ فَتُصْحَبُوا عَلَىٰ مَا فَعَلْتُمْ نَادِمِينَ (6)

Means: “O ye who believe! If an evil-liver bring you tidings, verify it, lest ye smite some folk in ignorance and afterward repent of what ye did.”(QS. Al-Hujurat:6).

Allah SWT. Command (the believers) to thoroughly examine the message of the wicked, and let them be cautious in accepting it and do not take it for granted, which consequently will reverse the reality. The person who receives just the news from him, means the same as following in his footsteps. While Allah SWT. Has forbidden the believers from following the path of the corrupted.

Departing from this understanding there are some scholars who forbid us to receive news (history) from unknown persons, because perhaps he is a wicked man. But some other scholars would accept it on the grounds that we were only

ordered to examine the truth of the wicked news, while the unknown (majhul) still has not proved his wickedness because he is unknown.

So in this study, researchers will create a system of information retrieval on article websites that will use a thesaurus for query expansion method TFIDF and Vector Space Model. The study is expected to improve the accuracy of information retrieval on article sites. So that the system information retrieval that is built to provide appropriate information and in accordance with the Surah al-hujurat: 6 above in order to examine the information obtained.

#### 1.2. Research Question

- a. How fast does the proposed crawler derive full content of each website?
- b. How high the precision, recall and F-measure of the proposed method to retrieve desired article from the Internet utilizing the combination of vector space model and cosine similarity?

#### 1.3. Research Objectives

- a. Measuring speed the proposed crawler derive full content of each website
- b. Measuring high the precision, and Accuration of the proposed method to retrieve desired article from the Internet utilizing the combination of vector space model and cosine similarity

#### 1.4. Research Scope

- a. The research focuses to retrieve document only in Bahasa Indonesia.
- b. Duration time for document retrieval from the Internet is limited due to bandwidth constrain of Internet provider.

## CHAPTER II

### LITERATURE REVIEW

#### 2.1. Web Crawler

Qian, Zhang, & Zhao (2014) discussed a topic-specific intelligent Web crawler based on Web content and structure mining. The method takes advantage of the characteristics of the neural network and introduces the reinforcement learning to find the relativity between the crawled web pages and the topic. When calculating the correlation, they just select the important tags of HTML makeup of the Web page, to analyze the web page's content and structure. The experiments show that their method improves the efficiency and accuracy clearly. They analyzed the development of topical crawler methods has received significant attention in nowadays. And then they introduced and performed their topic specific intelligent Web crawling algorithm based on Web content and structure mining. The method takes full advantage of the characteristics of the neural network and parallel calculation. They introduce the reinforcement learning to calculate the relativity between the crawled web pages and the topic. The experimentation proved that their method can improve the efficiency and accuracy of collected information clearly.

Sharma & Gupta (2015) served a various web crawling strategies have been studied by keeping different perspectives of the crawler in mind. As more and more information becomes available, it will be progressively more important to collect it effectively. Graph traversal can be the other term for Web Crawling. The web includes nodes and hyperlinks. It is basically a large graph in which pages

work as nodes and hyperlinks act as edges. To start the crawling process, these nodes and edges play a vital role. A crawler is an essential component of Search Engine that downloads the web pages over www by following the hyperlinks in the pages. It starts with a few of the seed urls and then reaches out to the others by following the links. The process of fetching a page and extracting the links within it is analogous to expanding a node in graph search. In a short period of time, millions of websites are contacted by the crawler and in this the web crawler consumes extremely large network, storage and memory resources. The limit of existing hardware is pushed by these loads and this task should be carefully coordinated and partitioned among processes.

Siddiqui (2015) developed a web crawler which used content and structural similarity of web pages to order them. The content similarity is calculated on basis of frequency of keyword among the crawled pages, while the structural similarity is calculated by pairing in-neighbors to a node, as defined in SimRank algorithm. The crawler eliminates duplicate URLs, thus providing with unique URLs. A set of similar websites were given as input to crawler to crawl and the outputs were compared on the set of parameters such as top URLs, precision, crawling time, ordering time and similarity scores. The developed Web crawler shows web pages that are relevant to a query on basis of content and structural similarity.

Gupta & Anand (2015) reviewed of focused crawler approaches have been presented which is classify in to five categories: Priority base crawler, Structured base crawler, Leaning base crawler, Context base crawler and Other focused crawler. In terms of precision and efficiency General Crawler has some limitation because of its generality, no specialty. Precision and recall of expert search on

web is improved by focused crawler. With the help of Focused crawler only selective and retrieve relevant page is collected in of all the pages. In search engine application Web crawling is one of the main components. Comparison is done between standard and References focused web crawlers to understand which one is better and also discussed about the merits of various approaches like priority based as well as contextual based focused crawling. The advantages of focused crawler are that we spend less money, time & effort processing WebPages that are most unlikely to be of value or worth. Also the manner in which focused crawler proceeds has been illustrated and documented on their paper.

Bahrami, Singhal, & Zhuang (2015) proposed a web crawler architecture based on cloud computing. The proposed web crawler is implemented in Windows Azure Cloud Platform. They used Azure Cloud Queue to make a connection among distributed web crawler's agents globally. Each agent provides its own results by fetching a URL from the queue and then by crawling the fetched URL. Each agent could add indexed pages by MapReduce technique in Azure Table storage which is based on a NoSQL database. Moreover, they suggested Azure Blob for collecting and storage unstructured data on cloud servers. Blob allows a web crawler to collect a massive amount of unstructured data, such as video, binary files and images. They discussed and compared our proposed web crawler architecture against traditional distributed systems and we explained the advantages of the proposed architecture.

Agre & Mahajan (2015) introduced extraction of URLs based on keyword or search criteria. It extracts URLs for web pages which contains searched

keyword in their content and considers such pages only as important and doesn't download web pages irrelevant to search. It offers high optimality comparing with traditional web crawler and can enhance search efficiency with more accuracy. Main advantage of using keyword focused web crawler over other web crawler is that it works intelligently, efficiently and doesn't need relevance feedback. It reduces number of extracted web pages thus it takes less time for crawling as it downloads relevant web pages only. Intension is to retrieve relevant web pages and discards the irrelevant web pages. They have developed an ontology based crawler using best knowledge path which fetches web pages according to relevancy decision mechanism. The added of delivered by the effort are reduction in number of extracted searched web page URLs and reduction in turnaround time for crawling process. This algorithm is potential and helpful to solve day to day troublesome for internet user and can save searching time.

Kapoor (2016) attempted to improve upon the time-efficiency of a map-reduce based crawler. Using bloom filter, based upon comparison against different parameters our results indicate the efficiency and effectiveness of the proposed approach. The anatomy of Apache Nutch has been studied and a proposed technique of bloom filters has been implemented in its architecture. The technique was found to be more efficient than the existing approach in case of generate phase but less efficient for update phase. This variant of Nutch could be very useful for developers, researchers, internet surfers in general who want faster offline dumps or faster web crawling technique. It will scale very well even for more than one website in seeds, and by allowing inter-host crawling. The bloom approach can be further enhanced by populating bloom filter only after the URL

has been fetched in fetcher phase (rather than populating beforehand in generate phase). Some variants of bloom filter like dynamic bloom filter (to avoid predefining the size of filter) and counting bloom filter (to support incremental crawling) make for an interesting research work. Their approach currently tested on a standalone system could also be extended to distributed framework. Trying bloom filters on some other cryptographic and non-cryptographic hash functions, individually as well as in combination may also be experimented. The combination of bloom filters and web crawling can be explored even for the deep web to keep track of the similar search interfaces, database content, thus helping in query formation.

Lawankar (2016) reviewed some of the techniques which are used for web crawling optimization. To overcome repetitive and irrelevant links in web crawler result, many techniques are used, of which techniques like maintaining personalized history, collaboration, classification which helps the focused crawler to distribute the data into various section for optimization are discussed in this paper. For optimization, algorithms like Page Rank, FP growth, genetic algorithm, or Apriori algorithm can be used to rank and index the web pages. In personalized history all the information of the user is stored, on that basis links can be optimized and only the profile related links are served in result. In classification, there are various pre-defined classes in which links are classified and more relevant contents are shown in search engine result. In collaboration technique only the frequently used links are stored by the web crawler which are directly served on the search engine by using genetic algorithm. In these techniques, mostly the page rank algorithm and normalization techniques are used to optimize



the URLs. By using these techniques specific topic related search and different URLs with similar text can be sort out which will increase crawling efficiency and eliminate repetitive display of same result.

## 2.2. Information Retrieval

Rasyidi & Romadhony (2013) researched on the application of thesaurus to developed request expansion in Indonesia Hadith Retrieval System. Dictionary construction is done automatically by using analysis of co-coccurrence followed by a manual validation process. Documents thesaurus will be used in the process of expansion of demand when seeking process. The results of the study stated that the implementation of the query using thesaurus expansion can improve the quality of search results. Quality improvement is particularly apparent in the measurement of recall. This means that query expansion has been successful to find more relevant documents. As for precision, although the amount is not significantly different, there is still no improvement. So if we consider the recall is more important, or the goal is to get more relevant documents, apply the query expansion using a thesaurus is a good choice.

Y. Gupta, Saini, & Saxena (2015) proposed and implemented a new fuzzy logic based ranking function to enhance the performance of Information Retrieval system. The proposed ranking function is based on the computation of different terms of term-weighting scheme such as term frequency, inverse document frequency and normalisation. Fuzzy logic is used at two levels to compute relevance score of a document with respect to the query in present work. All the experiments are performed on CACM and CISI benchmark data sets. The experimental results reveal that the performance of their proposed ranking

function is much better than the fuzzy based ranking function developed by Rubens along with other widely used ranking function Okapi-BM25 in terms of precision, recall and F-measure.

Khari (2016) presented an analysis of the three basic models of information retrieval. Vector Space Model, Latent Dirichlet Allocation Model and Latent Semantic Indexing Model. Analysis of the various IR models where each one of them has their set of advantages and disadvantages. While the VSM can deal with partial matches, the LDAM and LSIM can deal with polysemy and synonymy. The major application domain of each one of these models is also mentioned. The future scope of these models lies in improving their efficiency and effectiveness in various domains. The VSM can be improved by eliminating the highly common terms and stop words. Other methods that can be adopted for doing the same include stemming the terms of their corresponding roots, using the passage vectors in the case of long documents and defining a cosine threshold for retrieval. To improve the performance of LDAM, some alterations were done which resulted in the development of Probabilistic Latent Semantic Model. It offers various advantages over LSIM such as greater efficiency, increased throughput, and lesser computational complexity.

Science, Technology, & Samarahan (2016) discussed an approach to designing and developing new representation Information Retrieval System which uses ontologies. Experiments have been performed on keyword-based approach and ontology-based approach to validate the retrieval of documents by using five queries. Preliminary experimental results show that the proposed ontologies improve the precision and recall of the documents retrieval. As conclusion of this

work they would like to highlight that semantic retrieval approach can provide better search capabilities, thus achieving an improvement over keyword-based retrieval by means of the introduction and exploitation of ontologies.

Alshari, Azman, Mustapha, Doraisamy, & Alksher (2016) investigated the problem of predicting rating based on the comments from the Internet users. A classifier based on information retrieval and the Vector Space Model are proposed to solve the problem. The effect of integrating sentiment analysis model into the classifier is also investigated. The outcome promising showing that sentiment analysis has positive effect to the performance of the classifier in predicting rating form textual comments of the users. In future, more models of information retrieval, such as the probabilistic model or the statistical language model, can be adopted as the classifier to improve the prediction performance. In addition, the sentiment analysis model can be improved with advanced NLP techniques.

Ayetiran, Aruleba, Ekong, & Science (2016) proposed a new document ranking technique in IR with the intention of retrieving context information ranked according to information relevance. The proposed adaptive DROPT technique is to improve the relevance of retrieved documents. The DROPT technique adapts itself with individual user information needs based on environment and search context. In this respect, their algorithm can judge the relevance of the current search results according to the relevance weight of the retrieved documents. The technique demonstrated in providing a limited number of ranked documents in response to a given users' query. The new DROPT technique combined approaches suitable for user profiling and mining of user interest for the enhancement of IR system performance. The DROPT technique is

designed purposely to overcome some of the limitations (e.g. low precision and recall, and not adaptive to users) of existing traditional ranking algorithms that ignore the semantic analysis of the document itself.

Ahmad & Ali (2016) proposed a framework for tweets retrieval from Twitter. The system acquires information from Twitter by using Twitter search API and develops a corpus of user's contents by removing noisy and ambiguous elements from the retrieved collection of tweets. Their system accepts user's query and return the results relevant to the query in order to their decreasing similarity score calculated by cosine similarity measure. There are some limitations of their system, it uses only vector space model for information retrieval and for relevance ranking only cosine similarity is used.

V. K. Sharma & Mittal (2016) proposed a Wikipedia API based query translation approach. Queries are tokenized and multi-words query terms are created using N-gram technique. Wikipedia title and inter-wiki link features are exploited for query translation. Target language documents are retrieved using vector space retrieval model and BM25 retrieval algorithm. Experiment results shows that the proposed approach achieves better results without exploiting any language resources. Various Wikipedia issues are identified during experimentation. Poor coverage of Hindi Wikipedia article, unavailability of inter-wiki links, wrong target language articles have very bad Impact. Wrong target language articles are identified by cross verification of inter-wiki links. Other Wikipedia features such as the article, category, infobox, subsection, hyperlinks will also be utilized in future to solve poor coverage issue of Wikipedia knowledge base.

Kauer & Moreira (2016) proposed SABIR a method for polarity classification of tweets based on an Information Retrieval system. Their goal was to leverage the information on the class of (labelled) similar tweets to classify new unlabelled posts. They have proposed and tested 24 features that are based solely on the ranking provided by a search engine in response to queries consisting of the tweets we wish to classify. Since the labels for the indexed posts can be generated without human intervention, their method can be completely automatic. Comparative experiments on a number of baselines have shown promising results. The accuracy of SABIR is not statistically different from the accuracy of their best baseline. In comparison to the state-of-the-art results, they reach similar scores relying solely on the 24 proposed features. The scope of this article was limited, and therefore number of alternatives are still open for exploration.

Goswami, Gaussier, & Amini (2017) found good IR scoring functions by exploring the space of all possible IR functions. Earlier approaches to do so however only explore a small sub- part of the space, with no control on which part is explored and which is not. They aim here at a more systematic exploration by first defining a grammar to generate possible IR functions up to a certain length (the length being related to the number of elements, variables and operations, involved in a function), and second by relying on IR heuristic constraints to prune the search space and filter out bad scoring functions. The obtained candidate scoring functions are tested on various standard IR collections and several simple but promising functions are identified. They perform extensive experiments to compare these functions with classical IR models. It is observed that these functions are yielding either better or comparable results. They also compare the

performance of functions satisfying IR heuristic constraints and those which do not the former set of functions clearly out performs the latter, which shows the validity of IR heuristic constraints to design new IR models.

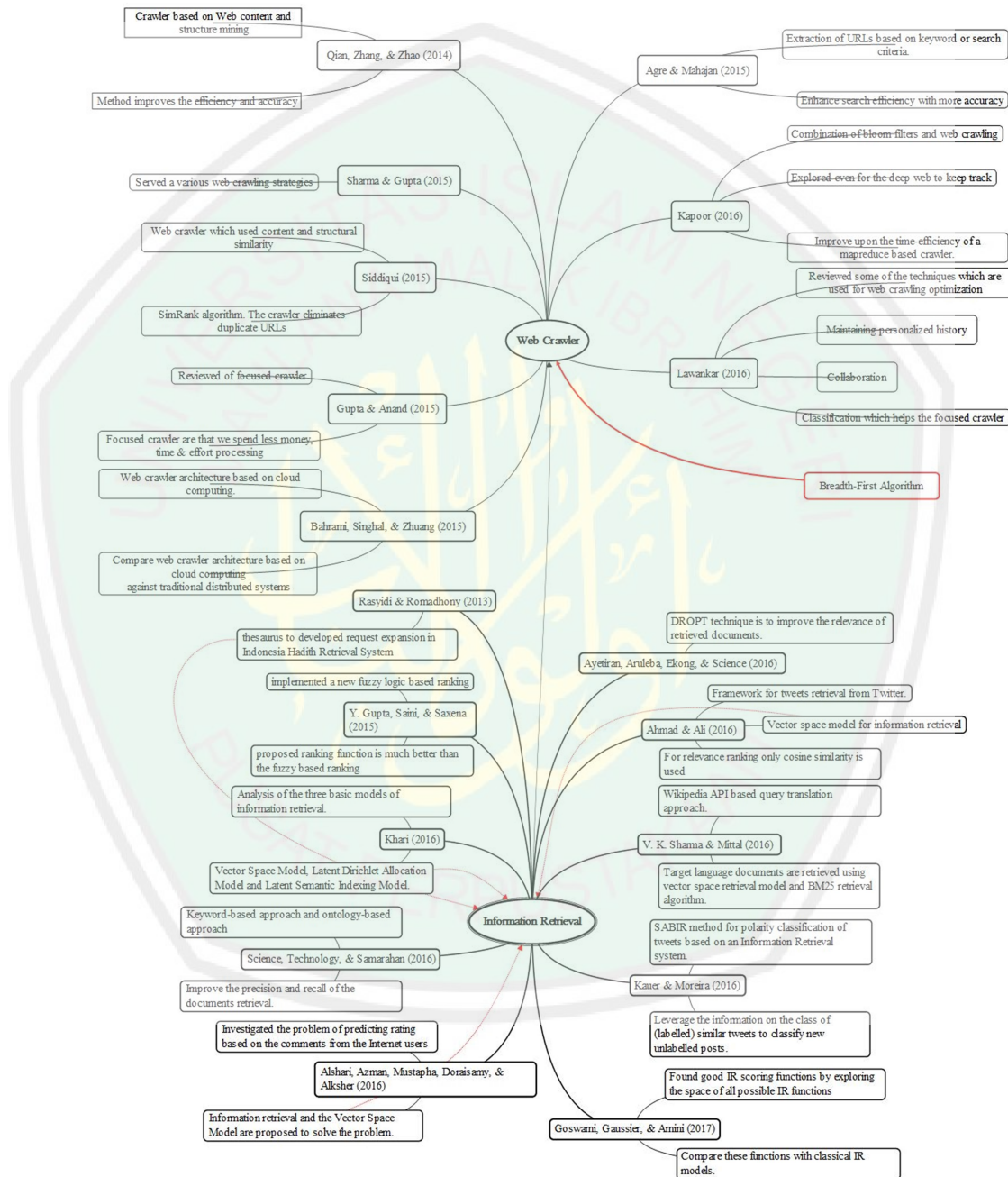


Figure 2.1 Map of Literature

## CHAPTER III

### RESEARCH METODOLOGY

#### 3.1. System Design

##### 3.1.1. Process Design

System design from this research is given in the Figure 3.1. The elaboration of the process in Figure 3.1 is given as follow : 1) Develop web crawler apply breadth-first algorithm to collect articles from website. Breadth-First algorithm will be make web crawler start from the first node from the left until right and will be continued to next node until the end of node to collect data. 2) Documents article as a dataset through its initial stages of pre processing. The documents contains the HTML code will be extracted. It aims to clean The HTML code and get the contents are needed, from this research we get the all URL from document to use in web crawler process and article content to use in information retrieval system. 3) In the preprocessing stage the article content of document is cut into pieces word (tokenizing), eliminating the word including stoplist (stopword removal), and converted into the root words (stemming). 4) After that, do the words of the term weighting the results of the preprocessing stage, from the calculation of the value of TF (Term Frequency) and IDF (Inverse Document Frequency) multiplied use Equation 3.1. 5) The weight of each term is modeled in vector space model to the calculation of the distance between query from user and each of documents using the cosine similarity . Each vector will be measured the similarity use Equation 3.2.

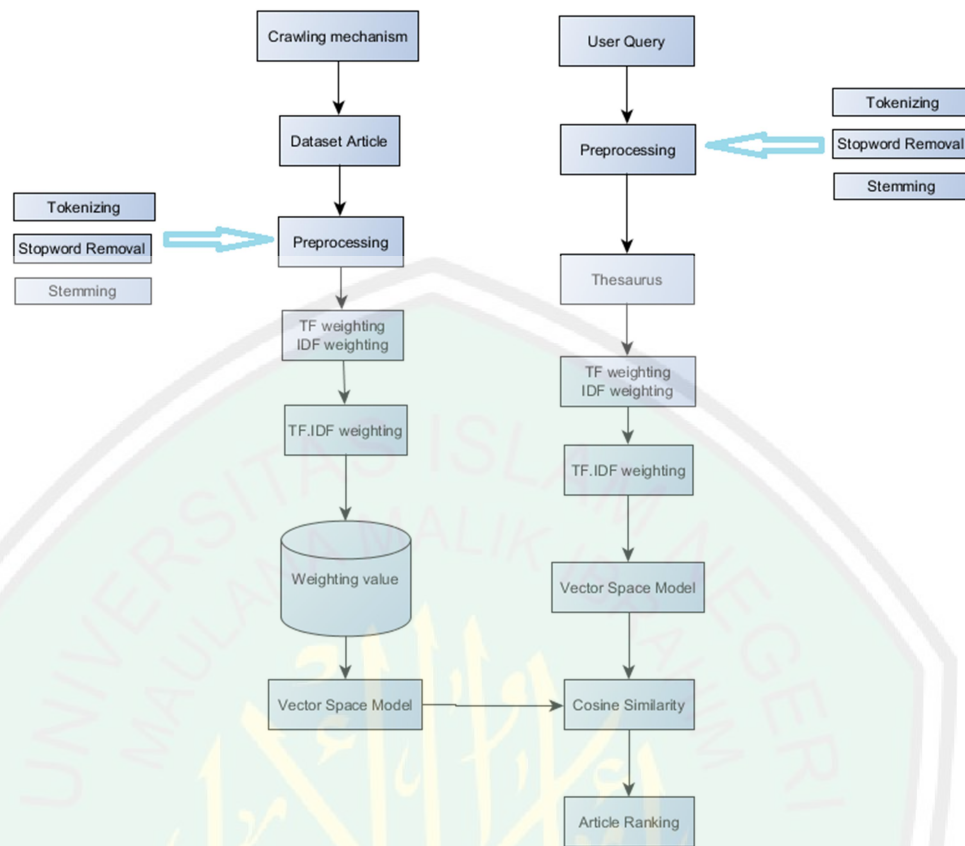


Figure 3.1 System Design

### ***Crawling Mechanism***

Web crawling is done to collect articles from various websites. Article collection process performed with web crawlers. Web crawlers are programs that automatically visit and download web pages, following hyperlinks in web pages.

The crawling process increases web traffic to a large extent. In order to minimize the network traffic, the web administrator may implement robot exclusion protocol on their websites. So do intervals on each request so that the traffic on the website is not too big.

Crawler can be categorized mainly into three general types batch crawler, incremental crawler and focused crawler.



A batch crawler crawls for a set of web pages periodically. For instance, the crawler is configured to crawl a set of 200 web pages every month. At the beginning of each month, the crawler will crawl the set pages in order to update the pages collection in its repository. After it has accomplished its task, it proceed to its next task.

An incremental crawler is a type of crawler that will never stop crawling, as it regards that the web is constantly updating and pages fetched will eventually expire. Besides crawling new web pages, an incremental crawler has a built-in scheduler that schedules for the re-visiting of previously crawled websites in order to fetch the site contents for any updates. This differs itself from other batch crawlers, which does not have any re-visit policy implemented in it.

While batch crawler and incremental crawler crawl pages regardless of content relevancy, a focused crawler seeks to crawl relevant and topic-specific web pages as requested by individuals or organizations. Such task can be accomplished by reading the H1 tag in the web page document, counting number of appearance of certain keywords in a document or implementing learning automata in the crawler itself.

The mechanism of crawler from this research is given in the Figure 3.2. The elaboration of the process in Figure 3.2 the given as follow : 1) User set the starting URL to crawl and the address of URL inserted to a list of addresses. The format of the list containing URL address is given as Table 3.1.

Table 3.1 Format list of web crawler

URL	Visited Status
example.com	visited
Example.com/news	Not visited

2) Web crawler started crawler process by sending request the page from URL to web server. The web crawler validate the HTML code and links found on websites visited. 3) After performing validation, web page and link is downloaded and then parses it into the main storage area. The data carried by the web crawler is simple only in the form of text and metadata. 4) While data links found on web pages visited will be placed in a data repository URL into the web crawler queue next visit. This step use Breadth-First algorithm to determine the next URL will be visit. This algorithm requires a queue q to save the node that has been visited. The nodes is needed as a reference for the visit the nodes Neighboring. Each node has visited entered into the queue only once. This algorithm also requires a Boolean table to store the nodes that have been visited so no vertex visited more than once. Breadth-first process of crawling on all nodes that are at the same level or hierarchy before proceeding searching on the node at the next level. BF crawling process sequence shown in Figure 3.2 are: A, B, C, D, E, F, ..

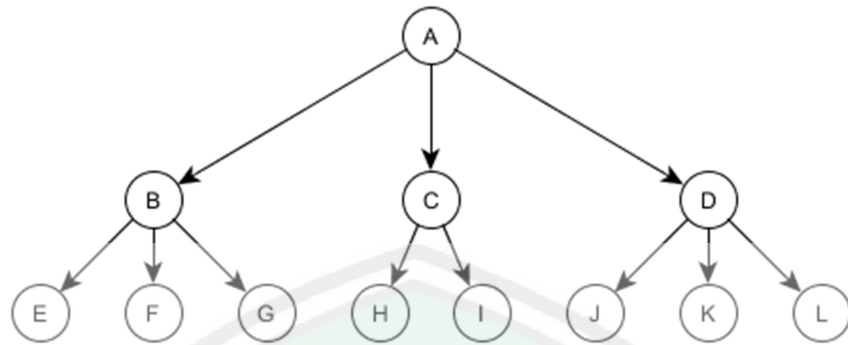


Figure 3.2 BF Crawling Process Sequence

Then using Breadth First algorithm, which will be crawled these are as follows:

Table 3.2 BF Crawling Queue

0	A = example.com	Visited
1	B = example.com/page/1	Visited
2	C = example.com/page/1	Visited
3	D = sample.com/	Visited
4	E = example.com/finance.html	Not visited
5	F = example.com/sport.html	Not visited
6	G = example.com/news.html	Not visited
7	H = example.com/politik.html	Not visited
8	J = sample.com/article/1	Not visited
9	K = sample.com/article/2	Not visited
10	L = sample.com/article/3	Not visited

- 5) Simultaneously web crawlers visit sites whose address is contained in the URL queue until the data runs out or terminated by the administrator.

The architecture of crawling mechanism is given in the Figure 3.3:

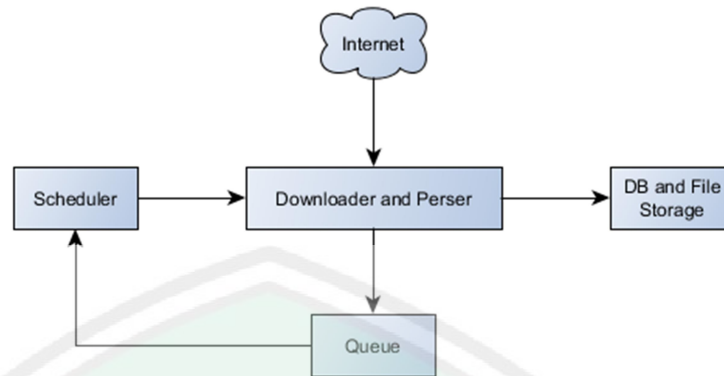


Figure 3.3 Architecture of Crawling Mechanism

#### ***Dataset***

The data used in this study is the document article in Bahasa Indonesian obtained by crawling the internet. The document article contains the HTML code will extract to get all of the url in each page and article Bahasa Indonesia without HTML code. Url extract to be used in the process of crawling and the article Bahasa Indonesia without HTML code from extracted will be save to the new file text. The new file is to be calculated weight.

#### ***Input***

The keyword that is used as an input are in the from of a single word, phrase or stanza. These keywords will be used as query. Then query work be developed by doing a query parsing process after the initial query through the process of preprocessing.

#### ***Preprocessing***

Each of these documents through the preprocessing stage. Implementation preprocessing consists of several stages, including tokenizing, stopword removal and stemming.

Tokenizing is done to break down the entire contents of the document into a single word. The following is a flow chart of the tokenizing process:

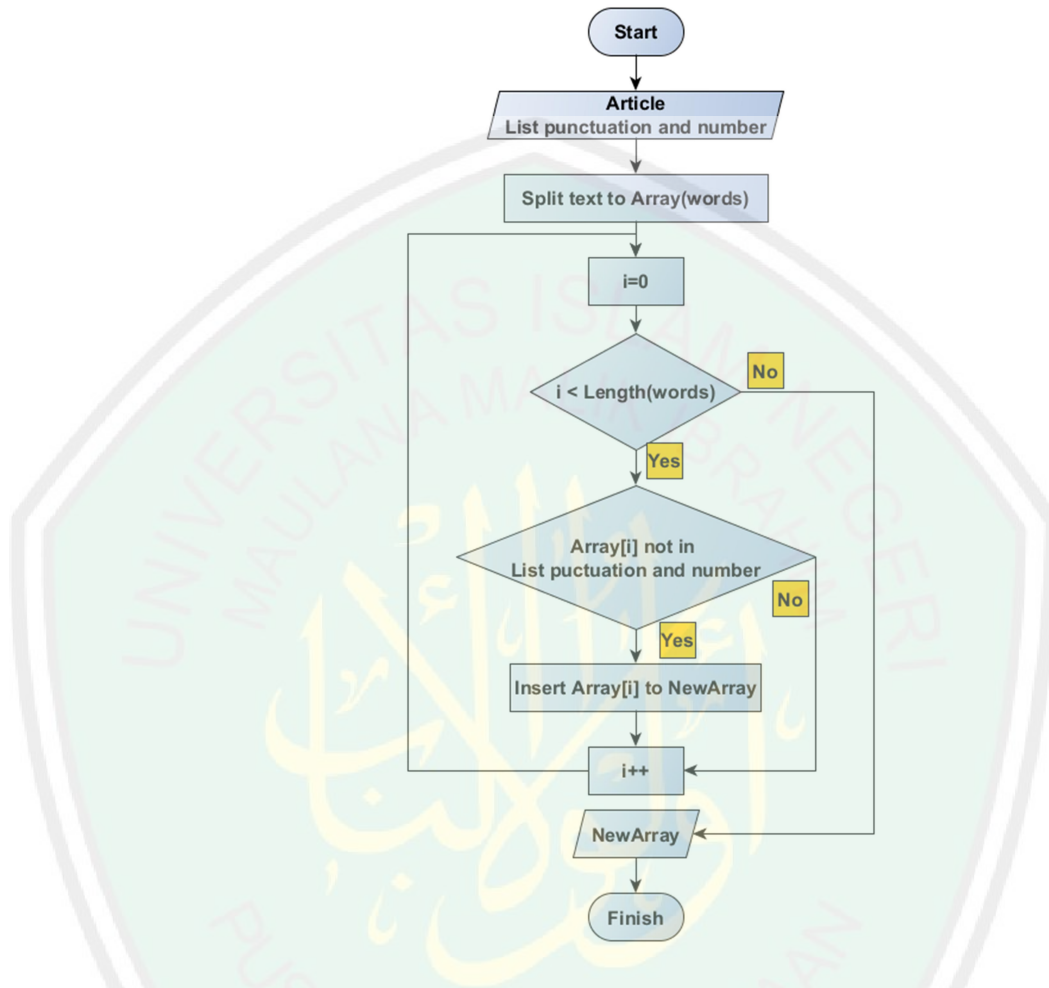


Figure 3.4 Flowchart Tokenizing

After the documents are broken down into single word then the stopword removal process. The elimination phase stopword done to eliminate words that often appear in the document but has no meaningful value in a document. . The following is a flow chart of the stopword process:

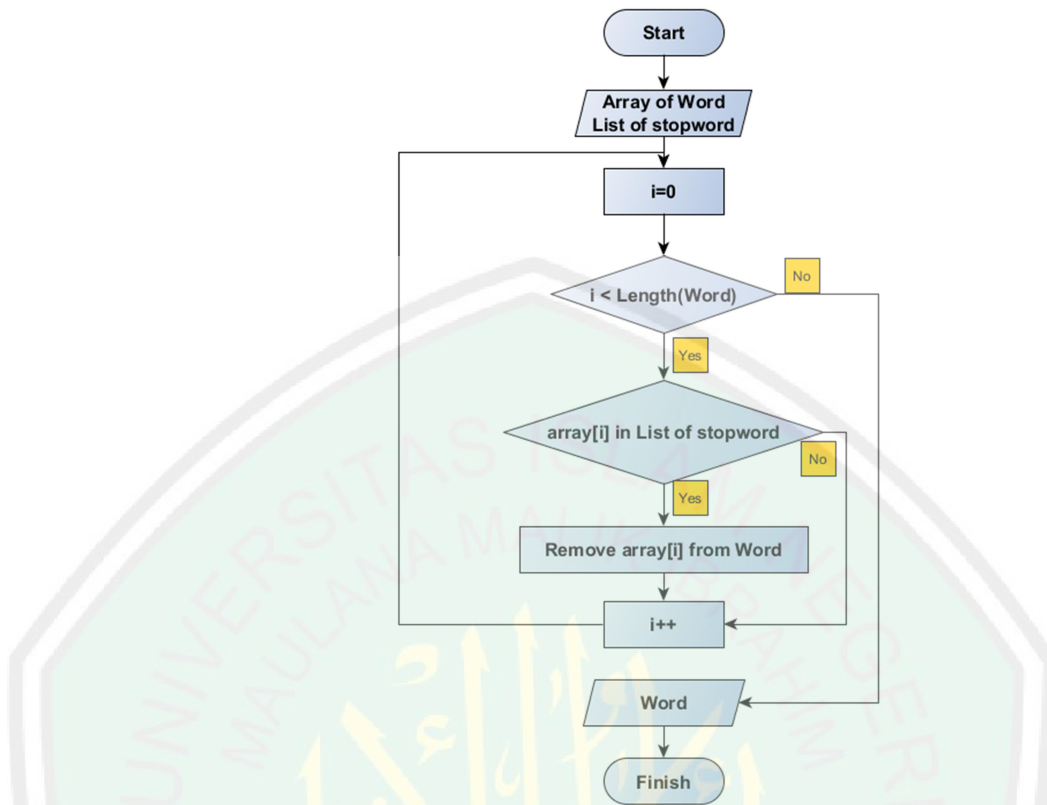


Figure 3.5 Flowchart Stopword Removal

The next stage is stemming used to obtain the basic words of each word by searching the base word (root) and removal affix and suffix. In the implementation of this stemming Sastrawi Stemmer that are used in literary library (<http://giuhub.com/saastrawi/>).

### ***Thesaurus***

Thesaurus is a set of terms and concepts which has similar or most similar meaning. Those definition is the closest to main purpose of using thesaurus in Information Retrieval System, because basically thesaurus could contains all types of relationships between words. The use of thesaurus in Information Retrieval System is very important, because it can improve the relevance of search result. Thesaurus used in indexing and searching process. There are two ways to build

thesaurus : manual and automatic. Manually built thesaurus can have very comprehensive content, thus need much effort. For a specific domain, manually built thesaurus will improve the system performance a lot. On the other side, automatically built thesaurus can be done in three ways. The first way is building thesaurus from document collection. The second way is merging existing thesaurus, and the last is capturing information from users.

One way to increase retrieval system performance is by implementing query expansion. In query expansion, users give additional input on query words or phrases. Some search engine such as Google provide candidate terms to be added by user. The source of candidate terms could be thesaurus, which contains synonym or related terms with query .

A thesaurus is a reference work that lists words grouped together according to similarity of meaning containing synonyms and generally lists them in alphabetical order. Unlike dictionary, thesaurus does not give the definition of words. Purpose of this step is to add of some term or word in the query in order to improve the performance of information retrieval from thesaurus. . The following is a flow chart of the thesaurus process:

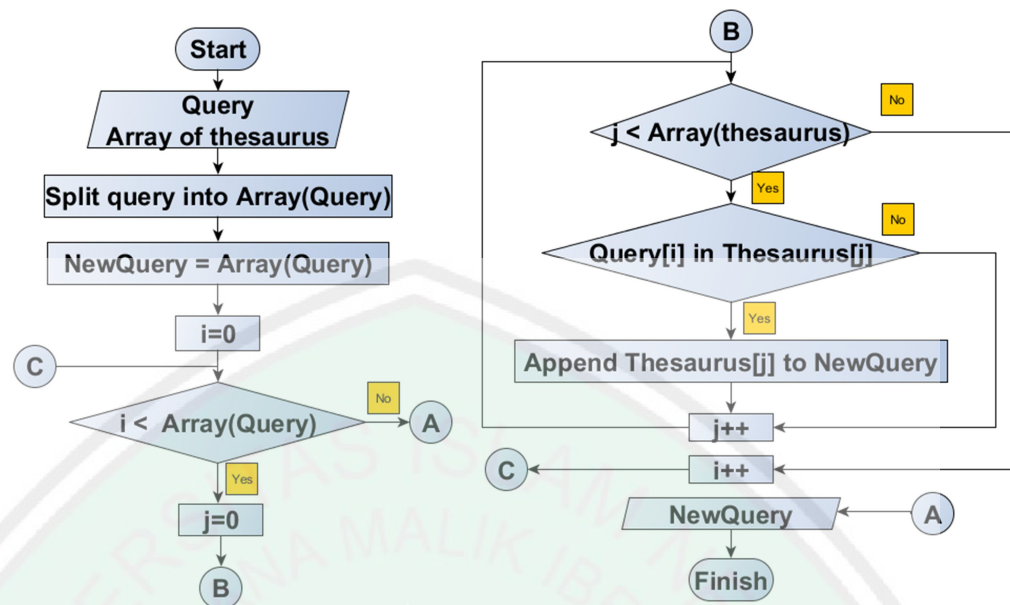


Figure 3.6 Flowchart Thesaurus

### Term Weighting

Stage after preprocessing and thesaurus is a term weighting. Nadu (2014) explained that term weighting is an important aspect of modern text retrieval systems. Terms are words, phrases, or any other indexing units used to identify the contents of a text. Since different terms have different importance in a text, an important indicator – the Term Weight – is associated with every term.

$$\begin{pmatrix}
 & T_1 & T_2 & \dots & T_t \\
 D_1 & w_{11} & w_{21} & \dots & w_{t1} \\
 D_2 & w_{12} & w_{22} & \dots & w_{t2} \\
 \vdots & \vdots & \vdots & \ddots & \vdots \\
 \vdots & \vdots & \vdots & \ddots & \vdots \\
 D_n & w_{1n} & w_{2n} & \dots & w_{tn}
 \end{pmatrix}$$

Figure 3.7 Vector term weighting



- 1) Term Frequency: It is defined in the simplest case as the number of occurrences of the term in the document. We would like to compute a score between a query term  $t$  and a document  $d$ , based on the weight of  $t$  in  $d$ . The simplest approach is to assign the weight to be equal to the number of occurrences of term  $t$  in document  $d$ . This weighting scheme is referred to as term frequency and is denoted  $t_{f,d}$ , with the subscripts denoting the term and the document in order. The following is a flow chart of the TF process:

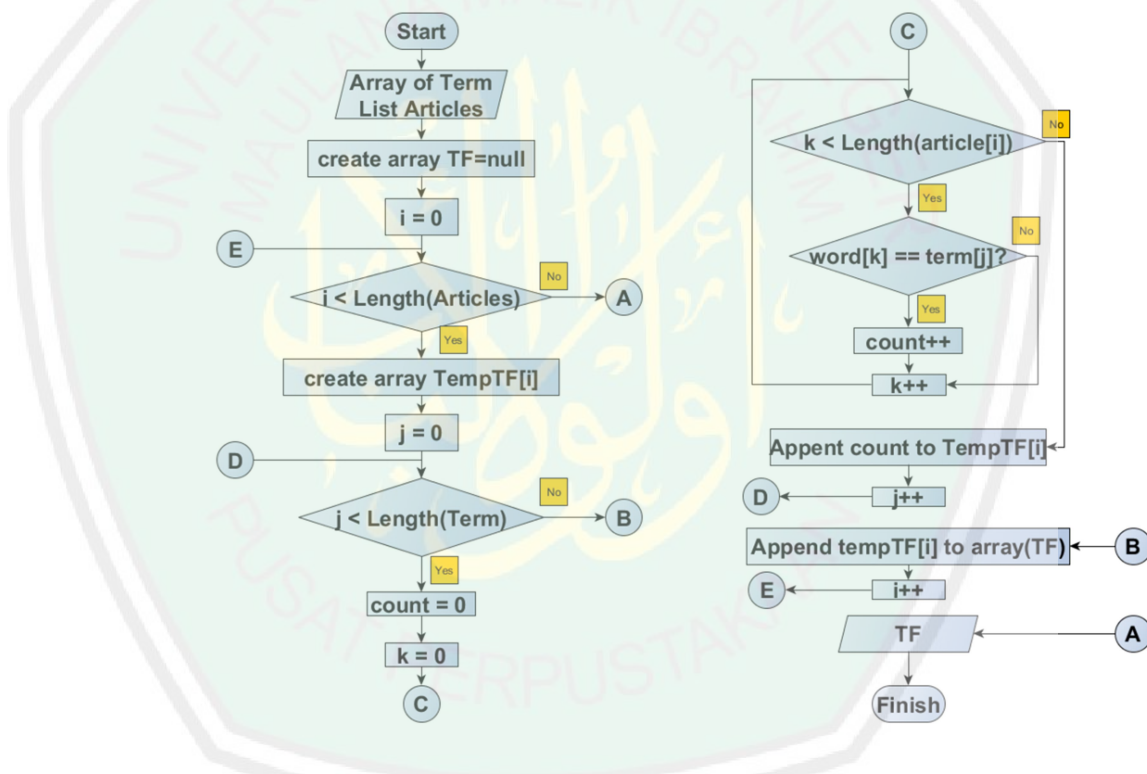


Figure 3.8 Flowchart TF

- 2) Document Frequency: It is defined to be the number of documents in the collection that contain a term  $t$  and is denoted  $d_{f,t}$ .
- 3) Inverse document frequency: It is a measure of the general importance of the term (obtained by dividing the total number of documents by the

number of documents containing the term, and then taking the logarithm of that quotient).

$$\text{idft} = \log \frac{N}{\text{dft}} \text{ or } \text{idft} = \frac{1}{\text{dft}} \quad (3.1)$$

Where, N : total number of documents in a collection

dft: Document frequency

The following is a flow chart of the IDF process:

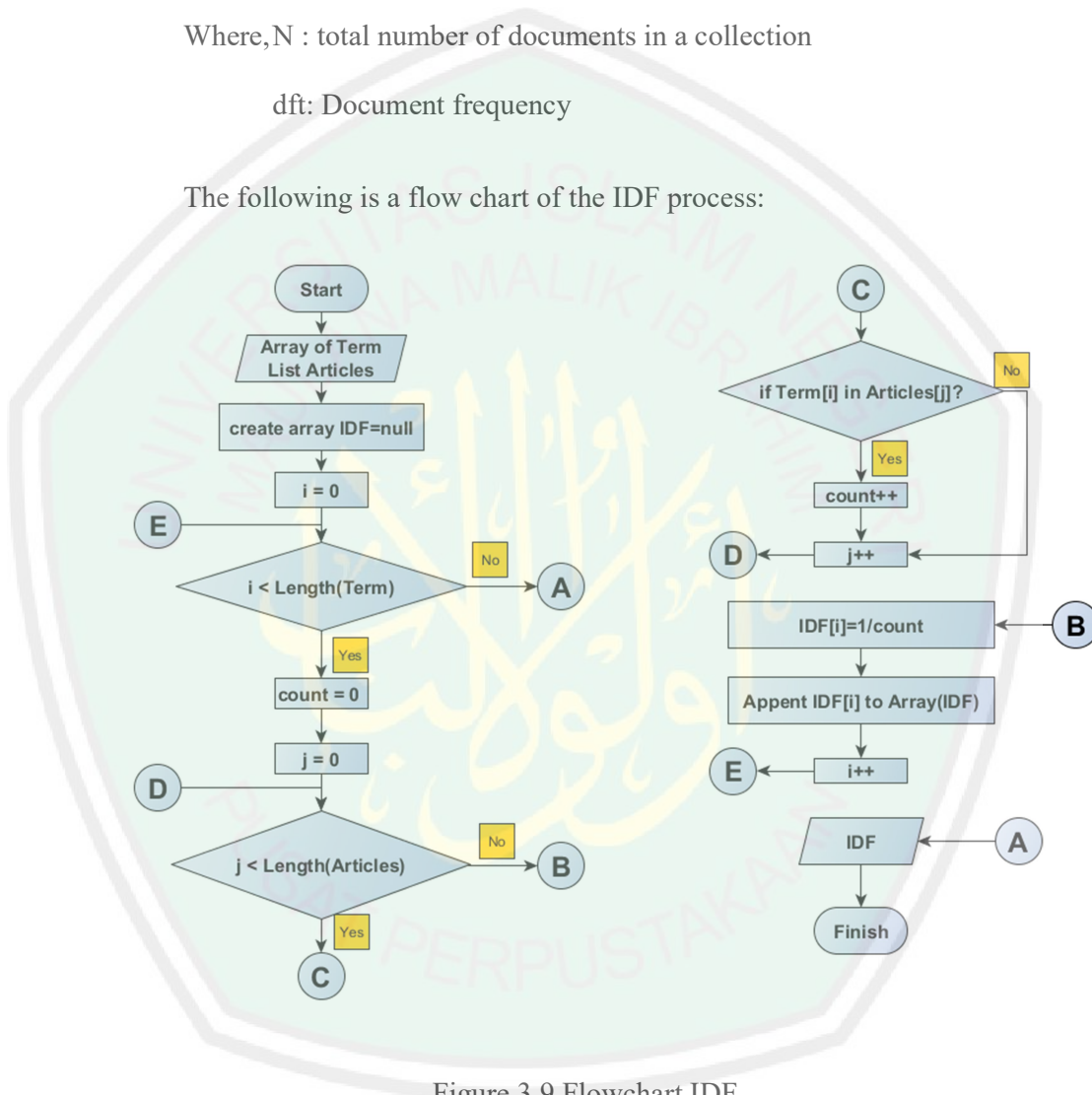


Figure 3.9 Flowchart IDF

We now combine the definitions of term frequency and inverse document frequency, to produce a composite weight for each term in each document. The tf idf weighting scheme assigns to term  $t$  a weight in document  $d$  given by

$$\text{tf} - \text{idf}, d = \text{tft}, d \times \text{idft} \quad (3.2)$$

In other words,  $tf-idf_{t,d}$  assigns to term  $t$  a weight in document  $d$  that is

1. highest when  $t$  occurs many times within a small number of documents (thus lending high discriminating power to those documents).
2. Lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);
3. Lowest when the term occurs in virtually all documents.

The weight of a term in a document vector can be determined in many ways. A common approach uses the so-called  $tf \times idf$  method, in which the weight of a term is determined by two factors: how often the term  $j$  occurs in the document  $i$  (the term frequency  $tf_{i,j}$ ) and how often it occurs in the whole document collection (the document frequency  $df_j$ ). Precisely, the weight of a term  $j$  in document  $i$  is

$$W_{i,j} = tf_{i,j} \times idf_j = tf_{i,j} \times \log \frac{N}{df_j} \quad (3.3)$$

where  $N$  is the number of documents in the document collection and  $idf_j$  stands for the inverse document frequency. This method assigns high weights to terms that appear frequently in a small number of documents in the document set. The following is a flow chart of the TFIDF process:

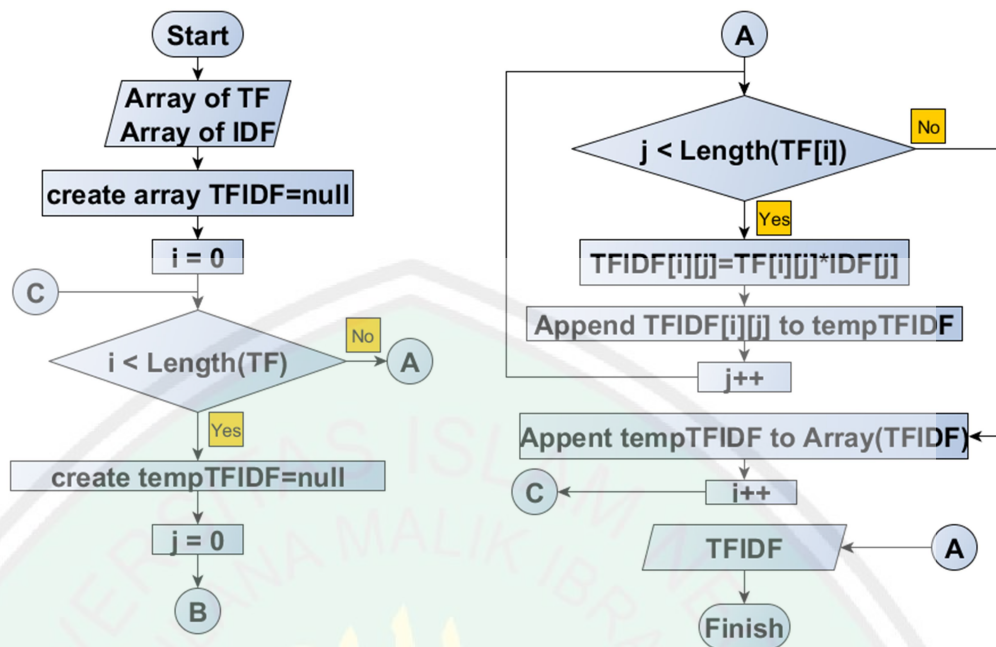


Figure 3.10 Flowchart TFIDF

**Vector Space Model (VSM)**

Vector space model (or term vector model) is an algebraic model for representing any objects as vectors of identifiers, such as, for example, index terms. It is used in information filtering, information retrieval, indexing and relevancy rankings.

Chen & Song (2009) explained that VSM model which is applied to the index has a great advantage. To search terms and the calculation of the similarity between the texts is very convenient. The calculation of this similarity using between texts is also very natural. There are forms of distance measures. Sometimes we call it distance function or similarity function. Vector inner product, cosine measure, Correlation distance, Spearman distance, Euclidean distance, City Block distance, Mahalanobis distance, Minkowski distance,

Tanimoto distance, Hamming distance, Jaccard distance and so on. We use these distance function to find the similarity of documents.

$$\text{sim}(D_j, D_k) = \cos\theta = \frac{\sum_{i=1}^n w_{ij}w_{ik}}{\sqrt{\sum_{i=1}^n w_{ij}^2 \sum_{i=1}^n w_{ik}^2}} \quad (3.4)$$

In the vector space model, we represent documents as vectors. The success of the vector space method is based on term weighting (Nadu, 2014). This space has dimension as the number of terms or in other words to a document which has N word is required n dimension. Each vector is represented in accordance weights of the terms that exist. The terms are there to be the coordinate axes as many as the number of terms, while the vector is a point whose position based on the value of these axes. Eg for documents with two dimensions, then when represented in Cartesian field will be a point that consists of x and y. Values x and y depends on the weight of term x and y. in its implementation, Vector Space Model is represented as two-dimensional matrix with columns as the term and the document as a line, as well as term as the weight of the contents of a matrix. Making Vector Space Model for dokumen is to make a two-dimensional matrix with columns as the terms and documents line. The contents of the matrix is a weight value term (term weight) of each term to each document. Term is the Vector Space Models are drawn from across the term unique to the entire document. The following is a flow chart of the get long vector process:

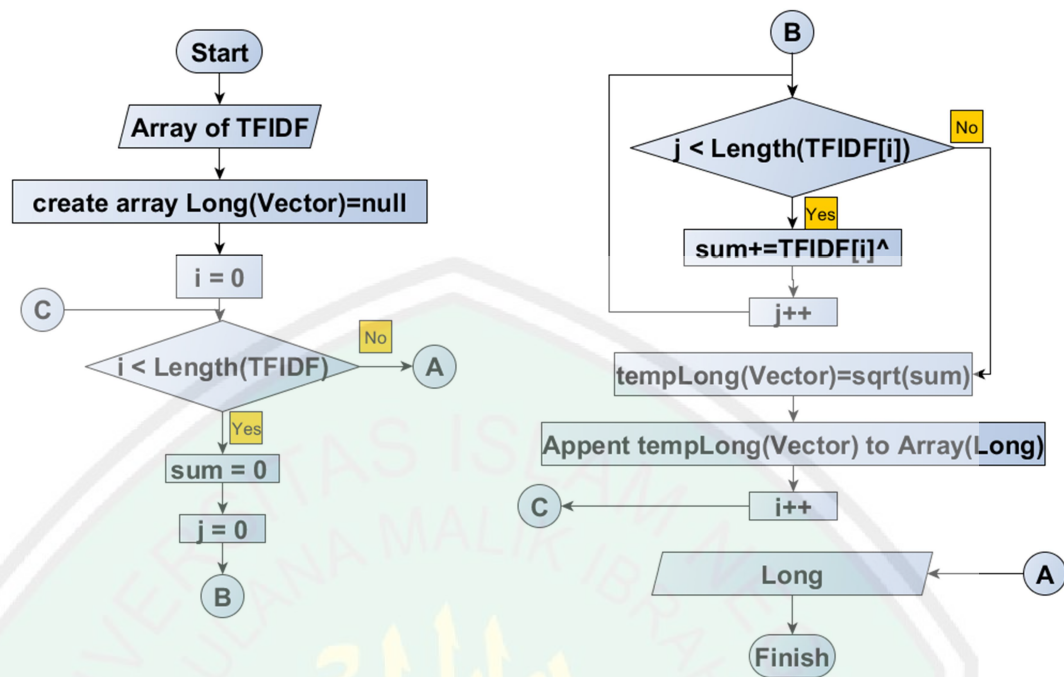


Figure 3.11 Flowchart Get Long Vector Space Model

### *Cosine Similiarity*

Cosine similarity is employed to conduct similarity that it is done by formatted the weight terms contained in the vector space model. Cosine similarity used to calculated each vector space model of the article and query using similarity according to the Equation 3.5. The results of these calculations give a similarity value between the range of 0 to 1. A value close to 1 indicates a high level of similarity. so that of the results of this calculation if sorted would produce documents sequentially.

$$\cos(q, d_j) = \frac{\sum_{t_i} [w(t_i, q)] * [w(t_i, d_j)]}{\sqrt{\sum |w(q)|^2 * \sum |w(d_j)|^2}} \quad (3.5)$$

In the equation  $\cos(q, d_j)$  is the cosine between query and document  $j$ ,  $w(t_i, q)$  is the weight of term  $t_i$  TF.IDF on the query, and  $w(t_i, d)$  is the weight for each term

TF.IDF  $t_i$  documents based on distribution  $j$  term in the classroom. While  $|w(q)$  and  $|w(d_j)|$  each the length of the vector  $q$  and the length of the vector document  $j$ . The following is a flow chart of the Cosine Similarity process:

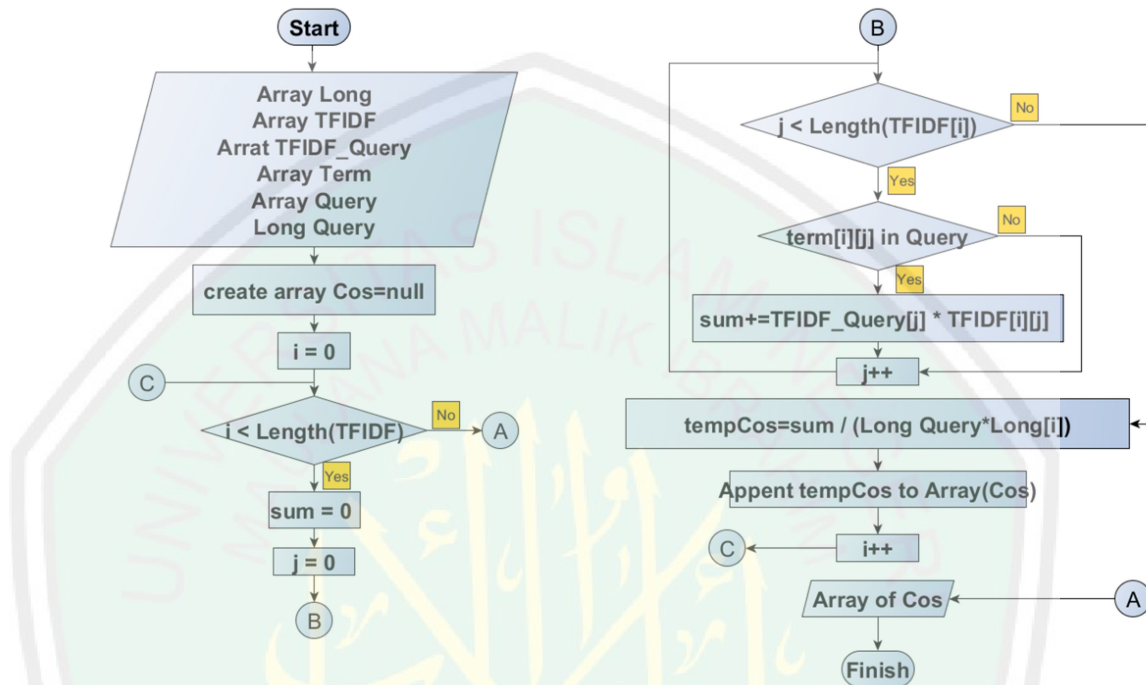


Figure 3.12 Flowchart Cosine Similarity

### 3.1.2. Interface Design

Designing a program for information retrieval would be applied to the desktop platform. The following is the interface design application to be made.

#### a. Form data collection and training

form data collection and training data is the form to do web crawling and weighting articles to be stored in a vector. This form displayed all web crawling data collection results. Input can be either IP address or URL of the website will start crawling.

Figure 3.13 Form Training

b. Form testing

Testing form is the form to do a search in accordance with article query. input form of query that may contain words, phrases or sentences. after the process of ranking the documents will be displayed sequence most relevant to the query.

Figure 3.14 Form Testing 2



### 3.2. Experiment

First Experiments is conducted by evaluating the results of the crawler to derive the whole content of each website. Tests carried out on a number of website. starting website 1 to the n-th. Of each website can be obtained time crawler and sum articles as the value of evaluating the time required to derive the whole content of each website.

Second Experiments is conducted by evaluating the results of the search invitation of ranking the document. Tests carried out on a number of user input, starting input 1 to the n-th. Of each input can be obtained effectiveness value as the value of evaluating the ability of the system based on the following parameters

Precision is the number of relevant documents retrieved divided by the total number of documents retrieved (Lee et al., 1997). TF.IDF methods with thesaurus will be compared with TF.IDF method without thesaurus. This test demonstrated done by calculating the number of relevant documents in retrieving and the number of documents in retrieving. Then look for the value of presicion.

$$\text{precision} = \frac{\text{The number of relevant documents in retrieving}}{\text{The number of documents in retrieving}} \times 100\% \quad (3.6)$$

Accuracy is a measure of how close a measurement results to the correct or acceptable value of the quantity being measured.

$$\text{Accuracy} = \frac{\text{The number of relevant documents in retrieving}}{\text{The number of documents in the database}} \times 100\% \quad (3.7)$$

## CHAPTER IV IMPLEMENTATION AND TESTING

In this section is explained about the implementation of each step presented from chapter 3. Then show the results of the test according to the test scenario of calculating the level of precision and akusau on the proposed method. Then will be presented analysis and discussion of test results at the end of the chapter.

### 4.1. Implementation

The proposed method is implemented with python and java languages on the 1.8.0 development kit platform and IDE Netbeans 8.1. The application is built on the Microsoft Windows 10 platform with an Intel Core i7-4510U processor and 8 GB of memory.

Implementation of the algorithm is done by making the functions of the stages that have been described in chapter 3. in this section is displayed pieces of important scripts in the section.

#### 4.1.1. Web Crawler

Web crawlers are built to retrieve website content by implementing the Breadth first algorithm. The first step is to define the initial url to be clawed. The next step the web crawler will take the HTML code from the initial url to diextrak url and its content. The given url is set to as the next url to visit and the content it gets will be stored. The web crawler source code is shown in Figure 4.1.

```

self.sql_domain = "SELECT id,link from data where id_domain=%d and status=0 and
(SELECT count(link) from `data` where id_domain=%d and status=1)<=10"
%(self.id_domain,self.id_domain)
row_count=self.konek.cur.execute(self.sql_domain)
hdr = {'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.11 (KHTML, like
Gecko) Chrome/23.0.1271.64 Safari/537.11',
'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
'Accept-Charset': 'ISO-8859-1,utf-8;q=0.7,*;q=0.3',
'Accept-Encoding': 'none',
'Accept-Language': 'en-US,en;q=0.8',
'Connection': 'keep-alive'}

while(row_count>0):
self.start_time_collection = time.time()
result=self.konek.cur.fetchone()
if result != None:
print result
link=result[1]
_id=result[0]

try:
req = urllib2.Request("http://"+link, headers=hdr)
htmltext=urllib2.urlopen(req,timeout=3).read()
# htmltext=urllib2.urlopen("http://"+link,timeout=3).read()
print htmltext

#remove character http dkk
set_except=['http://','https://','/','www.']
for item in set_except:
if item in link:
link=link.replace(item,"")

#set path for file
path=link
if "<" in path:
path=path.replace('<','$11111$')
if ">" in path:
path=path.replace('>','$22222$')
if ":" in path:
path=path.replace(':', '$33333$')
if "" in path:
path=path.replace("", '$44444$')
if "/" in path:
path=path.replace('/', '$55555$')
if "\" in path:
path=path.replace("\"\", '$66666$')
if "|" in path:
path=path.replace('|', '$77777$')

```

```

if "?" in path:
path=path.replace('?','$88888$')
if "*" in path:
path=path.replace('*','$99999$')
path+=" .txt"

# print path
#save htmltext to file
f=open(self.pathDataSave+"\\"+path,"w")
f.write(htmltext)
f.close()

soup=BeautifulSoup(htmltext, "lxml")
# print soup
list=soup.find_all('a', href=True)
# print list
#set status link 1

sql="UPDATE `data` SET `status`=1, `collection_time`='%s' WHERE id=%d"
%(str(time.time() - self.start_time_collection),_id)
self.konek.cur.execute(sql)
self.konek.con.commit()
#just write code below
#=====

#insert all url from current page to database
# print "panjang list : ",len(list)

for url in list:
self.start_time_verify = time.time()
link=url['href']
for item in set_except:
if item in link:
link=link.replace(item,"")
#check website is exists
try:
#cek apakah link termasuk dalam domain
parsed_uri = urlparse('http://' + link)
domain = '{uri.scheme}://{uri.netloc}/'.format(uri=parsed_uri)
domain=domain.replace("http://","")
domain=domain.replace("/","")
# print domain," = ",self.domain
if domain==self.domain and self.notduplicate(self.id_domain,link):
try:
req = urllib2.Request("http://" + link, headers=hdr)
response=urllib2.urlopen(req,timeout=3)
# response=urllib2.urlopen("http://" + link,timeout=3)

```

```

# print response.getcode()
#cek apakah link valid
if response.getcode() < 400:
# if response.status_code < 400:
sql = "INSERT IGNORE INTO `data` (`id`,`id_domain`,
`link`,`create_at`,`status`,`verify_time`) VALUES (NULL, %d,'%s',
CURRENT_TIMESTAMP,0,'%s')" %(self.id_domain,link,str(time.time()) -
self.start_time_verify))
self.konek.cur.execute(sql)
self.konek.con.commit()
except:
sql = "INSERT IGNORE INTO `data` (`id`,`id_domain`,
`link`,`create_at`,`status`,`verify_time`) VALUES (NULL, %d,'%s',
CURRENT_TIMESTAMP,4,'%s')" %(self.id_domain,link,str(time.time()) -
self.start_time_verify))
self.konek.cur.execute(sql)
self.konek.con.commit()
elif self.notduplicate(self.id_domain,link):
try:
req = urllib2.Request("http://"+link, headers=hdr)
response=urllib2.urlopen(req,timeout=3)
# response=urllib2.urlopen("http://"+link,timeout=3)
# response=requests.get("http://"+link,timeout=3)
# print response.getcode()
#cek apakah link valid
if response.getcode() < 400:
# if response.status_code < 400:
sql = "INSERT IGNORE INTO `data` (`id`,`id_domain`,
`link`,`create_at`,`status`,`verify_time`) VALUES (NULL, %d,'%s',
CURRENT_TIMESTAMP,3,'%s')" %(self.id_domain,link,str(time.time()) -
self.start_time_verify))
self.konek.cur.execute(sql)
self.konek.con.commit()
# print ("Input Success")
# print (self.konek.cur._last_executed)
except:
sql = "INSERT IGNORE INTO `data` (`id`,`id_domain`,
`link`,`create_at`,`status`,`verify_time`) VALUES (NULL, %d,'%s',
CURRENT_TIMESTAMP,4,'%s')" %(self.id_domain,link,str(time.time()) -
self.start_time_verify))
self.konek.cur.execute(sql)
self.konek.con.commit()

# except urllib2.HTTPError, e:
# print(e.code)
# except urllib2.URLError, e:
# print(e.args)
except:

```

```

pass
# self.konek.con.rollback()
# print ("Input failed ", sys.exc_info())
# print (self.konek.cur._last_executed)

#=====
#just write code above
# self.stop()
row_count=self.konek.cur.execute(self.sql_domain)
self.konek.con.commit()
# print (self.konek.cur._last_executed)
except:
self.konek=koneksi.Koneksi()
row_count=self.konek.cur.execute(self.sql_domain)
self.konek.con.commit()
# print (self.konek.cur._last_executed)
# print ("Link failed ", sys.exc_info())
else:
try:
row_count=self.konek.cur.execute(self.sql_domain)
self.konek.con.commit()
# print (self.konek.cur._last_executed)
except:
self.konek=koneksi.Koneksi()
row_count=self.konek.cur.execute(self.sql_domain)
self.konek.con.commit()
# print (self.konek.cur._last_executed)
# print ("Link failed ", sys.exc_info())

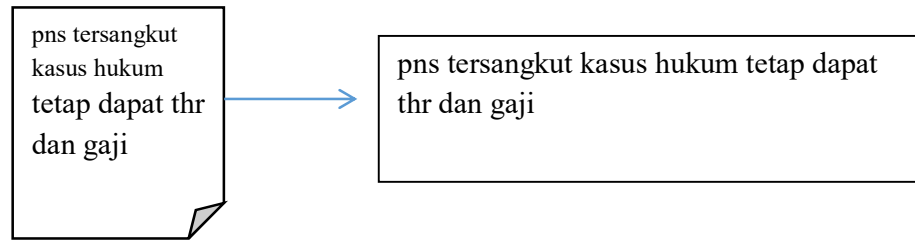
#sql = "INSERT INTO `time` (`id`,`id_domain`,`time_execution`,`create_at`) VALUES
(NULL, %d,%s, CURRENT_TIMESTAMP)" %(self.id_domain,str(time.time() -
Crawler.start_time))
sql = "UPDATE `domain` set time_execution = '%s' , create_at=CURRENT_TIMESTAMP
where id_domain=%d" %(str(time.time() - Crawler.start_time),self.id_domain)
self.konek.cur.execute(sql)
self.konek.con.commit()

```

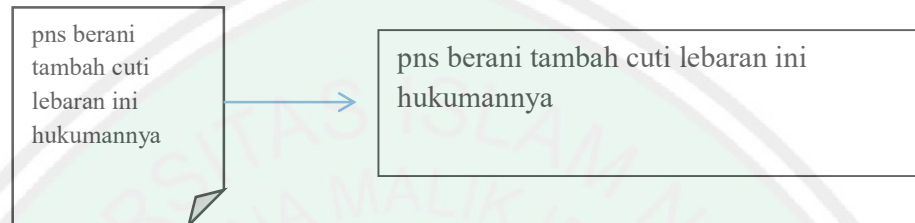
Figure 4.1 Source Code Web Crawler

#### 4.1.2. Process preprocessing documents

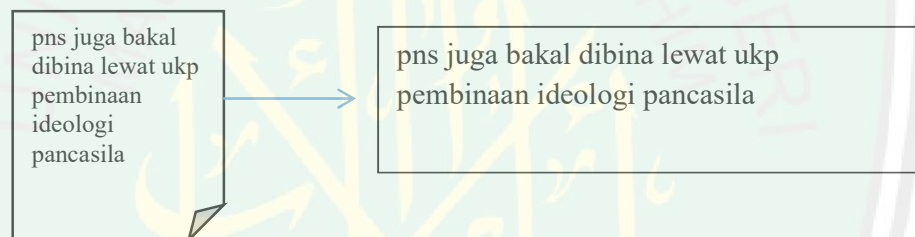
On each document to be indexed first done preprocessing to facilitate the next process. Implementation of preprocessing consists of several steps, namely tokenizing, stopword removal and stemming. Example from 3 documents below:



Document 1



Document 2



Document 3

### 1. Tokenizing

Tokenizing is done to break the entire contents of a document into a single word. The tokenizing process is shown in Figure 4.2.

```
def tokenizing(text):
    wordList = re.sub("[^\\w]", " ", text).split()
    arrWordlist = []
    for word in wordList:
        if word not in arrWordlist:
            arrWordlist.append(word)
    return arrWordlist
```

Figure 4.2 Source code Tokenizing

This process gives the following results

pns tersangkut  
kasus hukum  
tetap dapat thr  
dan gaji

Document 1

pns  
tersangkut  
kasus  
hukum  
tetap  
dapat  
thr  
dan  
gaji

pns berani  
tambah cuti  
lebaran ini  
hukumannya

Document 2

pns  
berani  
tambah  
cuti  
lebaran  
ini  
hukumannya

pns juga bakal  
dibina lewat ukp  
pembinaan  
ideologi  
pancasila

Document 3

pns  
juga  
bakal  
dibina  
lewat  
ukp  
pembinaan  
ideologi  
pancasila

## 2. Stopword Removal

The stopword removal process is eliminating the term that often appears in the document but has no meaningful value in a document. The removal mechanism is by way of whether term includes stopword, if term is included in stopword then the term will be deleted. Leaving a word that

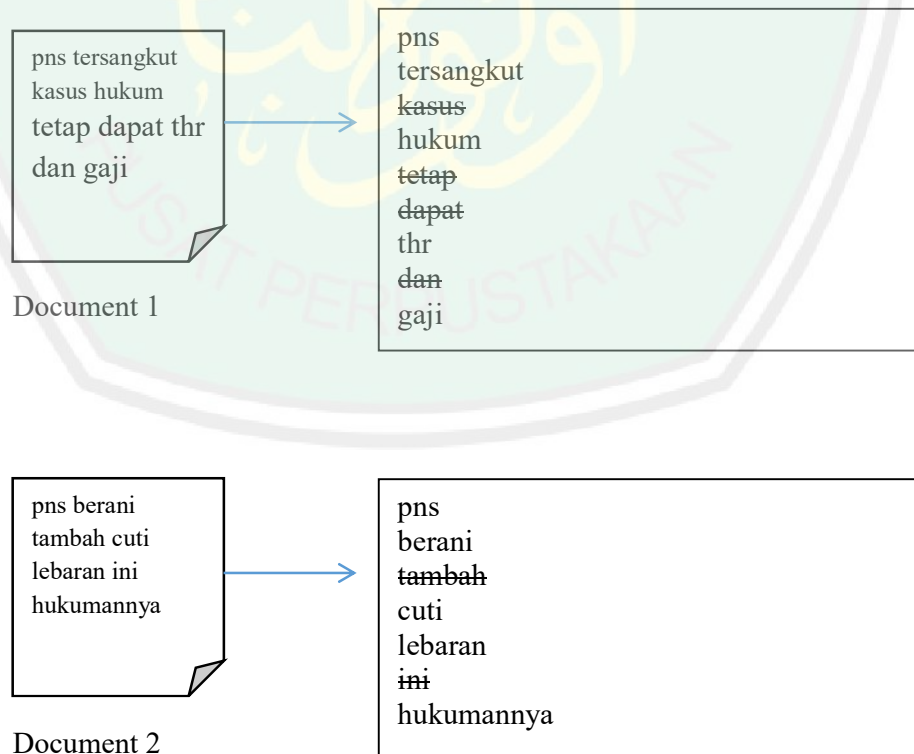


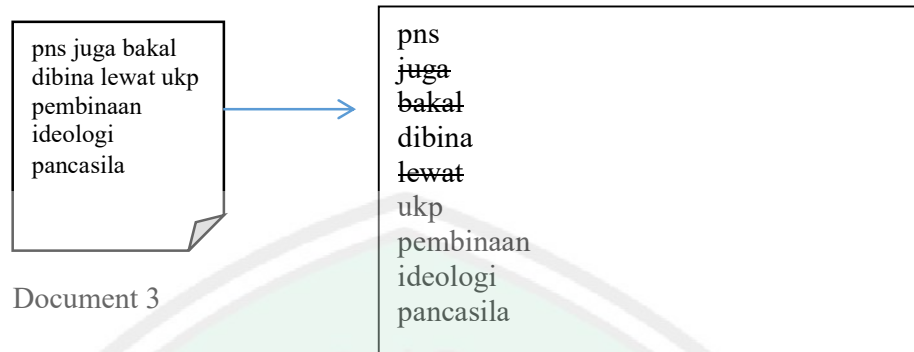
has influence in a document. The stopwords removal process is shown in Figure 4.3.

```
def stopwordsremoval(wordList):
    s=open(os.path.dirname(__file__)+'\data\stoplist.txt','r')
    sl=s.read()
    stoplist=sl.split(", ")
    stopword=[]
    #menghiangkan kata yang tidak penting (stopword removal)
    for item in wordList:
        if item not in stoplist and item!="":
            stopword.append(item)
    return stopword
```

Figure 4.3 Source Code Stopword Removal

This process gives the following results





### 3. Stemming

Stemming is the process of converting term into basic word form.

Stemming used to obtain the basic words of each word by searching the base word (root) and removal affix and suffix. In the implementation of this stemming Sastrawi Stemmer that are used in literary library (<http://giuhub.com/saastrawi/>). The stemming process is shown in Figure

4.4.

```

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

def stemming(stopword):

    factory = StemmerFactory()
    stemmer = factory.create_stemmer()

    sentence=" ".join(stopword)

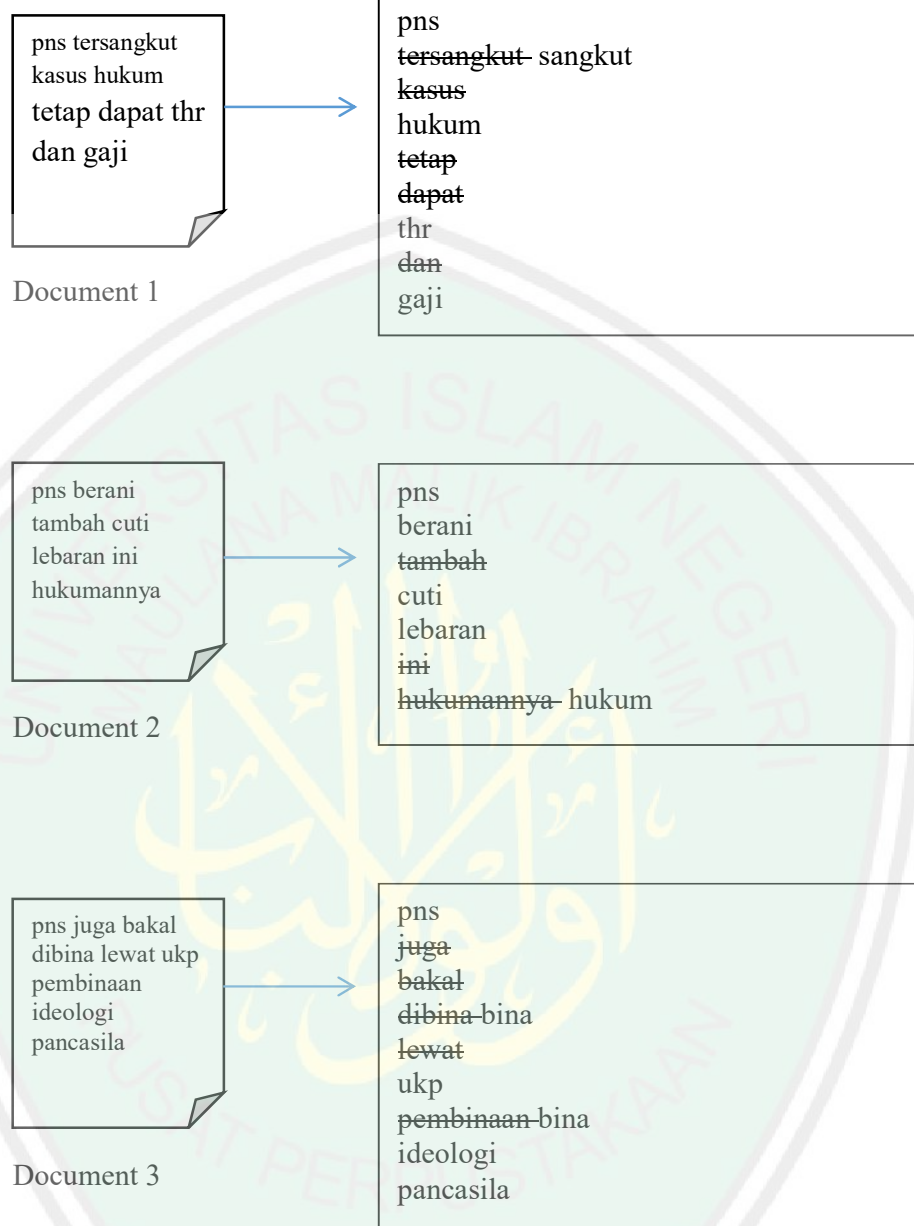
    output = stemmer.stem(sentence)

return output;

```

Figure 4.4 Source Code Stemming

This process gives the following results



#### 4.1.3. Pembobotan TF.IDF

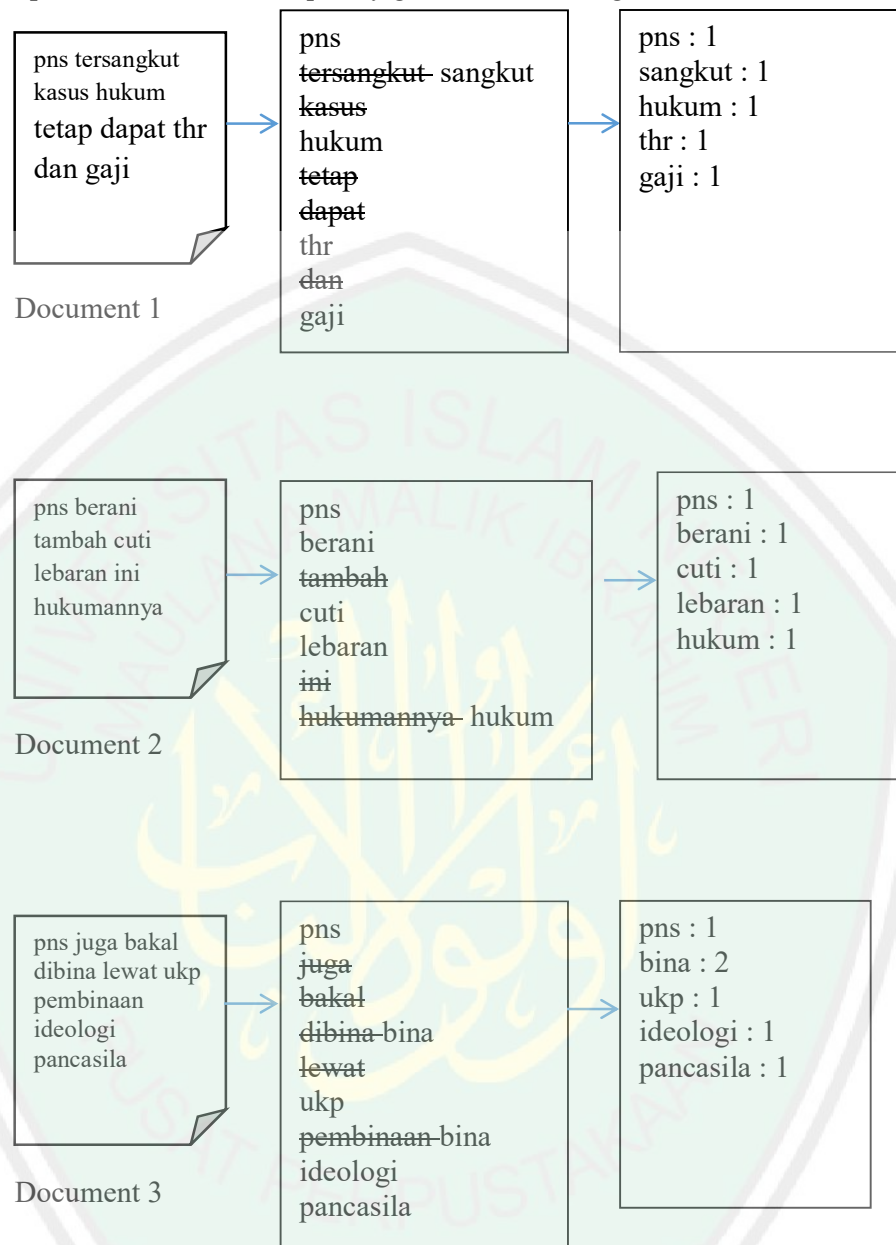
The next step after preprocessing is the term weighting. This weighting is done by calculating TF (Term Frequency) and IDF (Inverse Document Frequency). After we get the value of TF and IDF from each term, then weighing TF.IDF.

The calculation of TF is done by counting the number of occurrence frequencies of term in each document. Implementation of TF calculation and term already stored as TF in group based on id article and term. The TF process is shown in Figure 4.5.

```
def getTerm(docs):  
    term=[]  
    for doc in docs:  
        for word in doc:  
            if word not in term:  
                term.append(word.lower())  
    return term  
  
def getTf(docs,term):  
    arrTf=[]  
    for doc in docs:  
        tf=[]  
        for wordterm in term:  
            count=0  
            for worddoc in doc:  
                if worddoc == wordterm:  
                    count+=1;  
            tf.append(count)  
        arrTf.append(tf)  
    return arrTf
```

Figure 4.5 Source Code TF (Term Frequency)

This process count term frequency gives the following results



The calculation of the IDF is done by calculating the term distribution throughout the document. The term IDF term value of a term is inversely proportional to the number of documents containing the term. IDF weighting implementation is done by counting the number of documents containing a term.

Then calculated IDF with formula  $idf = \log \frac{N}{df}$  or  $idf = \frac{1}{df}$ . The IDF process is shown in Figure 4.6.

```
def getDf(term,docs):
    arrDf = []
    for wordterm in term:
        count = 0
        for doc in docs:
            if wordterm in doc:
                count += 1
        arrDf.append(count)
    return arrDf
def getIdf(arrDf,docs):
    arrIdf = []
    for df in arrDf:
        arrIdf.append(math.log(float(len(docs))/float(df)))
    return arrIdf
```

Figure 4.6 Source Code IDF (inverse Document Frequency)

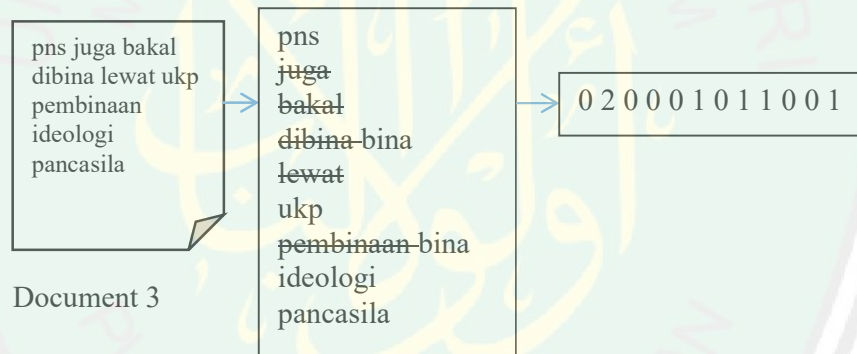
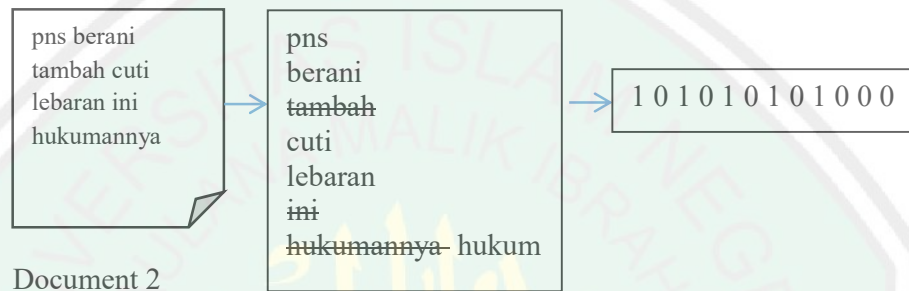
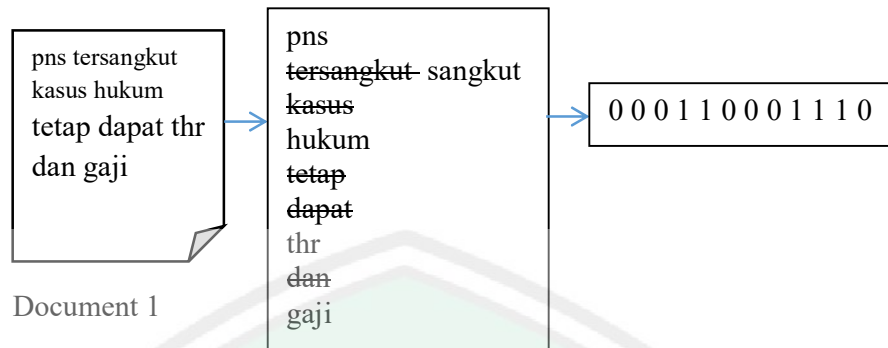
After getting the value of TF and IDF then multiply the weight of TF.IDF.

The TF.IDF process is shown in Figure 4.7.

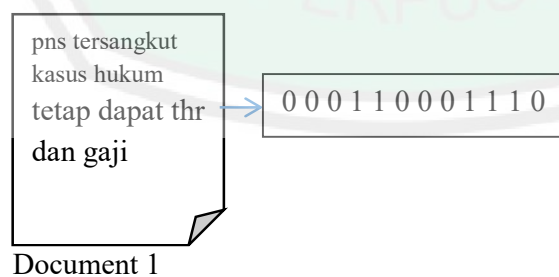
```
def getTfidf(arrTf,arrIdf):
    arrTfidf = []
    for tf in arrTf:
        tfidf = []
        for x,y in zip(tf,arrIdf):
            tfidf.append(float(x)*float(y))
        arrTfidf.append(tfidf)
    return arrTfidf
```

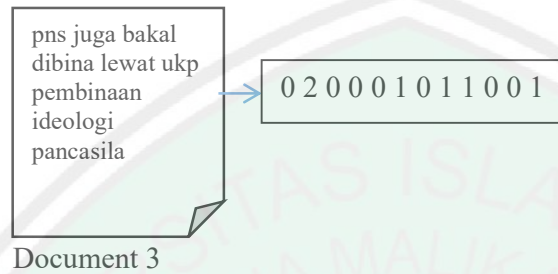
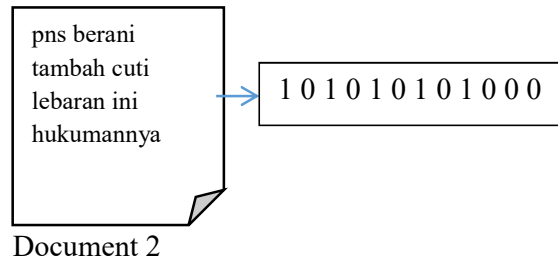
Figure 4.7 Source Code TF.IDF

This process to get the term frequency



This process to get the document frequency

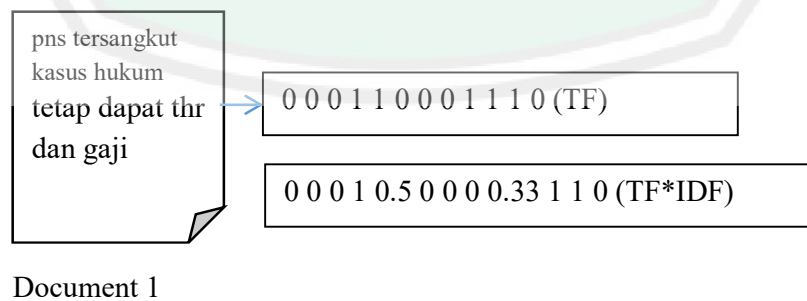




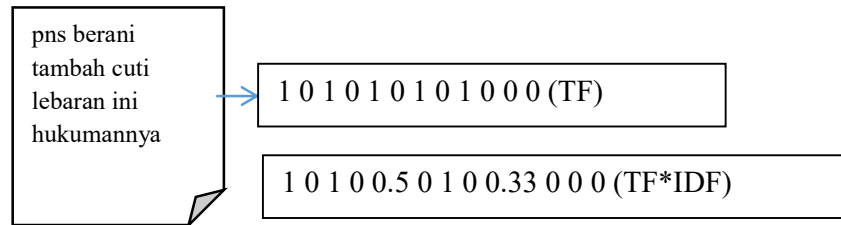
Tabel 4.1 Result of Document Frequency

ID	Word	Document frequency
1	berani	1
2	bina	1
3	cuti	1
4	gaji	1
5	hukum	2
6	ideologi	1
7	lebaran	1
8	pancasila	1
9	pns	3
10	sankut	1
11	Thr	1
12	ukp	1

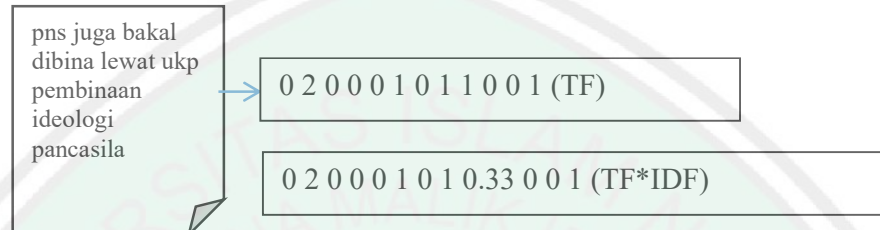
This process to get the inverse document frequency







Document 2



Document 3

#### 4.2 Result of Inverse Document Frequency

ID	Word	Document frequency	IDF
1	Berani	1	1
2	Bina	1	1
3	Cuti	1	1
4	Gaji	1	1
5	hukum	2	0.5
6	Ideology	1	1
7	Lebaran	1	1
8	Pancasila	1	1
9	Pns	3	0.33
10	Sankut	1	1
11	Thr	1	1
12	Ukp	1	1

#### 4.1.4. Thesaurus

Thesaurus is the process of expanding search queries by searching for query-related words in prepared dictionary thesaurus. The Thesaurus process is shown in Figure 4.8.

```

    querys = sys.argv[1].split(" ")
    querys = preprocessing.stemming(querys).split(" ");
    if sys.argv[2] == "no":
        saveDatabase(querys)
        return "Done"

    arr=[]
    f=open(os.path.dirname(__file__)+'\Data\newthesaurus.txt','r')
    texts=f.read().split("\n")
    arrQuery=[]
    for query in querys:
        arrQuery.append(query)
        for text in texts:
            arrText = text.split(",")
            arrText = [item.strip() for item in arrText]
            if query in arrText:
                arr.append(text)

    for item in arr:
        if "|" in item:
            item = item.replace("|", ",")
        words = item.split(",")
        words = [word.strip() for word in words]
        for word in words:
            if word != "" and word not in arrQuery:
                arrQuery.append(word)

```

Figure 4.8 Source Code Thesaurus

#### 4.1.5. Cosine Similarity

Query search is done by calculating cosine similarity between query and each document. Cosine similarity calculation is based on weighting TF.IDF. The cosine similarity calculation is done by comparing the proximity between the vector space matrix of the query model and the vector space matrix model of each document.

Cosine similarity calculation is to calculate the multiplication between the vector of each document query then calculate the length of vector. Length of vector is the root value of the sum of squared weights on each document or query. The Get Long Vector process is shown in Figure 4.9.

```

arrPanjang = []
for tfidf in arrTfidf:
    jumlah=0
    for tf in tfidf:
        jumlah=jumlah+float(tf*tf)
    arrPanjang.append(math.sqrt(jumlah))
return arrPanjang

```

Figure 4.9 Source Code Get Long Vector

The cosine similarity value is obtained by dividing the number of multiplication weights by multiplying the length of the vector query and the length of the document vector. The Cosine Similarity process is shown in Figure 4.10.

```

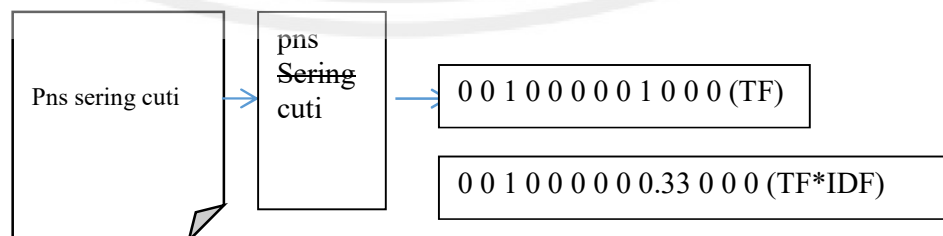
def cosineSimilarity(arrPanjang,arrTfidf,tfidfQuery,arrTerm,panjanVectorQuery,query):
    arrCos = []
    for x in xrange(0,len(arrPanjang)):
        panjangDoc = arrPanjang[x]
        tfidfDoc = arrTfidf[x]
        jumlah = 0
        for tfidf_q,query in zip(tfidfQuery,query):
            for term,tfidf_d in zip(arrTerm,tfidfDoc):
                if term == query:
                    jumlah = jumlah+(float(tfidf_q)*float(tfidf_d))
        arrCos.append(jumlah/(float(panjanVectorQuery)*float(panjangDoc)))
    return arrCos

```

Figure 4.10 Source Code Cosine Similarity

Illustration of this process we will give an example with a query “pns sering cuti”.

From the above query do the preprocessing process.



4.3 Result of TF.IDF

Term	tf					Idf	tf*idf			
	Q	D1	D2	D3	df	1/df	Q	D1	D2	D3
Berani	0	0	1	0	1	1	0	0	1	0
Bina	0	0	0	2	1	1	0	0	0	2
Cuti	1	0	1	0	1	1	1	0	1	0
Gaji	0	1	0	0	1	1	0	1	0	0
hukum	0	1	1	0	2	0.5	0	0.5	0.5	0
Ideology	0	0	0	1	1	1	0	0	0	1
Lebaran	0	0	1	0	1	1	0	0	1	0
Pancasila	0	0	0	1	1	1	0	0	0	1
Pns	1	1	1	1	3	0.33	0.33	0.3	0.3	0.3
Sankut	0	1	0	0	1	1	0	1	0	0
Thr	0	1	0	0	1	1	0	1	0	0
Ukp	0	0	0	1	1	1	0	0	0	1

#### 4.4 Result of Weight Document & Long Vec

Term	WDQ*WDi			Panjang Vector			
	D1	D2	D3	Q	D1	D2	D3
Berani	0	0	0	0	0	1	0
Bina	0	0	0	0	0	0	4
Cuti	0	1	0	1	0	1	0
Gaji	0	0	0	0	1	0	0
Hukum	0	0	0	0	0.3	0.3	0
Ideology	0	0	0	0	0	0	1
Lebaran	0	0	0	0	0	1	0
Pancasila	0	0	0	0	0	0	1
Pns	0.1089	0.1089	0.1089	0.1	0.1	0.1	0.1
Sangkut	0	0	0	0	1	0	0
Thr	0	0	0	0	1	0	0
Ukp	0	0	0	0	0	0	1
Total	0.1089	1.1089	0.1089	1.1	3.4	3.4	7.1
				1.1	1.8	1.8	2.7

Apply cosine similarity formula. Calculate the similarity of Q with D1, D2 and D3.

$$\text{Cos}(Q,D1) = 0.1089/1.1 \times 1.8 = 0.0564265$$

$$\text{Cos}(Q,D2) = 1.1089/1.1 \times 1.8 = 0.57457623$$

$$\text{Cos}(Q,D3) = 0.1089/1.1 \times 2.8 = 0.03878649$$

The results of these calculations are shown in the following table:

#### 4.5 Result of Cosine Similarity

D1	D2	D3
0.0564265	0.57457623	0.03878649

Sort results of similarity calculations, obtained:

#### 4.6 Result of Sorting Cosine Similarity

1	2	3
D2	D1	D3

#### 4.1.6. Design and Implementation GUI

GUI design described the utility of existing components in web crawler applications and information retrieval.



Figure 4.1 GUI Web Crawler

Explanation of the view among others:

1. Text Area input domain

This Textarea serves to write down the domain that will crawling.

2. Start Button

This button works to run web crawling process.

3. Stop Button

This button works to stop web crawling process

4. Cleaning Button

This button works to run cleaning process to remove tag HTML before indexing process.

5. Indexing Button

This button works to run Weighting process using TFIDF method.

6. Tematik Button

This button works to switch to Information retrieval desktop

7. List Domain

Function displays the list of domains that have been crawled and weighting

8. List Url

Function displays a list of urls that have been crawled and weighting from each domain

9. Text Area Detail

Function displays articles from the selected url

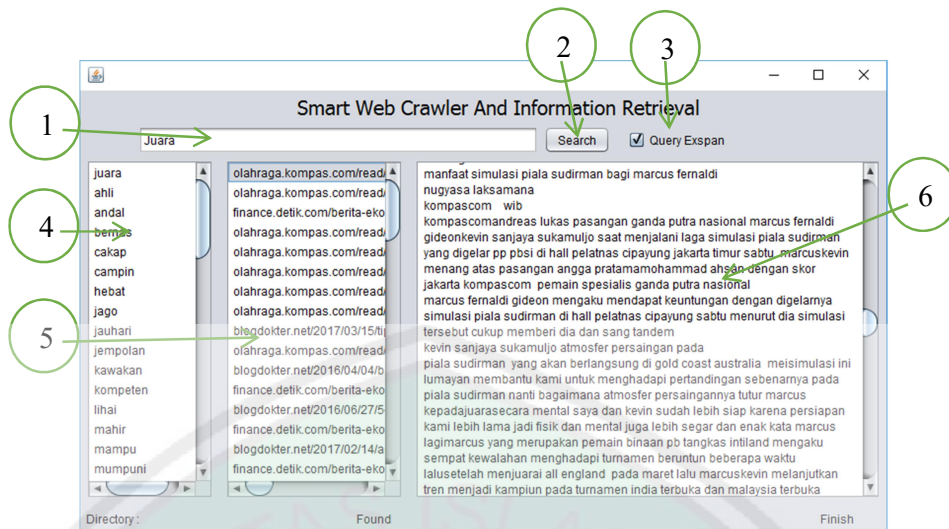


Figure 4.12 GUI Information Retrieval

Explanation of the view among others:

1. Text Area input query

This Textarea serves to write down the query that will search.

2. Search Button

This button works to run search process to find articles.

3. Checkbox Query Expansion(Thesaurus)

Function to choose whether to use thesaurus or not.

4. List Query

Function displays the list of words used for the search process of the article.

5. List Url

Fungsi menampilkan daftar url yang telah ditemukan.

6. Text Area Detail

Function displays articles from the selected url.

## 4.2. Testing Phase

Breadth First Algorithm that was implemented on web crawler, tested for getting the speed of web crawler. The speed obtained from the average of crawler time each website. Crawler time consist of verify time and collection time. Verification time is time used to verification the all url in website. Verification used to know that url is exist or broken. Collection time is time used to obtain the contents of website. The content will be used to testing information retrieval system that implement vector space model method.

Information retrieval system will be test with some query for getting the precision, recall and f-measure. The performance of information retrieval system is known by compare the result of system and the result from questionnaire. Questionnaire used to know how people sequence the information according the query. The result of comparison will show the performance of information retrieval system is approaching human or not.

## 4.3. Testing Result and Analysis

This section presents the results of the testing method in this study and then presented the analysis of the test results obtained. The first test is a trial speed web crawler in collecting data with breadth first method. Then the second test is a trial query to obtain the results of the retrieval which will then be compared with the results obtained from the source.

### 4.3.1. Crawler Testing Result

Based on crawler testing from some website. For result speed of crawler as follow .



Table 4.7 Crawler Testing Result

No	Domain	Verification Time Average	Collection Time Average	file size (Byte)	Node Routing	Connection Tpye
1	Bill.web.id	1.7256522	1.45938	194808	13	http
2	Informatika.uin-malang.ac.id	1.4811163	1.2641	26981	9	http
3	elfarqy.my.id	3.1556321	1.185	72502	11	http
4	mnafian.net	5.1186909	1.97705	100534	10	https
5	charisfauzan.net	4.4803362	1.19214	133745	13	http
6	blogdokter.net	4.3683853	2.8231	61511	15	https
Total		20.329813	9.90076			
Speed of Crawler		3.3883022	1.65013			

Information from table 4.1 the time average of web crawler to verification each url is 3.3883022 seconds. The time average for verification https is 4 seconds upwards while to verification http is 4 seconds down. The result showed that time to verification URL with https type connection more long than http.

Collection Time average of web crawler is 1.65013 seconds. The speed of web crawler to collection website is affected by two factors. First factor is type connection, http is more fast than https to get each page. Collection time average for http is 1.185 second until 1.45938 second while collection time for https start from 1.97705 seconds until 2.8231. the maximum collection time average from http is not reach the minimum collection time average of https. So we can conclude that http is more fast to crawl than https. The second factor is routing node, The more nodes it will take longer web crawler to get each page from

website. The table show collection time average for http in 9 node just need 1.2641 seconds and for 13 node need 1.45938 second there are any increased 0.2 seconds.this also applies for https connection, collection time average for 10 node need 1.97705 seconds and for 15 node need 2.8231 seconds there are any increased 0.9 seconds. So the conclude is more node will take longer to get each page from website.

#### **4.3.2. Information Retrieval Experiment Result**

Based on the tests that have been done on the data that has been prepared.

The data is 100 articles. The following results are obtained



Table 4.8 Supervised without query expansion

No	Query	Kode Artikel														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Juara Dunia	94	95	72	86	68	88	90	96	99	98	81	73	80	100	70
2	Pembalap	89	81	83	80	91	86	90	71	77	98	79	78	84	88	99
3	Sepak Bola	85	86	88	68	90	69	96	94	75	97	71	95	99	92	77
4	Pertandingan	97	35	72	69	36	94	92	87	70	85					
5	Bertarung	100	68	86	88	90	69	96	94	75	97	71	95	99	92	77
6	Kejuaraan	94	95	72	86	88	68	90	96	99	98	81	73	80	100	70
7	Kemenangan	68	69	75	95	92	81	80	70	88	90	96	94	4	97	39
8	Pelatih	86	88	90	99	5	15	96	27	95	23	77	41	81	73	78
9	Pemain	5	96	73	69	1	92	87	82	7	85	13	75	95	70	93
10	Gelar	68	75	95	80	70	93	85	72	86	88	90	69	96	94	97
11	Cara mendapatkan uang	52	60	55	62	50	47	58	54	49	64	37	66	59	61	36
12	Uang	58	54	52	49	64	37	66	60	61	36	67	55	39	95	63
13	Gaji bulanan	41	45	58	54	53	49	64	37	66	60	61	36	67	55	39
14	Cuti	58	54	52	49	64	37	66	60	61	36	67	55	39	63	48
15	Kerja	54	64	58	52	49	47	61	55	62	56	6	8	51	83	29
16	Meningkatkan harga saham	60	51	44	39	54	43	40	35	63	49	64	37	66	65	61
17	Thr	58	54	52	49	64	37	66	60	61	36	67	55	39	63	48
18	Mengurangi kerugian	46	62	66	67	14	4	11	21	97	23	15	7	9	54	3
19	Strategi meningkatkan keuntungan	27	35	46	89	81	26	70	59	39	54	43	40	35	63	49
20	Penghasilan	21	54	47	36	55	63	87	89	98	62	73	80	53	65	25
21	Mencegah kehamilan	25	28	14	2	11	6	29	34	12	22	16	30	23	3	26
22	Rumah sakit	22	29	30	3	33	16	7	6	23	14	8	34	11	25	9
23	Terakhir kali sakit	22	29	16	23	25	30	3	86	33	88	90	17	99	1	15
24	Dokter	3	7	6	14	8	34	11	25	9	12	22	5	29	2	21
25	Peninggi badan	9	17	7	6	14	8	34	11	25	12	22	5	29	2	21
26	Cara meningkatkan berat badan	34	7	6	27	14	2	8	28	33	11	12	22	23	5	21
27	Makanan sehat	7	6	14	8	19	34	11	25	9	12	22	5	29	2	21
28	Penyakit menular	33	17	32	18	22	29	30	3	16	23	7	6	14	8	34
29	Daya tahan tubuh	25	27	60	36	63	23	38	26	69	7	96	94	72	71	95
30	Pengobatan	22	33	7	6	14	28	8	34	11	25	16	9	12	5	29

Table 4.9 Form testing without query expansion

No	Query	Kode Artikel									
		1	2	3	4	5	6	7	8	9	10
1	Juara Dunia	100	80	73	70	72	95	94	81	98	99
2	Pembalap	88	90	99	86	79	83	81	84	80	78
3	Sepak Bola	85	97	92	69	96	94	75	95	88	90
4	Pertandingan	97	94	92	87	85	70	69	72	36	35
5	Bertarung	100	68	97	69	92	94	77	95	71	75
6	Kejuaraan	72	73	70	100	96	95	94	86	88	98
7	Kemenangan	95	68	80	96	92	97	39	70	81	75
8	Pelatih	95	41	78	73	77	99	86	96	81	88
9	Pemain	96	70	73	92	82	69	87	7	85	13
10	Gelar	94	72	68	96	69	85	70	75	93	95
11	Cara mendapatkan uang	55	62	36	54	49	64	37	66	59	61
12	Uang	36	63	55	52	54	49	64	37	66	61
13	Gaji bulanan	49	58	55	45	60	61	66	37	64	41
14	Cuti	61	58	52	63	48	49	55	60	66	37
15	Kerja	64	58	51	8	62	47	56	61	52	49
16	Meningkatkan harga saham	66	37	35	40	44	51	60	39	43	54
17	Thr	55	52	58	49	54	61	64	36	48	67
18	Mengurangi kerugian	67	66	62	46	7	9	3	4	21	23
19	Strategi meningkatkan keuntungan	54	49	40	39	35	46	59	43	70	26
20	Penghasilan	62	54	55	73	36	47	87	80	53	98
21	Mencegah kehamilan	25	30	11	2	28	3	14	6	29	34
22	Rumah sakit	16	33	8	7	9	3	11	30	22	34
23	Terakhir kali sakit	22	33	16	99	17	90	88	86	23	30
24	Dokter	2	3	9	22	21	5	12	11	34	14
25	Peninggi badan	22	17	14	2	9	21	5	12	11	34
26	Cara meningkatkan berat badan	28	22	14	33	23	27	7	6	8	34
27	Makanan sehat	34	11	7	6	12	21	5	14	22	9
28	Penyakit menular	33	17	18	22	16	14	6	7	34	23
29	Daya tahan tubuh	23	7	26	27	25	94	72	71	95	60
30	Pengobatan	16	33	22	12	8	6	5	25	7	29

Table 4.10 Result testing without query expansion

No	Query	Kode Artikel									
		1	2	3	4	5	6	7	8	9	10
1	Juara Dunia					72	95	94		98	99
2	Pembalap		90		86		83	81		80	
3	Sepak Bola	85	97		69	96	94	75		88	90
4	Pertandingan	97	94	92	87	85	70	69	72	36	35
5	Bertarung	100	68	97	69		94				75
6	Kejuaraan	72				96	95	94	86	88	98
7	Kemenangan	95	68	80		92			70	81	75
8	Pelatih	95					99	86	96		88
9	Pemain	96		73	92	82	69	87	7	85	
10	Gelar		72	68			85	70	75	93	95
11	Cara mendapatkan uang	55	62		54	49	64				
12	Uang	36			52	54	49	64	37	66	61
13	Gaji bulanan	49	58		45	60		66	37	64	41
14	Cuti	61	58	52			49		60	66	37
15	Kerja	64	58			62	47	56	61	52	49
16	Meningkatkan harga saham			35	40	44	51	60	39	43	54
17	Thr		52	58	49	54	61	64	36		
18	Mengurangi kerugian	67	66	62	46				4	21	23
19	Strategi meningkatkan keuntungan	54			39		46	59		70	26
20	Penghasilan	62	54	55		36	47	87			98
21	Mencegah kehamilan	25		11	2	28		14	6	29	34
22	Rumah sakit	16	33		7		3		30	22	
23	Terakhir kali sakit	22	33	16				88	86	23	30
24	Dokter		3	9				12	11	34	14
25	Peninggi badan		17	14		9				11	34
26	Cara meningkatkan berat badan	28		14	33		27	7	6	8	34
27	Makanan sehat	34	11	7	6				14		9
28	Penyakit menular	33	17	18	22	16					
29	Daya tahan tubuh	23	7	26	27	25					60
30	Pengobatan		33	22		8	6		25	7	

Table 4.11 Calculation testing without query expansion

No	Query	n(A)	n(B)	n(D)	C = B ∩ D	Precision	A	B	D	Accuration
1	Juara Dunia	100	10	15	5	50	80	5	10	85
2	Pembalap	100	10	15	5	50	80	5	10	85
3	Sepak Bola	100	10	15	8	80	83	2	7	91
4	Pertandingan	100	10	15	10	100	85	0	5	95
5	Bertarung	100	10	15	6	60	81	4	9	87
6	Kejuaraan	100	10	15	7	70	82	3	8	89
7	Kemenangan	100	10	15	7	70	82	3	8	89
8	Pelatih	100	10	15	5	50	80	5	10	85
9	Pemain	100	10	15	8	80	83	2	7	91
10	Gelar	100	10	15	7	70	82	3	8	89
11	Cara mendapatkan uang	100	10	15	5	50	80	5	10	85
12	Uang	100	10	15	8	80	83	2	7	91
13	Gaji bulanan	100	10	15	8	80	83	2	7	91
14	Cuti	100	10	15	7	70	82	3	8	89
15	Kerja	100	10	15	8	80	83	2	7	91
16	Meningkatkan harga saham	100	10	15	8	80	83	2	7	91
17	Thr	100	10	15	7	70	82	3	8	89
18	Mengurangi kerugian	100	10	15	7	70	82	3	8	89
19	Strategi meningkatkan keuntungan	100	10	15	6	60	81	4	9	87
20	Penghasilan	100	10	15	7	70	82	3	8	89
21	Mencegah kehamilan	100	10	15	8	80	83	2	7	91
22	Rumah sakit	100	10	15	6	60	81	4	9	87
23	Terakhir kali sakit	100	10	15	7	70	82	3	8	89
24	Dokter	100	10	15	6	60	81	4	9	87
25	Peninggi badan	100	10	15	5	50	80	5	10	85
26	Cara meningkatkan berat badan	100	10	15	8	80	83	2	7	91
27	Makanan sehat	100	10	15	6	60	81	4	9	87
28	Penyakit menular	100	10	15	5	50	80	5	10	85
29	Daya tahan tubuh	100	10	15	6	60	81	4	9	87
30	Pengobatan	100	10	15	6	60	81	4	9	87
Total						2020				2654
Average						63.33				88.47

Table 4.12 Supervisedwith query expansion

No	Query	Kode Artikel														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Juara Dunia	42	75	18	85	56	61	72	95	98	38	82	45	70	81	47
2	Pembalap	89	81	83	80	91	86	90	71	77	98	79	78	84	88	99
3	Sepak Bola	85	22	33	17	92	26	7	6	14	8	34	11	25	9	12
4	Pertandingan	94	42	72	18	82	96	92	6	1	61	12	74	89	62	97
5	Bertarung	42	72	61	97	92	99	62	87	38	70	35	83	85	84	69
6	Kejuaraan	70	95	38	94	96	82	69	87	31	85	19	60	23	36	26
7	Kemenangan	95	75	69	96	41	93	87	73	94	82	22	90	16	30	91
8	Pelatih	54	29	5	49	44	55	42	73	18	14	1	92	85	43	27
9	Pemain	93	22	85	41	27	70	29	1	49	5	21	47	23	96	82
10	Gelar	33	95	11	42	27	53	68	96	28	18	30	41	36	59	13
11	Cara mendapatkan uang	42	43	47	18	38	65	48	54	19	35	15	44	50	49	3
12	Uang	53	45	42	35	36	56	48	2	54	58	8	64	52	49	55
13	Gaji Bulanan	52	45	56	22	53	26	20	36	39	8	92	46	38	49	60
14	Cuti	7	11	61	45	57	58	54	52	49	64	37	66	60	36	67
15	Kerja	47	65	22	63	42	51	50	31	54	48	49	29	7	32	10
16	Meningkatkan Harga Saham	50	62	52	18	46	20	7	56	16	30	61	32	96	47	78
17	Thr	58	54	52	49	64	37	66	60	61	36	67	55	39	63	48
18	Mengurangi Kerugian	20	19	45	21	4	26	7	55	15	81	50	25	3	6	86
19	Strategi Meningkatkan Keuntungan	50	18	20	62	46	47	32	96	78	61	15	1	83	26	42
20	Penghasilan	11	44	65	31	20	30	12	22	56	14	25	13	17	4	35
21	Mencegah Kehamilan	28	29	9	17	19	33	25	16	39	15	40	3	22	70	20
22	Rumah Sakit	62	3	17	56	69	33	42	1	23	65	28	85	52	16	5
23	Terakhir Kali Sakit	10	19	15	47	4	42	33	30	17	55	52	20	27	65	29
24	Dokter	3	7	6	14	8	34	11	25	9	12	22	5	29	2	21
25	Peninggi Badan	23	65	2	9	18	1	42	19	75	3	96	54	49	51	58
26	Cara Meningkatkan Berat Badan	42	65	50	18	47	20	43	19	48	15	9	1	22	3	26
27	Makanan Sehat	20	26	15	1	31	23	25	29	39	54	52	16	58	18	43
28	Penyakit Menular	33	17	3	69	28	1	16	99	12	88	22	27	41	13	81
29	Daya Tahan Tubuh	65	15	23	1	42	18	9	24	20	58	47	22	26	25	31
30	Pengobatan	22	33	7	6	14	28	8	34	11	25	16	9	12	5	29

Table 4.13 Form testing with query expansion

No	Query	Kode Artikel									
		1	2	3	4	5	6	7	8	9	10
1	Juara Dunia	72	95	81	75	70	82	98	85	38	42
2	Pembalap	83	71	99	90	88	86	80	79	78	77
3	Sepak Bola	85	92	22	7	34	26	12	6	11	8
4	Pertandingan	94	82	92	96	89	72	62	12	42	6
5	Bertarung	92	69	87	70	85	99	84	97	72	83
6	Kejuaraan	85	87	94	69	82	96	70	95	60	36
7	Kemenangan	73	94	87	69	95	96	82	75	91	41
8	Pelatih	92	85	73	27	1	55	29	44	43	54
9	Pemain	93	96	82	70	85	1	21	41	47	23
10	Gelar	95	68	96	18	53	42	59	41	30	36
11	Cara mendapatkan uang	35	44	42	54	43	49	38	65	47	50
12	Uang	54	49	52	55	45	58	35	56	42	36
13	Gaji Bulanan	52	49	45	56	46	36	60	38	22	26
14	Cuti	61	45	54	58	64	49	52	67	66	7
15	Kerja	51	47	42	54	31	10	63	65	48	50
16	Meningkatkan Harga Saham	62	47	46	56	78	96	50	52	61	20
17	Thr	49	52	54	55	58	61	39	64	36	60
18	Mengurangi Kerugian	454	55	6	19	20	4	21	6	15	7
19	Strategi Meningkatkan Keuntungan	78	62	47	42	96	61	46	20	26	15
20	Penghasilan	35	44	65	56	31	12	4	20	13	17
21	Mencegah Kehamilan	25	28	9	15	29	33	20	19	22	3
22	Rumah Sakit	17	16	33	28	23	5	1	56	3	52
23	Terakhir Kali Sakit	33	17	4	30	27	15	19	10	20	29
24	Dokter	34	12	22	2	7	9	11	8	3	25
25	Peninggi Badan	19	23	9	1	2	3	18	58	51	75
26	Cara Meningkatkan Berat Badan	20	19	9	22	43	15	1	3	18	50
27	Makanan Sehat	23	20	15	39	31	43	54	16	25	52
28	Penyakit Menular	17	33	16	13	27	1	22	12	3	28
29	Daya Tahan Tubuh	23	24	20	9	22	26	31	15	1	25
30	Pengobatan	33	34	16	12	22	7	29	8	11	5



Table 4.14 Result testing with query expansion

No	Query	Kode Artikel									
		1	2	3	4	5	6	7	8	9	10
1	Juara Dunia	72	95		75			98	85	38	42
2	Pembalap	83	71		90		86	80			77
3	Sepak Bola	85	92	22	7		26		6		8
4	Pertandingan	94	82	92	96		72			42	6
5	Bertarung	92		87	70		99		97	72	
6	Kejuaraan	85	87	94	69	82	96	70	95		
7	Kemenangan	73	94	87	69	95	96	82	75		41
8	Pelatih		85	73			55	29	44		54
9	Pemain	93			70	85	1		41		
10	Gelar	95	68	96	18	53	42				
11	Cara mendapatkan uang	35		42	54	43		38	65	47	
12	Uang	54				45	58	35	56	42	36
13	Gaji Bulanan	52		45	56		36			22	26
14	Cuti	61	45	54	58	64	49	52			7
15	Kerja	51	47	42	54	31		63	65	48	50
16	Meningkatkan Harga Saham	62		46	56			50	52		20
17	Thr	49	52	54		58	61		64	36	60
18	Mengurangi Kerugian	45	55		19	20	4	21	6	15	7
19	Strategi Meningkatkan Keuntungan	78	62	47		96	61	46	20		
20	Penghasilan		44	65	56	31	12		20		
21	Mencegah Kehamilan	25	28	9	15	29	33	20	19		
22	Rumah Sakit	17		33		23	5	1	56	3	
23	Terakhir Kali Sakit	33	17	4	30		15	19	10		
24	Dokter	34	12			7	9	11	8	3	25
25	Peninggi Badan	19	23	9	1	2	3	18			75
26	Cara Meningkatkan Berat Badan	20	19			43	15			18	50
27	Makanan Sehat	23	20	15	39	31		54		25	
28	Penyakit Menular	17	33	16			1		12	3	28
29	Daya Tahan Tubuh	23	24	20	9				15	1	
30	Pengobatan	33	34			22	7		8	11	

Table 4.15 Calculation testing with query expansion

No	Query	n(A)	n(B)	n(D)	$C = B \cap D$	Precision	A	B	D	Accuration
1	Juara Dunia	100	10	15	7	70	82	3	8	89
2	Pembalap	100	10	15	6	60	81	4	9	87
3	Sepak Bola	100	10	15	7	70	82	3	8	89
4	Pertandingan	100	10	15	7	70	82	3	8	89
5	Bertarung	100	10	15	6	60	81	4	9	87
6	Kejuaraan	100	10	15	8	80	83	2	7	91
7	Kemenangan	100	10	15	9	90	84	1	6	93
8	Pelatih	100	10	15	6	60	81	4	9	87
9	Pemain	100	10	15	5	50	80	5	10	85
10	Gelar	100	10	15	6	60	81	4	9	87
11	Cara mendapatkan uang	100	10	15	7	70	82	3	8	89
12	Uang	100	10	15	7	70	82	3	8	89
13	Gaji Bulanan	100	10	15	6	60	81	4	9	87
14	Cuti	100	10	15	8	80	83	2	7	91
15	Kerja	100	10	15	9	90	84	1	6	93
16	Meningkatkan Harga Saham	100	10	15	6	60	81	4	9	87
17	Thr	100	10	15	8	80	83	2	7	91
18	Mengurangi Kerugian	100	10	15	9	90	84	1	6	93
19	Strategi Meningkatkan Keuntungan	100	10	15	7	70	82	3	8	89
20	Penghasilan	100	10	15	6	60	81	4	9	87
21	Mencegah Kehamilan	100	10	15	8	80	83	2	7	91
22	Rumah Sakit	100	10	15	8	80	83	2	7	91
23	Terakhir Kali Sakit	100	10	15	7	70	82	3	8	89
24	Dokter	100	10	15	8	80	83	2	7	91
25	Peninggi Badan	100	10	15	8	80	83	2	7	91
26	Cara Meningkatkan Berat Badan	100	10	15	6	60	81	4	9	87
27	Makanan Sehat	100	10	15	7	70	82	3	8	89
28	Penyakit Menular	100	10	15	7	70	82	3	8	89
29	Daya Tahan Tubuh	100	10	15	6	60	81	4	9	87
30	Pengobatan	100	10	15	6	60	81	4	9	87
Total						2110				2672
Average						70.33				89.07

Where :

$S = \{\text{articles}\}$

$A = S - B \cup D$

$B = \{\text{reference rank}\}$

$D = \{\text{systems rank}\}$

$C = B \cap D$

Precision =  $(C/n(B)) * 100$

Accuration =  $((C + A) / (A + B \cup D)) * 100$

Table 4.16 Result of IR System

Query	Title
Juara dunia	Kalahkan China, Korea Juara Piala Sudirman
	Fakta Seputar Korea Selatan pada Piala Sudirman 2017
	Rika Wijayanti Raih Perunggu Kejuaraan Dunia Paralayang
	Valentino Rossi Kecelakaan Saat Latihan Motokros
	Canelo Alvarez Menang Mutlak Atas Cotto
	Valentino Rossi Boleh Tinggalkan Rumah Sakit
	Kondisi Membaik, Valentino Rossi Masih Jalani Perawatan di Rumah Sakit
	China Yang Berubah Pasca Era Li Yongbo
	Dinyatakan Fit, Rossi Bisa Ikut GP Italia
	Pebalap Astra Honda Racing Team Siap Beraksi di Sirkuit Suzuka
	Pedrosa: Hadiah Menyenangkan dari Rossi
	Timnas Basket Catat Kemenangan Beruntun di SEABA
	Rossi: Sayang Harus Pulang Tanpa Poin
	Cotto Akan Bertarung dengan Juan Manuel Marquez
	Manfaat Simulasi Piala Sudirman bagi Marcus Fernaldi

Table 4.16 is result from Information Retrieval by query “Juara Dunia”. Table 4.2 is the result of testing. Experimental results show a precision retrieval information system is 70.33% and accuracy retrieval information system is 89.07%. Table 4.5 is the result of testing with sports theme. Experimental results show a precision retrieval information system is 63.33% and accuracy retrieval information system is 88.47%.

From various experiments conducted obtained results in the precision and accuracy of information retrieval using query expansion and without using query

expansion. Precision system information retrieval without using query expansion is 63.33 % and using query expansion is 70.33%. Comparison of 2 results suggests that the system information retrieval using query expansion is better than not using query expansion. Because there is a precision increase of 7%.

Accuration system information retrieval without using query expansion is 88.47 % and using query expansion is 89.07 %. Comparison of 2 results suggests that the system information retrieval using query expansion is better than not using query expansion too. Because there is a accuration increase of 0.6 %.Of the results we can say that use of query expansion in system information retrieval can improve system performance.

From the calculation of precision and accuracy of thesaurus usage for query expansion in system information retrieval get better result than without using thesaurus. This shows that the use of thesaurus can improve system information retrieval performance

#### 4.4. Integration of Research with Al-Quran

Information retrieval is the process of returning information according to the needs of the user. So the computer is required to understand the information. In understanding an information then the first thing to do is read. This research teach computer to read article from each page. Al quran gives command to read in surah al alaq ayah1:

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ (1)

Means : “Read: In the name of thy Lord Who createth,”

One day he was surprised by the revelation of his arrival at the Hira Cave. The angel of the revelation entered into the cave to meet him, then said, "Read!" Rasulullah Saw. Continued his story, that he replied, "I am not a good reader." So the angel held me up and held me so much that I was really tired of him, after which he let go of me and said again, "Read!" Prophet SAW. Replied, "I'm not a clever person." The angel came back to me for a second time until I was really tired, then let go of me and said, "Read!" I replied, "I'm not a good reader." The angel came back to me for the third time until I was really tired, then he let go of me and said:

Read: In the name of thy Lord Who createth. (Al-'Alaq: 1) up to His word: what he did not know. (Al-'Alaq: 5)

This surah is the first to come down on the Prophet Muhammad SAW. And is also the first command given by God. The researcher implements this command to teach the computer how to read the article. and the computer can provide information from the readable article called information retrieval.

The purpose of Information retrieval is give information to user about something. Allah SWT command to validation the information as in surah al hujurat ayah 6:

يَا أَيُّهَا الَّذِينَ آمَنُوا إِنْ جَاءَكُمْ فَاسِقٌ بِنَبَأٍ فَتَبَيَّنُوا أَنْ تُصِيبُوا قَوْمًا بِجَهَالَةٍ فَتُصْبِحُوا عَلَىٰ مَا فَعَلْتُمْ  
 نَادِمِينَ

Means: "O ye who believe! If an evil-liver bring you tidings, verify it, lest ye smite some folk in ignorance and afterward repent of what ye did."

Allah SWT. Command (the believers) to thoroughly examine the message of the wicked, and let them be cautious in accepting it and do not take it for granted, which consequently will reverse the reality. The person who receives just the news from him, means the same as following in his footsteps. While Allah SWT. Has forbidden the believers from following the path of the corrupted.

Departing from this understanding there are some scholars who forbid us to receive news (history) from unknown persons, because perhaps he is a wicked man. But some other scholars would accept it on the grounds that we were only ordered to examine the truth of the wicked news, while the unknown (majhul) still has not proved his wickedness because he is unknown.

This surah means that we must be selective in receiving information. Because missing formation can harm a people. So the retrieval system that the researcher build must provide the correct information so as not to defame a people. As instructed in the above surah. By increasing the precision and accuracy of the information retrieval system. So the system can deliver the right results according to user needs.

## CHAPTER V

### CONCLUSION AND SUGGESTION

#### 5.1. Conclusion

From the results of implementation and experiments conducted by researchers got the conclusion:

- a. Breadth-First Algorithm Method Can be applied to web crawlers to collect articles from websites. Using the Breadth-First Algorithm Method obtained the average time required to collect one article is 1.65013 seconds.
- b. Based on precision testing results for system information retrieval without using thesaurus is 63.33% and 88.47% accuracy. While the precision for system information retrieval using thesaurus is 70.33% and 89.07% accuracy. Can be concluded by using a thesaurus can increase the precision of 7% and 0.6% accuracy on the system information retrieval.

#### 5.2. Suggestion

Some suggestions for further research are as follows:

- a. Web crawler still running on single thread. It is expected that in the next research can use multi thread to increase crawler speed.
- b. In this study, the query is split into one word, it is expected the next research, query is not only splitdown into one word but broken down based on the phrase as well.

## REFERENSI

- Agre, G. H., & Mahajan, N. V. (2015). Keyword focused web crawler. *2nd International Conference on Electronics and Communication Systems, ICECS 2015*, 1089–1092. <https://doi.org/10.1109/ECS.2015.7124749>
- Ahmad, W., & Ali, R. (2016). Textual Content based Information Retrieval from Twitter, 2668–2672.
- Alshari, E. M., Azman, A., Mustapha, N., Doraisamy, S. A. C., & Alksher, M. (2016). Prediction of Rating from Comments based on Information Retrieval and Sentiment Analysis, 32–36.
- Ayetiran, E. F., Aruleba, K. D., Ekong, D. O., & Science, C. (2016). Algorithm for Information Retrieval Optimization 1 1.
- Bahrami, M., Singhal, M., & Zhuang, Z. (2015). A cloud-based web crawler architecture. *2015 18th International Conference on Intelligence in Next Generation Networks, ICIN 2015*, 216–223. <https://doi.org/10.1109/ICIN.2015.7073834>
- Bai, Q., Xiong, G., Zhao, Y., & He, L. (2014). Analysis and detection of bogus behavior in web crawler measurement. *Procedia Computer Science*, *31*, 1084–1091. <https://doi.org/10.1016/j.procs.2014.05.363>
- Chen, M., & Song, Y. (2009). Summarization of Text Clustering based Vector Space Model, (i).
- Goswami, P., Gaussier, E., & Amini, M.-R. (2017). Exploring the space of information retrieval term scoring functions. *Information Processing & Management*, *53*(2), 454–472. <https://doi.org/10.1016/j.ipm.2016.11.003>
- Gupta, A., & Anand, P. (2015). Focused web crawlers and its approaches. *2015 1st International Conference on Futuristic Trends in Computational Analysis and Knowledge Management, ABLAZE 2015*, 619–622. <https://doi.org/10.1109/ABLAZE.2015.7154936>
- Gupta, Y., Saini, A., & Saxena, A. K. (2015). A new fuzzy logic based ranking function for efficient Information Retrieval system. *Expert Systems with Applications*, *42*(3), 1223–1234. <https://doi.org/10.1016/j.eswa.2014.09.009>
- Internet Research, Anti-Phishing and PCI Security Services | Netcraft. (n.d.). Retrieved August 20, 2016, from <https://news.netcraft.com/>
- Kapoor, A. (2016). Application of bloom filter for duplicate URL detection in a web crawler. <https://doi.org/10.1109/CIC.2016.40>
- Kauer, A. U., & Moreira, V. P. (2016). Using information retrieval for sentiment polarity prediction. *Expert Systems with Applications*, *61*, 282–289. <https://doi.org/10.1016/j.eswa.2016.05.038>



- Khari, M. (2016). Analysis of Various Information Retrieval Models, 2176–2181.
- Lawankar, A. (2016). A Review on Techniques for Optimizing Web Crawler Results, 3–6.
- Lee, D. L., Chuang, H., & Seamons, K. (1997). Document ranking and the vector-space model. *IEEE Software*, 14(2), 67–75. <https://doi.org/10.1109/52.582976>
- Nadu, T. (2014). TEXT PROCESSING IN INFORMATION RETRIEVAL SYSTEM USING VECTOR SPACE MODEL, (978), 0–5.
- Qian, R., Zhang, K., & Zhao, G. (2014). A topic-specific Web crawler based on content and structure mining. *Proceedings of 2013 3rd International Conference on Computer Science and Network Technology, ICCSNT 2013*, 458–461. <https://doi.org/10.1109/ICCSNT.2013.6967153>
- Rasyidi, I., & Romadhony, A. (2013). Indonesian Hadith Retrieval System using Thesaurus, 285–288.
- Science, C., Technology, I., & Samarahan, K. (2016). Ontology-Based Information Retrieval for Historical Documents, 55–59.
- Sharma, S., & Gupta, P. (2015). The anatomy of web crawlers. In *International Conference on Computing, Communication and Automation, ICCCA 2015* (pp. 849–853). <https://doi.org/10.1109/CCAA.2015.7148493>
- Sharma, V. K., & Mittal, N. (2016). Exploiting Wikipedia API for Hindi-english Cross-language Information Retrieval. *Procedia Computer Science*, 89, 434–440. <https://doi.org/10.1016/j.procs.2016.06.094>
- Siddiqui, M. A. (2015). URL Ordering based Performance Evaluation of Web Crawler, 4(1), 149–156.
- Wang, X. (2012). Text Clustering Based on the Improved TFIDF by the Iterative Algorithm, 140–143.