

UDC 004.021

SOFTWARE SYSTEM FOR TESTING STUDENTS' WORKS FOR PLAGIARISM BASED ON A LOCALITY-SENSITIVE HASHING ALGORITHM

Valeria Fekete, Ivan Fekete, Yaroslav Obukhov

PhD, Professor of the Department of Software Engineering Viktoriia Ruvinska
State University "Odesa Polytechnic", Ukraine

ABSTRACT. The paper proposes a study aimed at reducing the time required to search for similar texts in the corpus when testing for plagiarism by reducing the asymptotic complexity of the modified locality-sensitive hashing algorithm. As a result of the work, a system that provides functionality for checking works for plagiarism on the basis of the developed algorithm of locality-sensitive hashing is created.

Keywords: Locality-sensitive hashing algorithm, asymptotic complexity, check of texts for plagiarism

Introduction. There are many websites that perform the function of checking a work for plagiarism by comparing it with other works on the Internet. However, there are no freely available systems of this kind that can be used to check for plagiarism, not texts from the Internet, but those stored in some text corpus. The system being developed is intended for educational institutions. It stores student works from all years and compares them with one another. This will greatly simplify the teacher's work, as they can at any time check whether the system raises a suspicion that a given work was copied.

In existing systems, algorithms for finding similar texts are insufficiently efficient, having asymptotic complexity of $O(n^2)$ or $O(n \log_2 n)$, where n is the total length of the corpus texts [1]; moreover, some of them are not capable of comparing short texts, and also have no ability to save the results of their operation.

The goal of the study is to reduce the time required to search for similar texts in a corpus when checking for plagiarism by reducing the asymptotic complexity of the modified locality-sensitive hashing algorithm.

Main Body. The developed system performs the task of comparing two texts using the Locality-Sensitive Hashing (LSH) algorithm – a probabilistic method for dimensionality reduction of multidimensional data [2]. Its core idea lies in selecting hash functions such that similar objects will, with high probability, have similar hash function values. It is one of the ways to combat the "curse of dimensionality" in searching and analyzing multidimensional data: as the dimensionality of the source data grows, index-based search performs worse than sequential scanning. The method allows building a structure for fast approximate (probabilistic) search of n -dimensional vectors "similar" to a target pattern.

The algorithm's operation can be roughly divided into 2 stages: the shingling stage and the stage of computing the Jaccard coefficient between the given document and all documents. The $O(n \log_2 n)$ complexity is due to the use of balanced binary search trees to store sets [1], which is not justified in this algorithm and can be optimized.

Through the use of a counting sort algorithm for large numbers, as well as Fermat's little theorem and binary exponentiation, the asymptotic complexity was reduced to $O(n)$, where n is the total length of the corpus texts.

In the system for checking texts for plagiarism, developed on the basis of the modified LSH algorithm, the following user roles exist: teacher and student. A student can submit their work, which will be automatically checked for plagiarism using the above-mentioned algorithm. When a teacher wishes to find out which specific works a given work resembles, they can generate a detailed report. To generate a report comparing all similar works for the teacher, a script was written that independently generates a file with the .html extension, which will be opened automatically for the teacher in a browser. The teacher also has the option to save the report to a local computer in .pdf format using their browser's tools.

The system being developed is client-server [3]. A personal computer serves the role of the server. The main program with the user interface is the client. In order not to overload the computing resources of the users' computers, thin client technology is used. Thus, all data is stored, generated, and processed on the server, and the client's main task is to interact with the users.

As the database for storing information of the developed program, the relational database management system SQLite was chosen [4]. All classes are implemented using the C++ language. Given that the platform must be convenient to use and meet users' needs regardless of the device from which they log in, it has an adaptive design. To achieve this goal, the Qt Creator development environment was used in designing the program interface [5]. To generate the report comparing all similar works for the teacher, a script was written in Python 3 [6]. The local server was created using classes from the Qt library – QTcpServer and QTcpSocket; data transmission is carried out based on the TCP protocol [7].

To test the efficiency of the algorithm in the system, program code in C++ was created with a measurement of the program's running time, which was tested on randomly generated texts of varying lengths.

Table 1 shows a comparison of the running time of the existing locality-sensitive hashing algorithm and the modified algorithm. A substantial difference in favor of the developed algorithm can be observed for texts of various sizes.

Table 1 – Comparison of the running time of the existing locality-sensitive hashing algorithm and the modified algorithm

Lengths of the two texts being compared	Average running time of the modified LSH algorithm	Average running time of the existing LSH algorithm	Reduction in running time of the modified algorithm compared to the existing one
10 ⁵ bytes	31 ms	56 ms	44.6 %
10 ⁴ bytes	9 ms	11 ms	18.2 %

Conclusions. As a result of the work, the following tasks were completed: the locality-sensitive hashing algorithm was modified by applying efficient algorithms and techniques; a system for checking student work for plagiarism was designed and developed based on the modified locality-sensitive hashing algorithm; system testing and algorithm evaluation were conducted. The running time of the locality-sensitive hashing algorithm with asymptotic complexity of $O(n \log_2 n)$, where n is the total length of the corpus texts, was compared with the developed modification of the algorithm with asymptotic complexity of $O(n)$, where n is the total length of the corpus texts, on texts of length 10⁵ and 10⁴ characters. Thus, on average, the running time of the modified algorithm is less than that of the existing one by an average of 31.4%.

A possible direction for further work is improving the standardization of a document prior to similarity checking by reducing similar characters to a unified standard, and introducing plagiarism checking against external text corpora, such as, for example, program codes submitted by students during olympiads.

REFERENCES

1. Introduction to Algorithms / T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. – Kyiv: K.I.S., 2018. – 1286 p. – (3rd ed.).
2. M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. "Locality-sensitive hashing scheme based on p-stable distributions", in Proc. 20th Annu. Symp. Comput. Geometry, 2004, pp. 253–262.
3. Valery Korzhov. Multi-tier client-server systems. Published by Open Systems (June 17, 1997).
4. Arsirii, O. O., Glava, M. G., Kolonko, Matthias & Glumenko, A. O. "Information Technology of Supporting Architectural Solutions Using Polyglot Persistence Concept in Learning Management Systems". Applied Aspects of Information Technology. Publ. Nauka i Tekhnika. Odesa: Ukraine. 2020; Vol. 3 No. 2: 13–31. DOI: <https://doi.org/10.15276/aait.02.2020.1>.
5. Official Qt website — Electronic resource: <https://www.qt.io/>
6. Prokhorenok N., Dronov V. Introduction // Python 3. The Essentials, 2nd ed. — BHV-Petersburg, 2019. — P. 11. — 608 p.
7. Douglas Comer. TCP/IP Internetworking, Vol. 1. Principles, Protocols and Architecture. — Moscow: "Williams", 2003. — P. 880. — ISBN 0-13-018380-6.