A Length-variable Feature Code Based Fuzzy Duplicates Elimination Approach for Large Scale Chinese WebPages

Hongzhi Guo[1,2]
[1]Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, P.R. China
Email: hongzhi.guo@gmail.com

Qingcai Chen[1,2], Cong Xin[1], Xiaolong Wang[1,2]
[2]Dept. of Computer Science and Technology, Harbin Institute of Technology, Harbin, P.R. China
Email: qingcai.chen@gmail.com, wangxl@insun.hit.edu.cn

***Abstract*—Most of the existing Chinese webpage duplicate elimination approaches do not focus on noisy and fuzzy duplicates elimination. In this paper, we propose an efficient and noise-tolerant Chinese webpage duplicate elimination approach based on Length-variable Feature Code. First, an Independent Extraction Unit is defined to eliminate the impact of short paragraphs on feature code extraction. Then the concept of repeatability is introduced by using the longest common substring to enhance the noise tolerant capability. Experimental results on 10 million webpage dataset show that the proposed approach can efficiently deal with duplicates from massive WebPages with the duplicate elimination precision of 99.03%.**

***Index Terms*—duplicates, near-duplicates, Independent Extracting Unit, Length-variable Feature Code, repeatability**

I. INTRODUCTION

Due to frequent reprinting between different websites, duplicate WebPages are quite normal in the returned results from search engines. Duplicates in search engines mainly fall into two camps. One is that different returned pages share the same contents and display styles; the other is that the display styles are different, but most of their contents are the same [1]. Since there are plenty of duplicates and near-duplicates in the internet, users have to check the same information for many times. Lots of time and energy are wasted. Meanwhile a lot of network bandwidth and storage of search engines are taken up and wasted. More importantly, the really needed information might be buried in these duplicate or near-duplicate results. Duplicate WebPages not only reduce the searching efficiency, but also cause a great impact on the precision and the recall of search engines.

Different kinds of approaches have been proposed for duplicates detection and elimination in recent years [2-6]. However most of them do not focus on noisy and fuzzy duplicates elimination. In this paper, an efficient and noise-tolerant Chinese webpage duplicate elimination approach using Length-variable Feature Code is presented. The approach effectively solves the above problems with a high duplicates eliminating precision meantime.

II. RELATED WORK

Duplicates elimination originates from text copy detection [7]. Compared with text copy detection which only compares the text contents, more information can be used in duplicates elimination, including URLs, hyperlinks and Web contents [4]. According to the information used in duplicates detection or elimination, these methods can be divided into three types: URL-based [5, 8], hyperlink-based [9] and content-based. Among these three methods the content-based methods have been the mainstream in this field recently.

Content-based methods make good use of the contents and structures of the WebPages. A number of features are extracted as the representatives of the documents [10], and this process can be regarded as a compression of the documents. Then the similarity between documents is defined and computed, after that duplicates can be easily detected and eliminated. Content-based methods can be divided into two types according to the methods of extracting [7], syntax-based [10-12] and semantic-based [13, 14]. Considering the differences of Chinese documents with English documents, a special word segmentation step is always needed. Therefore most existing duplicates elimination approaches for Chinese WebPages adopt the syntactic-based method for the efficiency consideration.

In [15] a feature code based duplicates removal algorithm was developed. Word strings with a length of L/2 were selected from both sides of the periods, these strings constituted a feature code, and then the feature codes were indexed by B-Trees for detecting duplicates. A similar duplicates elimination algorithm based on feature strings was presented in [16] by Pingbo Wu et al.. Most of the above methods used punctuation marks or similar signatures in the texts to extract features, Chinese characters were extracted and combined as a feature to identify a web page. Word segmentation and feature selection using statistical computing were not needed in

doi:10.4304/jsw.7.11.2622-2629

these methods, and similar strategies to hash techniques were implemented to map a longer string to a short feature code instead.

These methods are fast and more suitable for Chinese WebPages. But if the symbols using for generating the feature codes do not appear or appear rarely, feature extraction and the duplicates elimination results will be badly affected. Overall feature code based methods still have many advantages over the others and are popularly implemented. Besides, there is still another kind of method, which takes the keywords extracted in the indexing as the feature items and uses them to eliminate the duplicates. The approach used in Peking Tianwang search engine [5], which uses the abstracts and keywords for duplicates elimination, serves as a representative. These methods use the words as the granularity for comparisons, the granularity is relatively small, and this brings a high recall and a lower calculation inversely.

Apart from the methods above, Jinyan Chen et al. also proposed a duplicates elimination method using Fourier transform. By Fourier transform of the series each web page was expressed as several Fourier coefficients, and then the similarity between two web pages was calculated based on the Fourier coefficients [17]. In addition some other duplicates elimination methods were also proposed aiming at some special types of web pages. For example, a series of methods which used the released time of web pages to reduce the comparisons were put forward for the news pages [18].

Feature code based duplicate elimination methods are widely used in Chinese field and obtained good results on unchanged reprinting pages. However, these methods are too sensitive to punctuation marks. In practice the reprinting pages are always displayed in different styles, sometimes, additional ads and comments are added, all these factors will affect the extracted feature codes. If some sentences or paragraphs are deleted again, the impact on extracted feature codes will be greater. At the same time, the exact matching of feature codes also severely restricts the noise-tolerance ability of these algorithms.

In this paper, a feature code extraction algorithm based on the importance of paragraphs is proposed aiming at solving the above problems. The importance of each paragraph is determined according the length and location of it. Impacts of the subject paragraphs are highlighted, and better duplicates eliminating results are finally achieved by effectively filtering various kinds of noises.

III. Duplicates Elimination Using Length-Variable Feature Codes

The duplicates elimination algorithm using length-variable feature codes first extracts a short string from each web page. The string, as a representative of the web page, is the feature code. Then the concept of repeatability is defined based on page feature codes. Dlicated web pages are finally detected using the computed repeatability between the pages.

A. Extraction of Feature Codes

According to the duplicates elimination targets in this paper, length-variable feature codes are adopted to represent web pages. A long web page corresponds to a long feature code, and vice versa. In anchor based feature code extraction certain special symbols (common punctuation marks and characters, such as “的”(of), “。”(period), and “?”.) are defined as the anchors, the characters at the left and right of these anchors are connected as a feature code. An illustration is shown below:

Texts: *系统采用的特征码提取算法是基于语法获取特征的方法。这种方法将网页内容看成字符流，以一些标点符号和常用汉字作为锚点，从网页内容中抽取出文字作为网页特征码。*

Feature code: *系法这流以点从码*

After analyzing kinds of purified web pages, we found that there were several problems in existing anchor based feature code extraction: first, the texts disposed in these algorithms were purified documents, in which kinds of ads and un-related information should be removed. However in practice there probably still are some noises left. For example, “[1] [2] [3] [我来说两句]”, “相关专题：法治在线节目实录” etc. These often occurred in the beginning of the texts; second, some of the ads and comments, which were added to the duplicated web pages, might not be removed completely during the purification. This caused that the purified documents weren’t the same. These differences always took place in the beginning and end of the texts; third, the web pages in which most paragraphs had the same contents but for a few short different paragraphs, were also our duplicates elimination targets. To solve these problems above, the concept of **Independent Extraction Unit** is introduced below to get a better feature code.

Definition 1. Suppose that the web page P contains n $(n\geq 1)$ independent natural paragraphs, each paragraph has a respective length of $l_1, l_2,\ldots,l_n$, the ith paragraph is considered to be an **Independent Extraction Unit**, if it satisfies the following conditions:

- $l_i \geq L_{pth},\ L_{pth} \geq 0$;
- $\dfrac{l_i}{\sum_{i=1}^{n} l_i} \geq \delta_p,\ 0 < \delta_p \leq 1.$

Where, L_{pth} and δ_p are the predefined thresholds.

The definition of Independent Extraction Unit mainly focuses on the paragraphs in the web pages. If the length of the text in a paragraph is longer than the threshold L_{pth}, or its proportion of the full texts is more than the threshold δ_p, the paragraph can be defined as an Independent Extraction Unit.

If a paragraph is long enough or it takes a big proportion of the full texts, it is an Independent Extraction Unit. When a paragraph is recognized as an Independent Extraction Unit, the anchor-based feature code extraction method is implemented, and the extracted feature code is served as a part of the whole feature code of the web page, otherwise it will not be attended to. The impact of short texts on the whole feature code is easily shielded. The details are described in algorithm 1 below:

Algorithm 1. **Feature Code Extraction Based on Independent Extraction Unit**

1. Suppose that the web page P contains n $(n\geq 1)$ independent natural paragraphs, $p_1, p_2, \ldots, p_n$.
2. Initialize the feature code of web page P as an empty string C;
3. Calculate the lengths of the paragraphs $p_1, p_2, \ldots, p_n$ in P, and count the Independent Extraction Units in page P, followed by $u_1, u_2, \ldots, u_m$, $1\leq m\leq n$;
4. Loop $1, 2, \ldots, m$, for each Independent Extraction Unit u_i $(1\leq i\leq m)$,
   a) Extract the feature code c_i of u_i using the anchor-based extraction methods;
   b) Append c_i into C;
5. Return the feature code C of page P.

Generally speaking, longer paragraphs are more likely to be the subject paragraphs, so we mainly concentrate on extracting feature codes from long paragraphs. If there is more than one long paragraph, the extracted feature codes are connected to form a whole feature code for the web page. Compared to general feature code extraction methods, which deal with the whole web page without any paragraph information, or look all paragraphs as the same, the approach in this paper emphasizes the importance of long paragraphs, and shields the impact of noises and short paragraphs. This makes our feature code matching have a preliminary fuzzy duplicates elimination capability, and the later experiments verify this. The location of a paragraph is another factor affecting its importance, a paragraph in the middle always has a higher importance than that in the beginning or the end of the text. And the impact is detailed in the definition of Repeatability in Sect. 3.2.

In our experiments we found that there were some paragraphs, which were consisted of many very short paragraphs, and due to the definition of Independent Extraction Unit there wasn't any paragraph having Independent Extraction Units. Besides, there were still a part of web pages which contained few anchor characters. And these caused that feature codes couldn't be extracted or the extracted feature codes were too short, and resulted in a higher collision. To solve the problem that no Independent Extraction Unit exists in the page texts, we propose a method which extracts the beginning characters from the beginning and the end paragraph, and connect them into a feature code.

Algorithm 2. **Feature Code Extraction Algorithm**[1]

Suppose that the web page P contains n $(n\geq 1)$ independent natural paragraphs, $p_1, p_2, \ldots, p_n$, the whole algorithm of extracting feature codes from P is detailed below:

1. For the purified text ***Content*** of page P, if the length of ***Content*** is greater than 1000, the first 1000 characters are intercepted as ***Content'***, otherwise the full content of ***Content*** is set as ***Content'***;
2. Preprocess ***Content'***, remove invisible characters and count the information of paragraphs according to the paragraph symbols in HTML such as " ", " " (or "
", " "), and divide them into $p_1, p_2, \ldots, p_n$. Determine whether $p_1, p_2, \ldots, p_n$ satisfy the condition of Independent Extraction Unit extraction, and record the satisfied ones as $u_1, u_2, \ldots, u_m$ $(1\leq m\leq n)$;
3.
   a) If $m\geq 1$, use **Algorithm 1** to extract the feature code;
   b) Else if $m=0$, $n\geq n_{th}$, there is no Independent Extraction Unit, and the number of the paragraphs is no smaller than the pre-defined threshold n_{th}, the beginning and the end characters in the first N and last N non-overlapping paragraphs are connected as the feature code;
   c) Else if $m=0$, $n<n_{th}$, there is no Independent Extraction Unit, and the number of the paragraphs is smaller than the pre-defined threshold n_{th}, so all paragraphs are connected to one paragraph, and the anchor-based feature code extraction method is implemented;
4. If the length of the obtained feature code in **Step 3** is smaller than a predefined threshold, the feature code is discarded. The first $4N$ characters of the texts are intercepted as the feature code, and return. Otherwise return the feature code directly.

B. Repeatability

Considering that there are a maze of reprinting web pages, kinds of added ads and comments, and different formats used in web pages, it is far from enough only relying on length-variable feature code extraction to achieve fuzzy duplicates elimination. If the content of a page is a subset of another page, we also hope to detect and eliminate it. So the definition of **repeatability** based on the longest common substring is given below hoping to provide supports for imprecise matching between the feature codes.

Definition 2. The **Repeatability** S_{ij} of web page P_j over P_i, is defined as the proportion of the content of page P_j included in that of page P_i. The formula is shown in (1),

$$S_{ij} = \frac{Len(lcs(C_i, C_j))}{Len(C_j)} \tag{1}$$

Where, C_i, C_j are the feature codes of page P_i and P_j respectively, $Len(lcs(C_i, C_j))$ is the length of the longest common substring between C_i and C_j, $Len(C_j)$ denotes the length of C_j.

Suppose that the repeatability threshold of our approach is set as δ, if $S_{ij}\geq\delta$ $(0<\delta\leq 1)$, page P_j is called a duplication of page P_i or page P_j duplicates page P_i, this is signed as $P_j \triangleright P_i$. The contents of web pages are represented as feature codes, and the duplicated contents are counted using the longest common substring between them, if the proportion above exceeds the predefined

[1] The thresholds n_{th}, N, are pre-defined according to our experimental statistics and the references [16-18].

threshold δ, page P_j is determined as a duplicate of page P_i.

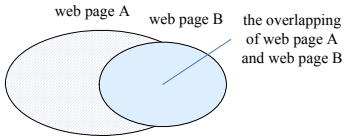


Figure 1. Diagram of duplicate WebPages A and B

Importantly the definition of repeatability is one-way and non-symmetrical, $P_j \rhd P_i \not\Rightarrow P_i \rhd P_j$. This could also be seen in the example in Figure 1, although the overlapping of pages A and B occupies a large part of B, it is less than half of A. This is because that in the computation of S_{AB} and S_{BA}, they have the same molecule but different denominators. In this paper not only the duplicates in common sense, but also the duplicates as shown in Figure 1, where the content of a short page is a subset of a longer page, can be detected using the definition of repeatability. This greatly improves the duplicates eliminating capability of our method, and help search engines save lots of storage. However since a page could be a duplicate of any page which has a longer or equal length, the comparisons increase a lot in detecting the duplicates, and these bring a larger computation.

According to formula (1) the size of the duplicated contents of two pages is measured by the length of the longest common substring between their feature codes. The longest common substring differs from the longest common subsequence (*LCS*), it requires continuous matching without interruption. The shorter the length of the longest common substring is, the smaller the final computed repeatability is. If mismatching happens in the beginning or the end, there are still opportunities to get a long longest common substring if the rest parts match well. Thus, the matching in the middle is more important than that in the beginning or the end of a page. In a word, the paragraphs in the middle have a greater impact than those in other positions. Whether the contents in the middle is the same or not, or how much the content in the middle duplicates the other, largely determines the finial duplicate relationship between two web pages.

C. Duplicates Elimination using the Suffix Tree

The previous two sections have given detailed discussions on our duplicates elimination algorithm. In this section we mainly focus on its implementation. And there are several problems: first, the similarity computation using the longest common substring is relatively time-consuming; second, a page could be a duplicate of any page which has a longer or equal length. In this way the comparisons could not be limited to a length around that of the current page, the range of comparisons might be extremely large. Therefore if we detect duplicates in traditional pairwise comparisons, a huge number of comparisons are needed, and the efficiency would be quite bad.

The generalized suffix tree structure is introduced to solve the above problems, and it is used to store the suffixes of the feature codes. The time complexity using the suffix tree to find the longest common substring of two strings is O(m+n), where n and m are the lengths of the two strings respectively. As a result the efficiency of the program is greatly improved. Each page to be detected only need once to find the longest common substrings between itself and other records, and the comparisons are greatly accelerated, the time complexity is reduced to O(n). Once a page is detected to be a duplicate, it will be dropped; otherwise the suffix of its feature code is inserted into the suffix tree waiting for comparing with other left web pages.

A page could be a duplicate of any page which has a longer or equal length. This means that in a data set, all S_{ji} ($\mathrm{Len}(C_j) \geq \mathrm{Len}(C_i)$, $i,j \in \{1,2, \ldots, m\}$) need to be computed to determine whether a page P_i is unique or not. Before eliminating the duplicates the feature codes are firstly sorted in descending order and then disposed from longer ones to shorter ones. The suffix tree only need to provide the repeatability of the current page over the pages in the tree, and inverse computation is not required since all of the lengths of the page feature codes in the tree are longer than that of the current page. According to the definition of Repeatability, the computation of S_{ji} only requires the feature code length of current page P_i and the lengths of the longest common substrings between P_i and the others. The lengths of the feature codes aren't needed in the suffix tree, the storage is saved and the design of the program is simplified. Meantime the comparisons between a page and any other pages which might be their duplicates are ensured.

D. Information Loss in Fuzzy Duplicates Elimination

Our approach adopts fuzzy methods to eliminate the duplicates. Once a page is determined as a duplicate it will be abandoned subsequently. Each time when a duplicate which isn't exactly the same with the ones in the suffix tree is eliminated, a certain information loss occurs. This part of information loss is inevitable. Citing figure 2 as an example, the web page B is a duplicate of page A and page B is eliminated in our duplicates elimination, the area 3 in figure 2 shows the information loss by eliminating page B.

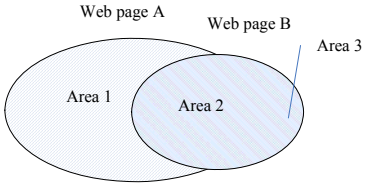


Figure 2. Schematic diagram of information loss

Suppose the information in the page A and B are *info(A)* and *info(B)* respectively, and the information loss of dropping the page B is *info(B)*(1-δ)* at the most (δ is the pre-defined repeatability threshold). The duplication relationship exists only when the proportion of the losing information over the page B is less than *info(B)*(1-δ)*. The losing information can be controlled by adjusting the repeatability threshold δ.

For two web pages with equal feature codes, if a duplicate is detected the latter one will be abandoned. In fact, the information loss is the same no matter which one is abandoned since *info(B)=info(A)*, $S_{AB}=S_{BA}$. However in

our implementation, we choose to discard the latter ones. Why don't we choose the first arrived to be abandoned? Whether are these two choosing equivalent?

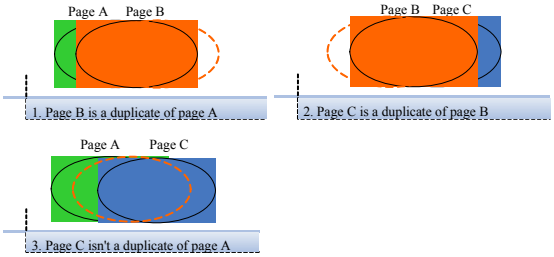


Figure 3. No transmission for duplicate relationship

Figure 3 gives a discussion of these questions. Since the duplication relationship is not transitive, that is $A \triangleright B, B \triangleright C \not\Rightarrow A \triangleright C$, if we want to know whether page A is a duplicate of page B, the repeatability of A must be computed, and we can't derive conclusions from other duplication relationships. The duplicates are detected while being abandoned. When we detect that $P_j \triangleright P_i$, the page P_j is discarded, and most information of P_j are contained in P_i which is retained in the results set. Assuming that we find a page P_k later, $P_k \triangleright P_i, P_i \triangleright P_k$, P_i and P_k have the same repeatability. The information loss of discarding any of them is the same. However since the duplication relationship isn't transitive, although P_i is a duplicate of P_k, it can't be ensured that P_j is the duplicate of P_k too. If P_i is eliminated instead of P_k, the information of P_j might not be kept.

Therefore we choose to retain the first arrived and abandoned the latter when they have the same repeatability. The strategy verifies that we can find another page which contains most part of the information of the abandoned web page in the reservation. The total abandoned information won't exceed a *1-δ* proportion of the full information of abandoned web pages. Irrational information losses are avoided and the total information losses are well controlled.

IV. Evaluation and Discussions

The precision and recall rate are used as the evaluation criteria in duplicates elimination. Unfortunately the duplicate web pages set can't be defined. In duplicates elimination using the symmetric similarity, if the page P_j is similar or duplicated with the page P_i, it is all feasible to retain the former or the latter. So there is no way to determine which one should be the duplicate and be stored into the duplicates set. Either of them could be the duplicate, and this relates to our strategies and processing sequence. In our approach due to the non-symmetry of the repeatability, if $P_j \triangleright P_i$ and $P_i \not\triangleright P_j$ happen, it is determinate that P_i is reserved and P_j is abandoned, that is P_j is one element of the duplicates set. Furthermore if $P_j \triangleright P_i$ and $P_i \triangleright P_j$ occur, we take the strategy that the first arrived is retained (Details on the reasons have been discussed in Sect. 3.4), in this case it relates to the input sequence which one should belong to the duplicates set.

Therefore in our experiments we mainly focus on the precision of detected duplicate pairs. The computation of the precision is shown in formula (2):

$$precision' = \frac{Num(DP_{right})}{Num(DP_{total})} \tag{2}$$

Where DP_{total} and DP_{right} denote the set of the detected duplicate pairs and the right ones of them, $Num(DP_{total})$ and $Num(DP_{right})$ are their numbers respectively.

In addition, the duplicates eliminated rate is computed as a complement to the precision. The computation is just as formula (3) shows:

$$removeRate = \frac{Num(rmW)}{Num(W)} \tag{3}$$

Where $Num(W)$ and $Num(rmW)$ denote the numbers of the web pages set and the set of the eliminated web pages respectively.

To evaluate our duplicates elimination approach using length-variable feature code, we have to compare the results to some ground truth. However there is no computer-processable ground truth in this area, we have to rely on manual evaluation. Experimental data are driven from the real internet environment. Owe to the huge data scale, a number of duplicates pairs (a proportion of *10%*) are randomly selected and manual labeling is carried out. In our experiments, the thresholds in extracting Independent Extraction Unit L_{pth} and δ_p are set as *300* and *0.75* respectively. Details of the thresholds are given in Sect. 4.3.

A. Evaluation on the Noise-Tolerance Ability of the Feature Code

In this part of the experiments a web page A is chosen as the benchmark, a series of operations such as insertion, deletion and modification, are carried out to generate the testing documents A_1', A_2' and A_3'. Feature codes are extracted from A, A_1', A_2' and A_3', the repeatability between the benchmark and the testing documents are computed, and all of them are implemented in evaluating the anti-noise of our feature code extraction algorithm. Web page A[2] is adopted as an exmaple, and its purified document is *A0.txt*. The extracted feature code is "*王示一差站高只事就示对题最机给害严府形象*".

The operations of insertion, deletion and modification are carried out in different positions of the page A. The experimental results cannot be fully listed, the following are the representatives: *(1)*. A_1' delete the hyperlinks in the end; *(2)*. A_2' append a long text to the second paragraph; *(3)*. A_3' replace the second paragraph by another short text. The short text used is shown below:

"中国制造"由于毒奶事件而在全球声誉下挫，在不顾公共道德追逐暴利的逆流之中，幸好还有"神七"成功飞天令世人对"中国制造"重拾信心。

The long text is:

在神七科研攻关中，哈尔滨工业大学承担着宇航员舱外宇航服的地面实验系统，学院气动技术中心课题组负责"水平舱环控系统改造"和"紧急复压系统"的研制。……而在神七发射前 38 分钟，上海的一名

[2] http://news.sina.com.cn/c/2008-09-30/181616385075.shtml

科研人员拔掉地面连接神七的最后一个重要插座，成为最后一位撤离发射架的人。[3]

Experimental results of the extracted feature codes are listed in TABLE I. $S_{A'A}$ is the repeatability of the benchmark over the testing documents and $S_{AA'}$ is the repeatability of the testing documents over the benchmark.

TABLE I.
EXPERIMENTAL RESULTS OF THE FEATURE CODES AFTER INSERTION, DELETION AND MODIFICATION ON THE PAGE

Testing documents		$S_{A'A}$	$S_{AA'}$
File	Feature code		
A_1'	王示一差站高只事就示对题最机给害严府形象	1	1
A_2'	在中哈服地统学责水造和海一七最座成架人	0.05	0.05
A_3'	王示一差站高只事就示对题最机给害严府形象	1	1

According to the experimental results above, the impact of shorter texts on feature code extraction can be well shielded using algorithm 1 in this paper. Inserting a short text or deleting a short text won't change the extracted feature codes, like A_1' and A_3'. But for a long text, if its length satisfies the requirements of Independent Extraction Unit extraction, no matter insertion, deletion, or modification will change the extracted feature code. In A_2', a replaced longer text in the middle changes the final feature code, and the repeatability becomes very small. In practice there are kinds of web pages and noises, the finial duplicates detection all depend on the situations.

B. Accuracy in Duplicate Elimination

Experimental Setting: The data set in this part contains 25,476 web pages, which were crawled from four websites[4]. These web pages are firstly purified and stored as *news_xc.ripes*, which is the input in this section.

The First Group of Experiments: This group of experiments contains duplicates elimination using the page title (the *title* field in the purified documents) and duplicates elimination using exact matching between page contents. And their duplicated results are compared with our method. This group of experiments is consisted of five small experiments:

Exp. I Duplicates elimination using exact matching between page titles;

Exp. II Duplicates elimination using exact matching between MD5s of the page contents;

Exp. III Duplicates elimination using exact matching between MD5s of the purified pages;

Exp. IV Duplicates elimination using exact feature code matching;

Exp. V Duplicates elimination using fuzzy feature code matching, the repeatability threshold is set to 0.75.

The experimental results of the above five experiments are listed in TABLE II.

TABLE II.
DUPLICATE ELIMINATION PERFORMANCE OF THE FIRST GROUP

	Effective records	Nums of duplicates	Duplicates removal rate	Precision	Time (s)
I	24633	1552	6.30%	34.84%	32
II	25476	684	2.68%	100.00%	36
III	25476	747	2.93%	100.00%	43
IV	25476	1410	5.53%	99.29%	36
V	25476	1769	6.94%	99.03%	44

The running time above contains the time of loading the purified documents and writing the results files. As shown in TABLE II Exp. I has the shortest running time and the lowest precision, too much mistakes make the method useless. Exp. II and Exp. IV have the same running time, and the running time of Exp. III and Exp. V are almost the same. However the rates of the duplicates eliminated in Exp. II and Exp. III are too low to fight the noises and cannot reach the requirements of duplicates elimination. Exp. IV and Exp. V all have high rates of duplicates eliminated and high precision, these two have the biggest advantages.

TABLE III.
DUPLICATE ELIMINATION PERFORMANCE OF THE SECOND GROUP

	M1	*M2*	*M3*
Effective records	25476	25226	25476
Nums of Duplicates	1769	2512	1172
Duplicates removal rate	6.94%	9.96%	4.60%
Precision	99.03%	85.37%	94.92%

The Second Group of Experiments: In this group, the duplicates elimination method in this paper is compared with the methods in References [15] and [16]. These methods are signed as *M1*, *M2* and *M3* for convenience's sake. The experimental results of their precision and the duplicates eliminated rates are listed in TABLE III and screened in figure 4.

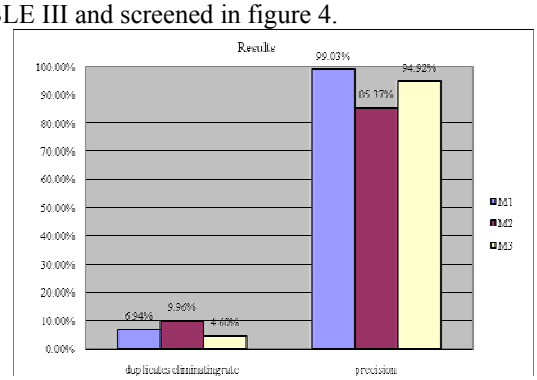


Figure 4. Comparisons of three methods on duplicates deletion rate and precision

To remove the influences of unrelated factors, the time of loading and writing files are removed from the running time listed above. In this group of experiments, *M2* has the highest duplicates eliminated rate (9.96%) and the lowest precision (85.37%); *M3* has a higher precision (94.92%) than *M2*, but its duplicates eliminated rate is unsatisfactory (4.60%); the duplicates eliminated rate of *M1*, the approach proposed in the paper, is situated between the two. The precision (99.03%) is the highest among these three methods, and the duplicates removal rate is satisfactory.

[3] *http://paper.wenweipo.com/2008/09/30/CH0809300012.htm.*
[4]http://news.sina.com.cn/,http://news.tom.com/, http://news.china.com/zh_cn/,http://news.sohu.com/.

(a) Example 1

(b) Example 2

Figure 5. Examples of duplicate web pages

In evaluating the precision, we find that most of the eliminated web pages in *M1* are subsets of the retained web pages, just as shown in figure 5(a). The feature code extraction in *M2* has only one form, and it cannot dispose those texts without a period. In this part of experiments, 250 extracted feature codes in *M2* are empty, but actually these web pages are not very short and contain lots of other punctuation. Furthermore after analyzing some samples we find that the fault duplicates pairs are mainly contained in the following situations: first, the sequence of the processed web pages are not sorted, and some longer web pages are eliminated as the duplicates of short pages; second, a part of the mistakes are similar to the one shown in figure 5(b), the web pages have very similar text and sentence structures, only some words or numbers in the middle of the sentences are different, and most of the characters around the punctuation are the same. So the extracted feature codes are nearly the same and fault duplicates happen. Compared to our approach (*M1)*, *M2* have more mistakes of this type, for their feature code extraction has only one form.

M3 has a high precision and a disappointed duplicates removal rate. This is because there are too many restrictions in reducing the comparison scope. It has little fuzzy duplicates elimination ability. A part of the fault duplicates in *M3* is caused by short texts, and the information in the extracted feature codes of them is too little to represent the web pages. Another part is also similar to the one shown in figure 5(b). Feature code based duplicates elimination methods cannot deal with these types of web pages, and specific ways should be designed.

C. Thresholds Setting in Duplicate Elimination

The repeatability threshold is the significant factor which affects the duplicates removal precision and the duplicates removal rate. Experiments in this part mainly focus on its influences in different values. *News_xc.ripes* with *25,476* records is still implemented as the data set, and seven tests are carried out in different repeatability threshold values. The experimental results are listed in TABLE IV.

TABLE IV.
RESULTS ON DIFFERENT REPEATABILITY THRESHOLDS

Thresholds of the repeatability	Nums of duplicates	Duplicates removal rate	Duplicates removal precision
0.70	1810	7.03%	98.78%
0.75	1769	6.94%	99.03%
0.80	1709	6.70%	99.06%
0.85	1659	6.51%	99.10%
0.90	1557	6.11%	99.17%
0.95	1452	5.70%	99.24%
1.00	1410	5.53%	99.29%

According to the experimental results above, the number of duplicates falls with the increasing of the repeatability thresholds, and the precision continues to rise. Therefore we can adjust the thresholds with different system requirements. If a higher duplicates removal precision is needed, the threshold can be set to a high value. And if the system needs to save the storage as much as possible with moderate requirements on the precision, you could set the threshold to a lower value. Whether the threshold is optimized or not depends on the specific circumstances.

V. CONCLUSIONS

In order to improve the noise-tolerance ability of the duplicates elimination algorithms, a length-variable feature code extraction method, which makes use of the importance of paragraphs, is proposed. On this basis the definition of web page repeatability is defined using the longest common substring. This greatly enhances the fuzzy duplicates removal ability of our approach. Furthermore, the generalized suffix tree structure is implemented to solve the problems of finding the longest common substrings between the web pages. Many extra pairwise comparisons are avoided. Experimental results show that the proposed algorithms can ensure a high efficiency with better duplicates results.

Besides, during our experiments we also found that our approach could not deal with some specific types of web pages. Most often the reason for this was that these pages had very similar text and sentence structures, only some words or numbers in the middle of the sentences were different. Syntax-based feature code extraction cannot deal well with these web pages. Semantic-based methods may be a good solution. For future work semantic-based duplicates removal or hybrid methods will be our focus.

ACKNOWLEDGMENT

This work is supported by Natural Science Foundation of China (No. 60703015 and No. 60973076). The authors are grateful for the anonymous reviewers who made constructive comments.

REFERENCES

[1] Daniel P. Lopresti, “Models and Algorithms for Duplicate Document Detection,” *In Proceedings of the Fifth International Conference on Document Analysis and*

Recognition. Bangalore, India, IEEE, pp. 297-300, 20-22 September, 1999.

[2] J. P. Kumar, P. Govindarajulu, "Duplicate and Near Duplicate Documents Detection: A Review," *European Journal of Scientific Research,* 32(4), pp. 514-527, 2009.

[3] Cao Yu-Juan, Niu Zhen-Dong, WangWei-Qiang and Zhao Kun, "The study on Detecting Near-Duplicate WebPages," *In Proceedings of the 8th IEEE International Conference on Computer and Information Technology (CIT 2008). Sydney, Australia,* IEEE, pp. 95-100, 8-11 July, 2008.

[4] Gurmeet Singh Manku, Arvind Jain and Anish Das Sarma, "Detecting near-duplicates for web crawling," *In Proceedings of the 16th international conference on World Wide Web. Banff, Alberta, Canada,* ACM, pp. 141-150, 2007.

[5] Wang Jian-Yong, Xie Zheng-Mao, Lei Ming and Li Xiao-Ming, "Research and Evaluation of Near replicas of Web Pages Detection Algorithms," *Acta Electronica Sinica,* 28(11), pp. 130-132, 2000.

[6] Rajiv Yerra, Yiu-Kai Ng, "Detecting similar HTML documents using a fuzzy set information retrieval approach," *In Proceedings of 2005 IEEE International Conference on Granular Computing, Beijing, China,* pp. 693-699, 25-27 July, 2005.

[7] Bao Jun-Peng, Shen Jun-Yi, Liu Xiao-Dong and Song Qin-Bao, "A Survey on Natural Language Text Copy Detection," *Journal of Software*, 14(10), 1753-1760, 2003.

[8] Deng, F., Rafiei, D., "Approximately detecting duplicates for streaming data using stable bloom filters," *In Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, ACM, pp. 25-36, Chicago, IL, USA: 2006.

[9] Jeffrey Dean, Monika R. Henzinger, "Method for identifying near duplicate pages in a hyperlinked database," *U. S. Patent 6138113*, USA, Oct 24 2000.

[10] K. Monostori, A. Zaslavsky and H. Schmidt, "MatchDetectReveal: Finding overlapping and similar digital documents," *In Proceedings of the Information Resources Management Association International Conference (IRMA2000)*, Anchorage Hilton Hotel, Anchorage, Alaska, USA, pp. 955-957, 21-24 May, 2000.

[11] A.Z. Broder, S.C. Glassman and M.S. Manasse, "Syntactic clustering of the Web," *Computer Networks and ISDN Systems*, 19(8-13), pp. 1157-1166, 1997.

[12] K. Monostori, A. Zaslavsky and H. Schmidt, "Parallel and distributed overlap detection on the Web," *In Proceedings of the Workshop on Applied Parallel Computing (PARA2000)*, Bergen, Norway, pp. 206-214, 18-21 June 2000.

[13] N. Shivakumar, H. Garcia-Molina, "SCAM: A copy detection mechanism for digital documents," *In Proceedings of the 2nd International Conference in Theory and Practice of Digital Libraries (DL'95)*, Austin, Texas, pp. 85-96, 11-13 June, 1995.

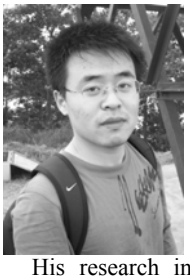
[14] A. Si, H.V. Leong and R.W.H. Lau, "CHECK: A document plagiarism detection system," *In Proceedings of the ACM Symposium for Applied Computing*, San Jose, California, United States, ACM, pp. 70-77, 1997.

[15] Zhang Gang, Liu Ting, Zheng Shi-Fu, Che Wan-Xiang and Li Sheng, "Fast Deletion Algorithm for Large Scale Duplicated Web Pages," *The twentieth anniversary Proceedings of the Chinese Information Processing Society of China (sequel)*, pp. 18-25, 2001.

[16] WU Ping-bo, CHEN Qun-xiu and MA Liang, "The Study on Large Scale Duplicated Web Pages of Chinese Fast Deletion Algorithm Based on String of Feature Code," *Journal of Chinese Information Processing*, 17(2), pp. 29-36, 2003.

[17] CHEN Jin-yan, SUN Ji-zhou and ZHANG Ya-ping, "Finding near replicas of Web pages based on Fourier transform," *Journal of Computer Applications*, 4(28), pp. 948-950, 2008.

[18] LUO Yong-lian, ZHANG Yong-kui, "Research on duplicated news web pages deletion method based on issue time," *Computer Engineering and Applications*, 43(6), 119-121, 2007.



Hongzhi Guo received his bachelor and master degrees from Dep. of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, respectively in 2004 and 2006. Since September 2006, he has been a Ph.D. candidate on Computer Application, at Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China.

His research interest covers information retrieval, search engine, and computational linguistics.

Qingcai Chen received the B.E. degree in engineering mechanics and M.E. degree in solid mechanics, both from Harbin Institute of Technology, China, in 1996 and 1998, respectively. In 2003, he received the Ph.D. degree in computer application from the Computer Science and Engineering Department of Harbin Institute of Technology.

From September 2003 to August 2004, he worked for Intel (China) Ltd. as a senior software engineer. Since September 2004, he has been with the Computer Science and Technology Department of Harbin Institute of Technology Shenzhen Graduate School as an associate professor.

His research interests include speech signal processing, information retrieval, natural language processing and machine learning.

Xiaolong Wang received the B.E. degree in computer science from the Harbin Institute of Electrical Technology, China, the M.E. degree in computer architecture from Tianjin University, China, and the Ph.D. degree in computer science and engineering from Harbin Institute of Technology in 1982, 1984, and 1989 respectively.

He joined Harbin Institute of Technology as an Assistant Lecturer in 1984 and became an Associate Professor in 1990. He was a Senior Research Fellow in the Department of Computing, Hong Kong Polytechnic University from 1998 to 2000. Currently, he is a Professor of Computer Science at Harbin Institute of Technology and the head of Intelligent Computing Research Center at HIT Shenzhen Graduate School.

His research interests include artificial intelligence, machine learning, computational linguistics, and Chinese information processing.