# HYBRID TECHNIQUE FOR SOFTWARE CODE CLONE DETECTION

Priyanka Batta
Research fallow, Department of Computer Science
Lovely Professional University, Jalandhar
priyankabatta@gmail.com

Miss Himanshi
Assistant Professor, Department
Of Computer Science
Lovely Professional University, Jalandhar
Himanshi19.rpr@gmail.com

## ABSTRACT
Software Clone detection is one of the hottest research area that helps in detecting duplicate code from an applications. The research has shown that 5% to 20% of software systems can contain duplicated code that is generated by simply copying the existing program code and pasting with or without minor modifications. Cloning creates problem when a bug is found in one code segment that was copied and pasted at several locations earlier. The objective of this study is to analyze the working of hybrid clone detection technique that design and analyze a hybrid technique for detecting software clone in an application. We will combine metric approach with text base (line of code) technique for the above said reason. A model will be designed to automate the concept of clone detection.

## General Terms
Cloud Computing, Standard deviation, open source clouds etc.

## Keywords
Software Clone, Clone Detection, Metrics etc.

## 1. INTRODUCTION

Code reuse is a standard practice in modern software development. The downside of code reuse via replication and copy-paste programming is that it leads to code bloat, increasing the technical depth of software products and making maintenance costly and time consuming [1].

In software development process, it is widespread to reuse code fragments by merely copying and pasting with or without minor variation. Due to copy and paste operation software systems often contain sections of code that are very similar to one another, called code clones. Code clone [5] is defined as the fragment of code in source code file that is identical or similar to one another. The contemporary research shows that code cloning plays a significant role in large software applications. Code clone detection is one of the most likely research areas in the field of software engineering. Clone detection is live problem in industry and an active research area with plenty of work on detecting and removing clones from software. Code cloning is a perception in which source code is duplicated. In technical terms code cloning is the progression of replicating code fragments. Cloning is mainly used to implement the concept of software reusability. Detecting code clones in a Code base is a very difficult problem. The research studies on open source and commercial code shows that around 66% of cloned code is modified, i.e. it's not an identical clone. There are various clone detection techniques (Text based, Token based, Abstract Syntax Tree, Program dependency Graph, Metrics based, Hybrid etc.) that came into subsistence in past decades.

The pressure [3] in software industry, persuade many software developers to use already implemented and tested software by cloning it and then adapting it to meet the needed functionality. When the entire library is of no use or when the elements to reuse are individually relatively small, copy-and-paste reuse is one of the most favorable methods of software cloning for software developers. Copy and paste is simple in nature, fast, and beneficial at a first look.

Clones do not come into existence itself. The reason behind the occurrence of clone in the software is developer himself. The various reasons of introduction code clones in the software application are time limit, language limitation, reuse, accidental, developer performance and risk in new code etc.

There exist numbers of clone detection techniques for detecting clone of software code. The different software clone techniques are categorized as follows [6] [7]:

String based clone detection technique divides the program into strings and then compare the string with each other to find any clone if exist. Token based clone detection technique divides the program or code into set or tokens and then compare each token with other to detect clones.

Parse tree clone detection technique develops a parse tree of code to find clone. Metric based software clone detection technique computes various metrics to analyze the delicacy or clone in the code. Hybrid clone detection technique use two or more existing clone detection technique to find clone of code.

A clone detection technique that uses a combination of syntactic and semantic characteristics is called hybrid clone detection technique. The objective of my research is to design and analyze a hybrid technique for detecting software clone in an application. Software metrics [2] gives physical or conceptual measure of program or data.

In this study of clone detection first of all various software metrics like line of code, cyclic complexity, number of

operators, number of unique operator, number of operand and number of unique operands will be computed, if two application

program under observation has any similarity in above said metrics the clone detection is formed that compare each line of code of one application program with another application program to find any duplicity in code segment. The different metrics [4] for measuring the size of a program uses different variables n1, n2, N1, N2 the number of distinct operators, number of distinct operands, number of operators and number of operands respectively. Cyclometic Complexity is one of the important measures of programming that measure the complexity in the program. The concept of cyclometic complexity is given by McCabe and computed as V(G)=e-n+p. beside this It is one the basic static metric that is used to measure the size of code segment. It helps in measuring the cost of project in an effective way.

## 2. LITERATURE REVIEW

**Javier Carretero, et al** in **IEEE 2011** describes a hybrid post-Si approach to validate a modern load-store queue. An effective error detection mechanism is used and an expandable logging mechanism to observe the micro architectural activity for long periods of time, at processor full-speed. Validation is performed by analyzing the log activity by means of a diagnosis algorithm. [8]

**Iman Keivanloo, et al** in **19th IEEE International Conference** presents a hybrid clone search approach using source code pattern indexing, information retrieval clustering, and Semantic Web reasoning to respectively achieve short response time, handle false positives, and support automated grouping/querying.[9]

**Rochelle Elva and et al** of **University of Central Florida** in **March 2012** presents a tool and algorithm for the detection of semantic clones in Java methods. For this purpose, semantic clones are defined as functionally identical code fragments. Thus, detection process operates on the premise that if two code fragments are semantic clones, then their input-output behavior would be identical. They adopted a wholistic approach to the definition of input-output behavior by including not only the return values of methods; but also their effects as reflected in the pre- and post-states of the heap. They referred to this as a method's IOE (input, output and effects) behavior and tool and algorithm were tested in a small case study using the open source database management software. [10]

**Mohammed Abdul Bari, et al** in **International Journal of computer Applications, April 2011** proposed the background concept of code cloning. They presented overall taxonomy of current techniques and tools, and classified evolution tools in two different format as static code cloning and dynamic code cloning, this together presented with program analysis, secondly as a solution the static code is divided into four parts as T1, T2, T3, T4, to finally develop a process to detect and remove code cloning.[11]

**Zhuo Li, et al.** in **International Journal of Advancements in Computing Technology, 2011** proposed an iterative process of building up a metric space. Based on it, clone detection can avail us of more convenience and flexibility. They exercised the approach in a real industry project and compare detection quality between Euclidean distance and cosine angle distance. Euclidean distance is proven to be a more native and accurate approach. [12]

**Bakr Al-Batran1, et al.** in**, Technische Universität München, Germany 2011** explained the techniques for the automatic detection of clones to improve their maintainability. As these approaches currently only consider syntactic clones, the detection of clones is limited to syntactically equivalent copies. Using the concept of normal forms, these approaches can be extended to also cover semantic clones with identical behavior but different structure. They presented a generalized concept of clones for Simulink models, described a pattern-based normal-Form approach, and discuss results of the application of an implementation of this approach. [13]

**D.Gayathri Devi, et al.** in **IJCSE 2011** compared the metrics based on speed, quality and their cost as these are the most that affects due to code clone. The strength and weakness of some of the metrics based are compared and evaluated. They suggested an approach based on metric and the quality goals are accessed by the use of metrics. [14]

**Kodhai Perumal and Kanmani** in **IJCCIS, 2010** proposed combination of textual and metric analysis of a source code for the detection of all types of clone in a given set of fragment of java source code. Various semantics had been formulated and their values were used during the detection process. This metrics with textual analysis provides less complexity in finding the clones and gives accurate results. [15]

**Mark Gabel , et al.** in **ICSE, May, 2008, Leipzig, German** explained the first scalable clone detection algorithm that is based on the definition of semantic clones .the main aim of their was the reduction of the difficult graph similarity problem to a simpler tree similarity problem by mapping carefully selected PDG sub graphs to their related structured syntax .they solved the tree similarity problem to create a scalable analysis .An algorithm have been implemented, that is a practical tool and performed evaluations on several million-line open source projects, including the Linux kernel.[16]

**NORFARADILLA BINTI WAHID** of **University Technology Malaysia, OCTOBER 2008** proposed the technique of ontology mapping to solve the problem of find out the possibility, but they were not using the real ontology for the clone detection. In their paper, the clone detection is using two layers of detection; i.e. structural similarity and string based similarity. The structural similarity is by using sub graph miner where it capable to get the similar sub tree between different files. And then they extracted all elements of that particular sub tree and treat the elements as a string. Two strings from different files then applied with similarity metric to know whether it is a clone pair. From their experimental result, it was found that the system is language independent but the result is good in precision but not so good recall. It was also capable to detect two main types of clone, i.e. identical clones and similar clones. [17]

**Filip Van Rysselberghe and Serge Demeyer IDE's supporting refactoring 2003** compared three representative detection techniques (simple line matching, parameterized matching, and metric fingerprints) by means of five small to medium cases and analyzed the differences between the reported matches. Based on this experiment, they conclude that (1) simple line matching is best suited for a first crude overview of the duplicated code; (2) metric fingerprints work best in combination with a refactoring tool that is able to remove

duplicated subroutines; (3) parameterized matching works best in combination with more fine-grained refactoring tools that work on the statement level. [18]

## 3. PROBLEM DEFINTION

We want to propose a hybrid clone detection model with user friendly interface that is capable of detecting clone in software by analyzing the various software metrics and then comparing each line of one file (application program) with every line of second file(application program). The various objective of this study are:

- To understand the concept of software clone
- To understand the need and working of software clone detection techniques
- To gain knowledge about hybrid clone detection technique.
- To design an algorithm for detecting clones in the program using hybrid clone detection technique
- To develop user friendly practical model for detecting clones

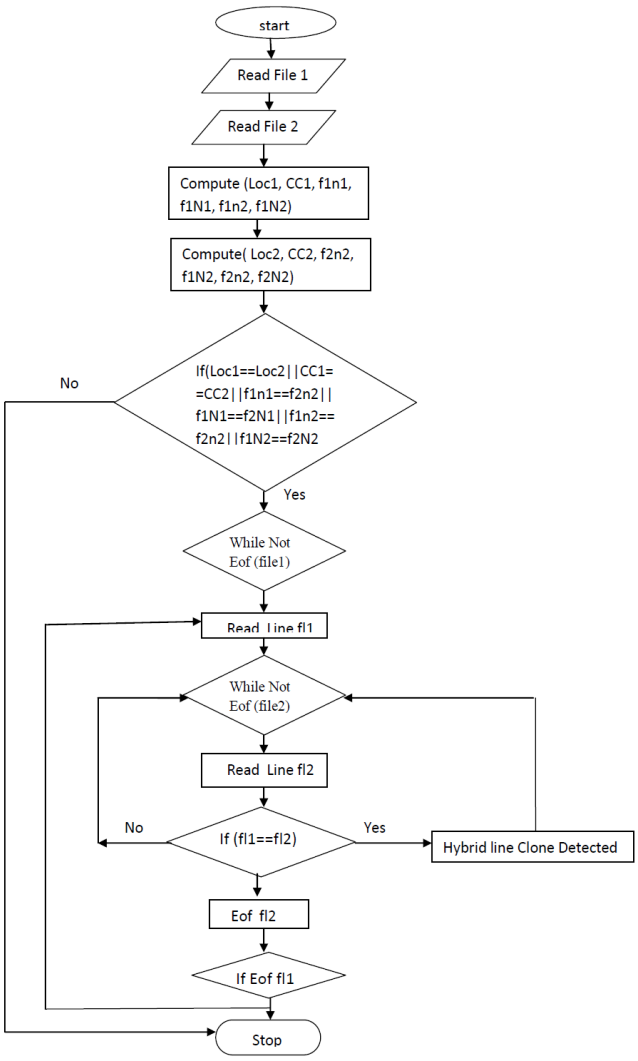
## 4. PROPOSED MODEL

We have developed the user friendly interface of hybrid clone detection by using one of most commonly used programming language i.e. Visual Basic 6.0. The code is based on the algorithm as given below:

```
Read file#1
Read file#2
Compute loc1, CC1, f1n1, f1N1, f1n2, f1N2
Compute loc2, CC2, f2n2, f2N2, f2n2, f2N2
If (loc1==loc2||CC1==CC2||   f1n1==f2n2
||f1N1==f2N1||f1 n2==f2n2||f1N2==f2N2) Then
While not EOF (File1)
        Read line fl1
        While not Eof (file2)
            Read line fl2
            If (fl1==fl2) Then
                Hybrid line Clone Detected
                Print line fl1
                Exit
            Else
                Read next line
            End if
        While end
        Read next line
    While end
Else
    Msgbox "Metrics does not match"
End if
```

In this algorithm, the line of code of first application program is compared with line of code of second application program. Here file#1 and file#2 are used to indicate first and second application program respectively. In this algorithm we are comparing every line of code of first application program with each line of second application program. But before actual comparison, the metrics of two application program are computed and based upon the analyzed metrics the comparison is carried out. So in this case metrics is of major consideration. After computation of the metric the actual comparison is carried out that will select the cloning of line code if exist.

## FLOW CHART OF PROPOSED MODEL




The visual user friendly interface based on above algorithm is as shown in the following diagrams:

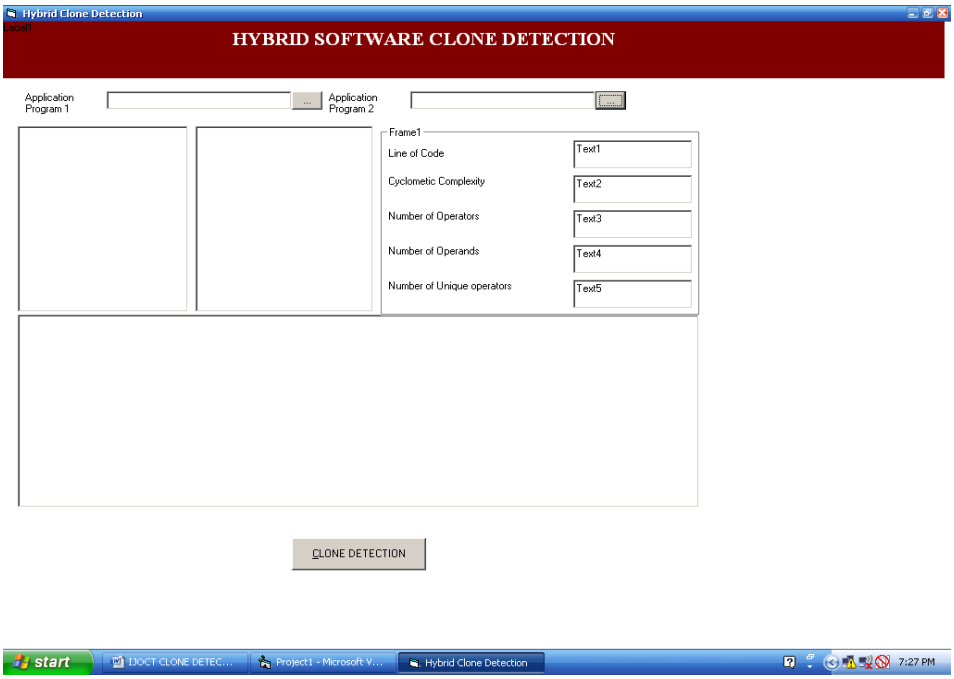



Figure1: Visual User Interface for Clone Detection

The following screenshot shows how the above designed model will upload the two application program, compute their required metrics and then comparing and analyzing it for any clone if exist. The interface will show the clones in the form of software code lines.

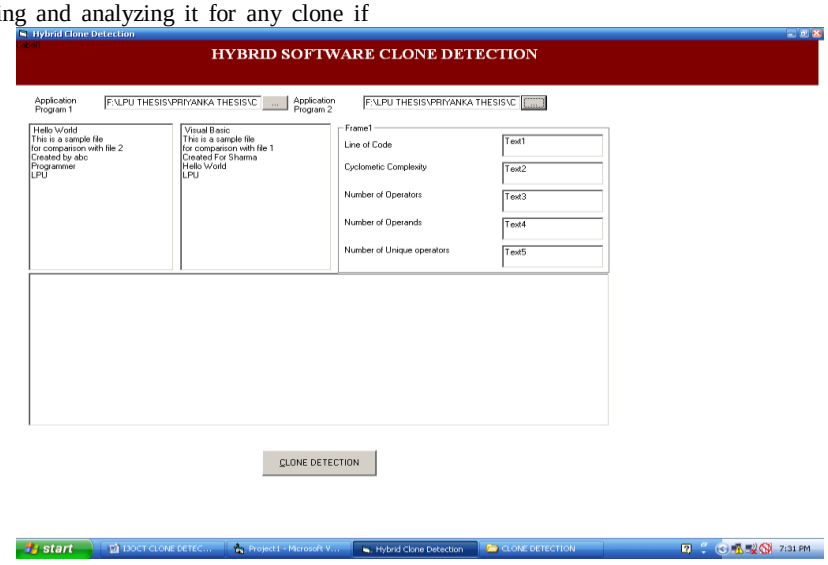



Figure: Uploading application program files

After uploading the required application program file the designed model will compute the various metrics as discussed above. The following screen shot show the analysis of various metrics:

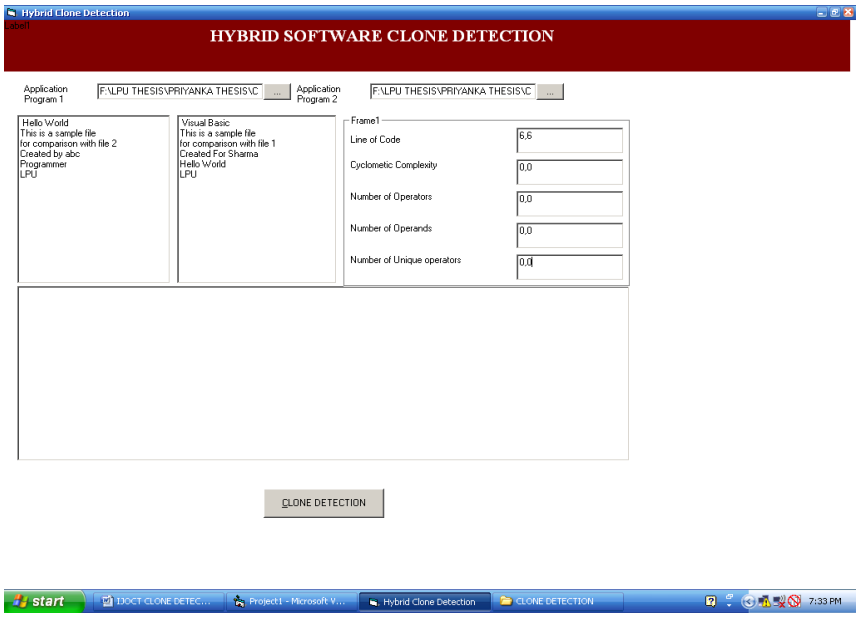



Figure: Metric Analysis

After computing the required metrics the designed model is used to detect clone by using this model

## 5. ANALYSIS:

Let us consider two input files prog1 and prog2 as given below:

Prog1

```
#include<stdio.h>
#include<conio.h>
Void main ()
{
    printf ("hello");
    Printf ("data in file 1");
    getch ();
}
```

Prog2:

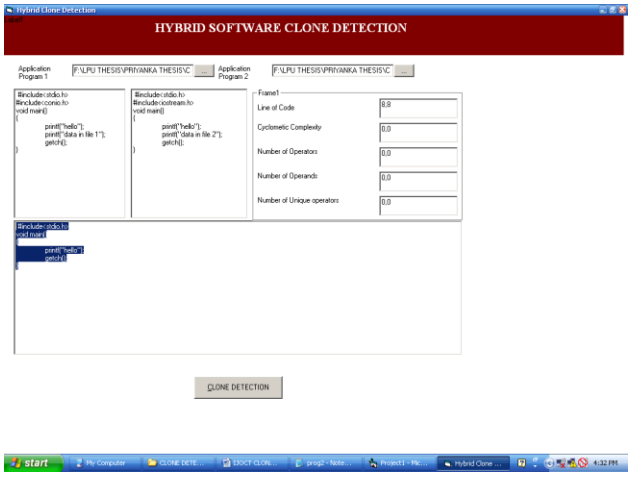
```
#include<stdio.h>
#include<iostream.h>
Void main ()
{
    Printf ("hello");
    Printf ("data in file 2");
    getch ();
}
```

The proposed system will compute the required metrics and detect the clone as an output as follow:

**Clone detected**

```
#include<stdio.h>
Void main ()
{
    printf ("hello");
    getch ();
}
```

The visual interface of above said complete procedure is given below:




## 6. CONCLUSIONS

Code duplication is an important factor that drives up maintenance costs. No software industry branch is immune from this problem. The problem is present in all areas where large software projects are developed, including the Open Source Software arena. Recent advances in code duplication techniques and tools make the detection and measurement of duplication possible with little effort. Such tools and techniques can bring high cost reductions for a little price, if used in the right way. We have designed and implemented a code clone detection system that helps in detecting clones in the code efficiently and effectively. The proposed system helps in removing the problems occurred due to software clone like increase maintenance work and cost, increased defect probability, increased resource requirement, increased probability of bad design etc.

## 7. FUTURE WORK

The proposed model is developed and implemented in visual basic 6.0 that runs under window platform. The designed system is able to detect code clone in C/C++ language only .The system can be modified to run on number of platform like windows, UNIX, Linux, Solaris etc. Further in future this system can be able to detect code clone in application developed under different programming languages like C/C++, Java, COBOL, PASCAL, SQL, and HTML etc. In the current analysis of hybrid clone detection model, we are using static metrics. Static metrics are being used for clone detection. These are simple to calculate and can be compared quickly. But can also lead to false positives. So, these can be further used as dynamic metrics for detecting clones in coding. A combination of these techniques can be used in future for detecting clones.

## 8. REFERENCES

[1] http://patterninsight.com/products/clone-detection (online).

[2] Manik Sharma, Chandni Sharma et. al. "Comparative Study of Static Metrics of Procedural and Object Oriented

Programming Languages", IJOCT, February 2012.

[3] http://www.solidsourceit.com/blog/?tag=code-clone (online)\

[4] Manik Sharma, Gurdev Singh, "A Comparative Study of Static Object Oriented Metrics"Vol. 3 No.1 (January 2012), IJoAT.

[5] Toshihiro Kamiya, Shinji kusumoto, "A Token based code clone detection toll ccfinder and its empirical evaluation", Technical report, 2000.

[6] Brooks, F., The mythical man month: essay on software engineering, Reading mass: Addison-Wesley Publishers.

[7] A.M. Leitao, "Detection of redundant code uses R2D2" in software quality journal, Vol. 12, No.4, pp. 361-382., 2004.

[8] Javier Carretero, et al " Hardware/Software-Based Diagnosis of Load-Store Queues Using Expandable Activity Logs", 2011 IEEE

[9] Iman Keivanloo, et al "SeClone - A Hybrid Approach to Internet-scale Real-time Code Clone Search", 2011 IEEE

[10] Rochelle Elva and et al "JSCTracker: A Semantic Clone Detection Tool for Java Code "March 2012

[11] Mohammed Abdul Bari, et al" Code Cloning: The Analysis, Detection and Removal" International Journal of Computer Applications (0975 – 8887) Volume 20– No.7, April 2011

[12] Zhuo Li, et al" A Metric Space based Software Clone Detection Approach and Case Study" International Journal of Advancements in Computing Technology Volume 3, Number 7, August 2011

[13] Bakr Al-Batran1, et al." Semantic Clone Detection for Model-Based Development of Embedded Systems"

[14] D.Gayathri Devi,et al."Comparison and evaluation on metrics based approach for detecting code clone" Vol. 2 No. 5 Oct-Nov 2011

[15] Kodhai Perumal.and Kanmani" Clone Detection using Textual and Metric Analysis to figure out all Types of Clones" *Vol2. No1. ISSN: 0976–1349 July – Dec 2010*

[16] Mark Gabel , et al."Scalable Detection of Semantic Clones" *ICSE'08,* May 10–18, 2008, Leipzig, Germany.

[17] Norfaradilla binti wahid"code clone detection using string based tree matching technique"

[18] Filip Van Rysselberghe and Serge Demeyer IDE's supporting refactoring" Evaluating Clone Detection Techniques"