Journal of Web Engineering, Vol. 2, No. 4 (2004) 247–261

RETRIEVING SIMILAR DOCUMENTS FROM THE WEB

ÁLVARO R. PEREIRA JR NIVIO ZIVIANI

Department of Computer Science
Federal University of Minas Gerais
Av. Antonio Carlos, 6627, Pampulha
Belo Horizonte, 31270-901, Brazil
{alvaro, nivio}@dcc.ufmg.br

Received March, 2004
Revised October 20, 2004

This paper presents a mechanism for detecting and retrieving documents from the web with a similarity relation to a suspicious document. The process is composed of three stages: a) generation of a "fingerprint" of the suspicious document, b) gathering candidate documents from the web and c) comparison of each candidate document and the suspicious document. In the first stage, the fingerprint of the suspicious document is used as its identification. The fingerprint is composed of representative sentences of the document. In the second stage, the sentences composing the fingerprint are used as queries submitted to a search engine. The documents identified by the URLs returned from the search engine are collected to form a set of similarity candidate documents. In the third stage, the candidate documents are compared to the suspicious document. The process of comparing the documents uses two different methods: Shingles and Patricia tree.

We implemented and evaluated the methods used for generating the document fingerprint and for comparing the suspicious document with the candidate documents. The experiments were performed using a collection of plagiarized documents constructed specially for this work. The best experimental result shows that in 61.53% of the tries the total number of source documents used in the composition were retrieved from the Web. In this case, in only 5.44% of the executions less than 50% of source documents used in the composition were retrieved from the Web. For the best fingerprint implemented, on average 87.06% of the documents were retrieved.

Keywords: retrieving similar documents, web, document similarity, fingerprint, plagiarism
Communicated by: R Baeza-Yates

1 Introduction

With the Internet, society has rapidly dived into a surge of plagiarism. From primary schools up to graduate courses, the ease to download and copy the information found has given rise to an outbreak of digital plagiarism. Donald McCabe, Professor of Administration at Rutgers University, in New Brunswick, US, investigated 4,500 undergraduate students from fourteen public and eleven private institutions [1]. In his study, 54% of the students admitted to having

used the Internet as a source to copy other people's work, later to be claimed as their own. Perhaps the most alarming consequence of this outbreak is to contribute for plagiarism to flourish as a practice in our educational culture. Students growing up with the Internet do not realize they are plagiarizing. The "copy & paste" action becomes more natural day by day. Students are getting used to simply repeating what someone else has done, with no creativity or innovation, and even worse, without learning from actual production.

Retrieving documents with a sought content is a complex task – particularly in a massive document repository such as the Web – to be performed by search engines that keep Web pages in their document database. The entire document database of a search engine is indexed as a data structure called inverted file, which enables the search task. The user sends the keywords related to the expected answer. By measuring the similarity between the keywords and each indexed document, the most similar ones are returned. For the current work, the problem remains that of retrieving documents from a large collection. The query, however, is not keywords, but a whole document.

This study presents a mechanism able to detect and retrieve documents from the Web that have a similarity relation with a given suspicious document. The process takes place in three main stages. The first stage is the generation of the document fingerprint that represents and identifies the suspicious document. It is composed of sentences from the text that are used in the second stage of the process. The aim of the second stage is to collect documents from the Web that are likely to present a similarity relation to the suspicious document. Each sentence in the fingerprint is then used as a query in a search system that returns the documents eligible to form the similarity candidate document database. In the third stage, each candidate document is compared to the suspicious document. Two methods are used with each pair: Patricia tree [2] and Shingles [3]. The three stages of the process are further detailed in Section 2.

We evaluated the process developed in two different ways: the capacity of the system to retrieve the documents used for composing the suspicious document, and its capacity to measure the similarity between the suspicious document and each original document used in the composition. In order to evaluate the process, we developed a plagiarized-document generator system, which can construct a plagiarized document from passages of Web documents. The system returns the *URLs* of the documents used in the composition and the *expected similarity* between the plagiarized document and each source document, measured by the number of terms used.

For the best fingerprint evaluated, in 61.53% of the tries the total number of source documents used in the composition were retrieved from the Web. In this case, in only 5.44% of the tries less than 50% of source documents used in the composition were retrieved from the Web. For the best fingerprint implemented, on average 87.06% of the documents were retrieved. For the stage of comparison between the suspicious document and each candidate document, the average of differences between the expected and obtained similarities for the best method implemented was 10.94%. We also evaluated the comparisons between the plagiarized document and topic-related documents *not* used in the composition. As expected, we obtained values close to zero, the difference being 2.06% on average.

Since 1994 several mechanisms for detecting similarity between documents have been proposed, using different models and for varied purposes. The SIF tool was the first one, and

treated the similarity problem not only for documents, but also for binary files. The COPS (COpy Protection System) [4] tool and different versions of SCAM (Stanford Copy Analysis Mechanism) [5, 6, 7] resulted from an important study on copy detection mechanisms in large document databases. The first version of SCAM [5] treated the problem considering locally stored documents. Later versions used the Web as the set of documents.

Another important work was KOALA [8], which presented some advantages in comparison to similar mechanisms proposed until that moment, such as noise resistance and a significant reduction of false matches. The CHECK [9] mechanism used new experiment metrics that proved to be satisfactory in terms of space and time. VAST (Visualization and Analysis of Similarity Tool) [10, 11] and CHITRA [12] are prototypes for visualizing the plagiarized parts of documents and of computer programming codes.

The MDR (Match Detect Reveal) tool shares a similar architecture with the tool proposed in this paper. Documents were searched for in an index containing a section of the Web and the documents candidates to similarity were later compared to the suspicious document. Another work [13] presented a new tool and compared it to some copy detection algorithms as proposed in [5, 8, 14, 15]. A suffix tree algorithm [16] was used. A special method for generating the fingerprint was presented [17]. Experimental results showed in most cases the document was not retrieved by using only the fingerprint proposed. Following, seven new fingerprints were presented and analyzed [18].

The system proposed in our work uses the concept of fingerprint for representing the document. The fingerprint is normally composed of sentences used as queries in the TodoBR[a] search engine. Once collected the results are compared to the suspicious document.

The main novelties of our model are: a) the use of metasearch for retrieving similarity candidate documents from the Web; b) the use and evaluation of two methods, Patricia tree and Shingles, for local comparison between the suspicious document and the similarity candidate documents; and c) the development of a mechanism for evaluating the system, which uses a collection of plagiarized documents constructed specially for the work.

2 System Specification

The system developed is divided into three stages. The first stage covers the fingerprint generation for the suspicious document. This fingerprint represents and identifies the suspicious document and is composed of sentences from the text. In the second stage, documents likely to present a similarity relation to the suspicious document are collected from the Web. In the third stage each candidate document is compared to the suspicious document. Two methods are used for comparing each pair of documents: Patricia Tree and Shingles. The three stages are detailed in the following Sections.

2.1 Fingerprint generation

Initially, a fingerprint is generated for the suspicious document. The main difficulty is to identify the best kind of fingerprint and its characteristics, as each sentence of the fingerprint is used as a query for searching and gathering candidate documents. For defining the best fingerprint, we should consider that we are not interested in searching the Web for the exact documents containing such fingerprint, as treated in [17, 18]. The aim of this work is to

[a] TodoBR is a vertical search engine that covers the Brazilian Web (http://www.todobr.com.br).

use the fingerprint for retrieving from the Web documents that might have been used in the composition of the suspicious document. Thus, searching for a list of scattered terms from the document or searching for the most frequent terms should result in low performance, due to the fact that the list of the most frequent terms from each document used in the composition of the suspicious document might not be the same.

We studied and implemented six different types of fingerprints. Most of them are composed of a list of terms in sequence, i.e., *sentences* obtained from the document. In some kinds of fingerprint we used specific terms as anchors in the text and each sentence was obtained by using the same number of terms to the left (including that term) and to the right of the anchor term.

Each of the fingerprints proposed can be altered in terms of granularity and resolution. We define *granularity* in terms of the number of terms in each sentence of the fingerprint. Every sentence from a fingerprint has the same granularity. The *resolution* is the number of sentences composing the fingerprint. Once each sentence in the fingerprint represents a query in the search system, the longest granularity considered was ten terms, which is the maximum limit accepted by most Web search engines. For the same reason, the resolution should be as small as possible. This means fewer queries to the search engine and fewer collected pages for composing the candidate document database. The strategy of sentence selection is specific for each fingerprint. The methods investigated are presented below:

(i) Frequency terms – FT: A fingerprint containing the highest-frequency terms in the document. Its resolution always refers to one sentence, whose granularity can vary.

(ii) Sentences with non-lexical terms – SNLT: The implementation of this fingerprint was motivated by the intuition that sentences involving non-lexical terms would provide good representatives of the document, since we believe in the existence of other documents with the same misspelled terms as rather unlikely. Using the GNU[b] program "ispell", every term not belonging to the Portuguese language dictionary is obtained and sorted from the longest term to the shortest. Since smaller terms can be only acronyms, the longest terms receive highest priority. The top terms from the list are the anchor in the text for obtaining the sentences that will make up the fingerprint.

(iii) Constantly distributed sentences – CDS: Equally distant distributed sentences are used for composing the document fingerprint. Independently of the size of the text, the same number of sentences is obtained, keeping resolution constant.

(iv) Proportionally distributed sentences – PDS: Equally distant distributed sentences are also used for composing the document fingerprint. However, the resolution is proportional to the size of the text, obtained according to Eq. (1), as follows:

$$res = k \times \log\left(\frac{charNum}{10}\right), \tag{1}$$

where $charNum$ is the number of characters in the text, k is a constant and res is the resolution.

(v) Frequency terms sentences – FTS: A list of the most frequent terms of the document is obtained. The top terms are used as anchors in the text for taking the sentences.

[b] http://www.gnu.org

(vi) Inverse frequency terms sentences – IFTS: Likewise, a list of the less frequent terms is obtained from the document and the top terms are used as anchors in the text as well. Due to the large number of terms with frequency one in most documents, the longest terms are chosen.

2.2 Searching and gathering candidate documents

The search uses a metasearch system for constructing the similarity candidate document database as the second stage in the process. Each sentence from a suspicious document fingerprint is used as a simple query in a search engine. A metasearch system consists of a program able to perform queries in search engines, using different services. Its architecture is much simpler than a search engine architecture, since it does not require indexing of Web documents, only searching on it through services available. The metasearcher we developed works in the following manner: the query is submitted to a query generator module for formating according to the TodoBR search engine style. TodoBR is the only service used in this work. The query is processed and returned to the merging module, which obtains the answer set (URLs) and retrieves the respective documents.

2.3 Comparison between the documents

The previous stages were important to build the similarity candidate document database. The third stage is meant to compare each candidate document to the suspicious document in order to check the similarity between the documents in each pair. We use two methods for detecting and evaluating the syntactic similarity between documents: Patricia tree [2] and shingles [3]. The Patricia tree is built over the suspicious document and the candidate documents have their contents searched on the tree, which allows us to detect occurrences of long similar passages in the suspicious document. The second method uses the "shingles" concept [3] for measuring syntactic similarity between each candidate document and the suspicious document, compared in pairs. The total number of shingles present and non-present in each pair of documents is used to calculate the similarity in that pair [19].

2.3.1 Patricia tree method

The Patricia tree (Practical Algorithm To Retrieve Information Coded In Alphanumeric) algorithm was presented in [2]. It is a binary digital tree in which individual bits from the key are used to decide the branch that should be followed. A "zero" bit indicates a branch to the left subtree and a "one" bit indicates a branch to the right subtree. Each internal node of the tree contains an integer that indicates which bit of the query might be analyzed for branching. The external nodes store key values [20]. The conventional Patricia tree construction algorithm has time complexity $O(n \log n)$, which n is the number of keys. A quadratic algorithm was proposed in [21] for secondary memory. In [22] a linear algorithm is proposed.

A semi-infinite string – sistring – is a subsequence of characters from the text, taken from a given starting point and going on as necessary to the right. As an example, for the text "a rose is a rose." we have five sistrings, considering the beginning of each term as being the indexing points: "a rose is a rose.", "rose is a rose.", "is a rose.", "a rose.", and "rose.".

The following example explains the Patricia tree method used. Consider the text "a rose is a rose is a rose." as representing the suspicious document and "never a rose is a rose and

a violet." as representing a candidate document. The algorithm starts by reading the two documents and storing the length of the candidate document, which is 36 characters long. Next the sistrings of the suspicious document are inserted in the Patricia tree. From this point on, the algorithm calculates the similarity by searching the sistrings of the candidate document on the Patricia tree. For each search, if the number of characters in the passage found is greater than fifteen, this value is used to compute the total similarity. The passages "never " and "and a violet." are not found in the tree. The passage "a rose is a rose " is found. Since 17 of the 36 total characters of the candidate document are found in the tree, $17/36 = 47.22\%$ of the candidate document is present in the suspicious document, according to the Patricia tree method implemented.

2.3.2 Shingles method

According to [3], two documents A and B can present relationships of "resemblance" and "containment". The w-shingling $S(D, w)$ of a document D is the set of total shingles with size w contained in D. This set represents the information used to calculate the similarity between documents. For example, the set of shingles from the text "a rose is a rose is a rose." with $w = 4$ is:

$S(D, 4) = \{$(a, rose, is, a), (rose, is, a, rose), (is, a, rose, is)$\}$,

resulting in three different shingles. In this work, the shingles that occur more than once in the text will appear only once in the answer set, as with the two first shingles from the example. Experiments demonstrate that a better performance is obtained for this situation.

From the distinct set of total shingles of two documents S and C, the absolute similarity between them is calculated using the concept of intersection and union of sets, as showed in Eq. (2):

$$r(S, C) = \frac{|S(S) \cap S(C)|}{|S(S) \cup S(C)|}, \tag{2}$$

in which $S(S)$ represents the set of total shingles of the suspicious document and $S(C)$ the set of shingles of the candidate document.

In practical terms, we have $S(S) \cap S(C)$ representing the total number of shingles occurring in the suspicious document and in the candidate document. $S(S) \cup S(C)$ represents the sum of the number of shingles occurring simultaneously in the two documents plus the number of shingles that occurs in each of the documents that do not occur in the other one. In the same way, it is possible to verify how much of a candidate document C is contained in a suspicious document S, as in Eq. (3):

$$c(S, C) = \frac{|S(S) \cap S(C)|}{|S(S)|}, \tag{3}$$

In this work we are interested in the percentage of the candidate document that is present in the suspicious document. Thus, we use the containment concept showed in Eq. (3).

We also explain the shingles method implemented by means of an example. Consider the text "a rose is a rose is a rose." as representing the suspicious document and "never a rose is a rose and a violet." as representing a candidate document. Also consider the size of the shingle $w = 3$. Every shingle from the suspicious document is obtained and inserted in a hash table. Each different shingle is inserted only once, even if it occurs more than once. For this example we have three different shingles: {(a, rose, is), (rose, is, a), (is, a, rose)}. Next, we

obtain shingles from the candidate document: {(never, a, rose), (a, rose, is), (rose, is, a), (is, a, rose), (a, rose, and), (rose, and, a), (and, a, violet),}, in a total seven different shingles. These shingles are searched in the hash table. Since three of the seven total shingles from the candidate document are found in the hash table, $3/7 = 42.86\%$ of the candidate document is present in the suspicious document.

3 Experimental Results

3.1 Construction of plagiarized document collections

For performing the experiments we developed a plagiarized-document generator system that uses passages from Web documents. The system is based on the intuition that someone using the Web for plagiarism does not make significant changes in the plagiarized text. Thus, such changes as replacing words with synonyms or substituting terms in a sentence, but keeping the original sense, are not treated by the system. The plagiarized document generator system simulates the *composition* of a document from original Web documents.

We created a synthetic set of documents as follows. We composed a set of documents from passages of documents available in the Web, whose themes are given by the words from the query. The aim of the system is to simulate a composition of a document made by a user from passages of documents from the Web. The number of documents used in the composition of the plagiarized document must be set, as well as the number of terms that the plagiarized document will have in relation to the size of the documents returned from the search. The new document formed is labeled "plagiarized document".

The system initially collects the first ten documents returned from a query performed by the search engine TodoBR. Following, the HTML document is parsed to obtain the text in ASCII format, which is separated into paragraphs. We consider a paragraph as being the text containing two characters "full stop" – that is, the concatenation of three sentences. Random paragraphs from each document are used to compose the plagiarized document, always maintaining the same percentage of common terms of the candidate document present in the plagiarized document. This information is the *expected similarity* of the plagiarized document related to that candidate document. The expected similarity represents how much text from the candidate document is present in the plagiarized document.

3.2 Fingerprint generation

The three experiments for evaluating the fingerprint generation stage had different objectives. Initially, the plagiarized document fingerprint is obtained. Each sentence of the fingerprint is used as a query in the metasearch system using the TodoBR search engine. The URLs of the documents returned for the query are compared to the URLs of the documents used in the composition of the plagiarized document. The percentage of documents retrieved for that fingerprint is returned, for each plagiarized document.

The resolution of the fingerprint influences the running cost for the system. Running experiments with different granularity and resolution values for different fingerprints is expensive for large collections of plagiarized documents. For this reason, we used a reduced collection of plagiarized documents in the first experiment, in which the best granularity value was chosen and used in the next experiments. For the same reason, in each experiment performed the fingerprint with worst result was excluded for the next experiments. These filters helped us

to reduce the cost of the experiments.

3.2.1 Choosing the best granularity

The first experiment aimed to filter the fingerprint used, choosing the best granularity for each fingerprint and excluding the fingerprint with the worst result. The six fingerprints proposed in the Section 2.1 were implemented. The experiment used a small collection of 350 plagiarized documents.

Except for the PDS and FT fingerprints, each one was tried out with 5, 10 and 15 sentence resolutions and 4, 6 and 10 term granularities, matching each resolution to a granularity value. For the PDS fingerprint only the granularity values varied, since the resolution was defined by Eq. (1), with $k = 2$. Resolution also does not apply to FT fingerprints.

The graph in Figure 1 compares different granularities for the fingerprints. It considers the average of the fraction of documents retrieved that were used in the composition of the plagiarized document, for different resolutions. We observe that the highest granularity experienced, which was ten terms, presented the best results (except for FT). Thus this granularity was selected for the next experiments. For the FT fingerprint we collected 10, 30 and 50 pages, the last one producing the best result, as showed in Figure 1. As this fingerprint presented a small percentage of documents retrieved, it was excluded from the next experiments.

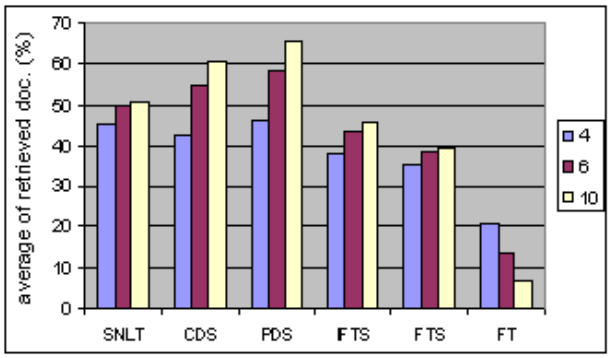

Fig. 1. Comparing different granularities across fingerprints.

3.2.2 Best fingerprints

The previous experiment was useful to filter the various possible compositions of a fingerprint. The aim of this experiment is to evaluate the quality of the fingerprints for a longer number of documents. We used a collection of 1,900 plagiarized documents for evaluating the performance of five different fingerprints: SNLT, CDS, PDS, FTS and IFTS; for three different resolutions: 5, 10 and 15 sentences. For the PDS fingerprint, resolution is defined by Eq. (1), with $k = 1$ and $k = 2$, presenting average resolutions of 5.84 and 12.15 sentences, respectively. The granularity is fixed in ten terms, for every fingerprint.

The graph in Figure 2 compares the average percentage of the retrieved pages, for each fingerprint, with the three different resolutions. Longer resolution fingerprints performed

better than shorter ones. This happens because longer fingerprints collect a great number of documents. Thus, they are better representatives of a given document, at a higher cost.

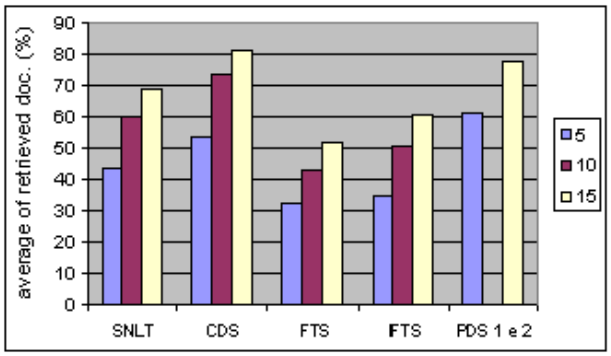

Fig. 2. Comparing different resolutions across fingerprints.

From Figure 2 we learn that the best fingerprint – CDS with resolution 15 – on average returned 81.28% of the documents used in the composition of the plagiarized document, followed by PDS with $k = 2$, returning 77.36% of the documents. Figure 3 presents the Pareto graph[c] for the best fingerprint, CDS, clustering the percentage of documents retrieved in intervals of 10% (except for 100%). We verify that in 46.75% of the cases, 100% of the documents in the composition were retrieved from the Web. In only 8.71% of the cases performance did not reach 50%.

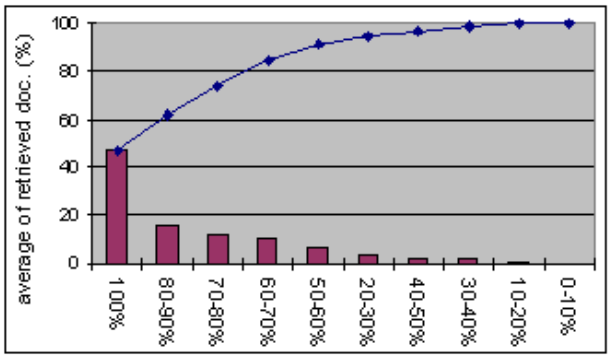

Fig. 3. Pareto graph for the CDS fingerprint with resolution 15.

As described in Sections 3.1 and 3.2, we used the URLs of the first ten documents returned from the search system. These URLs are used to verify the documents that were used in the composition of the plagiarized document. Since the gathering of a document represents some cost for the system, the experiments also analyzed the ranking position of the document returned, aiming to verify the possibility of collecting only a percentage of the top ten ranked

[c] Bar graphic that sorts the categories by decreasing order, from left to right.

documents for comparison. The graph in Figure 4 compares the average of the documents retrieved for the top ten ranked documents (as showed in Figure 2), to the top two ranked documents and to the highest ranked document. The graph considers the resolution of 15 sentences.

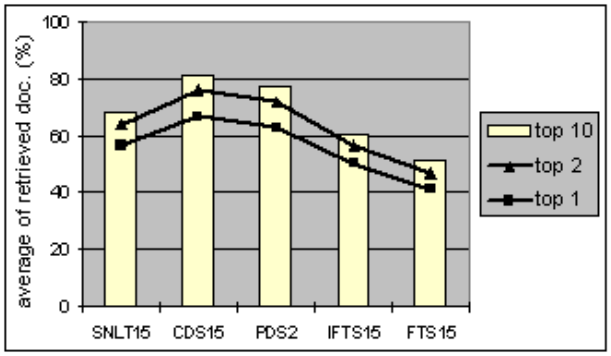

Fig. 4. Comparing the ranking positions of documents retrieved for the ten top, two top and the top ranked document.

We observed that, on average, 81.66% of the *documents retrieved* were in the top position (i.e., in 81.66% of the cases, the top ranked result of a query was a document that contributed to the plagiarized document) and 93.12% were in the top two positions, that is, in the top ranked or in the second ranking position. Thus, the performance of the system, measured by the average of the documents retrieved, is slightly reduced if the candidate database is composed of only the top two documents. Collecting only two documents for each sentence of the fingerprint would strongly reduce the cost of gathering candidate documents. Apart from this, there was indication, when the document was not found among at least the top two ranked documents, that a document with similar content had been found. This could not be verified in the experiment.

We undertook a manual analysis of a set of plagiarized documents from the CDS fingerprint with a resolution of 15 sentences, in which no document or only one document used in the composition was retrieved from the Web. We observed that most of the pages were (a) home pages with frames or a menu, (b) web blogs with special characters not recognized by the search engine used, (c) lists or (d) forms. These kinds of documents are not interesting for a user to compose a plagiarized document, since they do not have a sequential structured text. The plagiarized document generator system might have been affected by this fact. For the situations manually analyzed, the plagiarized document was composed of distributed terms or terms with special characters from the documents used in its composition.

3.2.3 Merging fingerprints

The previous experiment aimed to measure the performance of the system for the different fingerprints isolated. In this experiment the different fingerprints are merged in order to compose a new one with a greater capacity of retrieving similar documents. We used the same collection as in the previous experiment and the worst fingerprint for that experiment, FTS, was not considered. We considered 30-sentences as maximum resolution. Thus, it

was possible to merge all four fingerprints with resolution 5, or to merge three by three the fingerprints with resolution 10, or yet merge two by two the fingerprints with resolution 15.

Table 1 shows the average results of the possible combinations. We notice a slight increase in the average of retrieved documents, compared to the isolated fingerprints performed in the previous experiment. According to Table 1, the best merged fingerprint is "SNLT-CDS-PDS-10" (merging the fingerprints SNLT, CDS and PDS, each one with resolution 10), followed by "SNLT-CDS-15". In general, the average results were improved by using merged fingerprints.

Table 1. Different fusions of the fingerprints

New fingerprints	**Average of the experimental results**
All-four	82.91
SNLT-CDS-PDS-10	87.06
SNLT-CDS-IFTS-10	85.15
SNLT-PDS-IFTS-10	86.04
CDS-PDS-IFTS-10	86.56
SNLT-CDS-15	86.63
SNLT-PDS-15	84.80
SNLT-IFTS-15	80.35
CDS-PDS-15	86.39
CDS-IFTS-15	85.91
PDS-IFTS-15	84.19

Figure 5 shows the Pareto graph for the best merged fingerprint, "SNLT-CDS-PDS-10". Analyzing the graph we detect a significant improvement in the performance of the new fingerprint: in 61.53% of the cases 100% of the documents were retrieved from the Web, against 46.75% of the best isolated fingerprint, CDS, showed in the Figure 3. This means a improvement of more than 30% in the tries that returned *all* documents used in the composition of the plagiarized document. For the same merged fingerprint, "SNLT-CDS-PDS-10", only in 5.44% of the cases the performance fell below 50%.

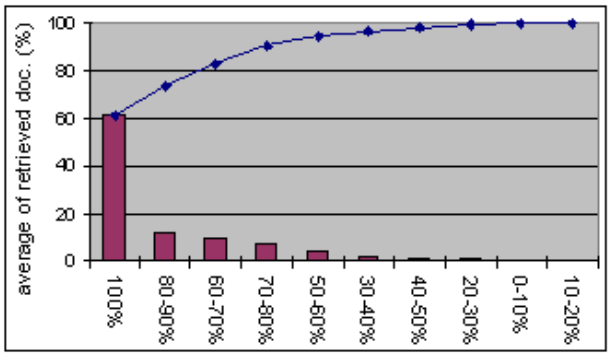

Fig. 5. Pareto graph for the merged fingerprint "SNLT-CDS-PDS-10".

As discussed in Section 3.2.2, we suspect that the situations in which fewer than 50% of the documents of the composition were retrieved back resulted from noises in the experiment, introduced by documents not desired in a composition like home pages, web blogs, lists and

forms. Not considering the tries in which fewer than 50% of the documents were retrieved, the average of documents retrieved for fingerprint "SNLT-CDS-PDS-10" grows up to 90.31%. For the merging "SNLT-CDS-15" this value is 89.91%.

3.3 Comparison between documents

The first three experiments evaluated the six different fingerprints used to retrieve from the Web the documents used in composing the plagiarized document, in order to form the similarity candidate document database. We shall now consider the document database as completed and proceed to evaluate the performance of the Patricia tree and Shingles methods used in the stage of document comparison.

3.3.1 Comparison between Patricia and Shingles methods

The first experiment to test the document comparison stage sought to determine which of the two methods would produce the best average results. At this point we worked with a collection of 900 plagiarized documents, each one made up of three to ten Web documents. In the different tries with the Shingles algorithm, w ranged from two to ten. For Patricia tree algorithm, we considered the beginning of each term as being the indexing points.

In this experiment, the system performance was measured based on the absolute differences between the expected similarity and the similarity obtained through the Patricia tree and Shingles algorithms. Table 2 presents the results obtained with both methods, taking into account different w values for the Shingles. The Shingles method clearly presented higher degrees of accuracy than the Patricia tree for some of the w values. The best result obtained was for $w = 4$, the difference between expected and obtained similarity being 4.13% on average. The same measurement for the Patricia tree was 7.50%. For this reason, only the Shingles method will be used in the next experiments.

Table 2. Average of absolute differences between expected and obtained similarity

	Shingles									**Patricia**
***w* values**	2	3	4	5	6	7	8	9	10	
Differences	8.97	6.72	4.13	5.04	7.34	9.67	11.42	12.71	13.78	7.50

3.3.2 Evaluating the Shingles method

As demonstrated in the previous experiment, in a comparative analysis of the two methods, the Shingles method presented the smallest difference across tries when we considered the absolute difference between expected and obtained similarities. In the experiment now being described, we considered the same absolute difference, but the values presented demonstrate the percentage of this difference related to the expected similarity. In this way, it is possible to analyze the results in terms of percentages, in which values close to zero stand for close to optimal results. We shall call this difference "relative" to the expected similarity. This time, we used a collection of 4,800 plagiarized documents, each one composed of three to ten Web documents in a total 25,000 comparisons. Figure 6 presents the average of relative differences for the main values of w. The best average results obtained were for $w = 3$, followed by $w = 4$, with differences of 12.95% and 16.59% on average, respectively.

Figure 7 shows the outliers graph for $w = 3$. We regarded as outliers all values three times higher than the standard deviation and took into account the average of relative differences for

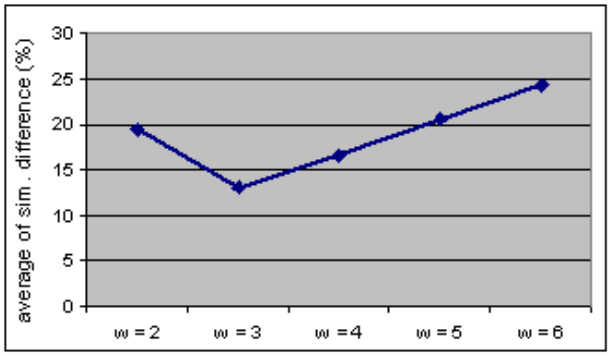

Fig. 6. Graph presenting average relative results for the following w values.

each set of documents used in making the plagiarized document. A large number of outliers were found, which we shall analyze below.

The similarity candidate document database then received a document *not* used to compose the plagiarized document, to be compared to and have its similarity measured against the plagiarized text. We selected a document retrieved from the Web in the same query that produced the documents used in the composition of the plagiarized document, i.e., a document dealing with the same subject matter. A very low degree of similarity was expected. Figure 8 presents the graph of outliers from the comparisons between the plagiarized document and the documents not used in the composition. As expected, this yielded values which were very close to zero and the average similarity reached 2.06% considering the outliers.

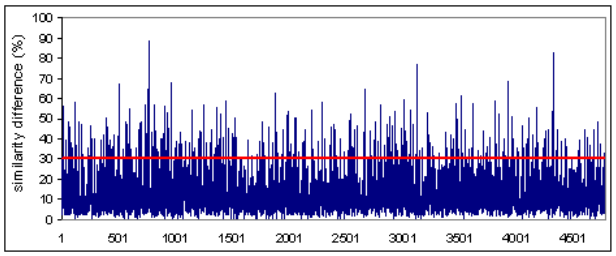

Fig. 7. Graph of outliers with the Shingles method, for $w = 3$.

In both kinds of comparisons, outliers occurred when the documents used (or not) to compose the plagiarized document shared certain passages, either because they were duplicates, mirrors, new versions or examples of sheer plagiarism. This provoked the plagiarized document generator to register wrong information regarding how much of the source document was used in the composition, since while processing the expected similarity, it only analyzed the sentences which were taken from the documents and disregarded passages common to the documents. This fact strongly suggests that the outliers emerged from unwanted interference in the experiment, instead of a fault in the process or in the method used. Once we eliminate the outliers from the data analyzed, the averages of relative differences for $w = 3$ and $w = 4$

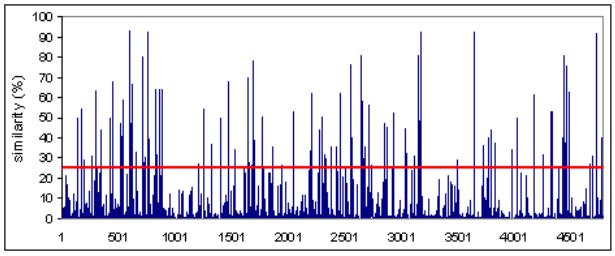

Fig. 8. Graph of outliers when comparing documents not used in the plagiarized document.

rise to 10.94% e 14.28% respectively.

4 Conclusions

We have proposed and implemented a process for detecting and retrieving similar documents from the Web. Through the construction of a plagiarized document collection in which each document contained passages from Web documents, it was possible to measure and evaluate the performance of this process. In the first of three stages involved, the fingerprint of the suspicious document was generated. The second stage used sentences of the fingerprint for searching documents and composing a similarity candidate document database. The third stage compared the suspicious document to each candidate document.

This paper presents experiments measuring the performance of the methods used to generate a document fingerprint and comparison between the suspicious document and each candidate document. For the best fingerprint evaluated, on average 87% of the documents used in the composition of the plagiarized document were retrieved and inserted in the similarity candidate document database. For the merged fingerprint "SNLT-CDS-PDS-10", in almost 62% of the tries, *all* documents used in the composition of the plagiarized document were retrieved and 93% of these documents retrieved were among the top two ranked documents.

For the comparison between the suspicious document and each candidate document, two methods were implemented: Patricia Tree and Shingles. The methods were evaluated based on the difference between expected and obtained similarity. The Shingles method performed more efficiently than the Patricia Tree method, proving itself satisfactory for use in the detection of similarity across documents.

Acknowledgements

This work was supported in part by GERINDO Project–grant MCT/CNPq/CT-INFO 552.087/02-5, CYTED VII.19 RIBIDI Project and CNPq Grant 30.5237/02-0 (Nivio Ziviani).

References

1. M. Stricherz (May 9, 2001), *Many teachers ignore cheating, survey finds*, J. Education Week on the Web, http://www.edweek.org/ew/ewstory.cfm?slug=34cheat.h20.
2. D. R. Morrison (1968), *Practical Algorithm to Retrieve Information Coded in Alphanumeric*, Journal of the ACM, Vol. 15, Num. 4, pp. 514-534.

3. A. Broder (1998), *On the Resemblance and Containment of Documents*, Compression and Complexity of Sequences (SEQUENCES'97), IEEE Computer Society, pp. 21-29.
4. S. Brin and J. Davis and H. Garcia-Molina (1995), *Copy detection mechanisms for digital documents*, ACM SIGMOD Annual Conference, pp. 398-409.
5. N. Shivakumar and H. Garcia-Molina (1995), *SCAM: A Copy Detection Mechanism for Digital Documents*, 2nd International Conference in Theory and Practice of Digital Libraries (DL'95).
6. N. Shivakumar and H. Garcia-Molina (1995), *The SCAM Approach To Copy Detection in Digital Libraries*, D-lib Magazine, month 15, http://www.dlib.org/dlib/november95/scam/11shivakumar.html.
7. H. Garcia-Molina, L. Gravano and N. Shivakumar (1996), *dSCAM: Finding Document Copies Across Multiple Databases*, 4th International Conference on Parallel and Distributed Systems (PDIS'96).
8. N. Heintze (1996), *Scalable Document Fingerprinting*, USENIX Workshop on Electronic Commerce.
9. A. Si, H. V. Leong and R. W. H. Lau (1997), *CHECK: a document plagiarism detection system*, ACM symposium on Applied computing, pp. 70-77.
10. F. Culwin and T. Lancaster (2001), *Visualising Intra-Corpal Plagiarism*, 5th International Conference on Information Visualisation (IV'01), pp. 289-296.
11. T. Lancaster and F. Culwin (2001), *Towards an Error Free Plagiarism Detection Process*, 6th Annual Conference on Innovation and Technology in Computer Science Education, pp. 57-60.
12. R. L. Ribler and M. Abrams 2000, *Using Visualization to Detect Plagiarism in Computer Science Classes*, IEEE Symposium on Information Vizualization, pp. 173-178.
13. K. Monostori, A. Zaslavsky and H. Schmidt (2001), *Efficiency of Data Structures for Detecting Overlaps in Digital Documents*, Australasian Computer Science Conference (ACSC '01), pp. 140-147.
14. U. Manber (1994), *Finding Similar Files in a Large File System*, USENIX Winter 1994 Technical Conference, pp. 1-10.
15. A. Broder, S. Glassman, M. Manasse and G. Zweig (1997), *Syntactic clustering of the Web*, 6th International World Wide Web Conference, pp. 391-404.
16. W. Frakes and R. Baeza-Yates (1992), *Information Retrieval: Data Structures and Algorithms*, Prentice-Hall (North Virginia).
17. T. A. Phelps and R. Wilensky (2000), *Robust Hyperlinks: Cheap, Everywhere, Now*, Digital Documents and Electronic Publishing (DDEP00), pp. 13-15.
18. S. Park, D. Pennock, C. L. Giles and R. Krovetz (2002), *Analysis of lexical signatures for finding lost or related documents*, 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 11-18.
19. A. R. Pereira Jr and N. Ziviani (2003), *Syntactic Similarity of Web Documents*, First Latin American Web Congress, pp. 194-200.
20. N. Ziviani (2004), *Projeto de Algoritmos com Implementações em Pascal e C*, Pioneira Thomson, second edition.
21. G. H. Gonnet, R. A. Baeza-Yates and T. Snider (1992), *Information Retrieval: Data Structures and Algorithms*, Chapter *New Indices for Text: Pat Trees and Pat Arrays*, Prentice-Hall, pp. 66-82.
22. E. Ukkonen (1995), *On-line construction of suffix trees*, Algorithmica, pp. 249-260.