Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 7, July 2014, pg.155 – 160

RESEARCH ARTICLE

A Hybrid Approach to Perform Code Clone Analysis

Sonia[1], Anshu Arora[2]

[1]Student, M.Tech (CSE), Meri College of Engineering and Technology, Sampla, Haryana
Sonia.balhara28@gmail.com

[2]Asstt. Prof., M.Tech (CSE), Meri College of Engineering and Technology, Sampla, Haryana
Anshu86arora@gmail.com

Abstract: Code cloning is about to copy some existing code or content without permission. It is considered as the serious crime and become worst if the code is reserved to some other person or the firm. Because of this, while writing the code it is required to track the program not to use this copy-paste technique. In this work, a hybrid mechanism is presented to identify the code cloning over the program code. The presented work has used a statistical and keyword analysis approach to identify the copy code over the program or software system. In this paper, the algorithmic structures of this code cloning are explored.

Keywords: Code Cloning, Plagiarism, Data Mining, Code Analysis

I. INTRODUCTION

Copy-paste is one of the most used software development activity adapted my most of the programmer. It is basically used to reduce the coding effort and to reuse the existing cost. There are number of online portals that provide the software code for different languages free of cost. Programmersheaven.com, koders.com etc. are the leading code providers. But while doing this copy-paste, sometimes the coder forgets to take the precautions about using the copy-write reserved codes or identifying the actual source of the code. Because of this, the code used by the coder comes under the code plagiarism. The plagiarism is the serious crime that comes under the cyber security. Because of this it is required to use the selective code from the online sources. Another drawback of code cloning is the, lack of effectiveness and understanding of the coder. Most of programmers just use the code without getting the actual meaning of the code or without identifying the basic assumptions or warning that lies with the code. Because of this, the integration of such code with other modules is not compatible. It increases the integration cost and sometime rewriting of code is required. Code cloning is also one of the major stage of software testing and software analysis to identify the actual cost of the

development and to estimate the actual efforts of the programmers. Because of these all reasons it is required for a company to identify the code cloning as the subsequent stage to the software development. There are number of source code providers available over the internet. Some of these providers are listed in table 1.

Table 1: Different Source of Codes

Code Site	Code Site2
CodeFinder	Google Code
CodeBroker	Sourceforge
Mica	SPARS-J
Prospector	Sourcerer
Hipikat	Sourcerer API
xSnippet	CodeGenie
Strathcona	SpotWeb
AMC	ParseWeb
Koders	S6
Exemplar	

As shown in figure, the most common type of cloning is copy-cloning. It is the copy-paste type of cloning in which programming does not add any efforts with the copied code. The code take from the actual source is used without any change. It is most easy to detect and analyze. Another kind of cloning is renamed clone, in such type of code, no intelligent change is done in code. The programmer just change the name of variable, identifiers, function names, file names etc and use the code as it is. If the flow graph or the structural analysis. Code cloning is basically divided in four major categories shown in figure 1.

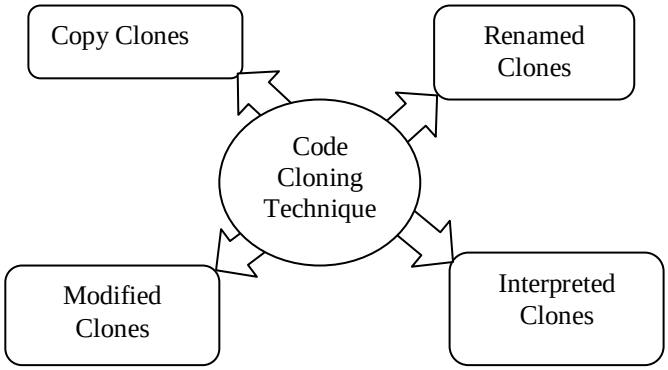

Figure 1 : Types of Code Cloning

Over the code is done, such kind of code cloning can be detected easily. Another type of code cloning is Modified Clones. In this kind of cloning, use the concept of renamed cloning as well as add some other code fragments in the code. Sometimes, the merge of two codes is also done. Sometimes, the sequence of the statements is also changed, but no change is done at the structural level. By performing the structural or component analysis such kind of detection can be identified easily. The most intelligent type of code cloning is the interpreted cloning. This kind of cloning having the features of modified cloning as well as change the structure of the code by changing the basic construct. Such as conversion of code from one structural form to other. This cloning type does the intelligent changes over the code that difficult to identify. Even the structural analysis fails to identify such kind of cloning.

A) CFG

CFG basically represents the flow of the program or the software in the form of graph. To identify the cloning using CFG, the component analysis of CFG is done. The CFG is a directed graph represented by G(V,E). Where V represents the vertices and E represents the Edges. To check the cloning, the degree of isomorphism is been analyzed over two CFG. Let G1(V1,E1) and G2(V2,E2) are two CFG of two different code that are been analyzed for code cloning. They are called isomorphic if 1:1 correspondence is present between the vertices edges. If the Code cloning using CFG is positive, then individual nodes are been analyzed for the both graphs. At this level, the actual code checking is performed. Here figure 2 is showing the example of CFG for some code shown in table 2.

Table 2 : Code Example for CFG

1:	WHILE NOT EOF LOOP
	Read Record;
2:	IF field1 equals 0 THEN
3:	Add field1 to Total
	Increment Counter
4:	ELSE
	IF field2 equals 0 THEN
5:	Print Total, Counter
	Reset Counter
6:	ELSE
	Subtract field2 from Total
7:	END IF
8	END IF

The code flow graph for the above listed code is shown in figure 2.

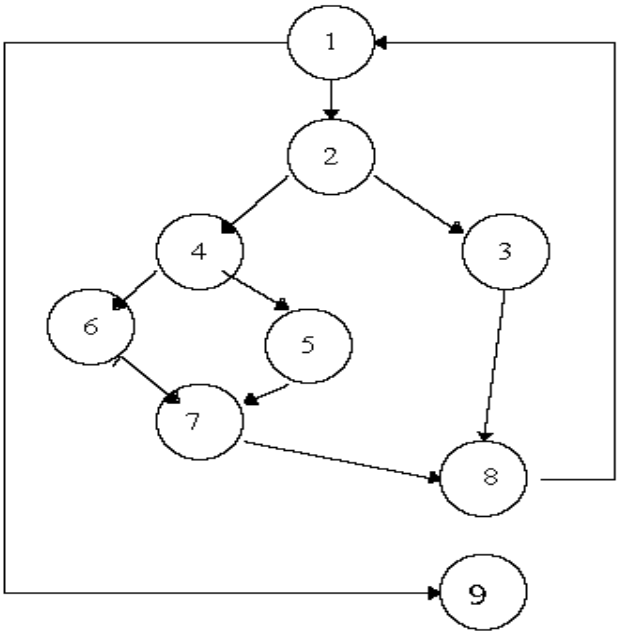

Figure 2 : CFG Example

II. EXISTING WORK

Lot of work is already done in the area of code clone detection and analysis. Some work done by the earlier researchers in the same area is discussed in this section. In Year 2011, Mohammed Abdul Bari[1] defined the basic taxonomy related to code cloning as well defined a deep analytical study of clone detection and its removal. Author defined the work for two major clone detection approaches called static and dynamic analysis. Author collect both these two approaches and defined an analytical solution. The objective of author was to perform a static check over the code and remove it. Another work in same area was performed by Simone Livieri[2]. Author defined the work by using some existing tool called CCFinder and D-CCFinder. It is the open source tool used for the clone anlaysis for larger codes. It is one of the complex tool adapted to perform the analysis over the large segment of codes. The tool performs certain number of tests so it takes lot of time for the analysis. Another study work on code clone approaches is done by Chanchal Kumar[3]. Author study the common constructs and phenomenon associated with code cloning as well as identify the type of cloning over the codes. Author also define different analysis approach and their evaluations. Author also identify the problems associated with code clone detection approaches. In year 2009, Tung Thanh[4] defined the study using another code clone detection tool called ClemanX. This tool is been implemented in real world to analyze the actual software projects. This tool perform the analysis under different parameters including the efficiency, preciseness, completeness and capabilities of the software projects. In year 2012, Xlaoyin Wang[5] define a work proposal to predict the harmfulness of code cloning in case of copy-paste kind of cloning. Author perform the study along with the exploration of different characteristics and features associated with code cloning. Author defined the Bayesian Network approach to predict the harmfulness of intended code cloning operations.

In year 2011, Yingnong Dang[6] define the effective code clone detection experience with the Microsoft engineers. Author has defined the development setting and experience so that the effective cloning will be done and to reduce the cloning errors. Randy Smith[7] has defined a detection and similarity analysis approach over the codes to identify the duplicate code blocks along with similarity threshold. Author defined the rank based similarity analysis approach for the code detection in C programs. In year 2010, Mu-Woong define[8] the code clone search for the large software systems. The author defined the code clone for large code segments and verified it for multiple open source projects. The author basically work on the efficiency to perform the detection process effectively and compromise with the accuracy. In Year 2011, G. Anil kumar[9] define a refactoring based approach for the code clone detection. Author performed the textual analysis and define the work under different clone detection approaches. Author perform the analysis and classification of different code cloning approaches. In year 2010, Elmar Juergens[10] defined a work on code clone detection to improve the accuracy. Author defined the outline based method for the clone detection. Author performed the industrial

case study to identify the accuracy and efficiency of the defined system. Toshihiro Kamiya[11] defined an integration of code clone detection approaches. Author defined the analytical approaches for cone detection. Author study the challenges faced during the integration of different software modules or the code fragments. Akito Monden[12] define an analysis on code clone analysis on the industrial point of few. The analytical study is done under different factors such as reliability, maintainability etc.

III. CODE CLONING APPROACHES

There are number of existing code clone detection approaches. These approaches identify the statements over the code, where the cloning is present. Once the statements are marked, the programmer can remove or modify these statements to remove the cloning. Most of these approaches perform the analysis of clone detection under different parameters. These parameters include the efficiency, accuracy, robustness etc. The analysis also includes the semantic analysis, syntax analysis over the program code. There are number of code clone detection approaches listed as under:

A) Text based code clone detection

In this clone detection approach, the code is considered as the simple text document and the text duplication is analyzed over the code. To perform the analysis, this approach divides the code in literals and ignores the comment lines and white spaces over the code. Once the literals over the code are retrieved, the suffix tree is generated. The suffix tree can be based on line based or literal based. This kind of analysis also uses some other approaches for the matching such as Matrix based Analysis, Binary Tree based Analysis.

B) Token based code clone detection

Another clone detection technique is based on the concept of data mining. Data mining is about th extraction of some quantitative information from the code. To perform this analysis, it is required to extract different components from the code such as loop, conditions, identifiers etc. Once the extraction is done, the structural information is derived such as number of loop etc. For each kind of code component a separate set is maintained. As the sets of two different codes are derived, the next work is comparing them under different data mining operations. The decision of the data mining operation is also the major thing that depends on the code size and the number of modules.

C) Token based Approaches

According to this approach, the complete code is represented in the form of tokens. The token includes the keywords, identifiers present in a source code. Once the tokens are derived, the next work is to generate the token sequence. A token sequence is the transformed using the defined set of rules that generalized the token in an abstracted way. Some heuristic search and the metrics is used to generate the token sequence. Once the sequence for two code segments is generated, the next work is to perform the comparative analysis. There some open source tools that work on the same phenomenon. CCFinder is one such tool that work on token based analysis. CCFinder also uses the lexical knowledge for the comparative analysis. It also provide the sophisticated visualization over the code so that appropriate method can be applied for the clone removal.

D) Pattern Matching Based Approach

There are some methods to perform the code cloning based using the pattern matching approaches. To perform this analysis the code is divided in smaller blocks. These blocks are identified by using the begin-end pairs or the parenthesis blocks. Once these all blocks are identified, some similarity metric is applied for the analysis and overall cloning over the code is identified. Once this information is obtained, the next work is to separate the high scoring pairs. This method compares the whole block of code and identifies the code cloning effectively.

E) Syntax Tree Based Approach

A Syntax tree based in approach based on the structural analysis. In this approach, the similarity between the code block is derived based on the variable names, literals etc. This method includes the syntax and syntactic analysis over the code. This kind of approach is beneficial to uncover the modified cloning. This method does not compare the identifier names exactly instead it is based on the meaningful component analysis along with its categorization.

IV. CONCLUSION

The presented work is about the exploration of code cloning, its type and approaches. In this work different analytical approach are described. In this work,

REFERENCES

[1] Mohammed Abdul Bari," Code Cloning: The Analysis, Detection and Removal", International Journal of Computer Applications (0975 – 8887) Volume 20– No.7, April 2011
[2] Simone Livieri," Very-Large Scale Code Clone Analysis and Visualization of Open Source Programs Using Distributed CCFinder: D-CCFinder".
[3] Chanchal Kumar Roy," A Survey on Software Detection Research".
[4] Tung Thanh Nguyen," ClemanX: Incremental Clone Detection Tool for Evolving Software", ICSE'09, May 16-24, 2009, Vancouver, Canada 978-1-4244-3494-7/09@ 2009 IEEE
[5] Xiaoyin Wang," Can I Clone This Piece of Code Here?", ASE '12, September 3 – 7, 2012, Essen, Germany Copyright 2012 ACM 978-1-4503-1204-2/12/09
[6] Yingnong Dang," Code Clone Detection Experience at Microsoft", IWSC'11, May 23, 2011, Waikiki, Honolulu, HI, USA ACM 978-1-4503-0588-4/11/05
[7] Randy Smith," Detecting and Measuring Similarity in Code Clones".
[8] Mu-Woong Lee," Instant Code Clone Search", FSE-18, November 7–11, 2010, Santa Fe, New Mexico, USA. ACM 978-1-60558-791-2/10/11
[9] G. Anil kumar," Code Clone Detection with Refactoring support Through Textual analysis", International Journal of Computer Trends and Technology- volume2Issue2- 2011 ISSN: 2231-2803
[10] Elmar Juergens," Achieving Accurate Clone Detection Results", IWSC2010 May 8, 2010, Cape Town, South Africa ACM 978-1-60558-980-0/10/05
[11] Toshihiro Kamiya," Interoperation Potential: Integration of Code-Clone Detection Methods with Other Analysis Methods".
[12] Akito Monden," Software Quality Analysis by Code Clones in Industrial Legacy Software".
[13] Chanchal K. Roy," Comparison and Evaluation of Code Clone Detection Techniques and Tools: A Qualitative Approach".

