INFORMATION TECHNOLOGY JOURNAL

Information Technology Journal 13 (14): 2309-2317, 2014
ISSN 1812-5638 / DOI: 10.3923/itj.2014.2309.2317
© 2014 Asian Network for Scientific Information

A Fingerprint of Paragraph Generation Approach for Detecting Similar Document

[1,2]Haitao Wang, [1]Shufeng Liu and [2]Zongpu Jia
[1]College of Computer Science and Technology, Jilin University, Chang Chun, China
[2]College of Computer Science and Technology, Henan Polytechnic University, Henan, China

Abstract: This study proposes a kind of fingerprint of paragraph generation approach for similar documents detection which mainly includes three parts, select long sentence, obtain feature set of paragraph by calculate weight value of sentence and generate the fingerprint of paragraph. As a result, a document is turned into a set of paragraph fingerprint and a similarity measure is then applied, the similarity degree is calculated between the input document and each document in the given collection. A similarity function based on consine function is employed to determine whether a document is a near duplicate to the input document by similarity degree between them or not. This study presented can better reveal the characteristics of the document and reduce the noisy influence on feature selection effectively. The experiment demonstrates that the method which offered has the preferable precision and recall ratio, relatively less running time comparing to the other method.

Key words: Similarity degree, detecting, weight value, fingerprint, paragraph

INTRODUCTION

With the development of network, the information which the human face exposes the explosive growth. Although the convenience feature of Internet enable the copies and transmission of information to be quick and cheap, especially hot topic news and blogs, nevertheless, it's an indisputable fact that there are a large amount of near duplicate or similar web pages which is produced behind the convenience feature. According to the reporter of CNNIC (China Internet Network Information Center) which declared in 2009, the statistical data indicate that the similarity rate between web pages reaches up to 40% to some extent. These near duplicate files not only consume numerous storage resources which are unnecessary in fact and increase the cost of constructing index but also greatly raise the expenditure and burden of acquiring the useful information from results which search engineer obtained because of a large number of duplicate content. Therefore, detecting the near duplicate issue (De Carvalho *et al*., 2012) on massive files receives sustained concern in literature and is becoming an import research hot topic in informative field.

Utilizing the similar document detecting approach on one hand, the use rate of storage space can be optimized and the same file or data block can be eliminated which distributed in storage system, on the other hand, the data volume can be reduce which transmitted among the network and then, lower the energy consumption and net expenditure, save the net band width for the other data transmission.

A system for detection of near duplicate document faces a number of challenges. First and foremost is the issue of scale: Search engines index billions of file, this amounts to a multi-terabyte database, second: Decision to mark a newly file as a similar of an existing file should be quickly and simple, finally, the system should take advantage of new mechanism as possible, e.g. Map reduce computing model.

The focus of this study, after studying the related work of file detecting, is to put forward a novel paragraph fingerprint generation approach based on weight factor of long sentence which is used to detect similarity degree for the large scale document and drastically lessen the storage room, greatly improve the usage rate of resource. This study is organized as following: The next section presents related work and our own focus, the third section details the paragraph fingerprint generation algorithm and the detecting course of similarity, the fourth is dedicated to analyze the experiment result and choose the appropriate parameter in algorithm, the final part provides our conclusion and points out the further research direction.

RELATED WORK

Duplicate data detection techniques via., (Hajishirzi *et al*., 2010) are the basis of data de-duplication. These techniques divide the files (usually, the old version and the new version of a same file) into a number of parts, compare corresponding parts between files via., hash techniques and find out redundant data. So,

Corresponding Author: Shufeng Liu, College of Computer Science and Technology, Jilin University, Chang Chun, China

Broder *et al*. (1997) proposed a shingling algorithm, coined DSC, as a method to detect near duplicates by computing a sketch of the document. A subset of shingless or n-grams, from each document is chosen as its sketch and similarity between two documents is computed based on common Jaccard overlap measure between these document sketches. To reduce the complexity of shingling for processing large collections, the use of "super shingles" (DSC-SS) was later proposed by Broder in DSS-SS makes use of meta-sketches, i.e., sketches of sketche, with just a minor decrease in result precision. Hod and Zoble investigate a variety of approaches for filtering good shingles, while Büttcher and Clark focus on information theoretic measures such Kullback-Leibler divergence in the more general context of search. Recently, Henzinger (2006) combined two algorithms for detecting near duplicate web pages, namely Broder's DSC and Charikar's random projection algorithm. Henzinger improved on precision compared to using the constituent algorithms individually. Moreover, sophisticated clustering schemes (Huang *et al*., 2008; Narayana *et al*., 2009; Kim and Lee, 2012) allow for incorporating additional knowledge in the form of explicit constraints to the clustering process. Incorporating information about document attributes or the content structure for near duplicate clustering has also been suggested.

Another scheme for detecting similar documents is fingerprinting based on work by Manber (1994) and subsequent work by Brin *et al*. (1995), (Qiu and Zeng, 2010; Sood and Loguinov, 2011). A document fingerprint is typically a collection of integers that represent some key content in the document, where in hashes of words or entire sentences are concatenated to some bit string to generate a characteristic fingerprint of the document. These fingerprinting schemes vary in the specific hash function used, as well as in the choice of strings used for hashing. Hash value based schemes like pick strings whose hash values are multiples of an integer. Position-based schemes, on the other hand, select strings based on their offset in a document. Wang and Chang (2009), Zhao *et al*. (2011), Xiao *et al*. (2011) and Valles and Rosso (2011). Choose word strings with high Inverse Document Frequency (IDF) using collection statistics. Their I-Math algorithm makes use of external collection statistics and improved on recall by introducing multiple fingerprints (based on different lexica) per document.

Locality Sensitive Hashing(LSH), proposed by (Theobald *et al*., 2008; Manku *et al*., 2007; Wang *et al*., 2000), is an approximate similarity search technique that scales to both large and high-dimensional data sets. LSH can be turned by concatenating k signatures from each data object into a single hash value for high precision and by combining matches over l such hashing steps- using independent hash functions for good recall. Min-hashing investigated has the interesting property that the probability of a match (i.e., a hash collision) between two data objects exactly confirms to the Jaccard similarity. More self-turning, iterative LSH approaches like LSH-Tree or Hamming-LSH, however, typically increase the number of signature extraction and hashing steps involved. While behaving significantly more robust than a single LSH step, the increasing amount of hashing may become the actual delimiting factor for runtime performance.

Among the techniques what discussed above, shingling suits processing large collections of documents, however, when dealing with the similar web pages which include the noise content, the phenomenon of missing detection often take place, while the technique based on feature code and string depends excessively on the corpus and connect with the special domain strongly, so lacks semantic feature which can excavate documents content deeply. Although, Simhash technique solves the issue of detecting large scale web page similarity (Pang *et al*., 2010; Huang *et al*., 2010), there are exists defects which can't be ignored, e.g., single fingerprint, a certain amount of data information loss. The method which Literature proposed utilizes long sentence as the feature and calculates similarity degree, even if owning the advantage of full feature information, however, because of complex computing task, it's not wise choose to performance near-duplication detection for massive data set (Zhu and Wang, 2006; Li and Bao, 2007; Li *et al*., 201a).

Considering on the above advantage and shortage of techniques (Yong and Zheng, 2009), the contribution of this study is mainly listed as follows: (1) Present a formula which calculates the weight of position, length and keyword of sentence and adequately extract and excavate the semantic feature of document, (2) Offer a generation algorithm for paragraph fingerprint and address the problem that fingerprint can't be on behalf of feature of document fully and (3) The algorithm which the study presents well fits near duplication detection for massive files because adopting dimensionality reduction technique. In addition, experimental results show that proposed formula and algorithm is reasonable and feasible.

DETECTING SIMILARITY OF WEIGHTING LONG SENTENCE

Definition 1: Feature item set, the fundamental linguistic unit set emerges in document-D can indicate the content of document which is expressed by D $(T_1,T_2,\ldots,T_n)$, where $T_k(1 \le k \le n)$ is the feature item which consists of words and phrase. For example, there are four feature items (a, b, c, d) in a document, which means that this document is expressed by D(a,b,c,d) .

Definition 2: Feature item weight, assume that a text includes n feature item and every feature item is given by a certain value which called weight and indicated the degree of importance for feature item. For example, a document $D = D(T_1,W_1; T_2,W_2;\ldots,T_n,W_n)$ which is marked as $D = D(W_1,W_2,\ldots,W_n)$, where $W_k(1 \le k \le n)$ is the weight of T_k.

Long sentence of paragraph selecting: When comparing with two documents, the basic unit which will be detected calls the chunk of text. The coarsest granularity chunk is to regard the whole document as the one, while this method just detect similarity degree between texts which are the completely same duplicate and can't detect similarity degree between texts which are modified, inserted or copied any part of document. The finest granularity chunk is to regard a character as the one, while it's unfeasible to adopt this scheme because any text which consists of character of alphabet which means all documents derive from the duplicate. Therefore, it's important to select an appropriate length of sentence which the value of chunk range should be arranged between the finest granularity and the coarsest granularity. At present, the most detection algorithms for similar document adopt the feature extraction from the whole document which causes that feature distribution can't reflect the architecture of paragraph well. So, to settle the above defects, it's important to adopt every paragraph of file as a unit, then, choose the long sentence of paragraph as feature item.

In the course of choosing the long sentence of paragraph, the length of sentence (k) indicates the number of words which is contained in the sentence, so, the long sentence is the collection of sentence that the value of length is less than or equals k. The partition token is often the semicolon, the question mark, exclamatory mark or full stop. Using this approach to define and choose has obvious merits, on the one hand, the long sentence includes the more content of paragraph and has the strong discrimination degree between paragraphs, e.g., the granularity choosing is appropriate exactly, the on the other hand, extracting long sentence relies on punctuation mark and its course is simple and convenient.

To ensure the efficiency in the course of extracting long sentence, the threshold value is set by k, if the number of long sentence in paragraph is more than k, choose k long sentences which are the longest, if the number is less than k, keep all of the long sentences, if the long sentence which length is k doesn't exist, choose the k sentences which are the longest among the whole document.

Feature obtaining: According to the relevant statistic data in lecture, the most subject of article appears in the beginning or end of paragraph which starts or summarizes the main idea of article. So, the length and location of sentence all make impact on the semantic of document, the corresponding weight value need to compute, respectively. There are three cases corresponded to weight value calculation as follows:

- **Subject keyword:** The theme or the subject keyword can summarize the semantic of document preciously, even if the modification in article takes place by someone, the keyword also generalizes the content of document exactly and express the theme of article truly. Therefore, handle the subject and keyword and discard the stop words, then obtain the subject keyword collection. $s_w = \{kw_1, kw_2, \ldots kw_n\}$, where n denotes the number of keywords

Given the sentence $S_{i,}$ the $W_{kw}(si)$ stands for the weight value of keyword for sentence, $W_{kw}(s_i)$ is calculated by the following Eq. 1:

$$W_{kw}(s_i) = \frac{\sum_{j=1}^{n} F_{i,j} \times |kw_j|}{|s_i|} \tag{1}$$

where, |*| denotes the length of keyword, $F_{i,j}$ represents the emergence frequency of keyword kw_j in sentence s_i. From the expression of Eq. 1, the weight value of subject keyword is directly proportional to the number and emergence frequency of keyword, while is inversely proportional to the length of keyword. The more number of keyword and emergence ratio is, the more convenient the similarity degree detection is:

- **Sentence length:** The longer the sentence is, the more amount of information contained is and then it deemed that the paragraph has the more representativeness and distinctiveness. For a sentence s_i, Equation $W_{len}(s_i)$ stands for the weight value sentence length, the calculation Equation is listed as follows. $W_{ln}(s_i) = \lg |s_i|$, where |*| denotes the length of string
- **Sentence position:** According to the grammar usage and feature, the beginning and ending of paragraph is often the generalization and summarization sentence, therefore, these sentence has the more representativeness for content of document

For a special sentence, Equation W $_{po}(s_i)$ represents the weight value of sentence position which the calculation is listed as follows:

$$W_{po}(s_i) = \begin{cases} 0.5 & \text{sentence locates at beginning or ending} \\ 0 & \text{other} \end{cases}$$

So, for a sentence which needs processes, the final weight value should be the sum of every weight value which listed above, the calculation Equation is list as following:

$$W(s_i) = W_{kw}(s_i) + W_{ln}(s_i) + W_{po}(s_i)$$

According to the above approach detailed, every long sentence of paragraph is calculated by means of the Equation 1, subsequently the set of long sentence for paragraph s_k is obtained by the weighting and listed as follows:

$$S_k = \{<s_1, W(s_1)>, <s_2, W(s_2)>, \cdots, <s_t, W(s_t)>\}$$

when s_k denotes the feature set for a paragraph, there exist two advantages of this method, one is that the feature long sentence contains the information of semantic and grammar for a paragraph, the other is that each of feature sentence has a weighting value which donates the importance degree of feature sentence.

Algorithm for generating paragraph fingerprint: Although, feature obtaining proposes a reasonable approach which obtains the feature units of paragraphs, the comparison processing between the feature units need huge computing and storage resource on the massive data set. To improve the efficiency, this study puts forward a fingerprint algorithm for the long sentence of paragraph which is list as follows in algorithm 1.

Algorithm 1 function: Generation fingerprint of paragraph
Input: Units for the long sentences of paragraph S_k
Output: n bits fingerprint for the paragraph

- Initialize n-dimension vector $\vec{v}$ equals zero, a n-bits binary number f is zero
- Generate n-bits hash value-h_j (s_i) by means of hash algorithm for the feature long sentence s_i of units-S_k.
- Set the j-th dimension value as v_j for the n-dimension vector $\vec{v}$, from 1 to n, calculate the v_j weighting value by the following Equation:

$$v_j = v_j + W(s_i) * g(h_j(s_i))$$

where, $W(s_i)$ adopts the calculation course of Eq. 1, $g(h_j(S_i))$ is expressed as follows:

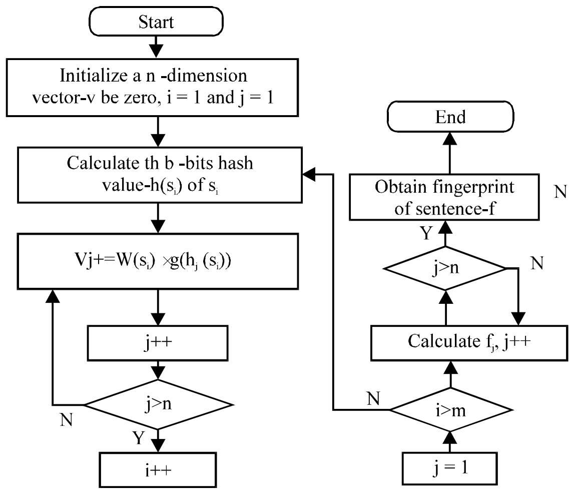
$$g(h_j(s_i)) = \begin{cases} 1 & \text{if } h_j(s_i)=1 \\ -1 & \text{if } h_j(s_i)=0 \end{cases}$$

- Repeat the above course for each element s_i of units s_k, if completing calculation, go to step 5
- Define f_j which identifies the j-th position of binary number, from 1 to n,
- Obtain the f value which is the fingerprint of paragraph

In the course of algorithm, the s_i hash value is calculated by Rabin fingerprint algorithm, the reason is that Rabin algorithm is an ideal choice when the requirement of efficiency is primary and urgent on the large scale data set, even if the safe property of Rabin is lower than the one of MD5.

If there are n feature sentences which are included among a paragraph, according the above algorithm, the course of fingerprint generation is listed in Fig. 1.

Similarity degree calculating: If the similarity degree value of two paragraphs is within a certain threshold one, it is deemed that two paragraphs are similar. For two files, e.g., M and N, the $f_{M,i}$ which is predefined stands for the i-th sentence fingerprint of M, the $f_{N,j}$ which is predefined for the j-th sentence fingerprint of N, the r stands for the threshold value which is set. If the values above three numbers meet the Equation, it's deemed that it's similar between the file M and N:



Start
Initialize a n -dimension vector-v be zero, i = 1 and j = 1
Calculate th b -bits hash value-h(s_i) of s_i
Vj+=W(s_i) ×g(h_j (s_i))
j++
j>n
N
Y
i++
End
Obtain fingerprint of sentence-f
N
Y
j>n
N
Calculate f_j, j++
N
i>m
j = 1

Fig. 1: Algorithm for generating fingerprint of sentence

$$S(M,N) = \frac{\sum_{i,j=1}^{n}(f_{M,i} \times f_{N,j})}{\sqrt{\sum_{i=1}^{n}(f_{M,i})^2} \times \sqrt{\sum_{j=1}^{n}(f_{N,j})^2}}$$

From the above Equation, we can see that the more different are the file M and N when the value of S(M, N) is lower, the identical are the file M and N when the value of S(M, N) is 1. Aiming to the massive data set, utilizing the above Equation calculates the similarity degree value of every two files among data set, respectively, which reaches the goal of similar detection for the massive set.

EXPERIMENTAL RESULT AND ANALYSIS

This section focus on the different parameters value affecting the algorithm of precision/recall rate, execute time, these parameters includes the sentence length-k, the number of the long sentence of paragraph-m, the bits of paragraph fingerprints-b and the similarity degree threshold value-r, where the precision/recall rate is defined as follows:

- Precision rate = Number of correct positive predications/Number of positive example
- Recall rate = Number of correct positive predications /Number of positive prediction

Experimental circumstance: This section experiment bases on the high performance computing platform of Henan polytechnic which consists of some DAWN blade computers, the hardware configuration of a single node includes Intel Xeon E5504 2.00GHz CPU, 8 G memory, 120 G hard disk, the software configuration is RedHat Linux, J2SE 6.0 etc. The experimental data adopts the Sougou news data sets which collects the related news data of 18 channels Sougou CS in 2010 and provides URL and main body information etc. (Li *et al*., 2010b; http://www.sogou.com/labs/dl/cs.html). Among these data sets, select 3000 files randomly, where the number of similar files reaches 850 approximately.

Value of k, m influence on precision/recall rate and times: Firstly, we focus on the different value of k, m influencing on precision/recall rate. When increasing the value of k, the number of long sentence of paragraph for a document which obtained will decrease, therefore the probability value which the number of long sentence of paragraph surpasses m will fall. So, select k value as 9, 11, 13, 15, respectively and m value as 6, 9, respectively, set b value as 32, r value as 0.8. On the circumstance of above situation, the result of precision/recall rate which obtained is shown in Fig. 2 and execution time is list in Fig. 3 as follows.

From Fig. 2 and 3, we can see that the recall rate is high when the k value is 9 which means it's a common phenomenon that the word number exceeds 9 for a general file, therefore it's a large number of feature vector for the paragraph fingerprint and it's longer for the execution time. When the value of k is 15, the precision rate increases but recall rate drops. This fact attributes to that there are no long sentences that the No. of word exceeds 15 in paragraph and no fingerprint of sentence are generated. When the value of k is 11 or 13, it's an ideal situation that the precision/recall rates of algorithm are

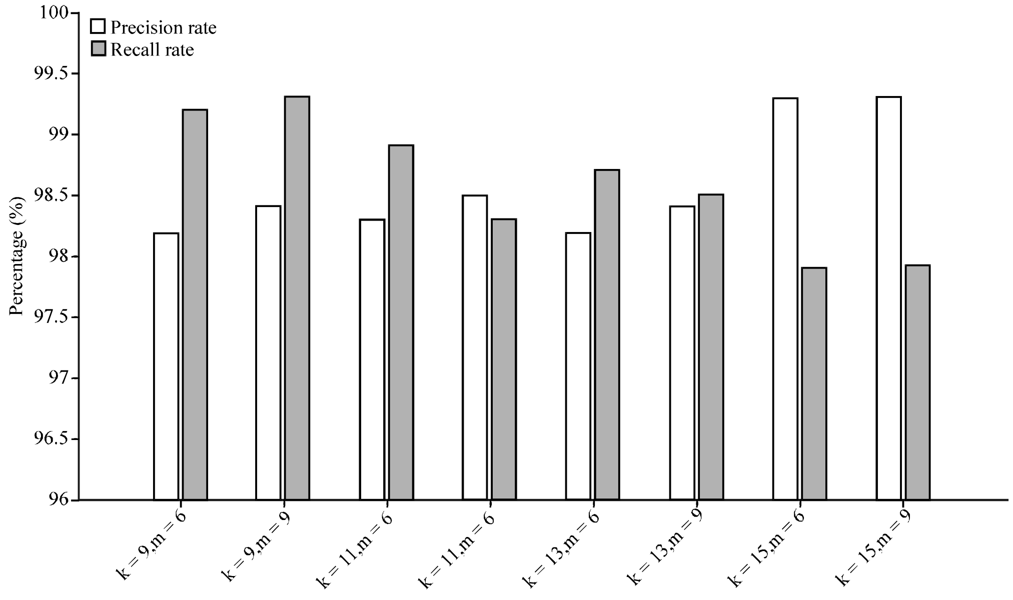


Fig. 2: Value of k and m influence on precision/recall rate

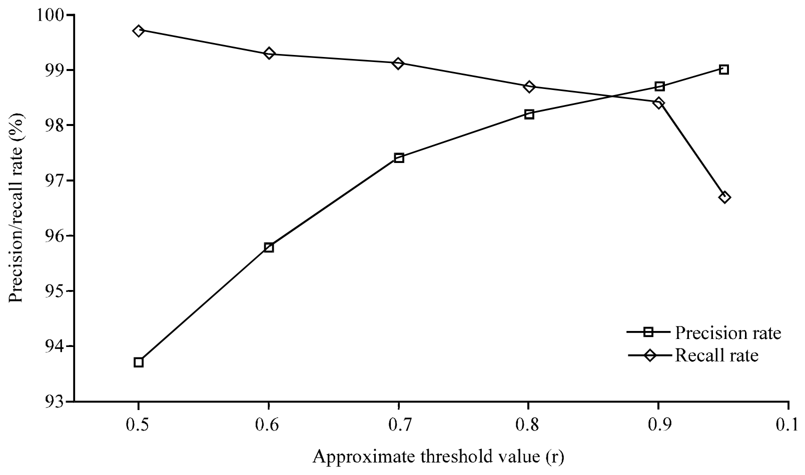


Fig. 3: Value of r influence on precision/recall rate

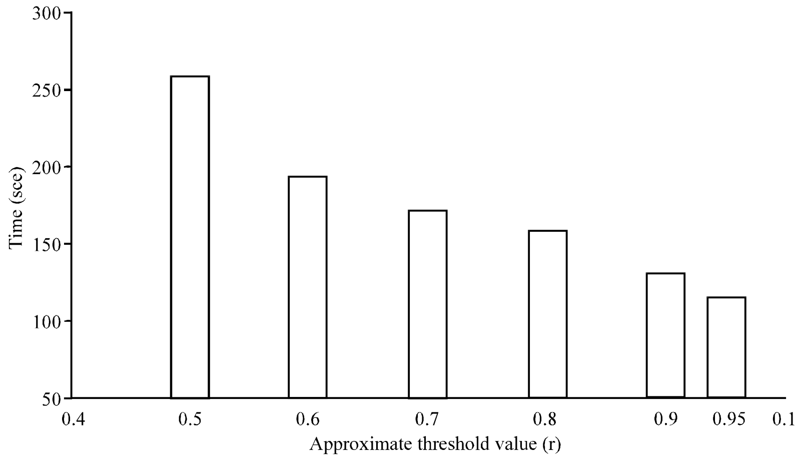


Fig. 4: Value of r influence on execution times

high relatively and execution time isn't high. On the whole, the value of k and m slightly affects precision/recall rates of algorithm.

Value of r influence on precision/recall rate and times: From value of k, m inflaence on precision/recall rate and time discuss, the algorithm can obtain the relatively high precision/recall rate when value of k is 11 and m is 9. This section focus on that the different value of r affects the precision/recall rate and execution time under the circumstance of fixed value of k and m which is 9 and 11, respectively. The result is shown in Fig. 4 and 5 as follows.

From the Fig. 4 and 5 it can be seen that it is an ideal situation for value of k which is 0.8 to obtain the relative high precision/rate and lower execution for the algorithm.

Comparison with other algorithm: This section focus on the parallel comparison with other algorithm which selects the Shingling algorithm, Simhash algorithm, long sentence of full file algorithm and long sentence of paragraph which is proposed by author. Among Shingling algorithm, utilize the string which is composed of the sequential six words, generate fingerprint by Rabin algorithm and extract fingerprint by means of random hash function, calculate similarity degree by Jaccard coefficient and set the threshold value as 0.7. Among Simhash algorithm, use 64 bits for storing fingerprints which generated, quantize the similarity degree by Hanmming distance which value is 3. Among the long sentence of full file, select the 18 sentences which length exceeds 9 words as the feature item, set threshold value 0.6. Among the long sentence of paragraph, choose the sentence length k which value is

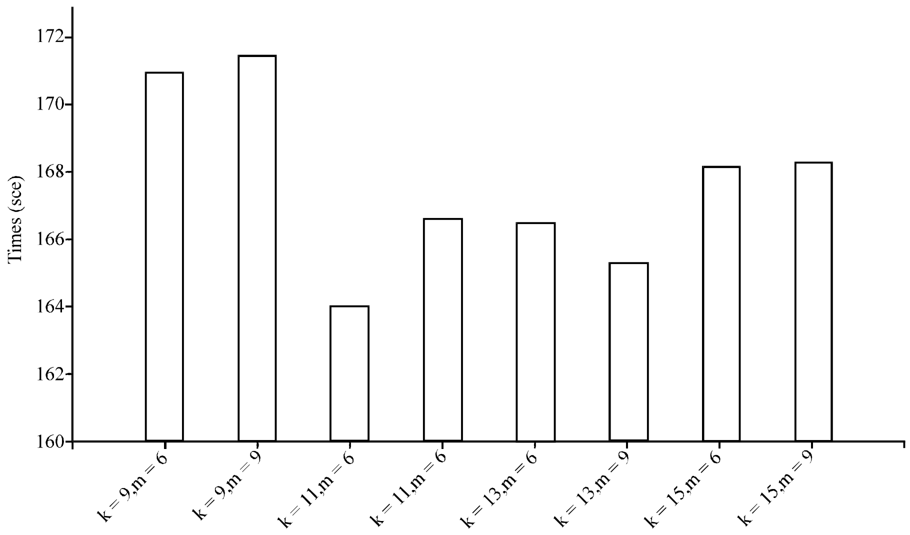


Fig. 5: Value of k and m influence on execution times

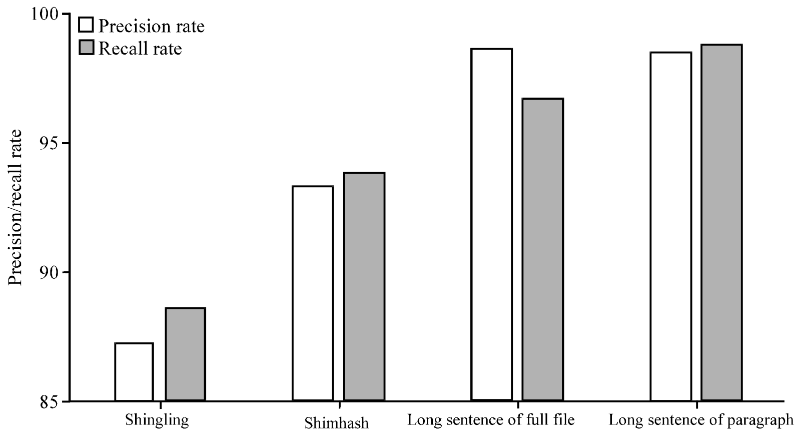


Fig. 6: Comparison with other algorithm in precision/recall rate

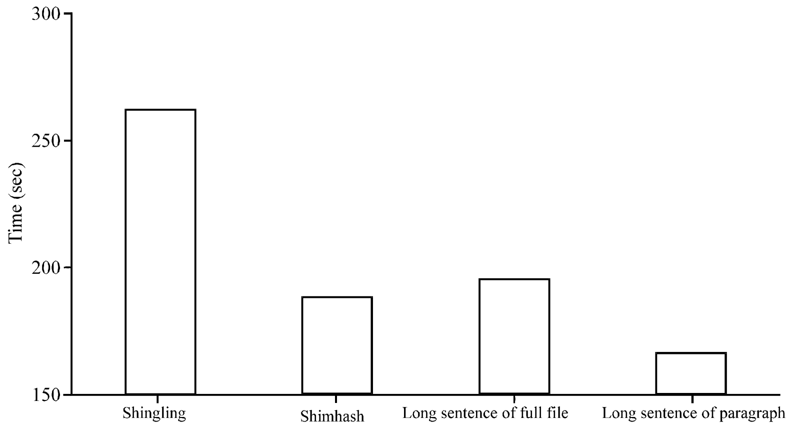


Fig. 7: Comparison with other algorithm in execution times

11, the No. of the long sentence of paragraph m which value is 9, the threshold value which is 0.8, respectively.

Perform the above four algorithm on experimental data set, the result is shown in Fig. 6 and 7.

In Fig. 6, it can be seen that the precision rate of long sentence of paragraph algorithm surpasses other three algorithms by 13, 5 and 0.4% , respectively, the recall rate by 11.6, 5.4 and 3.3%, respectively. The reason of poor precision/recall rate for Shingling lies on the fact that the frequent using words is selected as feature item for file, subsequently the misjudgment often take place. However, the precision rate of long sentence of paragraph which this study offered is lower than the one of long sentence of full file. The reason is that the later obtain this merit by means of long running times. Comparing the method which the author presented with the Simhash algorithm, there is no sharp increment phenomenon which reason is that Simhash adopts 64 bits to generate fingerprint.

With respect to the execution time, comparing with other three algorithms, it decreases by 38.5, 16.1 and 18.9%, respectively which reason is that sentence fingerprint is generated by hash function, therefore, for the massive file, this method not only save the storage space but also reduce performance time greatly.

CONCLUSION

This study proposed a paragraph fingerprint generation algorithm for detecting similar files which fully takes into account the emergence frequency of subject keyword, the length and position of long sentence. Considering these important factors, the fingerprint generation algorithm can prevent the noisy information from influence on precision/recall rate and reduce the execution efficiently. Comparing with other algorithm on experimental dataset, the feasibility and efficiency of algorithm is demonstrated, at the same time, experiments also proves that it's vital for the parameter k, m and r in algorithm to choose a rational value in the course of detecting similarity degree. Of course, the algorithm mainly focus on the text data set, how to mine the feature vectors of various data type, detect the duplicate data quickly and accurately with the low overhead of storage space is the advance direction for author.

ACKNOWLEDGMENT

This research is supported by the National Nature Science Foundation of China (No. 61300216), the National Innovative Approach Special Project (No. 2012IM010200), the Science and Technology Research Projects of Henan Province (No. 132102210123), the Open Laboratory Project of Mine Information Key discipline of Henan University, the Doctor foundation of HPU (B2009-21).

REFERENCES

Brin, S., J. Davis and H. Garcia-Molina, 1995. Copy detection mechanisms for digital documents. ACM SIGMOD Rec., 24: 398-409.

Broder, A.Z., S.C. Glassman, M.S. Manasse and G. Zweig, 1997. Syntactic clustering of the web. Comput. Networks Syst., 29: 1157-1166.

De Carvalho, M.G., A.H.F. Laender, M.A. Goncalves and A.S. da Silva, 2012. A genetic programming approach to record deduplication. IEEE Trans. Knowl. Data Eng., 24: 399-412.

Hajishirzi, H., W.T. Yih and A. Kolcz, 2010. Adaptive near-duplicate detection via, similarity learning. Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, July 19-23, 2010, Geneva, Switzerland, pp: 419-426.

Henzinger, M., 2006. Finding near-duplicate web pages: A large-scale evaluation of algorithms. Proceedings of the 29th Annual International ACM SIGR Conference on Research and Development in Information Retrieval, August 6-10, 2006, Seattle, WA., USA., pp: 284-291.

Huang, L., L. Wang and X. Li, 2008. Achieving both high precision and high recall in near-duplicate detection. Proceeding of the 17th ACM Conference on Information and Knowledge Management, October 26-30, 2008, Napa Valley, CA., USA., pp: 62-72.

Huang, R., S. Fen, J.Y. Yang, Y. Liu and M. Ao, 2010. Web page duplicate removal algorithm based on main body and long sentence of text. Comput. Appl. Res., 27: 2489-2497.

Kim, J. and H. Lee, 2012. Efficient exact similarity searches using multiple token orderings. Proceeding of the IEEE 28th International Conference on Data Engineering, April 1-5, 2012, Washington, DC., pp: 882-833.

Li, X.Y. and C.J. Bao, 2007. Near-duplicate recorders detecting Approach in data warehouse. Electron. Sci. Technol. Univ. J., 36: 1273-1275.

Li, R., K. Chao, W. Zheng and J.M. Wang, 2010a. Near-duplicate text detecting based on mapreduce framework. Comput. Res. Dev., 47: 400-406.

Li, A., J. Shu and M.Q. Li, 2010b. Data de-duplication technology. J. Electron., 21: 916-918.

Manber, U., 1994. Finding similar files in a large file system. Proceedings of the USENIX Winter 1994 Technical Conference on USENIX Winter 1994 Technical Conference, January 17-21, 1994, San Francisco, California, pp: 2.

Manku, G.S., A. Jain and A.D. Sarma, 2007. Detecting near-duplicates for web crawling. Proceedings of the International World Wide Web Conference, May 8-12, 2007, Canada, pp: 141-149.

Narayana, V.A., P. Premchand and A. Govardhan, 2009. A novel and efficient approach for near duplicate page detection in web crawling. Proceedings of IEEE International Advance Computing Conference, March 6-7, 2009, Patiala, India, pp: 1492-1496.

Pang, X.W., Z.L. Yao and Y.J. Li, 2010. High efficient detection approach of near-duplicate recorders for massive data. HuaZhong Sci. Technol. Univ. J., 38: 8-10.

Qiu, J. and Q. Zeng, 2010. Detection and optimized disposal of near-duplicate pages. Proceedings of the 2nd International Conference on Future Computer and Communication, Volume 2, May 21-24, 2010, Wuhan, China, pp: V2-604-V2-607.

Sood, S. and D. Loguinov, 2011. Probabilistic near-duplicate detection using simhash. Proceeding of the 20th ACM International on Information and Knowledge Management, October 24-28, 2011, Glasgow, UK., pp: 1117-1126.

Theobald, M., J. Siddharth and A. Paepcke, 2008. SpotSigs: Robust and efficient near duplicate detection in large web collections. Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 20-24, 2008, Singapore, pp: 563-570.

Valles, E. and P. Rosso, 2011. Detection of near-duplicate user generated contents: The SMS spam collection. Proceedings of the 3rd International Workshop on Search and Mining User-Generated Contents, October 24-28, 2011, Glasgow, Scotland, UK, pp: 27-34.

Wang, J.Y., Z.M. Xie and M. Lei, 2000. Research and evaluating on near-duplicate mirror web page detecting algorithm. Electronic J., 28: 130-132.

Wang, J.H. and H.C. Chang, 2009. Exploiting sentence-level features for near-duplicate document detection. Proceeding of the 5th Asia Information Retrieval Symposium on Information Retrieval Technology, October 21-23, 2009, Sapporo, Japan, pp: 205-217.

Xiao, C., W. Wang, X. Lin, J.X. Yu and G. Wang, 2011. Efficient similarity joins for near-duplicate detection. ACM Trans. Database Syst., Vol. 36. 10.1145/2000824.2000825

Yong, F. and J.H. Zheng, 2009. Research on web pages de-duplication approach. Comput. Eng. Appl., 45: 141-143.

Zhao, Z., L. Wang, H. Liu and J. Ye, 2011. On similarity preserving feature selection. IEEE Trans. Knowl. Data Eng., 25: 619-632.

Zhu, H.M. and N.S. Wang, 2006. A improved detection approach for similar and near-duplicate recorder. Control Decision, 21: 805-808.