Implementation of HTSCCDUToLiA (Hybrid Technique for Software Cloning Code Detection using Token and Line based Approach)

Sheetal
Department of Computer Science & Engg
Sri Sai College of Engg and Technology,
Manawala, Amritsar, Punjab, India

Rimmy Chuchra
Department of Computer Science & Engg
Sri Sai College of Engg and Technology,
Manawala, Amritsar, Punjab, India

ABSTRACT

The criteria or procedure of software development is going to be changing continuously by changing time. Some of the new functions may be added, modified or deleted according to the requirement of the users. Through the continues changes, the concept of software evolution is originated. These regular changes in any software during development are a very complex process. As the presence of same code more than one time is if increases then the quality of that particular software are automatically degraded. As founded by the authors in their previous study the maximum use of copy paste programming during software development may also effect on some other factors as an example software maintainability, software reusability, software performance, maintenance cost and overall software performance. The presence of code clones makes the software maintenance extremely difficult. So, to detect software clones during the development of any software or after the development of that particular software is a mandatory or priority for any software developer. Code clones identification thus becomes extremely necessary for improving the maintenance, reusability, performance and in other words, you may say for improving the overall quality of any software. Several studies show that about 5% to 20% of software systems can contain duplicated code even without doing little or minor modifications. As I studied in my journey of research there are various types of techniques are available in the software industry which are used for software clone detection viz. Token based Approach, Line based Approach, Lexical based Approach, Program Dependency Method, Abstract Syntax Tree based Method, Text Based Method and Metric Based Method. Every technique has following different criteria for detecting different type of clone viz. Type 0, Type 1, Type 2, Type 3 and Type 4.Everytype of software clone lies in own distinct software class so that can be described with different definitions as an example Type 0 performs exact cloning, Type 1 clone is used to detect identical copies except comments, Type 2 clone is used to detect some of the user defined names as like literal names and function names, Type 3 clone is used to detect added, deleted or interchanged lines and Type 4 Clone is used to detect unintentional or un-knowingly presence of similar code or in other words some software developers say auto-generated code which is too difficult to detect. This research paper has focused on to detect and eliminate the software code clones that are present in any software more than one time. I proposed a new methodology which is termed as “A Hybrid Technique for software code clone detection by using Token Based and Line Based Approach” (HTSCCDUToL$_i$A) whose main purpose is to detect different types of clones viz. Type 0, Type 1 and Type 2 clones etc. This newly designed methodology working is based on two different types of software cloning approaches viz. Line based Approach and Token Based Approach. This new designed hybrid method will produce more efficient results than already existing techniques. The major objective of this paper is to remove redundant code or free space which is covered by the comment lines especially. The main significance to propose this hybrid technique is to automatic detection of different software code clones within minimum duration of time. Different parameters are considered for software code clone detection in different tables as an example size of code, type of clone, efficiency and portability etc. In addition, at the last the percentage of code clone detection is also calculated by utilizing a different comparison parameter. In addition, the major benefit to design this new hybrid technique is to save software developer time, computer memory space as well as developer effort. By utilizing this new designed methodology the amount of code clones under a specific project or specific application can be easily reduced or removed up to some extent that will ultimately increases the overall performance of the software. In this way, this new designed methodology, in future will helpful for producing more consistent or more efficient results. And hence, the different software parameters viz. software reusability, software maintenance, software performance and overall software quality can be easily improved and easily managed.

Keywords

Line Based Approach, Token based Approach, Hybrid Technique Time, Space, Software Developer, Lines of Code (LOC), Types of clone, clone percentage, Software Cloning, Input File, Efficiency and Portability.

1. INTRODUCTION

As the dependency of the users on the cyber world is increases because of most of the tasks are handled over the cyber media. Several different types of software’s are available in a variety of disciplines as example software for a web development, mobile development, application development, back-end development. Data science, API (application program Interface) development, Software Tool development, embedded system development and security software development etc. Software developer’s uses different types of programming languages in the front end as a platform for developing several types of software applications. As noticed in the journey of review of literature most of the times copy paste programming is preferred because of it saves developer time as well as effort but it actually degrades the quality of the software. So, authors suggest to avoid this type of copy paste programming. During software development there may be a case where the same line of code (LOC) is used more than a one time. This duplicate code generates clone which most commonly developers called “Software Clone” [1]. The presence of clone increase maintenance cost, Decrease reusability and decrease reliability which will overall degrade the quality of the developed software. So, to overcome this problem and improves reusability, reduced maintenance cost, save developer time, developer effort and computer memory space. A new technique is proposed "Hybrid Technique for software code clone detection by using Token Based and Line Based Approach" (HTSCCDUToL$_i$A). The main motive of this newly proposed hybrid technique is to detect different types of clones viz. type 0, Type 1 and Type 2 clone [12]. The complete working of this new designed technique is initially based on Line based approach and Token based Approach. The 2 main parameters consideration is taken into account by the authors is an important factor in this research paper. The improved percentage of detected clone, improved efficiency and improved portability of newly designed hybrid technique shows it's more satisfactory or effective results than already existing software cloning techniques viz. Text based, Token Based[11], Lexical based, Line Based, Program dependency Graph[8], Metric based[11][13] and Abstract syntax Tree[11] etc. The main motive to implement this new hybrid technique is to detect several different types of clones' viz. type 0, Type 1 and Type 2[12]. The major objective of this research paper is to improve software reusability, software maintainability and software code understandability. The main benefit of this new designed hybrid technique is to save developer time & effort, computer memory space. In the way, by reducing maintenance cost and improving software reusability performance of the software is automatically improved which will later on helpful for improving the overall quality of the software (QOS). Hence, the major objective of this research paper is achieved. When IT Professionals or software developers utilize this newly proposed hybrid technique they will definitely achieve more effective or satisfactory results while saving their time and effort. Hence, the overall quality of the software is automatically improved.

2. REVIEW OF LITERATURE

FazalulHaque& Syed Mohd et al (March-April 2017) [5]:- This paper proposes a clone identification technique whose main purpose is to search and detect the parts of software clone which is identical. The major involvement of clones actually degrades the quality of the software, readability, changeability etc. The main aim of this paper is to reduce the time and effort of software developer. In addition, this paper also proposes generic technique which is used to detect clone from various input source codes as an example from web and disk etc.

Sandeep Bali and Sumesh Sood et al (May-June 2017) [2]:- This paper discusses about Hybrid Technique for Clone Detection that actually combines further different-2 clone detection techniques for giving better results in terms of precision and recall for achieving more accuracy in results. For detailed studies authors considered two different types or levels of metrics for improving proposed hybrid approach used for clone detection as an example function level metrics and class level metrics. The parameters considerations for different types or levels of metrics must be different as an example for function level metrics software developer considers LOC (Lines of Code), NI (Number of Invocations), NTIG (Number of times Invoked by methods non-local to its class) and NOS (Number of Statements) etc & for class level metrics software developer considers NOC (Number of Classes directly inherited from the given class), Pri A (Number of Private attributes), Pro A(Number of Protected attributes), NA (Number of Attributes), RFC (Response set of a class consists of set 'M' of methods of class), WMC (Weighted Method for Class), SIZE 2 (number of Attributes + Number of Local Methods), LOC (Lines of Code) etc.

Sreenivasa Reddy and Syed MohdFazalul Haque et al (July 2017) [4]:- Authors proposes a framework named Extensible Software Clone Detection Framework using ontology concept whose main motive is to detect clones with the help of ontologies concept. The main significance to propose such type of framework is it must be of user friendly nature and may supports multi-language.

Jahid Ali and Gurwinder Singh et al (September 2017) [1]:-In this paper, authors discusses about various existing clone detection methods as an example textual comparison, token comparison, Comparison of abstract syntax trees, Suffix trees and program dependency graphs. The main motive of authors is to present survey of all existing software clone detection techniques and develop a tool which must be user friendly, easy to maintain. The main significance to propose such type of tool is to provide a platform for independent system.

Gurwinder Singh and Jahid Ali et al (2017) [3]:- Authors proposed an efficient Clone Detection Tool which is used to detect clones in different programming languages. The main motive to develop such type of tool is enhancing the performance metrics such as recall and precision. The results demonstrate that the proposed tool helps to outperform as compare to traditional tools, which are shown by simulations using Netbeans. In addition, authors also review about different types of clone detection tools as an example Dup which is Token Based, CLAN which is metrics based, CC Finder which is also works for Token based, Dup LOC which is work for text based etc.

Heejo Lee and Hakjoo Oh et al (2017) [6]:- A new technique named "VUDDY" is proposed in this research paper. The main benefit to propose such type of technique is to detect security vulnerabilities. A billion lines of code is pre-processed very shortly. The main significance to propose this technique is to improve the readability of code.

Sukhpreet Kaur and Manpreet Kaur et al (April-June 2017) [7]:- This paper utilizes two different techniques for clone detection viz. ant colony optimization and Neural Network Classifier. The different parameters considerations are taken by different authors in the MATLAB simulation environment. As previous authors study founded, Back Propagation Algorithm gives more accurate results with faster training and testing of Neural Network by considering various performance metrics as an example false acceptance rate(FAR), False Rejection Rate (FRR), Recall, Precision and accuracy etc.

Nguyen H.A et al. (2017) [8]:-A new technique has been introduced which computes tree editing script, to detect and update clones of code. The previous study shows JSync is just like open source system which shows its better efficiency and accuracy in the clone detection concept.

Zibran M.F et al. (2016) [9]:- An effort model is proposed for refactoring clones of code in Object oriented and procedural source code. The risks of refactoring can be detected by priority scheme. Different types of techniques are used for refactoring clone. A combination of AI and OR has been maximum used as data collected by the authors in their survey because of it produces effective results when solving scheduling problems.

Meena Bharti and Rajan Goyal et al (December 2014)[10] :- In this paper, authors discusses about repeatedly this clone finding activity degrades the quality of the software and hence these duplicate fragments automatically decrease the overall maintenance cost of that particular software.

3. RESEARCH DESIGN

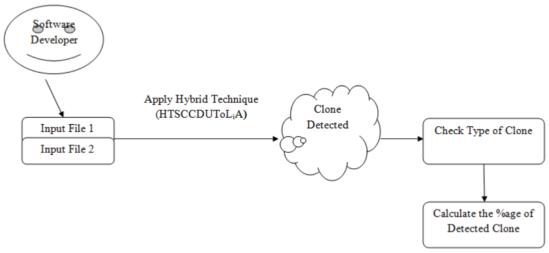


Figure.No.1:- A Process for Software Clone Detection using HTSCCDUToLiA(Hybrid Technique for Software Cloning Code Detection Using Token and Line Based Approach).

4. PROPOSED METHODOLOGY (HTSCCDUToL$_i$A)

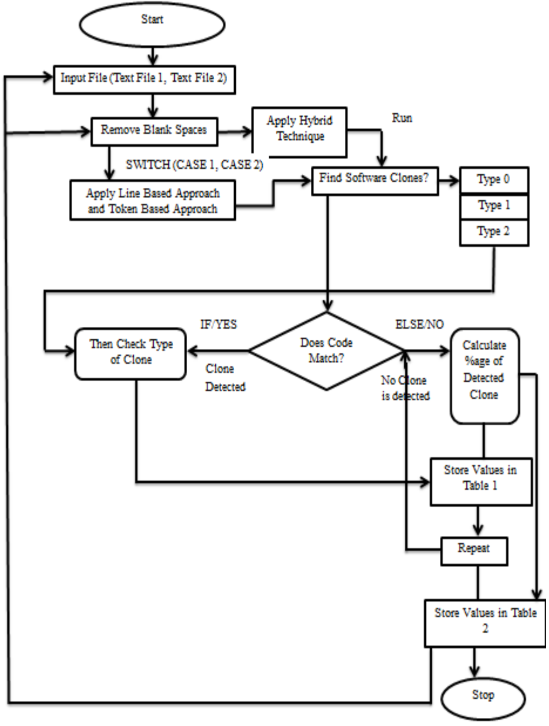


Figure 1: A Roadmap for Hybrid Technique for software code clone detection by using Token Based and Line Based Approach (HTSCCDUToL$_i$A).

5. STEPS ENABLED FOR PROPOSED METHODOLOGY (HTSCCDUToL$_i$A)

The following Steps of Proposed Technique is given below:-

Step-1) Input Text (File 1, File 2).

Step-2) Remove Blank Spaces.

Step-3) Apply SWITCH (CASE 1: Line Based Approach, CASE 2: Token Based Approach).

Step-4) Find Software Clone Types Type {0, 1, 2}.

IF (Code: = Matched)

Then Check Type of Clone.

ELSE Calculate the %age of Detected Clone.

Step-5) Store values of Detected Clones in Table 1.

Step-6) After that, Repeat ELSE Portion of Step 4 and Store Values in Table 2.

Step-7) Repeat Step 1 & Step 2.

Step-8) Apply Hybrid Technique.

Step-9) Repeat Step 4 to Step 6.

Step-10) End of the Algorithm.

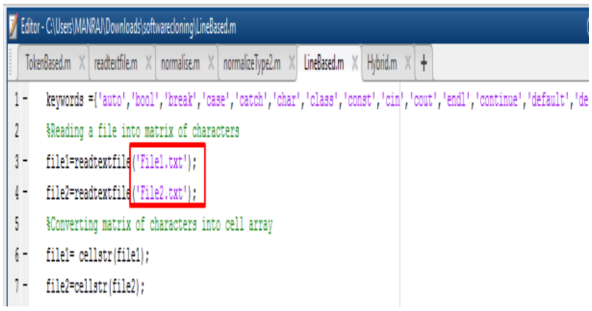
6. IMPLEMENTATION OF HYBRID TECHNIQUE (HTSCCDUToL$_i$A)

This implementation portion is divided into 2 sections. In section 1, results of this newly proposed technique is discussed and in the next section 2, discussion over these results is performed.

In this result section, the newly proposed technique named Hybrid Technique for software code clone detection by using Token Based and Line Based Approach (HTSCCDUToL$_i$A) is shown step by step. The main motive of this new technique is to improve the overall performance of the algorithm. The code at front end is written by the developer in any programming language viz. C, C#, Fortran and Python and at backend data is stored in the form of table's viz. Table 1 and table 2 in the database.

The following Steps of Proposed Hybrid Technique is given below:-

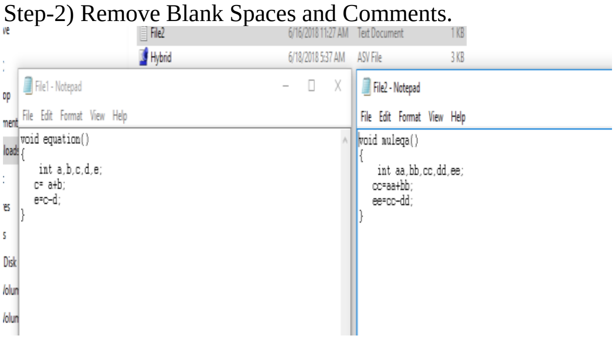
Step-1) Input Text (File 1, File 2).



```
keywords ={'auto','bool','break','case','catch','char','class','const','cin','cout','endl','continue','default','de
%Reading a file into matrix of characters
file1=readtextfile('File1.txt');
file2=readtextfile('File2.txt');
%Converting matrix of characters into cell array
file1= cellstr(file1);
file2=cellstr(file2);
```

Figure.1: Screenshot for Input File 1 & 2.

Step-2) Remove Blank Spaces and Comments.

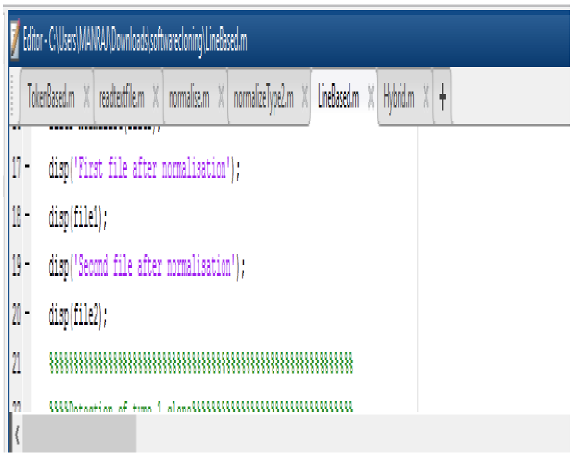

File2 6/16/2018 11:27 AM Text Document 1 KB
Hybrid 6/18/2018 5:37 AM ASV File 3 KB
File1 - Notepad
File Edit Format View Help
File2 - Notepad
File Edit Format View Help

```
void equation()
{
   int a,b,c,d,e;
  c= a+b;
  e=c-d;
}
```

```
void muleqa()
{
   int aa,bb,cc,dd,ee;
  cc=aa+bb;
  ee=cc-dd;
}
```

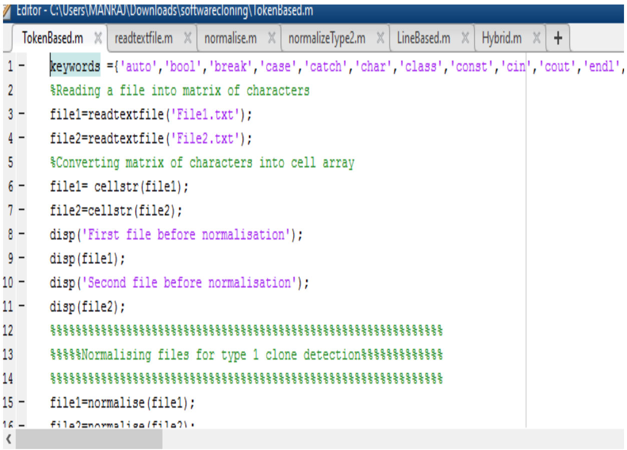
Figure.2: Screenshot for Removed Blank Spaces.

Step-3) Apply SWITCH (CASE 1: Line Based Approach, CASE 2: Token Based Approach).



```
disp('First file after normalisation');
disp(file1);
disp('Second file after normalisation');
disp(file2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Detection of type 1 clone%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure.3: Screenshot for Line Based Approach.



```
keywords ={'auto','bool','break','case','catch','char','class','const','cin','cout','endl',
%Reading a file into matrix of characters
file1=readtextfile('File1.txt');
file2=readtextfile('File2.txt');
%Converting matrix of characters into cell array
file1= cellstr(file1);
file2=cellstr(file2);
disp('First file before normalisation');
disp(file1);
disp('Second file before normalisation');
disp(file2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%Normalising files for type 1 clone detection%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
file1=normalise(file1);
file2=normalise(file2);
```

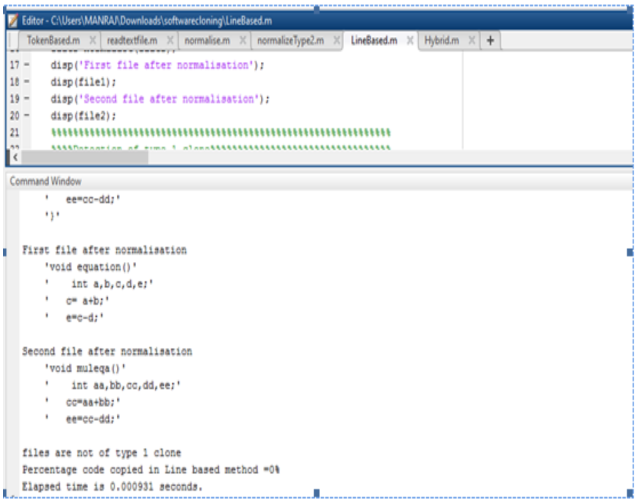
Figure.4: Screenshot for Token Based Approach.

Step-4) Find Software Clone Types Type {0, 1, 2}.

IF (Code: = Matched)

Then Check Type of Clone.

ELSE Calculate the %age of Detected Clone.



```
disp('First file after normalisation');
disp(file1);
disp('Second file after normalisation');
disp(file2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Detection of type 1 clone%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

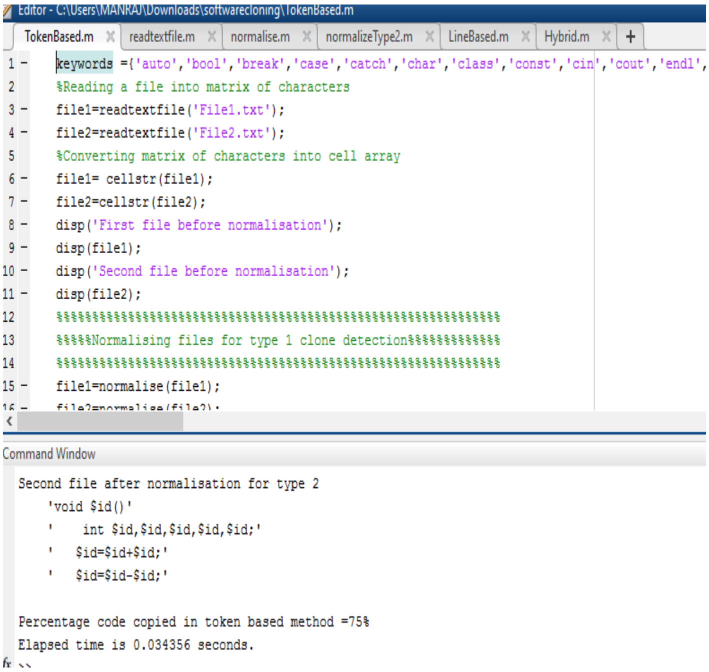
```
    '   ee=cc-dd;'
    '}'

First file after normalisation
    'void equation()'
    '    int a,b,c,d,e;'
    '   c= a+b;'
    '   e=c-d;'

Second file after normalisation
    'void muleqa()'
    '    int aa,bb,cc,dd,ee;'
    '   cc=aa+bb;'
    '   ee=cc-dd;'

files are not of type 1 clone
Percentage code copied in Line based method =0%
Elapsed time is 0.000931 seconds.
```

Figure.5: Screenshot for %age Copied Clone in Line Based Method.



```
keywords ={'auto','bool','break','case','catch','char','class','const','cin','cout','endl',
%Reading a file into matrix of characters
file1=readtextfile('File1.txt');
file2=readtextfile('File2.txt');
%Converting matrix of characters into cell array
file1= cellstr(file1);
file2=cellstr(file2);
disp('First file before normalisation');
disp(file1);
disp('Second file before normalisation');
disp(file2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%Normalising files for type 1 clone detection%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
file1=normalise(file1);
file2=normalise(file2);
```

```
Second file after normalisation for type 2
    'void $id()'
    '    int $id,$id,$id,$id,$id;'
    '   $id=$id+$id;'
    '   $id=$id-$id;'

Percentage code copied in token based method =75%
Elapsed time is 0.034356 seconds.
fx >>
```

Figure.6: Screenshot for %age Copied Clone in Token Based Method.

Step-5) Store values of Detected Clones in Table 1.

Table 1 :- Nomenclature for Software Clone Detection.

S.NO.	Software Clone Detection Technique	Input File Name	Input File Size	Type of Clone
1	Line Based Approach	Text File 1	1KB	0
2	Token based Approach	Text File 1	1KB	1
3	Hybrid Technique	Text File 1	1KB	1,2
4	Line Based Approach	Text File 2	1KB	0
5	Token based Approach	Text File 2	1KB	1
6.	Hybrid Technique	Text File 2	1KB	1,2

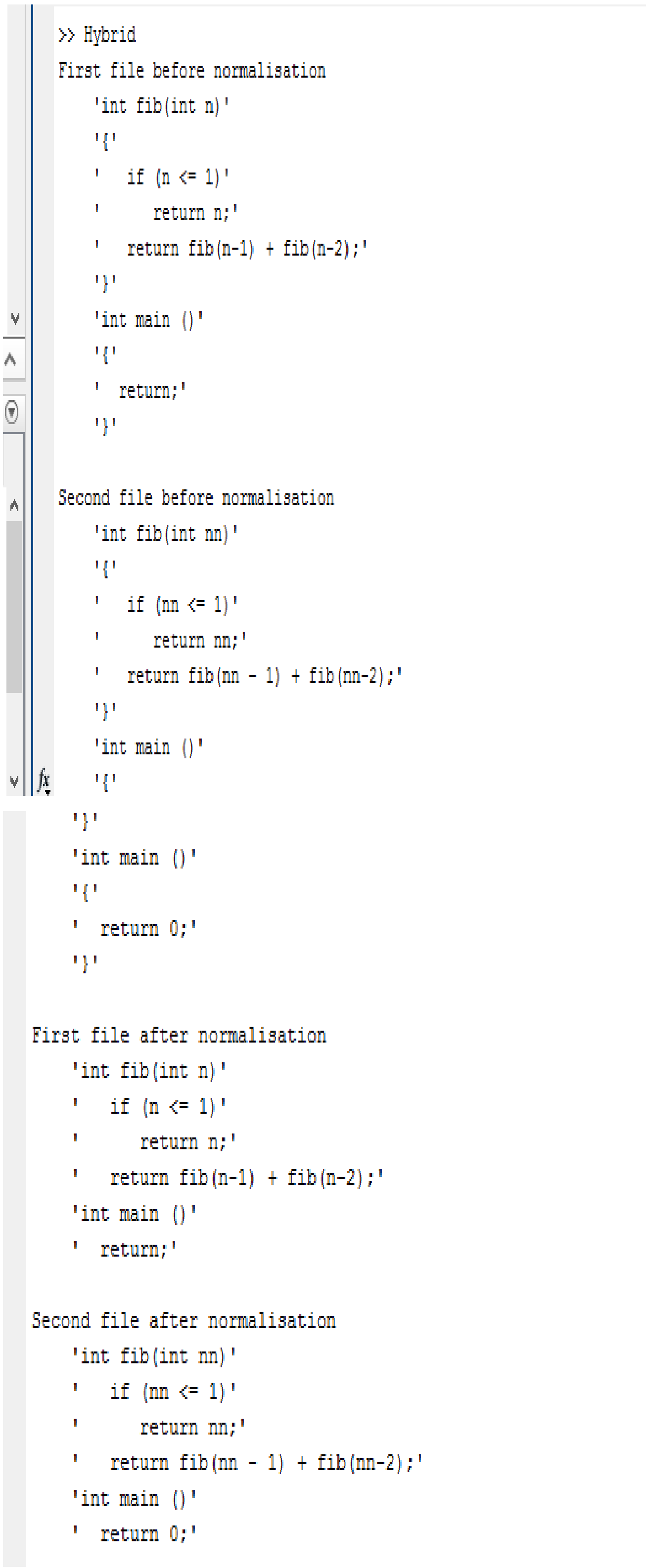
Step-6) After that, Repeat ELSE Portion of Step 4 and Store Values in Table 2.

Table 2:- Calculated %age of Detected clones by applying Different Techniques.

S.NO.	Software Clone Detection Technique	%age of Detected Clone	Efficiency(High, Average, Low)	Portability(Good, Average, Poor)
1	Line Based Approach	0%	Low	Poor
2	Token based Approach	75%	Average	Average
3	Hybrid Technique	83%	High	Good

Step-7) Repeat Step 1 & Step 2.

Step-8) Apply Hybrid Technique.

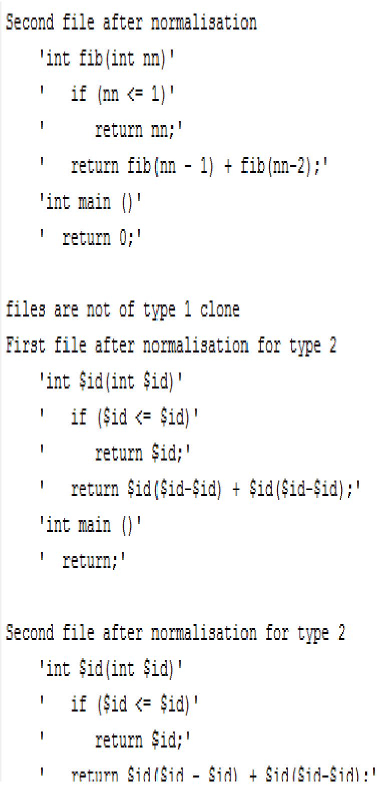


```
>> Hybrid
First file before normalisation
    'int fib(int n)'
    '{'
    '   if (n <= 1)'
    '      return n;'
    '   return fib(n-1) + fib(n-2);'
    '}'
    'int main ()'
    '{'
    '  return;'
    '}'

Second file before normalisation
    'int fib(int nn)'
    '{'
    '   if (nn <= 1)'
    '      return nn;'
    '   return fib(nn - 1) + fib(nn-2);'
    '}'
    'int main ()'
    '{'
    '}'
    'int main ()'
    '{'
    '  return 0;'
    '}'

First file after normalisation
    'int fib(int n)'
    '   if (n <= 1)'
    '      return n;'
    '   return fib(n-1) + fib(n-2);'
    'int main ()'
    '  return;'

Second file after normalisation
    'int fib(int nn)'
    '   if (nn <= 1)'
    '      return nn;'
    '   return fib(nn - 1) + fib(nn-2);'
    'int main ()'
    '  return 0;'
```

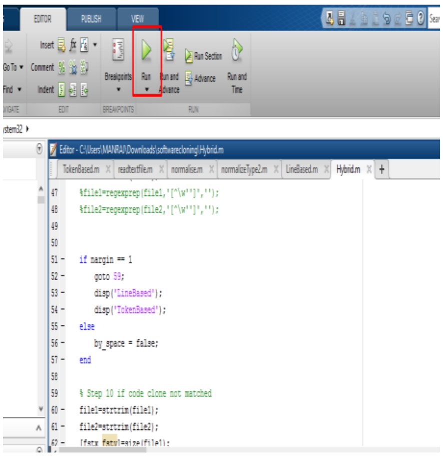


```
Second file after normalisation
    'int fib(int nn)'
    '   if (nn <= 1)'
    '      return nn;'
    '   return fib(nn - 1) + fib(nn-2);'
    'int main ()'
    '  return 0;'

files are not of type 1 clone
First file after normalisation for type 2
    'int $id(int $id)'
    '   if ($id <= $id)'
    '      return $id;'
    '   return $id($id-$id) + $id($id-$id);'
    'int main ()'
    '  return;'

Second file after normalisation for type 2
    'int $id(int $id)'
    '   if ($id <= $id)'
    '      return $id;'
    '   return $id($id - $id) + $id($id-$id);'
```

Figure.9: Screenshot for Proposed Hybrid Technique.

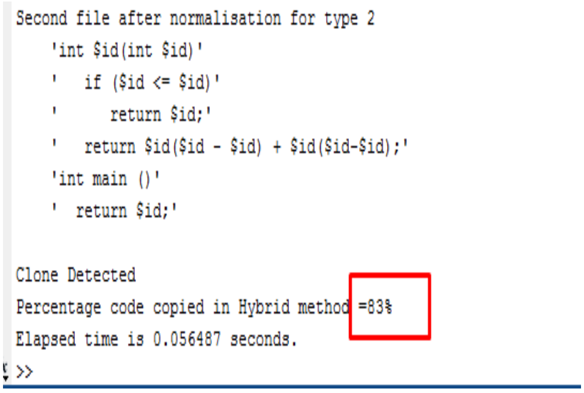


```
%file1=regexprep(file1,'[^\w'']','');
%file2=regexprep(file2,'[^\w'']','');


if nargin == 1
    goto 59;
    disp('LineBased');
    disp('TokenBased');
else
    by_space = false;
end

% Step 10 if code clone not matched
file1=strtrim(file1);
file2=strtrim(file2);
[fstx fsty]=size(file1);
```

Figure.10: Screenshot for Newly Proposed Hybrid Technique.



```
Second file after normalisation for type 2
    'int $id(int $id)'
    '   if ($id <= $id)'
    '      return $id;'
    '   return $id($id - $id) + $id($id-$id);'
    'int main ()'
    '  return $id;'

Clone Detected
Percentage code copied in Hybrid method =83%
Elapsed time is 0.056487 seconds.
>>
```

Figure.11: Screenshot for %age Copied Clone in Hybrid Technique.

Step-9) Repeat Step 4 to Step 6.

Step-10) End of the Algorithm.

7. DISCUSSIONS

This research paper is completed in two different sections. In section 1, I compared two different software code clone detection approaches viz. Token Based Approach and Line Based Approach. And In section 2, I propose a new Hybrid Technique for software code clone detection. This newly proposed technique initial working is based on two existing approaches viz. Token Based and line Based. The percentage of detected clone is calculated by applying mathematical formula in MATLAB simulator tool by considering 2 input text files named text file 1 and text file 2. Here, the result section shows 0% software code clone are detected in Line based Approach, 75% software code clones are detected in Token based Approach and 83% software code clones are detected in newly proposed Hybrid Technique. Hence, the performance of newly proposed hybrid technique is better or more convenient than other two existing techniques viz. Line based and Token Based. Hence, by utilizing these different software cloning techniques 3 different types of software code clones are detected viz. Type 0, Type 1 and Type 2. In addition, the maximum use of proposed hybrid technique automatically improves software reusability, reduce maintenance cost, improve software quality and improve overall performance of the software. Such type of services consumes less developer time & effort and will be helpful for producing more efficient results.

8. CONCLUSION

The presence of code clones makes the software maintenance extremely difficult. Code clones identification thus becomes extremely necessary in order to avoid the problems caused by them. Different types of Software code clone detection techniques are discussed in my thesis. A Hybrid Technique for software code clone detection by using Token Based and Line Based Approach” (HTSCCDUToL$_i$A) is proposed in my thesis. This hybrid technique is a combination of two different approaches viz. Line based approach and Token based approach. The main function of this new designed methodology is to save application space, developer time as well as developer effort. The main objective of this new designed Hybrid technique is to remove different software code clones viz. Type 0, Type 1 and Type 2 form any application or project and improve software reusability, software reliability, software maintenance, reduce maintenance cost, improve software performance and more importantly improve the overall quality of the software. The main significance to propose this hybrid technique is automatic detection of different software code clones viz. Type 1 and Type 2 within minimum duration of time. Different parameters are considered for software code clone detection in a table as an example size of code, type of clone, efficiency and portability etc.In addition, at the last the percentage of code clone detection is also calculated by utilizing a different comparison parameter. The results of three different techniques considered different parameters viz. Line based approach, Token Based Approach and Hybrid technique can be shown in three different tables having names table 1 and 2. The major benefit to propose this new designed methodology is automatic detection of different software code clones viz. Type 0, Type 1 & Type 2 within minimum duration of time. The simulated environment is taken into account in this work because of the absence of actual environment. With help of MATLAB, the proposed method is designed and implemented by using data analysis tool. The comparative analysis between Token Based Approach and Line Based Approach individually clearly shows the proposed Hybrid technique (which is a combination of Line based approach and token Based Approach) gives more effective results than already existing techniques separately which can be easily measured by considering different parameters as mentioned above. It is concluded that newly proposed hybrid technique can solve many problems like maintenance, reliability, performance, complexity of the code, reusability and also helps to improve the overall quality of the software. Hence, this new designed hybrid technique produces more efficient results than traditional/existing techniques.

9. FUTURE SCOPE

In future, this work will be extended by considering different techniques or on the other hand software Industry professionals may consider a combination of 3 or more techniques (merge techniques) in a hybrid technique and detect different types of software code clones viz. Type 0, Type 0, Type 1 and Type 2 etc. Most importantly, Type 3 and Type 4 will be easily detected by extending this newly designed Hybrid technique. Different types of parameters consideration are taken for Token based Approach; Line based Approach and Hybrid Technique later on.

10. REFERENCES

[1] Jahid Ali and Gurwinder Singh, September 2017. A Novel Composite Approach for Software Clone Detection, International Journal of Computer Applications.

[2] Sandeep Bali and Sumesh Sood, ,May-June 2017. A Composition of Clone Detection Technique: - A Hybrid Approach, International Journal of Emerging trends & Technology in Computer Science.

[3] Gurwinder Sigh and Jahid Ali, 2017.Study and analysis of Object oriented Languages Using Hybrid Clone Detection Technique”, Advances in Computational Sciences & Technology, Research India Publications.

[4] Sreenivasa Reddy and Syed MohdFazalulHaque, July 2017. XSCDF: - Towards a Framework for Comprehensive Software Clone Detection & Visualization Using Ontology, International Journal of Latest Technology in Engineering and Management & Applied Science.

[5] FazalulHaque& Syed Mohd, March-April 2017. Generic Code Cloning Method for Detection of Clone Code in Software Development, International Journal of Computer Engineering and Technology.

[6] Heejo Lee and Hakjoo Oh,2017. A Scalable Approach for Vulnerable Code Clone Discovery, IEEE Symposium on Security & Privacy, Korea University, Korea.

[7]Sukhpreet Kaur and Manpreet Kaur, April-June 2017. Code Clone Detection Using Metrics based Technique & Classification Using Neural Network, International Journal of Research in Electronics and Computer Engineering.

[8]Nguyen H.A, September-October 2017.Clone Management for Evolving Software, IEEE transactions on software engineering.

[9]Zibran M.F. and Roy C. K., 2016. Conflict-aware optimal scheduling of prioritized code clone refactoring, IET Software.

[10] Meena Bharti and Rajan Goyal, December 2014 Software Cloning & its detection methods.

[11]K. Rainer,F. Raimar, F. Pierre, 2006. Clone Detection Using Abstract Syntax Suffix Trees, Working Conference on Reverse Engineering.

[12]H. Yoshiki, U. Yasushi, 2011. Incremental Code Clone Detection: A PDG-based Approach, IEEE18th Working Conference on Reverse Engineering.

[13]M. Hiroaki, H. Keisuke, 2012. Folding Repeated Instructions for Improving Token-based Code Clone Detection, 2012.IEEE 12th International Working Conference on Source Code Analysis and Manipulation, Trento.