ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Clone-Chunk Algorithm for Non-Contiguous Clone Detection

Sonam Gupta[1], Dr. P. C Gupta[2], Dr. Ripuranjan Sinha[3]

Research Scholar, Suresh Gyan Vihar University, Jaipur, Rajasthan, India

Associate Professor, Department of Computer Science & Informatics, University of Kota, Kota, Rajasthan, India

Associate Dean Research, Suresh Gyan Vihar University, Jaipur, Rajasthan, India

ABSTRACT: A research [1] says "as a program evolves, it becomes more complex, and extra resources are needed to preserve and simplify its structure". Basically, the main reason for the occurrence of clones is copy-paste practice since it is easier to write an already developed code and make some changes in it rather than writing it from scratch, if the implementation is almost similar. Software maintenance is the longest and crucial stage of the software lifecycle, since in order to fulfil the requirements of the user, the system needs to be updated and bugs should be fixed timely. [2] So software evolution is the critical phase of the software maintenance. [3,4] found that due to maintenance more than half time is consumed almost 40% of effort is done on enhancements and extensions in the evolution phase. So, if the cloned codes can be identified correctly than the cost and effort of software development can be reduced to a much lower extent. This paper extends the clone chunk extraction algorithm given in [9] by using the ANTLR grammar thereby giving the view in chain view. Along with this the algorithm provides various statistics so that the programmer can decide whether he want to check the code for cloning or not.

KEYWORDS: Software maintenance, Clone chunk, ANTLR, open source system.

I. INTRODUCTION

Clones are introduced within the systems to get many maintenance advantages. For example risk in developing new code, clean and comprehensible software system design, speed up maintenance, ensuring hardiness in life-critical systems. Cordy [6] reports that clones do regularly happen in programming framework as they quantify incessant redesigns/upgrades of the present framework to help comparative styles of new functionalities. The software engineer is regularly asked to use the present code by rehashing and adjusting to the new item necessities attributable to the high hazard (outcomes of programming framework mistakes will run into the millions amid a solitary day) of programming framework blunders in new sections and since existing code is as of now decently tried (70% of the product framework exertion inside the fiscal area is spent on testing). To make programming framework building design understandable, commonly clones are designedly acquainted with the framework [5]. As two cloned code parts measure independent of each option both grammatically and semantically, they will develop at very surprising paces in seclusion while not influencing the inverse and testing is scarcely expected to the changed part. Keeping cloned parts inside the framework could hence accelerate support; especially once programmed relapse tests are truant [8]

ANTLR is used in the algorithm because it peruses syntax and creates a recognizer for the dialect characterized by the linguistic use. On the off chance that there are no language structure lapses, then the default activity is to just passageway without printing any message. Any actions can be connected to language structure components in the linguistic use by are writing in the programming dialect in which the recognizer is being produced. At the point when the recognizer is being created, the activities are implanted in the source code of the recognizer at the suitable focuses. Activities can be utilized to manufacture and check image tables and to emanate directions in a target dialect, on account of a compiler. ANTLR peruses punctuation and produces a recognizer for the dialect characterized by the linguistic use (i.e. a program that peruses an information stream and creates a lapse if the data stream does not comply with the grammar tags.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

II. CLONE CHUNK ALGORITHM FOR CHAIN VIEW

The clone chunk algorithm is based on the algorithm given in the earlier work [9]. The work has been extended to find the more details of duplicated code. The algorithm will work as follows:

```
Step 1: make table fID-startpoint and startPoint-fID
        Find the project number of statement, N.
                Read each file from selected file and
                Add no of statements in file to the total project number of statement, N.
Step 2: Make a matrix of statements by using project number of statements, N.
        matrix= formedMatrix.
Step 3: Use the 'Collector' which had been initialized before, with the statementGroups
Step 4:  For each statement group
                Compare one statement with the others in the same bucket
                Size=size of present file statements.
                For i->0 to i->Size-1
                        Store statement in s= Statement(i),
                        For j->i+1 to j->Size
                                Store statement in s2= Statement(j)
                                Check if, s1 and s2 are same i.e s1=s2
                                        Then insert the result to matrix.
                                        Call putAMarkOnMatrix(i,j)
Step 5: Find the different chains from the statements.
        For line->0 to line->project number of statement, N.
                For column->line+1 to project number of statement, N.
                        If there is an entry in matrix at line,column then
                                initialize an index array that forms a chain.
                                maxLine= getMaxLine from the current stored lines.
                                maxCol = getMaxCol from the current stored Columns.
```

Now by the help of including the above algorithm in the collector module given in [9] the chain view of the clones can be available to the programmer so that he can identify it as a bad smell or not.

III. EXPERIMENT AND RESULT

In this paper the tool has been extended which was developed in [9] so that it can give the programmer more details about the cloned code. The number of duplicated code will be shown in three views i.e., Statistic view, chain view and file view as shown in figure 1. To check the performance of this tool the study has been done on JHotDraw which is open source software written in Java. The code files taken to detect the clones from JHotDraw are PERTProject, NETProject, DRAWProject and SVGProject. These files are same as taken in [9] so that clarity of the available clones can be seen. Figure 1 shows the detection of duplicated codes for PERTProject.java and NETProject.java in Statistic view.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

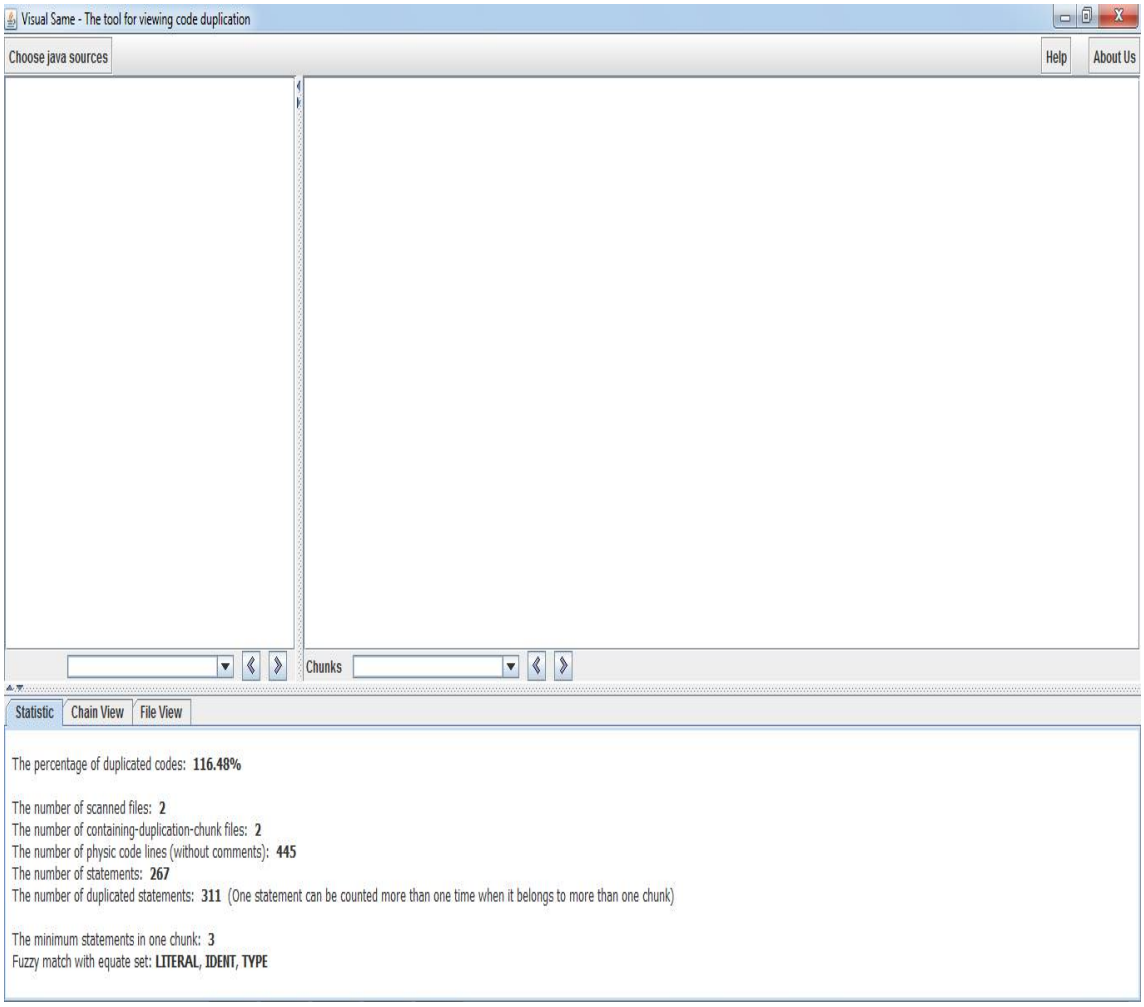

Figure 1 Statistic view

In this view there will be following information:

i) Percentage of duplicate code: It is the percentage of duplicated statements to the total number of statements in the system. The percentage will be more than 100% in the cases where a particular statement belongs to more than 1 chunk of duplicate statements. So, if the percentage is more than the user can get the alert there itself that the number of duplicated statements is much more in the system developed.

ii) Number of scanned files: This is the number of files that were selected from Choose Java source option.

iii) Number of containing-duplication-chunk files: It is the number of files having duplicate code.

iv) Number of physic code lines: It includes the total number of statements in the source code chosen.

v) Number of statements: This will tell the total number of statements in the code from which the duplicated code is to be detected.

vi) Number of duplicated statements: This statistics will show the total number of statements in the code which are duplicated. A single statement can be counted more than once if it belongs to multiple chunks.

vii) Minimum statements in a chunk: This is the count of minimum number of statements in the total duplicated chunks in the code.

Now if the user selects Chain View then the view will be as shown in figure 2. On the upper area the duplicated chunk will be displayed. Below this area there are two tabs namely chains and chunks. Along with this a table will be displayed which will show all the information regarding the duplicated chunks. The table is divided into five parts

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

namely ID which is the serial number, chains i.e. the information regarding the duplicated chunks, NDC which is the number of duplicated chunks for same type of code, NSC number of statements in a chunk which are duplicated and the percentage which tells how much portion of code is covered by a particular chunk.

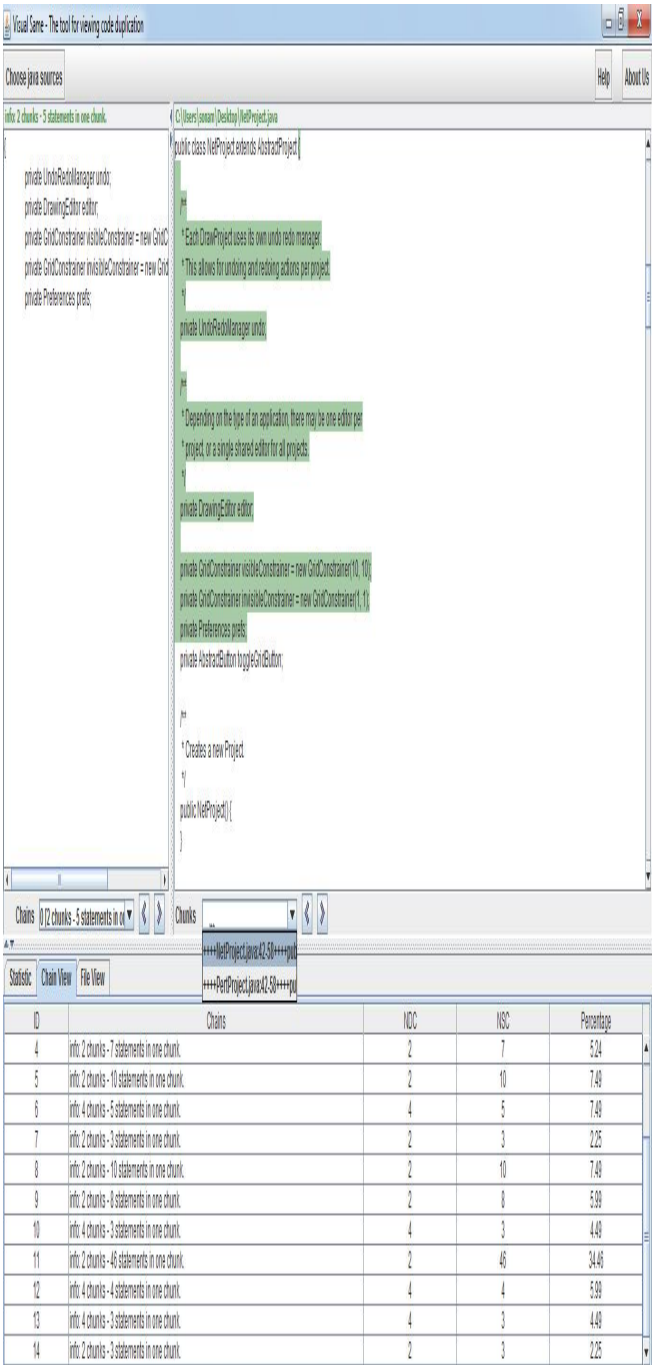

Figure 2 Chain View

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Similary the results of chain view can be analyzed as shown in table below.

	chunks	statements in a chunk
	SVGProject	
PERTProject	2	30
	4	4
	2	9
	4	5
	2	3
	2	10
	2	8
	4	3
	2	46
	4	4
	4	3
	NETProject	
PERTProject	2	5
	2	23
	3	4
	3	3
	2	7
	2	10
	4	5
	2	3
	2	10
	2	8
	4	3
	2	46
	4	4
	4	3
	DRAWProject	
PERTProject	2	30
	4	4
	2	9
	4	5
	2	70
	4	3
	4	4
	4	3
	2	3

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

The above table gives a clear view of the number of chunks of same code within two files selected for identifying the duplicated code. Along with number of chunks it also gives the number of statements in a particular chunk which contains cloned code. There are many detectors developed so far but none of them is able to give the view of cloned codes in such a clear manner so that the user can easily have the idea of presence of clones code along with its location.

IV. CONCLUSION

There are many approaches [7] that convert the detected clone code into a function and replace that cloned code from every file to that function. But the major disadvantage with this approach is that it causes to replace the unharmed clones also or the clones which have been inserted deliberately whereas the approach developed in this research helps the user to change the cloned codes according to his understanding. So by the help of this algorithm the programmer can easily analyze the cloned code with more details. Along with this the sixe of the chunks can also be calculated.
The results of the developed tool is based on the study of open source software system. So the future work will be to test the efficiency of the tool on the licensed software systems. The tool can detect the cloned codes written only in Java language only. This can be extended to detect the cloned codes from other languages as well.

REFERENCES

[1] M. M. Lehman and L. Belady, "*Program Evolution - Processes of Software Change*", London Academic Press, 1985.
[2] Lehman, M. M. and L. A. Belady (1985). Program Evolution: Processes of Software Change. London, Academic Press.
[3] Lientz, B. P. and E. B. Swanson (1981). "Problems in application software maintenance." Commun. ACM **24**(11): 763-769.
[4] Singer, J. (1998). Practices of software maintenance. Proc. Int'l Conf. on Software Maintenance (ICSM), IEEE Computer Society**:** 139-145.
[5] Cory Kaiser and Michael W. Godfrey. "Clones considered harmful". In Proceedings of the 13th Working Conference on Reverse Engineering (WCRE'06), pp. 19-28, Benevento, Italy, October 2006.
[6] J.R. Cordy. Comprehending reality: Practical challenges to software maintenance automation. In Proceedings of the 11th IEEE International Workshop on Program Comprehension (IWPC'03), pp. 196206, Portland, Oregon, USA, May 2003.
[7] Raghavan Komondoor. Automated Duplicated-Code Detection and Procedure Extraction. Ph.D. Thesis, 2003.
[8] Matthias Rieger. Effective Clone Detection Without Language Barriers. Ph.D. Thesis, University of Bern, Switzerland, June 2005.
[9] Gupta, Sonam, and P. C. Gupta. "Algorithm to Detect Non-Contiguous Clones with High Precision.", International Journal of Innovations in Engineering and Technology, Volume 5 Issue 1, February 2015.