Implementation of LSI Method on Information Retrieval for Text Document in Bahasa Indonesia

Jasman Pardede and Mira Musrini Barmawi

***Abstract*—Information retrieval system is a system that is used to obtain the information based on user's requirement. In this study, Latent Semantic Indexing (LSI) method is implemented in the system to search and to collect documents based on overall meaning of documents instead of individual's word. Typical of documents that needs to be retrieved in the system are text document in *doc, *.docx, or *.pdf formatted. In the text preprocessing phase, Nazief and Adriani Algorithm is used to eliminate the affix (prefix, suffix, etc.) of a word and then match them in database root word. To evaluate the quality of information retrieval performance, time response, values of recall and precision are measured. Implementation of multithreading from 'read document' to stemming process is required in order to improve time responses. The result shows by using multithreading, the greater number of term in document collection gives the more efficient in required time response. In term of the required time response, the document collection in docx format is the fastest, followed by doc and pdf format. For 80 documents and beyond, the system produces an error "OutOfMemoryError" at the matrix decomposition process. This means that the greater number of document in the collection, the greater memory is needed to execute retrieval process.**

***Index Terms*—effectiveness, LSI, precision, recall, multithread.**

I. INTRODUCTION

Typically, information is retrieved by literally matching terms in documents with those of a query. However, lexical matching methods can be inaccurate when they are used to match a user's query. Since there are usually many ways to express a given concept (synonymy), the literal terms in a user's query may not match those of a relevant document. In addition, most words have multiple meanings (polysemy), so terms in a user's query will literally match terms in irrelevant documents. A better approach would allow users to retrieve information on the basis of a conceptual topic or meaning of a document. LSI method tries to overcome the problem of lexical matching [1].

Since information becomes more huge and complex every day, it is impossible to collect and search information from documents manually. We need to build a system to help user in finding information, the system is called Information Retrieval System. An ideal Information Retrieval (IR) system is a system which can

1) Find relevant information
2) Only find relevant information and can't find irrelevant information.

Problem formulation arises such as: How can the Information Retrieval system read document files which have format doc, docx, and pdf. How can Information Retrieval System obtain relevant documents using the LSI method. How can Information Retrieval System, using LSI method, order the retrieved documents which have most relevant meaning with the query given by user. The objective of the research is to analyze and to implement LSI method in Information Retrieval System and to evaluate the quality of retrieval performance by measuring precision, recall and time value response.

II. THEORY

Information Retrieval system is used to retrieve back automatically information from information collection that most relevant to user's requirement [2], [3], [4], [5], [6]. According to Goeker and Davies, IR systems or information retrieval systems is part of computer science of information retrieval of documents based on content and context in documents themselves [1], [7], [8]. According to Gerald Kowalski, information retrieval system is a system that is capable to storage, retrieve, and maintain the information. The information in this context can consist of text (including numerical data and date), images, audio, video, and other multimedia objects [9].

According to Manning, the definition IR is how to find a document from unstructured documents that provide the information needed from a very large collection of documents stored in a computer [3]. The most common implementation of IR system is search engine applied in the internet. Users can find web pages based on requirements through search engine. Another example of IR system is library information systems.

Jasman Pardede is with the Department of Informatics Engineering, Faculty of Industrial Technology, Institut Teknologi Nasional (Itenas) Bandung, Indonesia. He can be reached at e-mail: jasman@itenas.ac.id.

Mira Musrini Barmawi is with the Department of Informatics Engineering, Faculty of Industrial Technology, Institut Teknologi Nasional (Itenas) Bandung, Indonesia. She can be reached at e-mail: sangkuriang26@ yahoo.com

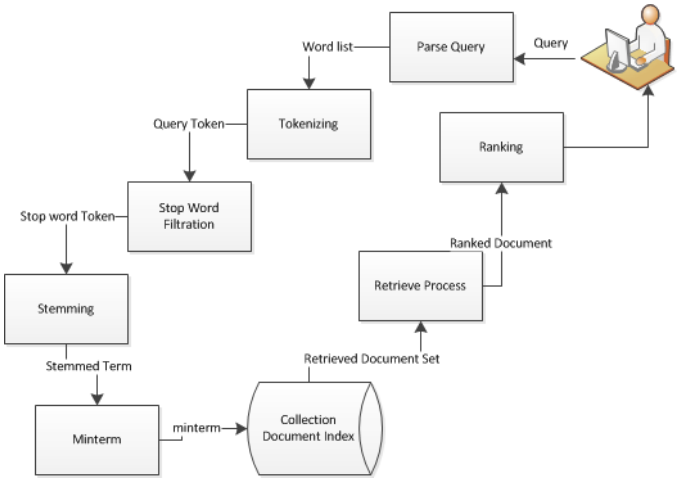

Fig. 1. Schema of IR system

IR systems have several parts that build the overall system as described in Fig.1. These parts can be described as follow:

a. Text operation. This operation includes word selection in query and document, and transformation those word into term index.
b. Query Formulation. We give weights in term index resulted from query.
c. Indexing. The database index from document collections is built,

Ranking, in this process, is the searching of documents that relevant with query is executed, and those documents are sorted based on degree of conformity of the document to the query.

III. METHODOLOGY

A. LSI Method

LSI method is a method that is implemented in IR System in order to search and to find information based on the overall meaning of a document not only the meaning of the individual words. In LSI method, document collection is built in form of vector space by using Singular Value Decompositions (SVD) technique, a technique from Linear Algebra [1].

In general, the process flow in LSI method can be described as follows (as shown Fig. 2):

a) Text operation in query and document collection includes: The process of reading the text (*. Doc, *. Docx, *.pdf), tokenization, filtration, stemming and parse the query and the document collection.
b) Documents Matrix is created based on the result of text operation in document collections. The row indicates words and the column indicates documents. The matrix element indicates the word frequencies. For example, matrix element in first row and second column indicates frequency of first word (T_1) in second document.

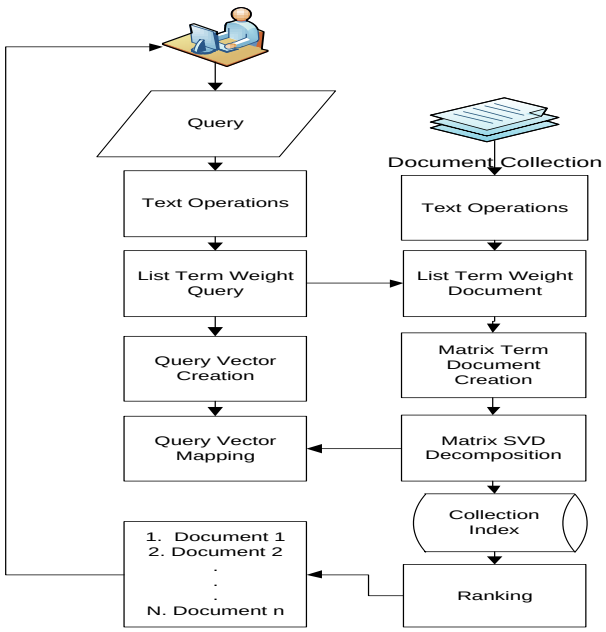

Fig. 2. The process flow in LSI method

c) SVD decompositions is processed as follows

i. The documents matrix, called $\boldsymbol{A_{mxn}}$, is decomposed into production of three matrices by using Singular Value Decompositions. So $\boldsymbol{A}$ can be rewritten as:

$$\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{S}\,\boldsymbol{V}^{\boldsymbol{T}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

ii. For further explanation, we assume that: $\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_n}$ as column vector of matric $\boldsymbol{U}$, $\boldsymbol{\sigma_1}, \boldsymbol{\sigma_2}, \boldsymbol{\sigma_3}, \ldots, \boldsymbol{\sigma_n}$ as element of diagonal of matric $\boldsymbol{S}$, $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}$ as column vector of matric $\boldsymbol{V}$. Thus,

$$\boldsymbol{A} = \boldsymbol{U}\ \boldsymbol{S}\ \boldsymbol{V}^{\boldsymbol{T}}$$

$$A = [u_1 \quad u_2 \quad \ldots u_n] \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & \vdots & \ddots \\ 0 & 0 & \sigma_n \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ v_3^T \\ \vdots \\ v_n^T \end{bmatrix} \ldots \quad (2)$$

iii. Rank of $\boldsymbol{A}$ is $\boldsymbol{k}$. The diagonal element of $\boldsymbol{S}$ that is $\boldsymbol{\sigma_1}, \boldsymbol{\sigma_2}, \boldsymbol{\sigma_3}, \ldots, \boldsymbol{\sigma_n}$. $\boldsymbol{k}$ is the amount of singular value of matrix $\boldsymbol{S}$.

iv. From these $\boldsymbol{k}$ of singular value of $\boldsymbol{A}$, we determine the biggest singular value, so $\boldsymbol{\sigma_1} > \boldsymbol{\sigma_2} > \boldsymbol{\sigma_3} > \ldots > \boldsymbol{\sigma_n} > 0$, where $\boldsymbol{r} < \boldsymbol{k}$.

v. New matric $\boldsymbol{A}$ can be rewritten as follows:

$$\boldsymbol{A} = \boldsymbol{U_r}\,\boldsymbol{S_r}\,\boldsymbol{V_r^T} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

with $\quad \boldsymbol{U_r} = [u_1 \quad u_2 \quad \ldots\ldots\ldots\ldots u_n]$

$$\boldsymbol{S_r} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & \vdots & \ddots \\ 0 & 0 & \sigma_n \end{bmatrix} \quad \text{and} \quad \boldsymbol{V_r} = \begin{bmatrix} v_1^T \\ v_2^T \\ v_3^T \\ \vdots \\ v_n^T \end{bmatrix}$$

d) Query Vector Creation
The column matrix, $\boldsymbol{q}$, is obtained from text operation. The Query vector can be represented as follows:

$$\boldsymbol{q} = \begin{matrix} T_1 \\ T_2 \\ \vdots \\ T_n \end{matrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

With $\boldsymbol{q_j}$, $\boldsymbol{j}$ = 1,2,3,…, $\boldsymbol{n}$ is frequency of $\boldsymbol{T_j}$ in query.

e) Query vector mapping
Based on value $\boldsymbol{r}$ from c (iv), vector query $\boldsymbol{q}$ is mapped into vector space with $\boldsymbol{r}$ dimension as follows:

$$\boldsymbol{A} = q^T\, U_r\, S_r^{-1} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

f) Ranking. The rank of documents depends on value of angle between query vector and document vector. The smaller angle between them, the more relevant document with query.
Document vector is represented by:

$$V_r = \begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_n \end{bmatrix}, D_j = \begin{bmatrix} d_{j1} \\ \vdots \\ d_{jr} \end{bmatrix}$$

where, $\boldsymbol{D_j}$ = document vector of $\boldsymbol{j^{th}}$ document.
Assuming that query vector can be represented as:

$$\mathrm{Q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_r \end{bmatrix}$$

g) Cosine similarities between document vector and query vector is calculated as follows:

$$similarity\,(Q, D_j) = \cos(Q, D_j) = \frac{Q \cdot D_j}{|Q||D_j|} = \frac{1}{|Q||D_j|} \sum_{i=1}^{r} q_j * d_{ji} \quad \ldots\ldots\ldots\ldots\ldots\ldots (6)$$

h) Final result
The cosine similarities value between $\boldsymbol{Q}$ and $\boldsymbol{D_j}$ is obtained and sorted in descending order. The largest cosine similarity value indicates the most relevant document with query.

B. Nazief and Adriani Algorithm

Nazief and Adriani algorithm is a stemming algorithm for Indonesian text [10]. The algorithm was described first time on unpublished technical report in University of Indonesia. This algorithm uses several morphological rules to eliminate affix (prefix, suffix, etc.) of a word and then matches them in database root (root word). This algorithm is used in text processing (Fig. 2), before the document matrix and the query vector are created.

C. Multithreading For Text Operation

The process can be divided into a number of separate units called as thread. Multithreading is the ability of a program or operating system to execute and to handle user requests more than one user without duplication or copying program that runs on a computer at the same time [11]. In other words, multithreading is the ability to process multiple threads from the many processes where each thread runs separately at the same time.

Fig. 2 shows that the process of text operation which consists of tokenizing, stop word, filtration, and stemming, is a process that is done repeatedly on each document that exists in the document collection and as user input query. Differences occur only on reading the user input query and process any documents readings from document collection. Therefore, in this study, the same process is repeated at the input query and the document will be carried out by a stand-alone process through the implementation of the thread. Based on the process flow in Fig. 2, this research implements a search process that uses the scheme of multithread in the LSI method as in Fig. 3.

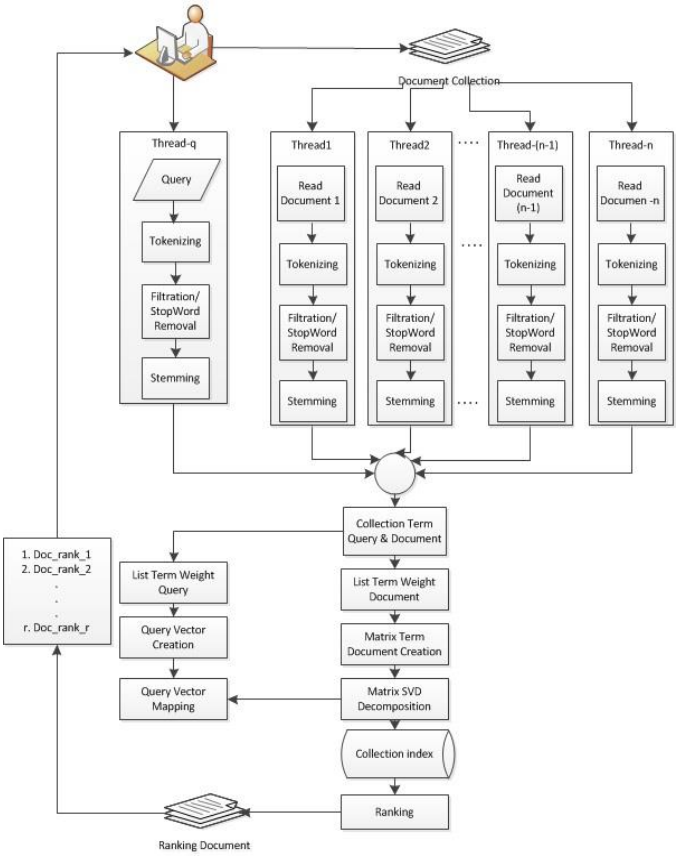

Fig. 3. The process flow in multithread for text operation on LSI method

IV. The Information Retrieval System Development

The Information Retrieval system is built by using java programming language. This system is called IRLSI

(Information Retrieval Latent Semantic Indexing) system. The development process is conducted on the basis of Watterfall software development methodology. Waterfall has 4 phases, that need to be followed, those are analysis, design, implementation and testing phase.

In analysis and design phases, object oriented design approach is used. Use case diagram of IRLSI system is defined to represent the features of IR system as represented on Fig. 4.

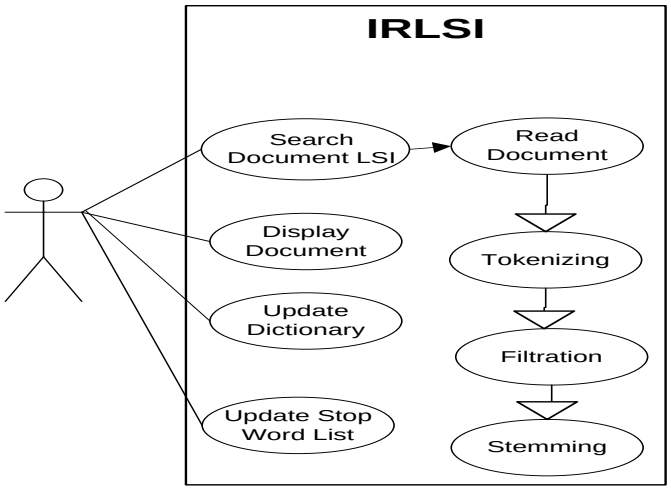

Fig. 4. Use case of IRLSI system

Tools and java class library used to develop the IR system are as follows:

a) JDK1.6.7 Java Programming language or higher version, NETBeans IDE 7.3.
b) Apache Poin function to read document in microsoft word format.
c) XML Beans to read word document file that have extension *.docx.
d) DOM4J function to read document file that have extension *.doc.
e) PDFBox function to read document file that have extension *.pdf.

Efficient Java Matrix Library (EJML) API function to create document matrix and SVD matrix.

V. Evaluation Of Information Retrieval System Performance

There are three parameters that used to evaluate the quality of information retrieval system performance, i.e: time response, recall and presicion. RECALL is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage. PRECISION is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved [12], [13], [14]. It is usually expressed as a percentage.

A testing was conducted to gain the time response, values of recall and precision. First, we prepare 10 folders filled with number of document collections. Those folders contain of 10, 20, 25, 30, 40, 50, 60, 70, 75 and 80 file documents in *.doc, *docx, *.pdf format. The testing result for docx format shows time response by using either multithreading or non multithreading. In both conditions show precision and recall for the documents are the same. The results are shown in Table I. Time response of doc and pdf format that uses multithreading is shown by Table II.

TABLE I
THE RESULT OF IRLSI SYSTEM TESTING DOCX FORMAT

No	Number of Documents	Time without Thread (s)	Time With Thread (s)	Precision (%)	Recall (%)
1	10	4.289	3.120	100	66.7
2	20	6.055	4.275	100	87.5
3	25	7.219	4.920	100	100
4	30	8.551	5.600	84.2	100
5	40	15.598	11.717	100	87
6	50	19.630	15.49	100	81.5
7	60	26.494	20.265	100	100
8	70	31.405	22.995	100	100
9	75	35.099	26.083	100	100
10	80	~	~	~	~

TABLE II
THE RESULT OF TIME RESPONSE OF THE IRLSI TESTING IMPLEMENTED MULTITHREAD TO DOCX, DOC, AND PDF FORMAT

No	Number of Documents	Format		
		docx	doc	pdf
1	10	3.120	3.198	3.573
2	20	4.275	4.290	4.508
3	25	4.920	5.135	5.320
4	30	5.600	5.787	6.130
5	40	11.717	11.889	12.545
6	50	15.490	15.943	16.645
7	60	20.265	20.373	21.59
8	70	22.995	23.135	23.852
9	75	26.083	26.411	27.159

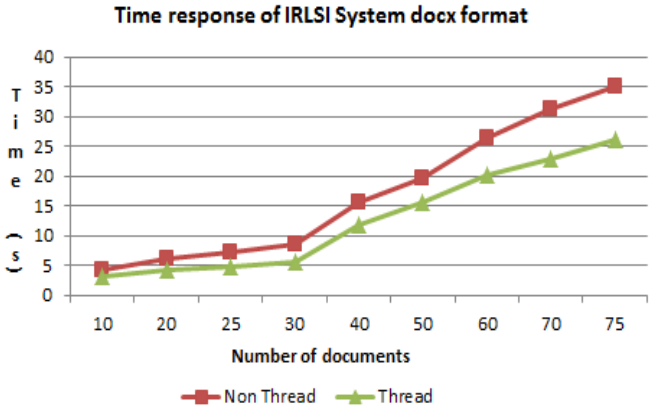

Fig. 5. Graph of time response of the IRLSI system comparing with-and without multithreading

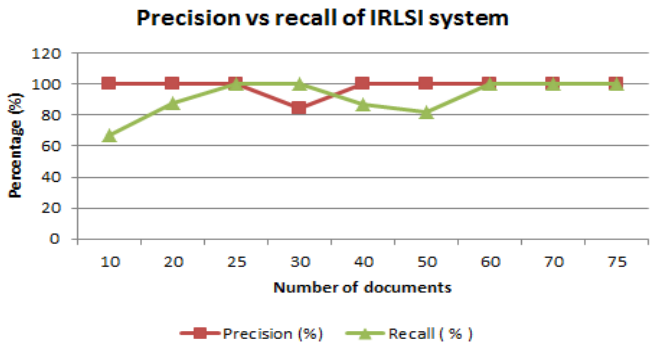

Fig. 6. Graph of recall vs. precision of the IRLSI system

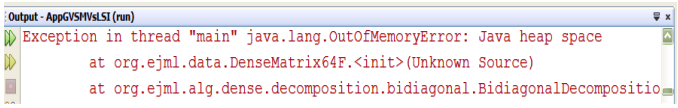

Fig. 7. OutOfMemoryError testing for 80 documents

The Fig. 5 indicates that the larger the document collections, the required time response is larger by the system to retrieve documents with using multithread or not.

In Fig. 6, the values of precision and recall are 84.2% and 100% respectively for test case no. 4 (number of documents is 30). This result occurred because the LSI method retrieves all relevant documents based on the overall meaning of document instead of similarities of individual words.

In our case, if the collections of documents reach the number of 80 then error message "out of Memory error" will be occurred. This means that the system needs more space memory to execute retrieval process of 80 documents. The result of IRLSI system testing for 80 documents is represented on Fig. 7.

VI. Conclusion

The Latent Semantic Indexing method was successfully implemented in IR system that can retrieve document of bahasa Indonesia in doc, docx, and pdf format. Speed of time response that required by the document collection in docx format was the fastest, followed by doc format and pdf format. The application of multithread for text operation on LSI method can increase time response required to about 28.69 % in average. The greater number of document in the collections resulted in the more efficient using multithread. The greater amount of terms in the document collections leaded to the greater response time required for the system to execute retrieval process. The document matrix with larger size resulted in greater memory requirement to retrieve documents. Based on our experiment, the system may produce "Out of memory" error.

References

[1] Rosario, B., *Latent Semantic Indexing : An Overview*, INFOSYS 240, Spring 2000, 2000.

[2] Ingwersen, I. And Järvelin, K.., *The turn: integration of information seeking and retrieval in context*, Springer, 2005.

[3] Manning, C., D., et al, *An Introduction to Information Retrieval*, Cambridge University Press, England, 2009.

[4] Soboroff, I. And Nicholas, C., *Collaborative Filtering and The Generalized Vector Space Model*, Athen, Greece, 2000, pp.351-353.

[5] Tsatsaronis, G. And Panagiotopoulou, V., *A Generalized Vector Space Model for Text Retrieval Based on Semantic Relatedness*, Proceedings of the EACL 2009 Student Research Workshop, Athen, Greece, 2009,pp.70-78.

[6] Wong, S.K.M, Ziarko, W., and Patrick, C.N.W., *Generalized Vector Space Model In Information Retrieval*, 1985, [Online]. Available : http://www.cs.odu.edu/~jbollen/IR04/ readings/p18-wong.pdf, access date 27 December 2014.

[7] Goker, A., and Davies, J, *Information Retrieval : Searching In The 21st Century*, A John Wiley and Sons, Ltd., Publication, United Kingdom, 2009.

[8] Yates, R.B., and Neto, B.R., *Modern Information Retrieval*, ACM Press, New York, 1999.

[9] Kowalski, G., *Information Retrieval System Theory and Implementation*, Kluwer Academic Publishers, United States of America, 1997.

[10] Nazief, Bobby dan Mirna Adriani, *Confix-Stripping: Approach to Stemming Algorithm for Bahasa Indonesia*, Faculty of Computer Science University of Indonesia.

[11] Oracle, *Multithreaded Programming Guide*, 2012, [Online]. Available : Oracle and/or its affiliates, access date 27 December 2014.

[12] Power, D.M.W., *Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation*, Journal of Machine Learning Technologies, 2011, pp.37-63.

[13] Robertson, S., *On Document Populations and Measures of IR Effectiveness*, Microsoft Research, Cambridge, UK, 2007.

[14] Smucker, M.D, and Clarke, C.L.A., *Time-Based Calibration of Effectiveness Measures*, SIGIR'12, Portland, Oregon, USA, 2012.

Jasman Pardede received Bachelor degree in science math from Universitas Andalas (Unand), Indonesia, in 2001, and the M.Eng degree in Informatic Engineering from Institut Teknologi Bandung (ITB), in 2005. He currently works at Institut Teknologi Nasional (Itenas), Bandung, as a lecturer.

Mira Musrini Barmawi received Bachelor degree in science math from Institut Teknologi Bandung (ITB), Indonesia, in 1990, and the M.Eng degree in Informatic Engineering from the same institute, in 2006. She currently works at Institut Teknologi Nasional (Itenas), Bandung, as a lecturer.