Implementation of Pattern Matching Algorithm for Portable Document Format

Anton Yudhana
Department of Electrical Engineering
Ahmad Dahlan University
Yogyakarta, Indonesia

Sunardi
Department of Electrical Engineering
Ahmad Dahlan University
Yogyakarta, Indonesia

Abdul Djalil Djayali
Master of Informatics Engineering
Ahmad Dahlan University
Yogyakarta, Indonesia

***Abstract*—Internet availability and e-documents are freely used in the community. This condition has the potential for the occurrence of the act of plagiarism against an e-document of scientific work. The process of detecting plagiarism in some cases seems to be done manually by using human power so that it has the potential to make mistakes in observing and remembering the checkpoints that have been done. The method used in this research is to represent two sets of objects compared in the form of probability. In order for the method to run perfectly, the Rabin-Karp algorithm is applied, wherein Rabin-Karp is a string matching algorithm that uses hash functions as a comparison between the searched string (m) and substring in the text (n). If both hash values are the same then the comparison will be done once again to the characters. The resulting system is a web-based application that shows the value of the similarity of two sets of objects.**

***Keywords*—Pattern matching; Rabin-Karp algorithm; data mining; web**

I. INTRODUCTION

Plagiarism turns out to infect developing countries like Indonesia. Some recent cases are even found in developed countries like the United States. The difference is that developed countries impose sanctions that do not play games with plagiarism, while Indonesia still seems shy to impose tough sanctions because most of the scientific work has not been protected by *Hak atas Kekayaan Intelektual (HaKI)* then plagiarism is classified as an academic crime that including as ethical violations and difficult to be criminalized. As the first step to prevent a similar case is needed how to detect the possibility of such plagiarism in the college environment that is primarily on the final outcome of undergraduate candidates and undergraduate thesis of master degree and doctoral dissertation candidates who are prone to plagiarism [1].

There are two main classes of methods used to reduce plagiarism: methods of preventing plagiarism and methods of detecting plagiarism. Prevention methods of plagiarism include ritual punishment and complementary procedures of plagiarism explanation. This method has a long-term positive effect, but it takes a long time to implement because they rely on social cooperation between different universities and departments to reduce plagiarism [6]. Plagiarism detection methods include manual methods and software. They are easy to implement but have a momentary positive effect. Both methods can be combined to reduce cheating and cheating. Although software is the most efficient approach to identifying plagiarism, the final assessment must be done manually [7].

To minimize the practice of plagiarism, detection of writing is required. To overcome the practice of plagiarism, it is not enough to simply remind the students that plagiarism is not well done. The detection of plagiarism practices is the best solution so that the fraudulent actions can be minimized. However, manual detection is difficult to do because of a large amount of writing. So the system needed to detect plagiarism. Methods for detecting plagiarism can be classified into three methods: full-text comparison method, fingerprinting document method and keyword equality method [1].

Rabin-Karp algorithm is a string-matching algorithm that uses hash functions as a comparison between the search string (m) and substring in a text (n). The Rabin-Karp algorithm is based on the fact that if two strings are equal then the hash value must be the same. But there are two problems that arise from this, the first problem is that there are so many different strings, this problem can be solved by assigning multiple strings with the same hash value. The second problem is not necessarily a string that has the same hash value matching to overcome it for each string that is assigned to do string matching by BruteForce [1], [3]

II. RESEARCH METHOD

Similarity measurement methods have been developed with various methods applied. Although each method has its own way of measuring but the results to be achieved remains the same that is to create a system that can measure the level of similarity in the text string in an optimal and effective [1].

There are three kinds of techniques that are built to determine the value of similarity (similarity) of documents, such techniques are [1], [2]:

- Distance-based similarity measure, which measures the similarity of two objects in terms of the geometric distance of the variables enclosed within the two objects. Distance-based similarity methods include Minkowski Distance, Manhattan/City Block Distance, Euclidean Distance, Jaccard Distance, Dice's Coefficient, Cosine Similarity, Levenshtein Distance, Hamming Distance, and Soundex Distance.
- Feature-based similarity measure, which is to calculate the level of similarity by representing the object into the form of features that want to be compared. The feature-

based similarity is widely used in classifying or pattern matching for images and text.

- Probabilistic-based similarity measure, which calculates the level of similarity of two objects by representing two sets of objects that are compared in the form of probability. Includes Leibler Distance Kullback and Posterior Probability

Rabin-Karp algorithm is included in the category from left to right. The Rabin-Karp algorithm implements a hash function that provides a simple method to prevent the time complexity Θ(m2). There are four categories of comparison process [3]:

- From right to left
- From left to right
- In specific order
- In any order

The key to the efficient Rabin Karp algorithm is in its hash value selection. One well-known and effective way is to treat each substring as a number on a specific basis. The hash function should provide at least four properties [4]:

- Able to perform computing efficiently
- High string discrimination
- The hash function (s[i+1...i+m]=s[i...i+m-1]-s[i]+s[i+m]) should be easy to compute from:

a) Hash (s[i...i+m-1])

b) Hash (s[i])

c) Hash (s[i+m])

- The Rabin-Karp algorithm marks the following steps:

a) Apply hash function

b) The preprocess phase in the time complexity Θ(m) and time constant

c) Search phase in time complexity Θ(m)

- Θ(n+m) estimates the active time

III. Proposed System

We use the Rabin-Karp algorithm to compare the pattern of files uploaded with servers on the server. This comparison yields a percentage value of the similarity of uploaded files to files contained on the server. This comparison is performed by preprocessing steps shown in Fig. 1: case folding, tokenizing, filtering and stemming.

A. Case Folding

In this process, we make changes to the words in the document into lowercase (a to z) [4].

B. Tokenizing

We do a cut to the input string based on the specified delimiter. Characters other than letters will be considered as delimiters and will be omitted or deleted for the process of getting text compiler words. From this process will be generated words string or text compilers or often called tokens or term [4].

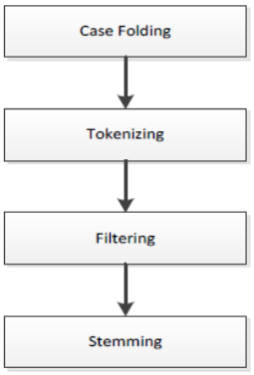

Fig. 1. Preprocessing.

C. Filtering

We remove the words that have been registered into the stop-word or stop-list. Stop-word is the words that often appear in the text in large numbers and is considered to have no significance [3].

D. Stemming

This process we do to get the basic word from a word. Stemming Nazief-Adriani is a stemming algorithm created by Bobby Nazief and Mirna Adriani [8].

E. Rabin-Karp

By seeing that the two strings are the same, the hash value must be the same. But there are two problems that arise from this, the first problem is that there are so many different strings, this problem can be solved by assigning multiple strings with the same hash value [5].

F. Similarity Value Measurement

Measuring similarity and distance between two information entities is a key requirement for the discovery of information. The first stage is dividing the word into k-grams. Second, group the term results from the same k-grams. Then to calculate the similarity of the word set then used the formula 1 Dice's Similarity Coefficient for the word pairs are used [9].

$$S = \frac{(2 \times G)}{(A + B)} \times 100 \quad (1)$$

G. Similarity Value Percentage

To determine the similarity between existing documents 5 types of understanding percentage similarity [5]:

- 0%: the 0% test result means the two documents are completely different in both the content and the sentence as a whole.
- < 15%: Test results less than 15% means the two documents have little in common.
- 15 - 50%: Test result means that the document includes a moderate plagiarism.

- > 50%: Test results over 50% means it can be said that the document detects plagiarism.
- 100%: Test results with a percentage value of 100% indicate that the document is a plagiarism because from the beginning to the end have the exact same content.

IV. RESULT

At the beginning of the application selected one of the detection methods, namely detection by using the title, the content of the content as in Table 1 below:

TABLE I. TEXT OF THE FILE

Text1	Berisi Text 1
Text2	Isi Dari Text 2

The first process, the process of preparation is done the tokenizing process, filtering and stemming process results shown in Table 2 below:

TABLE II. TOKENIZING, FILTERING AND STEMMING RESULTS

Text1	berisitext1
Text2	isitext2

The second process as shown in Table 3 below is a process of parsing K-gram with length K = 4.

TABLE III. RESULTS OF K-GRAM PARSING

No	Parsing Teks 1	Parsing Teks 2
1	beri	isit
2	eris	site
3	risi	itex
4	isit	text
5	site	ext2
6	itex	…
7	text	…
8	ext1	…

Here is a hashing calculation by converting char to decimal based on ASCII with K-gram = 4 and Modulo = 101. The result of this hashing calculation is shown in Table 4.

Pattern = 'beri'

Hashing = 98 * 103 + 101 * 102 + 114 * 101 + 105 * 100 = 109345 mod 101 = 63

Remainder = 109345/101 = 1082.623762 = 109345

And so on.

TABLE IV. CALCULATION RESULTS MODULO AND REMAINDER

Text 1			Text 2		
P	***H***	***R***	***P***	***H***	***R***
beri	63	109345	isit	1	117666
eris	41	113565	site	6	126761
risi	10	125755	itex	65	117730
isit	1	117666	text	55	127416
site	6	126761	ext2	80	114210
itex	65	117730			
text	55	127416			
ext1	79	114209			

The third process shown in Table 5 below is the result of calculating the values found in Table 4 that are matched by matching string by taking the value of match yes.

TABLE V. STRING MATCH RESULTS

Text 1			Text 2			Match
P	***H***	***R***	***P***	***H***	***R***	
itex	65	117730	itex	65	117730	Yes
text	55	127416	text	55	127416	Yes

The fourth process, to obtain similarity level information is weighted using Dice's Similarity Coefficient [10]:

P Similarity = ((4*2)/(8+5))*100%
= (8/13)*100%
= 61.53846154%
= 61.54%

The similarity values obtained from Text 1 and Text 2 are 61.54% and it can be said that the document detects plagiarism. With the time required in comparing text1 and text2 is 0.08 seconds. Testing the system produces the output as shown in Fig. 2 and 3 below:

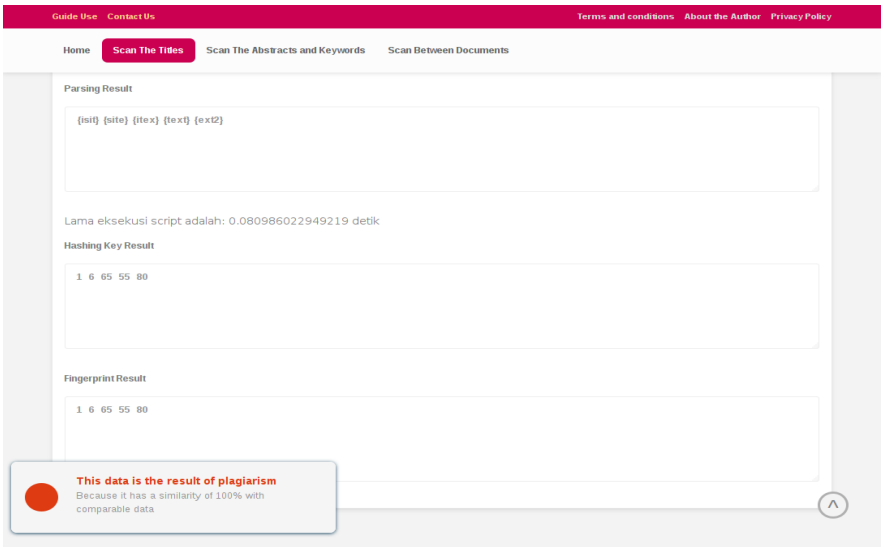

Fig. 2. Results of parsing, hashing key and fingerprint against PDF docs on our system.

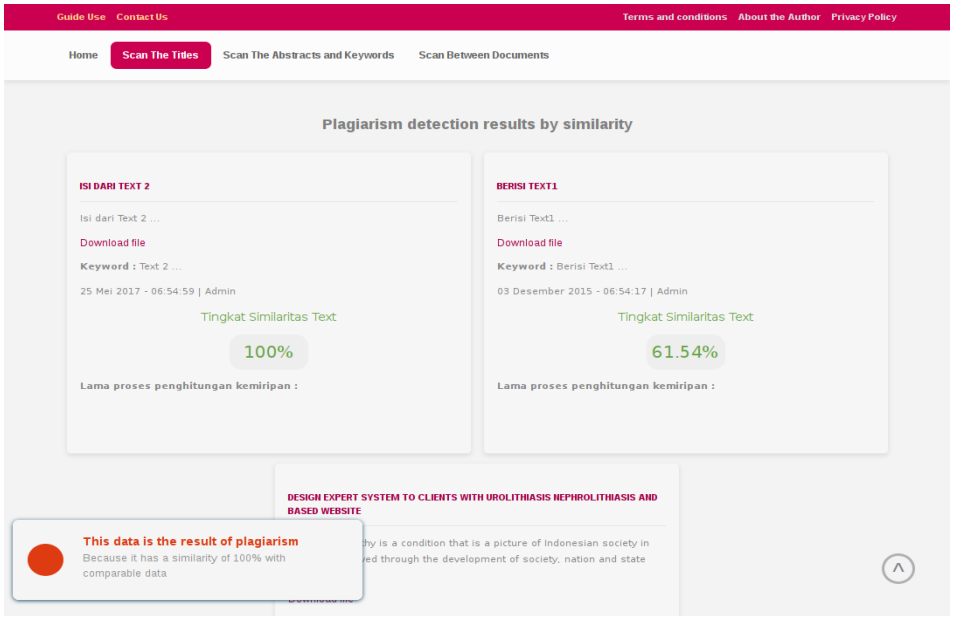

Fig. 3. Notice of plagiarism with color and result percentage of similarity in our system.

V. CONCLUSION

Based on the series of tests we have done, our system can provide a true value of scientific paper data by using k-gram and hashing parsing to find matches of the same word or phrase in the document being tested. Rabin-Karp algorithm modification of time processing process similarity (running time) better. The system has been able to check the title of scientific papers, abstractions or documents comparable with the existing comparative documents on the database with accurate. The checking system at document similarity level with Rabin-Karb algorithm gives a result of similarity percentage and detection notification.

REFERENCES

[1] M Thohir Yassin.; , "Aplikasi Pendeteksian Plagiat Pada Karya Ilmiah Menggunakan Algoritma Rabin-Karp", Research Report Faculty and Behavior Development, Gorontalo State University, Nov. 2012

[2] Zaka, Bilal.; "Theory and Applications of Similarity Detection Technique", Dissertation. Institute for Information Systems and Computer Media (IISCM), Graz University of Technology Austria, 2009

[3] Sahriar, Hamza.; M. Sarosa.; Purnomo Budi Santoso.; "Sistem Koreksi Soal Essay Otomatis dengan Menggunakan Metode Rabin-Karp", Jurnal EECCIS Vol. 7, No. 2. 2013

[4] Firdaus, Bagus.; "Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin-Karp", Makalah If2251 Strategi Algoritmik. 2008

[5] Mutiara, Benny.; A, Agustina.; Sinta.; "Anti Plagiarism Application with Algorithm Karp-Rabin" at Thesis in Gunadharma University, Gunadharma University, 2008

[6] Lukashenko R.; Graudina V.; Grundespenkis J.; , "Computer-based plagiarism detection methods and tools" : an overview [C]. In: Proceedings of the International Conference on Computer Systems and Technologies, pp. 14-15, 2007

[7] Gruner G.; Naven S, "Tool support for plagiarism detection in text documents" Proceedings of the ACM symposium on Applied Computing",pp. 13-17, 2005

[8] Pramudita. Penerapan Algoritma Stemming Nazief & Adriani dan Similarity pada Penerimaan Judul Thesis, Jurnal DASI, 2014, Vol. 14, No. 4.

[9] Kosinov, Serhiy. Evaluation of n-grams conflation approach in text based information retrieval. Unpublished journal. Computing Science Department, University of Alberta, Canada, 2001.

[10] Rizqi Bayu Aji P, ZK. Abdurrahman Baizal, Yaunar Firdaus. Automatic Essay Grading System Menggunakan Metode Latent Semantic Analysis, Seminar Nasional Aplikasi Teknologi Informasi (SNATI), 2011.