Scientific Journal of Riga Technical University
Computer Science. Information Technology and Management Science

2010
Volume 44

Research into Plagiarism Cases and Plagiarism Detection Methods

Maria Kashkur, *Riga Technical University*, Serge Parshutin, *Riga Technical University*,
Arkady Borisov, *Riga Technical University*

***Abstract* –In the age of information technology intellectual property becomes especially valuable. This is one of the causes why the cases of the plagiarism appear more frequently in all vital sectors. Due to that, there is a growing need for different instruments for the protection and verification of copyright for finding plagiarism. Before checking the document for plagiarism, reviewing algorithms and approaches for searching plagiarism, you must know and understand what constitutes the plagiarism. Therefore, in this paper we discuss definitions of plagiarism itself and give a look into most important types of plagiarism. As also the paper describes the most common plagiarism detection systems, methods used in those systems, and provides a description of several programs designed to compare documents and detect plagiarism.**

***Keywords* –algorithms for searching plagiarism, plagiarism, plagiarism detection systems, programs for plagiarism detection**

I. Introduction

Today plagiarism is found in almost all fields of human activity: literature, science, music, design, etc. That is why nowadays a lot of attention is given to identification and detection of plagiarism. However depending on the scope of activities in which plagiarism occurs, definition of plagiarism, its boundary and detection methods change too. For this reason the first part of this paper is devoted to the general definition of plagiarism and the types of student plagiarism in particular.

The second part of this paper presentsa classification of methods of plagiarism detection as well as outlines the most popular systems for plagiarism detection.

The next part of the paper is devoted to plagiarism detection in the program code. This part describes the most popular and effective algorithms used by systems to detect plagiarism in the code of programs.

The last partprovides examples of the programs accessible on the Internet;descriptions of both English and Russian programs are given.

II. Plagiarism

A. Definition

According to the Collins Dictionary of the English Language, plagiarism is ‘the act of plagiarising’, which means ‘to appropriate (ideas, passages, etc) from (another work or author)’[7].

Barnhart traces the etymology of the word plagiarism (‘literary theft’), from the earlier English word plagiary (‘one who wrongfully takes another’s words or ideas’), derived from the Latin *plagarius*(‘kidnapper, seducer, plunderer, literary thief’), from *plagium*(kidnapping) from *plaga*(snare, net) [2].

The verb “plagiarise” is defined in the *Shorter Oxford* as follows: ‘Take and use as one’s own (the thoughts, writings, inventions, etc., of another person); copy (literary work, ideas, etc.) improperly or without acknowledgement; pass off the thoughts, work, etc. of (another person) as one’s own’ [3].

However the border-line between plagiarism and research is surprisingly murky. After all, advanced research is only possible by using the already existing information.

B. Student Plagiarism

The core business of the knowledge industry is handling information and ideas from different sources, so there is inevitably great scope for plagiarism within the academic world. Here plagiarism occurs in a variety of settings, including collaboration or cooperation between students working together, unattributed use of other people’s writings by undergraduates, copying of graduate students’ work by supervisors or other members of academic staff and taking credit in research grant applications for work done by someone else. Plagiarism is the old problem in the highest education that was aggravated with the advent of the Internet [11].

However, it should be noted that when the student enters the university, he might be not informed about plagiarism or how to overcome it. Therefore, plagiarists share on types that define seriousness of their actions. Three types of plagiarists [8] can be identified:

- Accidental: lack of understanding, the student was unaware that it was wrong thus demonstrates poor academic practice;
- Opportunistic: aware of this being ‘wrong’ but does so due to some source of pressure or in the belief that it will result in higher marks;
- Committed: intentional (pre-meditated) cheating via misrepresentation.

Researches show that the majority of plagiarism carried out by students who don't understand the academic requirements, therefore the majority of students are accidental plagiarists.

III. Plagiarism Detection Methods

With the growing popularity of the Internet, many and various documents are available free. People can easily search for the required documents and make their copy instead of writing the documents themselves. These practices have an enormous impact on the education system. In addition, the problem is also supported by many servers, which offer a wide range of various topics.

Document protection techniques, which disable copy-pasteoperations and printing, are insufficient. A large database of existing documents is a better solution. The main idea of this protection rests in psychology because every plagiarized document can be easily identified when compared to the database. Most of the plagiarists only copy a part of a document and do not try to hide this activity. This is an evident case of plagiarism that can be easily identified because a large continuous text is copied. The consistent plagiarists copy some parts of sentences and sometimes exchange several words to cause confusion. This type of plagiarism is difficult to determine[4].

A. Classification of Plagiarism Detection Methods

The most general classification of copy detection methods is to free textor source code. The classification in Table I is intended just for free text plagiarism detection methods [4].

TABLE I
CLASSIFICATION OF FREE TEXT PLAGIARISM DETECTION METHODS

Type of classification		Description
Complexity of the used method	Superficial	The metrics is computed without any knowledge of the linguistic rules or a document structure.
	Structural	The metrics is computed with a partial understanding of documents, e.g. words are converted into their linguistic root, or replaced by a synonym.
Number of documents processed by the used method	Singular	A single document is processed to compute the metrics. Several Singular metrics can be employed to calculate how similar the documents are.
	Paired	Two documents are processed together to compute the metrics.
	Multidimensional	*N* documents from a corpus are processed together to compute the metrics.
	Corpal	All documents contained in a corpus are processed together to compute the metrics.

Almost all current free text or source code copy detection systems are Paired or Singular. This means that every document must be compared with any other possible documents to analyze the whole corpus. Therefore, Paired and Singular methods are suitable for seeking some possibly plagiarized documents, which are related to the concrete tested document. However, none of both methods is able to perform criss-cross comparison at a time.

The older systems, such as COPS or SCAM working on the term frequency, are purely Superficial. The current systems, which employ N-grams, are also rather Superficial than Structural. The reason is too time-consuming analysis of sentences whose grammar includes many linguistic rules. Fortunately, some modern approaches from the other fields of nature language processing give us new possibilities of modification and improving the current plagiarism detection methods [4].

B. Plagiarism Detection Systems

COPS (COpy Protection System) is a prototype of copy detection system developed at Stanford. The system sketches the common approach to plagiarism detection based on unit chunk hashing. A chunk is a sequence of consecutive units; a document may be divided into chunks in a number of ways, as chunks are allowed overlap or not cover the document entirely. A method of selecting chunks from a document is called a chunking strategy.

A system following the COPS methodology consists of two main functions. One which obtains chunks from a document via a selected chunking strategy and stores hashes of these chunks into a hash table. The second function is a function that realizes the violation test [14].

SCAM (Stanford Copy Analysis Mechanism) is a plagiarism detectionsystem developed at Stanford. Unlike COPS, it operates by assuminga vector space model for the registered documents. The difference to otherInformation Retrieval (IR) systems is in using a new similarity measure.This measure was developed to more accurately characterize copy overlap,while traditional IR systems look for semantic similarity [14].

The SCAM system, as well as COPS, is classified as paired and superficial system. It is tuned to discover small overlaps, which results in many false positives when word distributions are similar but the texts are still different [4, 15].

MOSS (Measure of Software Similarity) was developed at UC Berkeley in 1994. It is a free available plagiarism detection system for academic usage only. MOSS supports a lot of different programming languages and two platforms, UNIX and Windows. As the name suggests, its primary purpose is to detect programming assignment plagiarism and is used mostly by programming lecturers from computer science and engineering departments, although it also supports other text input types apart from code. Its aim is to detect the standard attempt at cheating, which consists of changing variable names, I/O prompts, statement spacing and comments. However, even this ’dumb’ attempt is enough to fool a simple file diff, rendering a careful manual comparison necessary. MOSS overcomes this problem by offering a script which, whenever run, emails a selected batch of programs to a Berkeley server for analysis. Response is usually obtained within the same day and consists of a set of html documents comprising a report. The report highlights pairs of programs that exhibit suspiciously high mutual similarity [5, 14].

YAP (Yet Another Plague) – token-based system that treats programs as a sequence of strings. The last version of

YAP (YAP3) introduces a totally novel algorithm to face the presence of block-moves in programs. Namely: the Running-Karp-Rabin Greedy-String-Tiling algorithm.Its aim is to find a maximal set of common contiguous substrings as long as possible, each of which does not cover a token already used in some other substrings [9, 14].

MDR (Match Detect Retrieval)is a prototype of a system capable of detecting overlapping documents.The approach used in MatchDetectReveal system avoids using a hash-function due to concern about hash collisions.The basic matching components uses string-matching algorithm based on suffix trees to identify the overlap. The algorithm used for building the suffix tree from the query document is a modification of Ukkonen's algorithm. As such, this system is only capable of locating exact copies of document parts. Once the suffix tree is built, all registered documents are compared against it [14, 15].

SID (Software Integrity Diagnosis, or alternatively Share Information Distance) is a system developed at University of California, Santa Barbara. The authors noted that plagiarism detection systems like MOSS and the YAP proceed by tokenizing the input sequence and then comparing the token sequences. A basic problem underlying the second phase is how to measure similarity of a pair of token sequences. If the metric is inappropriate, plagiarism may go unnoticed. If it is well-defined andnot universal, it can always be cheated. For example, the MOSS designers refuse to publish details of their algorithm openly on the website, fearing the cheaters would quickly learn to beat the system. Such an approach to security through withholding static information is not a good design choice.

Authors of SID therefore take a different approach, where they consider the sequence similarity from an information-theoretic perspective. The metric that measures the amount of information between two sequences (not necessarily program token sequences – many applications are imaginable, including DNA sequences or text documents) is based on Kolmogorov complexity and is universal. The universality guarantees that if there is similarity under any computable similarity metric, this metric will detect it [14].

CHECK is another plagiarism detection system that uses document structure to build a hierarchal representation of the document. Each document is viewed at multiple abstraction levels, which include the document itself, its section, subsection, subsubsections and finally paragraphs.For each level, the set of relevant keywords is extracted. Keyword extraction uses keywords frequency as well as italics and boldface formatting information to assign weights to keywords. At query time, the nodes of the query abstraction and that of the referential document are traversed, starting with the root node. Similarity is computed as cosine measure of the two node's keyword weight vectors. In this respect, CHECK follows the traditional IR approach. If the similarity exceeds a given threshold, the two node's children are processed recursively. The purpose of this step is to obtain pairs of document segments (represented by the lowest level of abstraction, i.e. paragraphs) that are similar to each other. The final step is to analyze these similar pairs of paragraphs sentence-by-sentence and report detected copies [14].

IV. Plagiarism Detection in the Source Code

A. Types of Code Interpretation

To use systems of plagiarism detection in the codes of programs, the source code sometimes is represented differently. For example, early systems of plagiarism detection represented the program as a point in *n*-dimensional space of the natural numbers, whose*i*-th coordinate is the quantitative characteristic of any property (attribute) of all program. For example, it can be average length of code line, a number of variables used, a number of operators of branching etc. If points of two programs are located side by side, one of them might be copied from another [5, 10].

Other systems consider the source code of programs such that it was in the beginning. For example, so do systems which work with the code the same as with the normal text. But they are extremely ineffective, as renaming of functions and variables or insignificant changes in the code become a serious obstacle for their correct operation.

Sometimes the parameterized representation of the code is used. For example, names of functions and variables are replaced at the first meeting in the code to zero, and at subsequent to the distance to the previous position [1].

One more type of interpretation of the code is tokenization. This interpretation is based on saving the essential information about the program and ignoring the surface information. It should be noted that tokenization process depends on the programming language used in the source code [13].

B. Plagiarism Detection Algorithms

One of the plagiarism detection algorithms is the **Heskel's algorithm** which is based on the sharing string on k-grams that is k-length substrings, and search for matches, focusing already on them. Nevertheless, this algorithm has a principal lack. In the big programs is a very small number of unique k-grams. Therefore many coincidence that don't contain such k-grams, will be ignored [5,10].

Another algorithm uses the**method of local alignment of strings** which has been developed for determination the similarity of strings of DNA (deoxyribonucleic acid). To use this method, two programs should be represented as a string of characters. Alignment of strings is obtained by inserting spaces in the strings so that their length became equal. It should be noted that there are a large number of different alignments of two strings [21].

It is necessary to consider a heuristic **algorithm of greedy string tiling**. It receives the input two strings of characters, and the output set of their common non-overlapping substrings, which is close to optimal. Substring is appearing in this set, called tile. There are quite a large number of optimizations of this algorithm which considerably increaseits high-speed performance. There is also a more radical improvement using the algorithm of Karp-Rabin of substrings search in the string. The main advantage of the algorithm can assume that the rearrangement of the large part of the code does not affect efficiency of the algorithm [10, 13, 19].

One more interesting algorithm is based on **Kolmogorov complexity**. The basis of this algorithm is a function of distance which is based on Kolmogorov complexity. The more close function of distance of two programs to zero, the more shared information these programs contain. As Kolmogorov complexity is not computable, the heuristic approach based on the use of compression algorithm is employed[21].

In the **fingerprinting method**, tokenization programs are represented as sets of prints so that similar sets of similar programs overlap. This method allows user to implement an effective search for large databases [21].

There are few algorithms that use the interpretations as a tree or a graph. Only two of these algorithms can be performed at reasonable times. Therefore, they are rarely used in practice.

For plagiarism detection, a **method of neural networks** can also be used. Plagiarism detection can be compared to the classification task in which a set of programs can be divided into classes, in each of which there will be only copied programs. Neural networks can be represented as the black box whose input data is the known information, the output data - the information that you would like to know. For example, the input data can be the set of programs, and the output data - the inference about plagiarism presence. This method, for example, uses the detector Sherlock [22].

The summarized information is given in Table II where it can be seen what interpretation of the code and what algorithm for plagiarism detection are used by each of detectors.

TABLE II
SYSTEMS AND ALGORITHMS

Detector	Interpretation of the code	Algorithm
Accuse	N-dimensional space	Method of specificities calculating
JPlag	Tokenization	Greedy String Tiling algorithm
SID	Tokenization	The metrics based on Kolmogorov complexity, ETokenCompress
SIM	Tokenization	Alignment of strings
YAP	Tokenization	Symbol comparison, Heskel's algorithm
YAP3	Tokenization	Greedy String Tiling methodoptimization with algorithm of Karp-Rabin
MOSS	Fingerprints	Fingerprinting method
Plan-X	XML format	Usage utility XML Store
Sherlock	Neural networks	Self-organized mapping of Kohonen

V. PLAGIARISM DETECTION PROGRAMS

A. *English-Language Plagiarism Detection Programs*

Turnitin [17] is the most popular service of plagiarism detection. It was developed by group iParadigms for teachers and educational institutions and was formerly known as Plagiarism.org. The service works on a commercial basis and requires pre-registration. Professors and teachers present student's works on site and in a day or two receive the results. The system compares these materials to the indexed Web-content, large databases containing texts from so-called «collections of essays» (they are sold in Internet for usage as school or university term papers), as well as previously reported materials [5, 23].

Recently the validity of using this service has been questioned: Turnitin after check includes works in the database, getting the economic benefits without payment of compensating to the students. Despite this defect, the Joint Information Systems Committee (JISC), representing the interests of all the universities of Great Britain, recently has organized on the basis of Turnitinits own plagiarism detection service [18].

Independent application **WordCHECK** [12] exposes more students copying from each other, than borrowing of external materials. To use this application the teacher downloads all documents in the internal archive where they are compared for detection of copying within educational group. Comparing is based on the profiles of keywords (a sort of linguistic equivalent of the fingerprint) and comparison of phrases [5, 23].

Although the system, strictly speaking, isn't calculated for plagiarism detection, it will be able to do it if you include in the internal archive texts from «collections of essays» and other similar materials. Unfortunately, according to the results of the tests of this tool performed in 2001 by request of committee JISC, its functional capabilities were recognized as unsatisfactory.

Program **EVE2** [6] — commercial application that when installed on the PC finds out whether the student has not copied material from the Internet. For every work, application generates the report containing instructions of percentage of loans, list URL and the annotated copy of the work in which the copied fragments are selected by red colour. It is possible to use several file formats, including plain text and Microsoft Word documents, but the annotated copies are created only for plain text [5, 23].

In essence, this tool provides the interface to the search engine in the Internet, but such simplicity doesn't restrict its efficiency. The unique lack of EVE2, noted in report for the JISC in 2001,is that search is fulfilled only for Web-content in HTML format, but the most part of a material in the World Wide Web is stored in other formats.

Program **WCopyFind** [16] — the free tool for detection of the facts of writing off by the students, developed by Professor Lu Bloomfield at the University of Virginia. It exists at least in two versions, most convenient of which has a simple graphic interface, allowing user to download a set of documents in internal archive (like WordCHECK). The documents are compared with each other and, at will, it is possible to separate archive of files (which the professor, probably, collected several years) to compare sentences. WCopyFind presents results in HTML format and connects hyperlinks common phrases in documents to specify, who of the students copied it. Although it also can't carry out search onthe Internet, the tool is very convenient, operates quickly enough and produces quite clear results [23].

It would seem that such a variety of tools is quite enough, especially if to consider that the list far isn't full. However the use of each of them has certain restrictions, and sometimes — demands expenses.

Any tool that can't perform search in PDF has restricted value, at least, in educational institutions. It is necessary to consider that the Internet is the unique tool for the majority of students at researches on their chosen topics.

In the light of the aforementioned tools,Turnitin and EVE2 can only appear useful, but each of them has the restrictions. Turnitin is the commercial service accessible only on a subscription which, most likely, is aimed at detection of texts from «collections of essays». Such texts are suitable at high schools or on low courses of colleges, but rarely approach for theses. EVE2 is less expensive means and it is not restricted to abstracts, but its possibilities envelop only a Web-content in HTML format [23].

B. Russian-language plagiarism detection programs

Program**AdvegoPlagiatus**[25] issimple in usage, but at the same time fast enough and exact on search results utility.

For determination of authenticity of text,AdvegoPlagiatus uses two methods of the analysis of uniqueness – simple and deep. The default is simple and fast. For the second method,more time is necessary as the search is for similar phrases.

It is possible to lower check time by means of adjustments of a threshold of uniqueness of the text that is to set a value at which it is already possible to suggest the fact that article is copied.

For the analysis it is possible to enter the text into a program window as well as to use the link to the material. This program will scan the code of page and recognizes the text. After that it is necessary only to push the button «Check up uniqueness» and to wait for result, the utility will do all.

The result will be deduced on termination of the analysis. It is the detailed report on the quality of uniqueness, a coincidence level, and the sources where the text has been borrowed.

It is important that the program is completely free, requires no installation, is permanently updated and defines plagiarism in most cases.

Double Content Finder [20] –another tool for plagiarism detection. It is quite fast, works independently, without any settings on specification of query parameters, thereby doing operation very simple. Other advantages of this program are that it is free and capable to perceive the Russian language.

However availability and simplicity, in most cases, implies imperfection and Double Content Finder isn't an exception. It is possible to consider as a program lack that it is directed only on search of identical texts, without accepting in attentions already the slightest changes.

Using the program is simple enough. It is possible to enter the checked text by any of three methods: to add the text from the clipboard, to load the text file or to specify the Internet address where the article is allocated. Search will occupy some time. In the end two types of answers are possible: the text is unique or the text isn't unique. In the second case is produced the list of addresses where text copies are allocated. The amount of produced addresses is restricted 50th. That is enough to make sure that the text is not unique.

The free program**Praide Unique Content Analyser 2** [24] isthe tool for determining the uniqueness of the text. This program is more advanced and customizable than described above Double Content Finder. Flexible configuration consists of three variants of data input available: work with the link to the web-page with the checked text, with the file in TXT or HTML format, and also with the text entered manually or pasted from the clipboard. At the choice of work with the link, the utility will check up the text and allow the user to correct the contents in case of need.

The disadvantage of the program is its slowness, which can only be explained by the large number of queries performed by the utility. But finally, the result of the program is pretty good.

Praide Unique Content Analyse in the verification process for the direct use of reliable search systems, such as: Yahoo, Yandex, Mail.Ru, Google, giving a choice with which searcher to work. Users can add their own search system.

Protection IP, background mode and connection through the proxy server are very useful possibilities. Especially the background mode as the search can take some time somewhere about an hour and the utility will not hinder in this mode to work with other applications.

VI. Conclusion

Nowadays the most effective methods for plagiarism detection are considered Kolmogorov complexity based approach and the fingerprinting method. The first - because theoretically it is most difficult to hide plagiarism from it, and the second - because it is the only one of the algorithms described in the article which can scan the big databases during a reasonable period of time. Thus, the most effective of the described systems are plagiarism detectors MOSS and SID.

References

[1] **Baker B.S**. On finding duplication and near-duplication in large software systems // The Second Working Conference on Reverse Engineering, Toronto, Canada, 14-16 July, 1995. – Washington: IEEE Computer Society, 1995. – P. 86.

[2] **Barnhart, R.K**. (Ed.)Chambers Dictionary of Etymology – Edinburgh: Chambers, 1988. – 1284 p.

[3] **Brown, L**. (Ed.) The New Shorter Oxford Dictionary on Historical Principles – Oxford: Clarendon Press, 1993. – 3801 p.

[4] **Ceska Z**. The Future of Copy Detection Techniques // The 1st Young Researchers Conference on Applied Sciences, Pilsen, Czech Republic, 13 November, 2007. – Pilsen: University of West Bohemi, 2007. – P. 5-10.

[5] **Clough P**. Plagiarism in natural and programming languages: an overview of current tools and technologies // The 20th Annual ACM Symposium on Applied Computing, Santa Fe, New Mexico, 13-17 March, 2005. – New York: ACM, 2005. – P. 776-781.

[6] EVE: Plagiarism Detection System. USA, 2000. [Online].Available: http://www.canexus.com/eve/index.shtml. [Accessed: June 12, 2010].

[7] **Hanks, P**. (Ed.) Collins Dictionary of the English Language. – London: Collins, 1979. – 1690 p.
[8] **Harvey J**., **Robson S**. The Accidental Plagiarist: An institutional approach to distinguishing between a deliberate attempt to deceive and poor academic practice // 2nd International Plagiarism Conference, Gateshead, UK, 19-21 June, 2006. – Newcastle: NorthumbriaUniversity Press, 2006. – P. 16.
[9] **Jadalla A**., **Elnagar A**. PDE4Java: Plagiarism Detection Engine for Java source code: a clustering approach // International Journal of Business Intelligence and Data Mining – Vol. 3, No. 2 (2008), P. 121-135.
[10] **Karp R.M**., **Rabin M.O**. Efficient randomized pattern-matchingalgorithms // IBM Journal of Research and Development – Vol. 31, No. 2 (1987), P. 249-260.
[11] **Park C**. In Other (People's) Words: plagiarism by university students - literature and lessons // Assessment and Evaluation in Higher Education – Vol. 28, No. 5, October 2003, P. 471-488.
[12] Plagiarism detection free detectors at wordchecksystems.com. [Online]. Available:http://www.wordchecksystems.com. [Accessed: Apr. 4. 2010].
[13] **Prechelt L**., **Malpohl G**., **Phlippsen M**. Finding Plagiarisms among a Set of Programs with JPlag // Journal of Universal Computer Science – Vol. 8, No. 11 (2002), P. 1016-1038.
[14] **Rehurek R**. Semantic-based plagiarism detection: PhD Thesis Proposal – 2007, P. 6-13.
[15] **Sorokina D**., **Gehrke J**., **Simeon W**., **Ginsparg P**. Plagiarism Detection in arXiv // The Sixth International Conference on Data Mining, Hong Kong, Japan, 18-22 December, 2006. – Washington: IEEE Computer Society, 2006. – P. 1070-1075.
[16] The plagiarism resource site. Charlottesville: Lou Bloomfield, 1997. [Online]. Available: http://plagiarism.phys.virginia.edu/Wsoftware.html. [Accessed: June 1, 2010].
[17] Turnitin: Plagiarism Checker to Ensure Academic Integrity. San Francisco: iParadigsm, 1998. [Online]. Available: http://www.turnitin.com/static/index.html. [Accessed: May 21, 2010].
[18] TurnitinUK. New Castle: iParadigsm, 2010. [Online]. Available: http://www.submit.ac.uk/static_jisc/ac_uk_index.html. [Accessed: June 1, 2010].
[19] **Wise M.J**. String similarity via greedy string tiling and running Karp-Rabin matching: Technical report No. 463 – The University of Sydney, March, 1993. – P. 3-8.
[20] Биржакопирайтинга. Russia.TextBroker, 2007. [Online]. Available: http://www.textbroker.ru/main/dcfinder.html. [Accessed: June 1, 2010].
[21] **Евтифеева О**., **Красс Л**., **Лакунин Н**., **Лысенко Е**., **Счастливцев Р**. // Научно-техническийвестник - №39 (2007), pp. 188-196.
[22] **Круглов В.В**., **Борисов В.В**Искусственныенейронныесети. Теория и практика. – Москва: Горячаялиния – Телеком, 2002. – 382 p.
[23] **Нейл К**., **Шанмагантан Г**. Web-инструментдлявыявленияплагиата // Открытыесистемы – №01 (2005), pp. 40-44.
[24] Проверкауникальноститекста в интернете. Russia, 2010. [Online]. Available: http://nado.su/downloads.html. [Accessed: June 4, 2010].
[25] AdvegoPlagiatus - проверкауникальноститекста. Russia,Advego, 2008. [Online]. Available: http://advego.ru/plagiatus/. [Accessed:June 1, 2010].

Maria Kashkur,M.Sc. student,Institute of Information Technology, Faculty of Computer Science and Information Technology, RigaTechnicalUniversity, 1 Kalku Street, Riga, LV-1658, Latvia. E-mail: kaskura@inbox.lv.
Maria Kashkur received her BSc degree in information technology from RigaTechnicalUniversity in 2010. Now she is an MSc student at the Institute of Information Technology.

Serge Parshutin, Ph.D. student, Department of Modelling and Simulation, Riga Technical University, 1 Kalku Street, Riga, LV - 1658, Latvia.
E-mail: serge.parshutin@rtu.lv.
Serge Parshutin received his MSc degree in information technology from RigaTechnicalUniversity in 2006. Now he is a PhD student at the Faculty of Computer Science and Information Technology and a Lecturer with the Department of Modeling and Simulation at the RigaTechnicalUniversity. His research interests include data mining and knowledge extraction, intelligent information systems, evolutionary computing and decision support.

ArkadyBorisov is Professor of Computer Science in the Faculty of Computer Science and Information Technology at RigaTechnicalUniversity. He holds a Doctor of Technical Sciences degree in Control in Technical Systems and the Dr.habil.sci.comp. degree. His research interests include fuzzy sets, fuzzy logic and computational intelligence. He has 205 publications in the area.
Contact information: Department of Modelling and Simulation, RigaTechnicalUniversity,1 Kalku Street, Riga, LV - 1658, Latvia. E-mail:arkadijs.borisovs@cs.rtu.lv

MarijaKaškura, SergejsParšutins, ArkadijsBorisovs. Plaģiāta gadījumu un to noteikšanas metožu pētīšana
Informācijas tehnoloģiju laikmetā intelektuālais īpašums kļūst īpaši vērtīgs. Tas ir viens no iemesliem, kāpēc plaģiāta gadījumi arvien biežāk notiek dažādās būtiskās nozarēs, tādās kā literatūra, zinātne, mūzika, dažādi mākslu veidi u.c. Sakarā ar to pieaug nepieciešamība pēc dažādu plaģiāta noteikšanas, autortiesību aizsardzības un apstiprināšanas instrumentu izstrādes un ieviešanas.
Atkarībā no pēc plaģiāta pārmeklētas sfēras, dinamiski mainās paša plaģiāta definīcija, un ar to mainās pārmeklēšanas telpa un metožu, kuru ir iespējams pielietot, kopa. Tāpēc ir svarīgi pirms uzsākt dažādu plaģiāta noteikšanas metožu un pieeju izskatīšanu un pirms uzsākt plaģiāta meklēšanu dažādās sfēras – dokumenti, mākslas darbi, programmu pirmkods, projekti u.c., ir nepieciešams saprast kas vispār ir plaģiāts, kas būs plaģiāts mūsu izvēlētajā gadījumā un kādām plaģiāta pārmeklēšanas metodēm ir vērts pievērst vairāk uzmanības. Līdz ar to šajā rakstā ir izskatīti dažādi plaģiāta definējumi un plaģiāta svarīgākie veidi, tajā skaitā studentu plaģiāts. Tiek diskutēts par studentu plaģiāta parādīšanas iemesliem. Rakstā ir piedāvāti populārāko plaģiāta noteikšanas sistēmu, kas tika izstrādāti un ir pielietoti tādās ASV universitātēs, kā Stenforda universitāte, Kalifornijas universitāte u.c., apraksti. Aprakstītas apskatītās plaģiāta noteikšanas sistēmās pielietoto algoritmu īpašības un dažas plaģiāta noteikšanai izmantojamas programmas, kas tiek pielietoti dokumentu salīdzināšanai.

Мария Кашкур, Сергей Паршутин, Аркадий Борисов. Исследование случаев плагиата и методов их обнаружения
В эпоху информационных технологий интеллектуальная собственность становится всё более ценной.Это является одной из причин проявления случаев плагиата в различных сферах таких, как литература, наука, музыка, различные виды искусства и др.В связи с этим возрастает потребность в разработке и внедренииразличных инструментах для защиты и подтверждения авторских прав и выявления плагиата.
В зависимости от исследуемой на возможность плагиата сферы, меняется и само определение термина «плагиат», и, соответственно, изменяются пространство поиска и методы, которые можно применять. Поэтому прежде, чем приступать к рассмотрению различных методов и подходов для выявления плагиата и поиску плагиата в различных областях – документах, произведениях искусства, проектах и т.д., необходимо чётко представлять, что в конкретном случае будет являться плагиатом и, соответственно, на какие подходы и методы обнаружения плагиата следует делать акцент. В данной работе рассматриваются различные определения плагиата иобсуждаются наиболее важные типы плагиата, в частности, студенческий плагиат, и возможные причины появления плагиата среди студентов. Также представлены описания наиболее распространённых систем выявления плагиата, разработанных и используемых в университетах США, таких, как Стэнфорд (Stanford), Калифорнийский университет (University of California) и др. Приведены описания особенностей методов, применяемых в выбранных системах обнаружения плагиата, а также - описания нескольких программ, используемых для сравнения документов при поиске плагиата.