KOPI Protection instead of Copy Protection

László Kovács, Máté Pataki
MTA SZTAKI
{laszlo.kovacs;mate.pataki}@sztaki.hu

Abstract

This paper describes how a document can be protected against plagiarism without the need to use heavy copy protection systems, as despite they are hard to break, they make the lives of legitimate users very difficult.

1. Introduction

Access restriction and detection of plagiarism are two ways of protecting documents. It is really hard to protect a document from unauthorized copying while ensuring that authorized people can access it easily. Most copy protection mechanisms can easily be broken, with ready-made tools freely available on the Internet. Others are harder to break, but the legitimate use is also made hard, because one has to install extra programs or tools, which most probably won't work on all systems, or in many cases are not worth to deal with because the content itself may not be that important. People with disabilities – who are using special tools to access the web – most probably won't be able to access these documents.

Plagiarism detection systems won't protect your document from illegal copying, but when used by a wide range of people, they can prevent others from presenting it as their own work. This protection is twofold: On the one hand, if a work is copied, this system can tell whom it was copied from. On the other hand, if the existence and use of the system is known to everybody, they most probably won't risk being exposed as a plagiarist.

2. Copy Detection Systems

The existing copy- and plagiarism-detection systems can be categorized as follows:

2.1. Watermark or Checksum

Many systems use watermarks placed in the formatting of the text, or checksums for the whole document. In most cases, when speaking of text documents, watermarks can be easily and automatically removed. Checksums can be easily deceived with some small alterations in the text and are **not good at detecting smaller overlapping** parts.

2.2. Authorship Attribution

A totally different approach for plagiarism search is authorship attribution and identification, which is a grammar style analysis, it can be used to detect whether two or more texts are written by the same author or not. However, it has two big disadvantages: The first is that in most cases the algorithm used for linguistic analysis is **language-dependent** so it has to be developed for each language used. The other problem is that it needs more texts from the same author, which are not available in many cases.

2.3. Open Search Engines

Meta-systems that use search engines (like Google) for plagiarism search [9], have good results in detecting works copied from the Internet, yet in most cases **the documents cannot be found on the Internet**. Only a few people put their work or theses on the Web, and the access to most digital libraries and collections are also restricted (deep web). Authors and publishers only seldom publish their books electronically, they are paid after each book sold, and would lose a lot of money if they were made available on the Internet.

2.4. Text Comparison

Comparing **two documents to each other** is an easy task, the best known word processor, Microsoft Word has also this feature built in, but even by

comparing 10 documents to each other this would take 45 comparisons, not to speak of comparing a document to a few thousand others.

2.5. Questionnaire

To detect plagiarism, there are programs that generate a questionnaire from the original text, where a given number of words are removed, and the gaps are to be filled in by the author. Most probably he will use the same words and expressions found in the text [4]. This solution could perhaps work at a school or university; however, a student is directly accused of plagiarism and has to fill in the questionnaire. This **takes a lot of time from both the student and the professor**, and more importantly the risk of false accusations may be too high.

2.6. Security through Obscurity

There are also a couple of commercial systems, like Plagiarism Finder [8] and EVE Plagiarism Detection System [3], but because of their kind, their working mechanisms and the **algorithms they use are unknown** (security by obscurity). This makes them hard to rely on, as it is not known how they can be deceived and what conditions the documents must fulfill to be suitable for plagiarism search. Moreover, many people and institutions cannot afford paying for such services.

3. KOPI Protection

The KOPI Online Plagiarism Search and Information Portal [5] is a free similarity search engine developed by the Computer and Automation Research Institute of the Hungarian Academy of Sciences [7].

The users can upload documents to the KOPI Portal; the documents will be converted, chunked and uploaded into a database. KOPI offers three plagiarism search services to the users: comparing documents to each other, comparing documents to the users own documents, and comparing them to all the documents uploaded to the system. This last one is the real KOPI protection for the uploaded documents, because plagiarizing from these documents can be easily detected by any user of the system.

The development of the portal began early 2003 and it is open to the public since the end of May 2004. The database of the documents in the system is getting larger with the increasing number of users, and the more documents the system has, the more effective the detection will be.

Contrary to the above described systems, the algorithms used by KOPI have the following advantages (the problems are addressed in the same order as in the previous section):

3.1. Detects Partial Overlapping

The heart of the similarity search engine is the chunking method used to chunk the given text into smaller pieces, which makes it possible to find smaller overlapping parts in the texts.

For chunking KOPI uses a new method [6] which is the marriage of word chunking and overlapping word chunking [1, 10, 11]. This new algorithm provides a fast and accurate search, while keeping the size of the database small.

3.2. Language Independent

This chunking method is independent of the language or style of the document. The only information it needs is which characters in the text are letters and which ones are not (symbols, numbers etc.), because the latter ones are not used by the chunk creation: they are discarded by the converter.

3.3. Can Protect Proprietary Documents

To make the system faster, the chunks are converted into numbers, and when documents are compared to each other only these chunks or their so-called compressed fingerprints are to be compared, to determine how many common parts the documents have. This has one other big advantage besides being faster, because KOPI uses the one way hash algorithm MD5, the document cannot practically be reconstructed from the fingerprints.

With this method KOPI is able to compare documents to each other, even if the texts are unknown to the system, it only needs the fingerprints. This could be used for example by a publisher, who could upload all his books into a protected database – where only the fingerprints are stored – so he can protect his materials without having the digital copies made public.

3.4. One to Many Comparison

By uploading all the fingerprints into a database a fast indexed search can be performed, which results in all the documents containing that particular chunk. Even with a couple of million documents in its database this method returns the result almost instantly.

For user visualization or a more precise result it is absolutely imaginable to use a one-to-one comparison, but only on the result set, which in most cases would consist of just a couple of documents.

3.5. Without User Intervention

As described above KOPI does not need any user intervention to process the documents, it could be even used in a way, that it automatically harvests the documents from given institutions, digital libraries and if a given ratio of similarity between documents is exceeded, it sends out a warning to the owner with a link to the similar documents.

3.6. Known Algorithm

The algorithm, used by KOPI for chunking, fingerprinting and search, has been published [13] and made public on the homepage, so it is available to anybody. Even with the code published, it is really hard to deceive the system and the alteration of the text cannot be done automatically.

KOPI uses a new method, a combination of word chunking and overlapping word chunking. Word chunking with parameter n means, that a new chunk is begun at every n^{th} word. The number of chunks within a document will be (where w_k is the number of words in document k):

$$ch_k = \left\lfloor \frac{w_k}{n} \right\rfloor$$

Two documents chunked with word chunking cannot be compared, because the chunks differ depending on where the chunking begun (phase shift problem). Overlapping word chunking solves this problem by beginning a new n-word-long chunk at each word (thus making each chunk overlap with the next one by n-1 words). This method produces about n times as many chunks as the normal word chunking:

$$ch_k = w_k - n - 1$$

In this new method, the documents uploaded to the database are chunked with word chunking, and the ones compared to it are chunked with overlapping word chunking. So the size of the database remains small, and the phase shift problem is also avoided. With this method, the documents in the database can not be compared to each other, because they could be phase shifted, but a document chunked with the same parameter but with overlapping word chunking can already be compared to the documents in the database.

Here are some examples, how different alterations to a text affect the results of the plagiarism search (each number represents one word, parameter n = 5):

- This is a 22 words long document :
  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
- This is the document chunked with word chunking (| represents the chunk border):
  1 2 3 4 5 | 6 7 8 9 10 | 11 12 13 14 15 | 16 17 18 19 20 | ~~21 22~~
- Overlapping word chunking would be
  1 2 3 4 5 | 2 3 4 5 6 | 3 4 5 6 7 | 4 5 6 7 8 | 5 6 7 8 9 | …

The possible differences could be:

- Deleting word(s) from the document in the database (phase shift):
  1 2 ~~3~~ 4 5 6 | 7 8 9 10 11 | 12 13 14 15 16 | 17 18 19 20 21
- Adding word(s) to the document in the database (phase shift):
  1 2 3 3b 4 | 5 6 7 8 9 | 10 11 12 13 14 | 15 16 17 18 19
- Altering word(s) in the document in the database:
  1 2 3b 4 5 | 6 7 8 9 10 | 11 12 13 14 15 | 16 17 18 19 20

The underlined chunks can still be found within the chunks of same document when chunked with overlapping word chunking. In each case only those chunks are affected in which the altering happened. Changing one word (delete, add or alter) can cause only one chunk to differ, two words can cause two chunks do differ in the worst case, and so on. The only possibility not to have same chunks is to alter every n^{th} word in the document.

To succeed in deceiving the system, which uses, for example, a parameter 10 for the word chunking, at least every 10^{th} word should be changed in the whole document. Paraphrasing a 50 page thesis this way would be in itself a great accomplishment, and there is still a risk that a system maybe uses a smaller parameter, and in that case the whole plagiarism would be discovered.

4. Distributed KOPI System

Currently we work on the new version of the KOPI system, which will be a distributed document store with the ability of identifying similarities between documents, independently from the location where they are stored.

Each participating institute (library, university, conference organizer…) will be able to install a

standalone KOPI system (a node). These systems would then be connected to each other with a standardized communicating protocol, and so a plagiarism search started at one place would go through all relevant nodes and look for similarities. This distributed search process will be transparent, thus the whole system would appear to the end-user as one. A similar architecture is used in StreamOnTheFly, the distributed archive network for radio programs [12], where the connection scheme used by the system is transparent, thus each node is equivalent to the end-user.

Larger institutions having a collection or document store with their own plagiarism search engine could also participate, without installing a new system, by implementing the standardized communicating protocol used by the KOPI nodes to communicate with each other.

5. User feedback

The system has, at the date of this writing, more than 2000 registered users, some of them have more than 100 uploaded documents, but most users uploaded only a couple of documents, like their theses, articles or scientific papers. We got a large number of feedbacks from our users, which we try to take into consideration when developing the next version of KOPI.

The most requested features are the visualization of the results and the ability to upload documents and to start plagiarism searches with an automated robot. This last one would be used by the owners of existing data sources to upload their content into the KOPI system.

6. Conclusion

With such a system in use, digital content owners could much more freely distribute their (KOPI protected) documents, which, in most cases, is good for them, for example if more people read a paper then most probably more articles will refer to it, which consequently will result in a higher impact factor.

The public and professionals could access documents easier, read the content they like. Even if some would succeed in plagiarizing a part of a work, having a lot of publicly available documents is a much bigger gain for the community than having a lot of documents that nobody reads because of the difficulties in accessing them.

7. References

[1] BAEZA-YATES Ricardo, RIBEIRO-NETO Berthier, *Modern Information Retrieval*, Addison Wesley, 1999.

[2] *Department of Distributed Systems*, http://dsd.sztaki.hu

[3] *EVE Plagiarism Detection System*, http://www.canexus.com

[4] *Glatt Plagiarism Screening Program*, http://www.plagiarism.com/

[5] *KOPI Online Plagiarism Search and Information Portal*, http://kopi.sztaki.hu

[6] MONOSTORI Krisztián, FINKEL Raphael, ZASLAVSKY Arkady, HODÁSZ Gábor, PATAKI Máté, *Comparison of Overlap Detection Techniques*, Computer Science ICCS 2002 International Conference, Amsterdam, April 21-24, 2002, Proceedings, Part I., Springer, 2002.

[7] *Computer and Automation Research Institute of the Hungarian Academy of Sciences*, http://www.sztaki.hu

[8] *Plagiarism Finder*, http://www.m4-software.de/en-index.htm

[9] *Plagiarism Search V 1.0.0*, http://baltic.cse.msu.edu/heynige1/Search/

[10] SHIVAKUMAR Narayanan, GARCIA-MOLINA Hector, *SCAM: A Copy Detection Mechanism for Digital Documents*, Department of Computer Science, Stanford University, CA 94305-2140, Proceedings of 2nd International Conference in Theory and Practice of Digital Libraries (DL'95), Austin, Texas, 1995.

[11] SHIVAKUMAR Narayanan, GARCIA-MOLINA Hector, *Building a Scalable and Accurate Copy Detection Mechanism*, Department of Computer Science, Stanford, CA 94305, 1996.

[12] *Stream on the Fly*, http://radio.sztaki.hu

[13] Máté Pataki, *Plagiarism Detection and Document Chunking Methods*, The Twelfth International World Wide Web Conference, Budapest, 2003, http://www2003.org/cdrom/papers/poster/p186/p186-Pataki.html