2014 5th International Conference on Information and Communication Systems (ICICS)

Approaches for Candidate Document Retrieval

Šimon Suchomel
Faculty of Informatics
Masaryk University
Brno, Czech Republic
Email: suchomel{at}fi.muni.cz

Michal Brandejs
Faculty of Informatics
Masaryk University
Brno, Czech Republic
Email: brandejs{at}fi.muni.cz

***Abstract*—Plagiarism has become a serious problem mainly because of the electronically available documents. An online document retrieval is weighty part of a modern anti-plagiarism tool. This paper describes an architecture and concepts of a real-world document retrieval system, which is a part of a general anti-plagiarism software. A similar system was developed as a part of nationwide plagiarism solution at Masaryk University. The design can be adapted into many situations. Provided recommendation stem from experience of the system operation for several years. The proper usage of such systems contributes to gradual improvement of the quality of student theses.**

I. Introduction

Standard systems for plagiarism detection, which operate on the basis of detailed document comparison, cannot detect similarities unless they possess both the source and the plagiarized document. An algorithm to evaluate a document similarity must build inner indices for a detailed document comparison. A modern plagiarism detection process can be divided into two main tasks. The candidate document retrieval; and the detailed document comparison, which can be reduced to pairwise document comparison. The pairwise document comparison is very computational demanding especially for real time plagiarism solutions which must evaluate millions of documents [1]. Figure 1 shows approach of a modern plagiarism detection. For an input suspicious document the outputs of the plagiarism revealing software are annotated passages of that document, which may have been plagiarized.

A candidate document retrieval is a process for anti-plagiarism software to be performed for each suspicious document before it computes pairwise document similarities to find potential sources of plagiarism. The goal is to enlarge the document database of anti-plagiarism system of relevant documents only. More relevant document means better opportunity to discover specific similarity. On the other hand, since the similarity computation is very time consuming it is not wise to maintain uselessly vast document database, for instance by crawling the web.

A. Candidate Document Retrieval

Having a suspicious document d_{susp} and a very large document collection D of potential source documents, the candidate document retrieval task is to select a small subset $D_{ret} \subseteq D$ of documents $d_{ret} \in D_{ret}$ which probably served as a source of plagiarism. For example, documents which have a sufficient probability of $sim(d_{susp}, d_{ret}) > 0$, where sim can represent any inter-documents measurable similarity. In a realistic scenario the D would contain all documents on the web and the access method would be through the standard interface of a modern search engine, where we do not have direct access to its internal index.

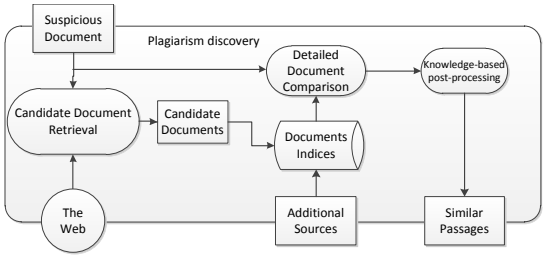

Fig. 1. A global view of a modern anti-plagiarism software.

The retrieved candidate documents are subsequently indexed for the purpose of the detailed document comparison.

II. Conceptual System Characteristics

The outer behaviour of the candidate document retrieval system should be as much like as is behaviour of a student who searches for documents on the web and reuse a text from them. Martin Potthast depicts a standard process of text reuse from the web as shown at figure 2.

Considering a standard plagiarism detection tool, we suppose that suspicious documents are single-themed. That is the most common situation. Such documents are for example theses or seminar papers. The majority of documents which are expected to be checked for plagiarism are single-themed. This assumption leads to possibility of extracting keywords from the whole document without significantly lowering the performance of automated keywords extraction. Keywords extraction is one of the fundamental features of the source retrieval system (see the following section for more details). Under this assumption an example of unsuitable use of an anti-plagiarism tool would be checking of one diverse-themed document like newspapers uploaded in a single file. Such a document should be then divided into separate documents according to the articles which the newspapers contain, which will result again in single-themed documents.

From a single document point of view, the system preprocess the input document and creates appropriate queries which are submitted to a search engine interface. The search results must be also processed accordingly. The system should follow several considerations: i) maximizing precision and recall; ii) minimizing the overall cost; iii) be scalable and robust.

978-1-4799-3023-4/14/$31.00 ©2014 IEEE

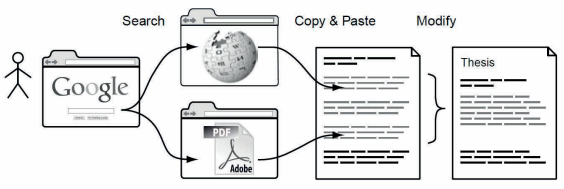

Fig. 2. The generic steps of text reuse from the web [2].

A. Retrieval Performance

The demand to maximize the recall and precision of retrieved documents is obvious. However, it is usually balanced with acceptable computational load of the system. It is also very difficult to measure precision and recall of a real-world web document retrieval system. Let D_{src} denote the set of documents that served as a source of plagiarism for document d_{susp}, and let D_{ret} denote the set of retrieved documents. Then the precision and recall can be defined as $prec = |D_{ret} \cap D_{src}|/|D_{ret}|$ and $rec = |D_{ret} \cap D_{src}|/|D_{src}|$ respectively. However this standard information retrieval calculation is far from being applicable. Namely because of the existence of so called near-duplicate documents on the web [3]. The source document retrieval system can select a near duplicate document d_{ret} which certainly is true positive detection and it does not have to be the same source document d_{src} which was plagiarized from. In order to measure a near duplicate, some characteristics must be defined. For an anti-plagiarism system the positive value of similarity $sim(d_{susp}, d_{ret}) > 0$ can be sufficient to consider the retrieved document d_{ret} as a true positive. The similarity can be any kind of likeness between two documents which is computed by the detailed document comparison subsystem of the anti-plagiarism software.

Organizers of PAN[1], competition on plagiarism detection, determine whether a document d_{ret} is a near-duplicate to any document from the set of source document D_{src} by three characteristics: i) whether they are actually equal $d_{ret} = d_{src}$; ii) whether they are similar according to an empirically set threshold of Jaccard similarity $sim_{jac}(d_{ret}, d_{src}) > n$; iii) or whether the passages in a suspicious document d_{plg} that are known to be reused from d_{src} are also contained in d_{ret} [4]. We can now observe that one document can be a near-duplicate of more than one source document and one source document can have more than one near-duplicate. Next, they denote a set D_{dup} of all near-duplicates of all source documents D_{src} of d_{plg} and a subset D'_{ret} of D_{src} containing documents having at least one positive detection in D_{ret}. Then precision and recall of set D_{ret} based on a suspicious document d_{plg} are defined as follows:

$$precision = \frac{|D_{ret} \cap D_{dup}|}{|D_{ret}|}, recall = \frac{|D'_{ret} \cap D_{src}|}{|D'_{ret}|} \quad (1)$$

This results in a fact that retrieving more than one near-duplicate document to a single source document does not increase recall and it does not decrease precision either. Retrieving the first of the near-duplicate documents into a single source document increases both recall and precision.

It is worth mentioning that in order to evaluate all near-duplicates we need to build an index of the whole corpus of all potential source documents, which could be searched via a given search engine. Therefore, such evaluation is infeasible in a real-world situation when the corpus of source documents is the web.

In a real-world scenario the recall is much more important than precision. If the precision is low it could affect time performance of the retrieved algorithm, since the system would process a lot of documents needlessly. It can also excessively extend the index for detailed document comparison, which is not a problem as long as the detailed document comparison is feasible according to user expectations. On the contrary, if the recall is low, the anti-plagiarism system may simply not be able to detect the plagiarized passage, since it may not have the source document retrieved and indexed in its database.

In addition to documents that contain similar passages with the suspicious document, we consider as a true positive retrieved result a document following the same theme as the source document. Themed documents are considered relevant. A theme can be detected by overlapping sets of equal keywords or keyphrases [5]. Existing themes are therefore defined by the characteristics of the suspicious documents within the database of the anti-plagiarism system.

B. Retrieval Cost

In a standard way the cost of the system consists of time and space requirements of all algorithms and data needed. Apart of that, the most costly component is the number of executed search queries, and secondly the number of Internet document downloads.

In any information retrieval system, there is always a correlation between the retrieval performance and the cost. Consider a system using an exhaustive search approach. For example querying every sentence from a suspicious document would result in high recall, but it is simultaneously too cost demanding to be applicable elsewhere than in an experimental environment. On the contrary, in real-world systems the number of search queries should be narrowed as much as possible, which can result in certain situations in executing only a single query per suspicious document.

It is crucial to reduce the number of queries since the real anti-plagiarism system must utilize modern search engines like Bing, Google or Yahoo. Furthermore, each search engine has strict rules about the amount of queries which can be submitted from one IP address, which prevents to use the exhaustive search. The query execution is usually not particularly time demanding, yet the time consumption is not negligible. The automated querying can often be attended by additional fees. In the document retrieval system design, the querying represents the most expensive part.

The second significant part of the system cost is the number of document downloads. The download alone is in todays system a cheap operation, but it can be very time and space consuming while considering a huge number of downloads. Also a post-processing of the downloaded documents is very time consuming operation. The number of downloads must be tuned according to the system computational possibilities and expectations.

[1] http://pan.webis.de/

C. System Scalability and Robustness

The purpose of the system determines its scalability. The modern anti-plagiarism systems[2] maintain database of millions of documents and are able to process new documents within hours or a day. The complete processing of a new document means that all results of candidate document retrieval together with the suspicious document must be already indexed for detailed document comparison. Afterwards the evaluation of similarities of that document is usually real-time (within seconds). The design of the candidate document system, which is further recommended, can scale easily by adding more computational nodes.

A need for robustness stem mainly from a huge diversity of Internet documents. It is discussed together with the detailed design of the system in the following sections.

III. System Architecture

The candidate document retrieval should run as several independent tasks, in order to be highly parallelizable and scalable. The tasks can share data via transactional relation database. The database represents a central point for process control. If it is accessible over a network, the computational power can be increased by adding more computer nodes. The database should be utilized in order to keep detailed information about the progress of document processing. The tasks could be divided according to the following functions into 4 main groups: 1) parsing an input document; 2) searching the Internet; 3) downloading the results; 4) the results post-processing.

A. Parsing an Input Document

Let us assume that an input of this stage is a textual representation of a suspicious document d_{plg}. Since the anti-plagiarism system needs to build data structures for the detailed document comparison, the plaintext format is needed anyway. Therefore the input document conversion into plaintext must be generally also considered. The output of this stage would be queries prepared for their execution.

Textual processing and keywords extraction algorithms may become quite time demanding. A standard algorithm optimization should be considered when needed. This stage, however, does not represent the most time demanding part of the overall candidate document retrieval process. Each suspicious document is processed independently, thus the system may scale by simultaneously processing suspicious documents.

The matters to consider in this stage include: i) document cleaning and preprocessing; ii) language detection; iii) chunking; iv) keywords extraction; v) query formulation; vi) permanent storage of extracted queries and the input document information into the database.

Cleaning of the document may comprise special characters removal; original document structure violation; existing citation detection; or in-text urls extraction for a direct download.

[2]For example anti-plagiarism system run by Masaryk University (http://theses.cz/, http://odevzdej.cz/) or Turnitin (http://turnitin.com/).

1) Language Detection: A modern anti-plagiarism system should also be able to detect similarities among and across multiple languages. This must be borne in mind during the system implementation. Many of the shelf tools for lingual processing or keyphrase extraction would not be possible to utilize.

Current effective automated language identification methods are based on frequency analysis, such as utilizing the principle of language-characteristic sequences of n-grams. For the usage of such methods one needs to construct a referential vector language model for each supported language. Other beneficial, less computational demanding method, can be language detection by stop-words matching. Only lists of language specific stop words are kept and the language with the highest number of matches is selected. This method works reliably for longer textual parts. Next, a method based on word relevancy can be utilized for shorter texts. It is also applicable for the web page language identification [6]. With supporting of multiple languages the automated language identification must also be applied on every retrieved web document.

Please consider that in many theses, there are usually small parts of text written in multiple languages, such as the abstract or summary. The detection method should detect the major language of the text or identify those language-different parts.

2) Chunking and Keywords Extraction: The purpose of chunking is to distribute focus of text processing algorithms evenly across the document and thus lower the possibilities of influencing the efficiency of that algorithms by unexpected characteristics of the text. Chunking is also applied in order to detect textual differences, where one cannot preset the exact boundary in a document, where the textual characteristics are changed. For this purpose, the principle of sliding window is usually used, where two primary parameters must be determined. The first stands for the size of the window and the second represents the size of the overlap between two neighbour windows during sliding. The size of the overlapping part also influences the detected characteristic differences between two chunks. If the overlapping size is too big the difference would probably pass unnoticed. On the other hand, using small size of that interval sharpens algorithm detection edges, but poses a risk of placing the window's centre on the textual characteristic boundary, resulting accidentally in no detection. It may be also considerable to process more than one pass of the algorithm with different sliding windows settings—a type of cross validation.

Other considerable chunking approaches are no chunking, paragraph based chunking, chapter based chunking, or sentence chunking.

The Keywords or keyphrase extraction is the most straightforward process for subsequent query formulation. The high quality keywords extraction is crucial for the proper query formulation. Modern keywords extraction methods are based on the word repetition allied to a statistical estimate of likelihood. Also the most widely used method in both PAN competitions on plagiarism detection (2012 and 2013) in the source retrieval subtask was keywords scoring by $tf.idf$ (term frequency—inverse document frequency) [7], [4].

Keyphrases can also be extracted from the selected chunks. However in the real-world scenario it appears to be more

beneficial to extract global keywords from the whole document. Such keywords are fully related to the document theme and should suitably describe the individual document. From a longer textual part, there can also be obtained more descriptive keywords than from the shorter part.

3) Query Formulation: The query formulation is the most important part of the candidate document retrieval system, since it has the highest impact on the overall performance and costs. In order to control the cost, a maximum number of queries submitted per document should be set. The total number of executed queries influences directly not only the cost, but also the time demands of the input document process and the number of Internet documents to be processed.

Suchomel et al. [8] propose a methodology based on the combination of three different types of queries. The first type of queries is constructed from keywords or keyphrases extracted from the whole document. They suggest to use 5 word long keywords based queries. The query length is important since it directly influences the number and the relevance of retrieved results. If the query is too long, it could be too specific, which will probably lead to no retrieved results. On the other hand, if the query is too short it will be too general resulting in retrieval of many irrelevant documents. The purpose of the keywords based queries is to retrieve theme bounded documents.

Other types of the proposed queries are extracted from different chunks of the suspicious document. It deepens the search for those specific parts and aims for retrieval of more text-related documents. They also suggest to detect suspicious passages of the document by evaluating textual characteristics with intrinsic plagiarism detection methods and deepen the search in those passages. Queries constructed from small text parts of the source document can be characterized as phrasal queries and are usually longer (up to 10 words).

The proposed methodology is applicable in a real-world document retrieval system and it also performed best in terms of the total system workload, while maintaining a good retrieval performance in PAN 2012 competition on plagiarism detection [7]. In 2013 this methodology was improved with enhanced download control and the third type of queries was changed from header based to paragraph based queries. The main idea remains the same, though [9].

This methodology is also expected to perform better in real-world scenarios while utilizing modern search engines, than in PAN competition environment. It is because the search engine used during the competition did not support phrasal search, which influenced a significant part of queries of the proposed methodology.

It also scales up to a single query per document. The first query is constructed from the keywords which obtained the highest score. In the next step, the keywords based queries are formulated from the consecutive extracted keywords sorted by their score up to the score threshold or a up to the preset maximum number of queries of this type. After that, the search can be deepened by phrasal types of prepared queries.

A multilingual search can be accomplished by a query translation, which is generally easier than the full sentence translation. It is sufficient to translate all the query terms consecutively, especially if the query is constructed from keywords only. Translation can be done by the dictionary associations. Still, a quality disambiguation may pose a problem for successful translations. It is therefore better to use words in their canonical forms in keywords based queries, since the search engine will not distinguish between different forms of one word. A different situation is at phrasal queries, they should remain unchanged, since the modern search engines will attempt to appraise the meaning of the search, like for example the new Google's searching algorithm called Hummingbird.

B. Searching the Internet

During this task the prepared queries are submitted to the search engine. A search control should be implemented in order to minimize the total number of submitted queries. As a consequence of the limited query budget, the queries should be processed stepwise and search results should be evaluated; in terms of a basic feedback for the search controller; after each query. During the searching the search controller can reschedule the submission of prepared queries, which may include query skipping or reformulation. The basic search control represents submission of queries according to their priority up to a specific number of submissions. Haggag and El-Beltagy [10] check subsequent queries against all previously downloaded documents, which were downloaded based on an analysis of one suspicious document, through a simple token matching. They skip queries which show 60% or higher tokens match. Suchomel et al. [9] submit document global keywords based queries at first. Next, according to obtained similarities between the suspicious and retrieved document, they skip queries covering by their position the portions of the suspicious document, which were already mapped to the source.

Issues to consider during this task include: i) the search control; ii) the feedback from retrieved documents; iii) the storage of query records and retrieved results with results filtering. The storage of the query record prevents from executing of the same queries if prepared for different input documents. Also the date and time of each query execution can be decisive for the eventual query resubmission.

C. Downloading the Results

Real Internet searches include many types of documents, such as textual rich formats or multimedia formats. A plaintext needs to be extracted from retrieved documents, therefore only plaintext convertable documents should be downloaded. Downloads can pose a huge bandwidth and disk storage demands. In the real-world, there is little information known prior to the download, which influences download decision making. The type and size of the document can usually be determined from headers of Internet documents prior to the full document download, which still pose a header request to a web server. This leads to having to leave the decision making about the quality of the search results to the post-processing phase.

The web is a very wild and volatile environment, thus more emphasis must be put on the robustness of the downloading subsystem. Addition to standard timeouts, other techniques should also be considered. For example, more attempts to

TABLE I. PERFORMANCE OF VARIOUS WEB DOCUMENT DOWNLOAD TECHNIQUES.

Δt [min.]	domain ordering	no domain ordering
one process	1:49	1:44
db dedicated thread	1:56	1:48
4 download dedicated threads	0:31	0:34

download a document should be carried out if the previous one was unsuccessful. A maximum file size limit should be set for *html* files, otherwise the downloader can get stuck in endless web files. On the other hand, certain types of documents (like *pdf* files) have to be downloaded completely in order to extract the text from them properly. The request for header can help to set the maximum file size of such types of documents. Unfortunately, Internet files headers do not always provide veracious information.

Various metadata of downloaded documents need to be stored permanently. The date and time of the last download and the original internet document are needed also to be able to ascertain plagiarism.

Assuming database driven data exchange, the time demands of the hypothetical simple downloader consist of database operations; establishing connections to the target web server; downloading the actual data; and saving the data. Our tests show that beneficial speedup of download is favourable by the process parallelization only. The connection establishment can be sped up for example by the DNS record caching. Not many crawler-like optimization can be performed, since the search results generally contain various web sites. All downloads can be sorted according to the hosting site in order to use cached DNS records. Table I presents averaged times of 2 passes of downloads of 137 different Internet documents obtained from searches based on different queries. 91 of those 137 documents were downloadable at the time of the tests. Others ended with various HTTP errors among which the HTTP 404 (Not Found) was the most abundant. The tests ran in homogenous network and hardware conditions. The *domain ordering* column shows times, when the downloader tried to optimize Internet requests by accessing the same sites consecutively. The times shown in the second column were obtained while accessing Internet documents in the order as they were added to the database for download by the search algorithm. The second data line of the table shows times of the changed downloader differing from the first line of the table in a threading approach. A dedicated thread was used for downloading the documents, and database operations together with the other logic remained in the main thread of the programme. The third line represents multi-threaded download process—4 threads were used for the documents download and the main thread remained unchanged. The results show that the database operations are negligible when compared to time demands for downloads. Also, the site ordering will not probably pay off, since it burdens the algorithm with the additional sorting. The downloads are certainly the most time demanding operations, but they are also easily parallelizable. The third row of the table shows that n additional threads will almost lineary n times decrease the total time for download.

D. The Results Post-processing

In the post-processing phase, the task is to evaluate the quality of the downloaded document and if the quality is sufficient, to pass the document to the indexer. Only among the indexed documents the similarities can be calculated, which represents the subsequent stage of the plagiarism detection process.

A plaintext needs to be extracted from every downloaded document in order to evaluate similarities. Several information must be obtained before the actual text conversion, which includes: file type identification—the file type given by a *http* response header cannot be trusted, therefore file type must be identified by MIME detection tools.

Sometimes the file type must be identified according to the file extentions if any, or according to other heuristics. The encoding of the file needs to be determined in pursuance of the correct tokenization and indexing. The language of the document should also be known supposing appropriate lingual classification.

A modern source retrieval system should be able to convert the most common web file formats which include: *html* and other markup languages document formats; Microsoft Office family file formats; Open Office file formats; and probably the most common *pdf* files.

From the nature of MS Office and Open Office formats it follows, that the plaintext conversion is possible, because those files carry a text source information. It is hidden in the proprietary structure of the files. Standard tools for those format creation can extract the text, the extraction must be fully automated, though. Not all documents are generally convertable, since those formats allow to lock or create password protected text. There are also publicly available programme modules and extraction tools for Open Office[3] and MS Office[4] documents.

The plaintext conversion of *pdf* file is more complicated since the *pdf* is not an easily editable file format. There are many tools for creating various versions of *pdf* files, thus the issue of the *pdf* text conversion is far from smooth and errorless process. Firstly, the standard methods of text extraction from the *pdf* text layer together with the text correctness should be performed. If the text is not well-formed or if the extraction fails, other conversion methods should be applied. Further possibility of text extraction is to pass the document to an OCR recognition[5]. The check for text correctness is important even if the extraction from *pdf* layer was errorless. Typically non ASCI characters can be damaged and a profile of the text must align with any of the supported language. If a plagiarist obfuscates plagiarism by braking the textual layer and keeping the document to display correctly, the use of an OCR is also inevitable. For example a student creates a plagiarized text and replaces every space in the text with any letter in white color, which will not be seen by a human reader. The text will have the character of a single huge meaningless word for the text extractor. The use of OCR will recreate the text correctly, since it looks at the document in the same way as the user does.

[3]http://freecode.com/projects/oo2txt

[4]http://www.adelton.com/perl/Docserver/

[5]http://en.wikipedia.org/wiki/Optical_character_recognition

The issue of text extraction from *html* family files is even more complicated. The majority of web pages include together with the main content also so-called boilerplate content. The boilerplate content is for example a navigation link, advertisement, header or footer. It is a meaningless content for document comparison. Having indexed all unchanged text from web pages would result in many false positives in evaluation of the document similarities. Internet documents would be spoiled with repeated parts of text, which do not carry needed information. For example, every page from Wikipedia contains the same footer. Therefore, the text extraction from *html* files should be accompanied by a boilerplate removal, for example by means of context based approaches to identify and remove the boilerplate [11].

For example, Masaryk University runs proprietary servers for plaintext conversions inside the network document storage of the university Information System. It includes dedicated client-server network hosted applications for MS Office, Open Office and *pdf*, including OCR, documents to plaintext conversion.

The plaintext conversion is generally very computational demanding, it also takes a lot of tools and technologies to convert many document types. From all, the *pdf* files are possibly the most computation demanding to convert and except for *html* files the *pdf* files represent the most preferable file format to be published on the Internet.

Having extracted a plaintext from the downloaded documents allows for subsequent document evaluation. A decision whether to actually index the document for the plagiarism detection must be made. Straightforward evaluation is to compare the retrieved document d_{ret} with the suspicious document d_{plg} for a document similarity. However, considering the real-world plagiarism detection for many input documents, the candidate document retrieval system can retrieve a theme bounded document based on a query created from a certain suspicious document, but the retrieved document could serve as the source for plagiarizing another document, which is also in the anti-plagiarim system database. Then the retrieved document is valuable, but evaluating it with only the document from which the query was constructed can result in no similarities. In such situations all retrieved good quality texts should be also indexed for all documents similarities.

The text extraction can also be parallelized on the document level, like downloading, but especially the optical recognition can still be very time consuming. It may currently take tens of minutes to complete, based on used hardware.

IV. Conclusion

This paper discussed the main points of the candidate document retrieval system architecture. The candidate document retrieval is an unexpendable part of a modern anti-plagiarism detection system. Such a system should firstly retrieve potential sources of plagiarism from the web. Consequently it evaluates document similarities among all documents which the system operates with together with the newly retrieved documents.

A user is usually provided with the percentual portion of document similarities between pairs of similar documents. The overall percentage can also be provided. However, the system do not decide about plagiarism, it only selects similar passages. The issue of plagiarism is far more complicated. It is always up to a user judge to decide, whether the given text is plagiarized or not. The system simplifies the tedious work of finding the sources of similar texts. For example a page in a thesis can be copied from the other text source, which would not be considered as a cheat if cited correctly.

There are also many types of plagiarism, for example paraphrasing, copying the structure of the document, copying the results or copying the texts. The overall quality of a plagiarism system can be evaluated using measurement based on what reused text obfuscation it can detect.

This paper summarized experience from the real-world operation of the anti-plagiarism system used at Masaryk University. It provided ideas, concrete methods, and concepts for a candidate document retrieval system construction. It also discussed concepts used on PAN competition on plagiarism detection. The research behind the competition provides additional topic-related information.

Acknowledgment

The authors would like to thank to the Information System of Masaryk University for creating an opportunity to improve the plagiarism issue in Europe.

References

[1] Jan KASPRZAK, Michal BRANDEJS, Miroslav KŘIPAČ and Pavel ŠMERK. *Distributed System for Discovering Similar Documents*. Brno: Faculty of Informatics, Masaryk University, 2008. 14 pg.

[2] Martin POTTHAST. *Technologies for Reusing Text from the Web*. Dissertation, Bauhaus-Universität Weimar, December 2011.

[3] Dennis FETTERLY, Mark MANASSE, and Marc NAJORK. *On the Evolution of Clusters of Near-Duplicate Web Pages*. In proceedings of the 1st Latin American Web Congress, LA-WEB 2003. IEEE, 2003.

[4] Martin POTTHAST, Tim GOLLUB, Matthias HAGEN, Martin TIPPMANN, Johannes KIESEL, Efstathios STAMATATOS, Paolo ROSSO, and Benno STEIS. *Overview of the 5th International Competition on Plagiarism Detection*. CLEF 2013 Evaluation Labs and Workshop, 2013

[5] Šimon SUCHOMEL. *Systems for Online Plagiarism Detection*. Thesis, Faculty of Informatics, Masaryk University, Brno 2012

[6] Radim ŘEHŮŘEK, and Milan KOLKUS. *Language Identification on the Web: Extending the Dictionary Method*. In *Computational Linguistics and Intelligent Text Processing, 10th International Conference, CICLing 2009, Proceedings.*. Mexico City, Mexico: Springer-Verlag, 2009., 12 pg. ISBN 978-3-642-00381-3.

[7] Martin POTTHAST, Tim GOLLUB, Mathias HAGEN, Johannes KIESSEL, Maximilian MICHEL, Arnd OBERLÄDNER, Martin TIPPMANN, Alberto BARRÓN-CEDEÑO, Parth GUPTA, Paolo ROSSO, and Benno STEIN. *Overview of the 4th International Competition on Plagiarism Detection*. CLEF 2012, Evaluation Labs and Workshop, 2012

[8] Šimon SUCHOMEL, Jan KASPRZAK, and Michal BRANDEJS. *Three Way Search Engine Queries with Multi-feature Document Comparison for Plagiarism Detection*. CLEF (Online Working Notes/Labs/Workshop), Rome, 2012

[9] Šimon SUCHOMEL, Jan KASPRZAK, and Michal BRANDEJS. *Diverse Queries and Feature Type Selection for Plagiarism Discovery*. CLEF 2013 Evaluation Labs and Workshop, Valencia, 2013

[10] Osama HAGGAG, and Smhaa EL-BELTAGY. *Plagiarism Candidate Retrieval Using Selective Query Formulation and Discriminative Query Scoring*. Notebook for PAN at CLEF 2013, Valencia, 2013

[11] Pomikálek, J.: *justext: Heuristic based boilerplate removal tool*, available at Google code, online *http://code.google.com/p/justext/*, (2013)