The analysis of source code plagiarism in basic programming course

D Maryono, R A Yuana and P Hatta
Department of Computer and Informatics Education, Universitas Sebelas Maret
Jl. Ahmad Yani No. 200 Pabelan, Kartasura, Sukoharjo, Jawa Tengah, Indonesia
E-mail: dwimaryono@staff.uns.ac.id

Abstract. The purpose of this research is to recognize the types of plagiarism that occurred in basic programming courses as a preliminary research in developing an intelligent system to detect source plagiarism in programming class. This research use a hierarchical clustering method which is implemented in term-document matrices. In this research, we propose the use of keywords in the programming language informing the term-document matrix. The result is normalized and then we use Euclidean distance to normalize data for similarity measurement. The result shows that this method is quite effective to classify objects of data indicated by plagiarism. Some types of plagiarism which are mostly occurred are formatting source code, changing comment, renaming identifier, and adding statements or redundant variables.

1. Introduction
Plagiarism can be defined as an illegal act of - using or copying complete or part of others’ documents. This definition also applies in source code context, where documents in source code are written in several programming languages. A plagiarism program is an original copy or modified program using various types of modifications [1].

According to [2] and [3], source modifications can be grouped into two main categories: lexical and structural modification. Lexical modifications do not require special programming knowledge and in many cases, they are easily detected. Typical examples of plagiarism which include lexical modification are formatting source code (L1), changing comment (L2), renaming identifier (L3), splitting or merging declaration variable (L4). structural modification, on the other hand, it requires a higher level of programming knowledge and skills so that detection in this case is more difficult. Typical examples of plagiarism include in the structural modification, they are changing the order of variables in statements (S1), changing the order of statements in blocks (S2), realigning code blocks (functions) (S3), adding statements or redundant variables (S4), modifying control structures (S5), changing data type, data structures modification (S6), redundancy (S7), temporary variables and subexpressions (S8).

Detecting source code plagiarism can be done manually, but it’s certainly time-consuming. Especially, if the task has a large scales. In order to help lecturers in detecting fraudulent practices like this, it is necessary to develop a system of plagiarism detection in order to keep students’ honesty values and increase their creativity in doing their tasks. Several plagiarism detection systems have been developed and can be accessed on a web-based basis such as Plagiarisma, DustBall, and Duplchecker. Some of these systems can be used as an initial investigation into the practice of plagiarism fraud, but detection errors often occur. Alsmadi et. al [4] stated that the error that occur is a false sign, where documents that are not indicated as plagiarism are detected as plagiarism.
In this research, the plagiarism detection is limited for basic programming course in Department of Computer and Informatics Education. The document checking is only limited to the tasks in the lecture, it is not necessary to compare the source code files on the internet, so that the resulting system is expected to be faster. In order to improve performance, it is also necessary to explore information about the type of plagiarism commonly done by students. It is important to determine what algorithm is suitable to be applied among existing algorithms such as string matching [5], coding factorization and pattern matching [6], and so on. As an initial step, this paper focuses on recognizing the kinds of source plagiarism that occurs in basic programming course, so that we can use an appropriate algorithm for identifying plagiarism and helping lecturers evaluating students’ assignments.

This research uses hierarchical clustering technique to find objects which are similar to each other, where it indicates as plagiarism. After the clustering results, then analyzed to see the types of plagiarism that emerged, in accordance with the theory [3]. Generally, document clustering uses a term-document matrix [7], where the values of the matrix entry are the number of times a term appears in the document. This can be done as a first step to see the similarities among source code. However, it should also be noted that there is a possibility that this plagiarism includes the change of identifier name, addition of comments, redundant variables and so on. Therefore, in developing the term-document matrix, we don’t use the entire term, but we just use programming keywords and character such as: begin; for, if:, =, write, writeln, read, readln, gotoxy, label, and, or, etc. We assume that when someone does source code plagiarism, the number of rows, programming structure, conditional structure, looping structure, and assignments will be relatively the same, so this idea can be used for identifying the type of plagiarism that might occur.

After the document term matrix is formed, hierarchical clustering will be carried out to see which documents are similar to the others, for further analysis of the type of plagiarism and the appropriate algorithm for the subsequent classification process is identified. However, before applying hierarchical clustering, data is normalized using the normal distribution function.

2. Methodology
This is a descriptive research, which having data by doing observation, where we want to identify the type of plagarism which occurs in basic programming class. For this purpose, we implement hierarchical clustering to the source document and then analyze all the document which are simmiliar. The steps of this research is as Figure 1.

3. Result and discussion

3.1. Datasets
We use 3 datasets which are taken from individual or group assignment in basic programming course. Dataset 1 consisted of 22 source code documents, dataset 2 consist of 16 source documents, and dataset 3 consisted of 12 source documents. All source code documents are in Pascal Language.

3.2. Result

3.2.1. Result on Dataset 1
The first dataset is taken from individual tasks. In this dataset, there are 22 source code files to be analyzed. Each document is labeled T1001, T1002, ..., T1022. The hierarchical clustering result on the term-document matix normalized can be seen in Figure 2.
From Figure 2 we can see the distance between documents in clusters, in which it can be used to analyze the type of plagiarism that occurs. Based on Figure 2, we analyze all the documents which are related each other in those clusters.
We look at 2 pairs of documents with numbers 8 and 22; and 7 and 21. Both pairs have a distance of nearly 0, so that it can be said to be almost the same. From the document we have, it’s obtained a fact that documents number 8 and 22 are exactly the same. So do documents number 7 and 21. The type of this plagiarism is L1.

For data pairs, 9 and 10, it turns out that it is also just copying the source code without changing. Meanwhile the pairs of documents 6 and 17 even though the distance between the two pairs is the same, at distance 1, but it shows the existence of new forms of plagiarism. By looking at documents 6 and 17 we found that there are some changes in variable names (L3), modification in the comments section (L2), and the addition of redundant statements (S4).

From Figure 2, we can get a big cluster includes documents numbered 4, 7, 21, 9, 10, 6, 17, and 14. From the documents, we found that all these documents come from the same source. This can be seen from the variable declaration, the sequence statement of programs starting from input, logic, and output. The types of plagiarism that appear are L2, L3 and S4. For documents numbered 13, 18, 16, it is indicated that they also come from one source. Plagiarism that appears is a change in comment (L2), change in identifier (L3), adding a redundant statement (S4).

From testing data set 1, it can be concluded that the use of a document-term matrix that only involves a few key words is good enough in clustering similar data object which are indicated as plagiarism. The most common types of plagiarism are L1, L2, L3 and S4.

3.2.2. Result on Dataset 2

Source code in this data is more complex compared to data sets 1. Therefore in this case, more keywords are used to build term document matrix. The results of the clustering on the normalized term-document matrix are as shown in Figure 3.

Based on Figure 3, there are several data objects in the same clusters, which are suspected as plagiarism. Based on the documents, objects 13 and 14 are very similar from the declaration section to the end of the program body, including variable names, procedure, and the statements on the program body. The difference only locates in one procedure. In this case, the type of plagiarism detected is L1.
Objects 9 and 10 are also very similar, including variable names, procedure names, and statements on the program body. However, there has been a slight modification, such as the code block rearrangement by dividing a procedure into two (S3) and adding redundant statements, such as adding textcolor (S4).

Objects 5 and 12 are also very similar, modifications that made are adding variables and redundant statements (S4). Object 6 and 10 in this data are not very similar, only the variables in the declaration section, and one of the procedures that looks similar, the rest of the structure and statement flow appear to be quite different. We can also see that object 17 are closed to objects 6 and 10, but this object is actually more similar to objects 9 and 10, especially in the declaration section, including giving the same variable name.

For this data set, the type of plagiarism which are accored is L1, S3, and S4.

3.2.3. Result on Dataset 3

The results of the clustering on the normalized term-document matrix for dataset 3 are as shown in Figure 4.

![Hierarchical clustering for Dataset 3.](image)

From Figure 4, it can be seen that there are 4 objects that are relatively close together, which are object numbered 3, 5, 6, and 7. From the documents we get fact that objects 3 and 5 are very similar to each other. Modifications are made only with the addition of redundant statements (S4), especially related to display statements.

Furthermore, object 6 is also similar to objects 3 and 5, the modifications made are the same, which is the addition or modification of additional statements (S4). For objects 7, it is also similar to object 6 but there is an elimination of a procedure, and changes in data types, so the similarity decreases with objects 3, 5, and 6. However, it is concluded from the object 3, 5, 6 and 7 are strongly indicated from the same source with some modifications. The type of plagiarism in this case are L1, L2, L3 and S4.

For object 4, the distance is quite far from the previous 4 objects, because it is basically different in the structure. Object 4 used pointer for the data structure, meanwhile the previous object doesn’t, so
structurally there is no indication of plagiarism with objects 3, 5, 6 and 7. So do object 9 and 4 so there is no indication of plagiarism.

From cases 1, 2 and 3, there is an interesting fact that the distance between clusters indicated as plagiarism is below 1.5. This is important to provide a threshold to consider objects in a cluster are indicated as plagiarism or not. This will be certainly very useful to conduct further research on the types of plagiarism that occur.

4. Conclusion

Based on the results above, the term document matrix by using keywords in the programming language is enough to detect the beginning of plagiarism with various types. From the results of the analysis found that the types of plagiarism that mostly occurs in the basic programming assignments are L1, L2, L3, and S4. By using hierarchical clustering, further checking can be done on the documents in the same cluster. This will certainly make the checking of plagiarism more effective. And furthermore, we can apply or develop special algorithms to detect the type of plagiarism which mostly occurs, especially plagiarism types of L1, L2, L3, and S4.

References

[1] Jones E L 2001 Metrics based plagiarism monitoring Journal of Computing Sciences in Colleges 16(4) 253-261
[2] Joy M and Luck M 1999 Plagiarism in programming assignments IEEE Transactions on Education 42(2) 129-133
[3] Roy C K and Cordy J R 2007 A Survey on Software Clone Detection Research Queens School of Computing TR vol 541 pp 64-68
[4] Alsmadi I, AlHami I and Kazakzeh S 2014 International Journal of Software Engineering and Its Applications 8(4) 23-34
[5] Delev T and Gjorgjevikj D 2017 Comparison of string matching based algorithms for plagiarism detection of source code Proc. of the 14th International Conference for Informatics and Information Technology 2017 (Mavrovo: Ss. Cyril and Methodius University) pp 151-157
[6] Chilowicz M, Duris É and Roussel G 2009 Electronic Notes in Theoretical Computer Science 238 47-62
[7] Yates R B and Neto B R 1999 Modern Information Retrieval (New York: Addison Wesley) p 144