Chapter 17
Plagiarism Detection Algorithm for Source Code in Computer Science Education

Xin Liu
Xiangtan University, China

Chan Xu
Xiangtan University, China

Boyu Ouyang
Xiangtan University, China

ABSTRACT

Nowadays, computer programming is getting more necessary in the course of program design in college education. However, the trick of plagiarizing plus a little modification exists among some students' home works. It's not easy for teachers to judge if there's plagiarizing in source code or not. Traditional detection algorithms cannot fit this condition. The author designed an effective and complete method to detect source code plagiarizing according to the popular way of students' plagiarizing. There are two basic concepts of the algorithm. One is to standardize the source code via filtration against to remove the majority noises intentionally blended by plagiarists. The other one is an improved Longest Common Subsequence algorithm for text matching, using statement as the unit for matching. The authors also designed an appropriate HASH function to increase the efficiency of matching. Based on the algorithm, a system was designed and proved to be practical and sufficient, which runs well and meet the practical requirement in application.

1. INTRODUCTION

With the development of computer technology, now in most courses of programming in colleges and universities, the traditional form of students' homework hand-written work, program in paperwork is replaced by computer programming. In such job checking system, normally source codes, are required to be submitted and then be compiled and run automatically, Grades of these home works are given per

DOI: 10.4018/978-1-5225-8057-7.ch017

the running results. Not only does this method reduces the workload of teachers effectively, but also it improves the students' actual programming skills, and thus it's applied by many colleges and universities.

According to Zhang, Liu, & Li (2010), Hou, & Liu (2011), and Li (2010), a widespread problem appeared that many students copying other people's code to submit as their own jobs. In order to detect such tricks, the system began to add plagiarism detection function, once two identical or sufficiently similar codes were detected, it'll be judged as plagiarism and the code submitted to the system later will be rejected. The plagiarists also use a variety of ways to modify the source code to evade the system's detection. In the past few decades several plagiarism detection methods, such as properties counting method (Halstead, 1977), structural measure approach (Damashek, 1995), shingling (Manber, 1994), simhash algorithm (Manku, Jain & Sarma, 2007), etc. are present by researchers. But these methods exist some shortcomings. Therefore, designing a high adaptability algorithm to check the source code is of strong practical significance.

The structure of this paper is as follows: In Section 2, we introduce the existing methods and their shortcomings. In Section 3, we illustrate the plagiarism detection algorithm for source code designed by us, and analyze the time complexity of our algorithm. In Section 4, we describe an actual job submission system based on the algorithm, and present some experiment result. We discuss the flaw of this approach and the future directions of our research in Section 5.

2. EXISTING METHODS AND SHORTCOMINGS

Back in the 1970s, researchers started research of the similarity detection technology against source code. Halstead (1975) proposed the first algorithm named property counting method. The algorithm counted the operators and operands statistics appeared in the source program, and used the results as main basis of detecting. Ottenstein (1976) implemented the first source code near-duplicates detection system for Fortran by using properties counting method. Since the attribute notation doesn't remain the program structure information, the method cannot meet practical requirements of short program due to high false alarm rate (definition in section 4).

In the mid-1990s, Verco and Wise (1996) added vector dimension technology to the properties counting method, but the effect is still not satisfactory. Damashek (1995) proposed structural measure approach, used program control flow as metrics, such methods are usually applicated with attribute notation. Such methods work well in checking large programs, because in handling complex problems, different programmers often have different ideas, probability of identical program control flow is extremely low, so the false alarm rate is relatively low, but experiments proved that when such algorithms applying on program designing jobs, it has a relatively high false alarm rate. Because programming as common work is simple and the fundamental knowledge is quite similar, so the students' main concepts of solving the problems are similar, thus the control flow of the program will be basically alike.

Since mid-1990s, research focus shifted to the investigation of natural language text. Manber (1994) proposed approximate fingerprint concept, the basic principle is measuring the similarity between documents through the string matching method. This principle was adopted by most researchers. Then, on this basis, the researchers increased the word frequency count, keyword extraction technology, done matching by calculating the hash-value of the text. The simhash algorithm (Manku, Jain & Sarma, 2007). COPS (Damashek 1995), SCAM (Shivakumar & Garcia-Molina, 1995), CHECK (Barrón-Cedeño & Rosso, 2009) and other systems were essentially based on the realization of the precious principle.