ADV MATH SCI JOURNAL
Advances in Mathematics: Scientific Journal **9** (2020), no.6, 3925–3931
ISSN: 1857-8365 (printed); 1857-8438 (electronic)
https://doi.org/10.37418/amsj.9.6.72 Spec Issiue on ICAML-2020

CODE CLONE DETECTION USING OBJECT ORIENTED METRICS

SIMRAN SHARMA [1], DHAVLEESH RATTAN, AND KARANDEEP SINGH

ABSTRACT. Code clone detection is the most important aspect for removal of repetitive code. Current software is based on larger codes, large source code takes more processing time and bigger memory to store. There require some automated tool that can identify different types of clones in the codes and mark the code, which has a clone. The programmer can remove the clones by making the general functions. These functions can be called repetitive. In the proposed object oriented metrics based technique for clone detection, different types of clones are detected. The proposed technique has been implemented using Java-based tool. It is a generalized tool, it can convert the code into sequence of the processing steps to identify the types of clones. The proposed technique detects all types of clones T-1, T-2, T-3 and T-4. The proposed approach achieves high accuracy 100%, 90%, 100%, 75%, respectively.

1. INTRODUCTION

In software design process, it is very common to reuse the code by using copy and paste activities. As a consequence of this, many sections of code look exactly similar, called code clones. In large software systems, duplication of code occurs frequently. Programmers select a block of code that matches their requirements and modify them to achieve the targets. Code cloning is beneficial in the sense that it leads to faster development of software code however it can cause serious problems related to software evolution and maintenance.

[1]*corresponding author*

2010 *Mathematics Subject Classification.* 68N30.
Key words and phrases. clone, metrics, QoS, types of clones.

1.1. **Reasons for Code Cloning.**

(1) **Time Limit:** The Primary reason for code cloning is the pressure of strict deadlines on programmers. To meet these deadlines, they follow the approach of copy and paste.
(2) **Language Limitation:** Clones are introduced when language does not have features to handle all real world problems [6].
(3) **Developers Performance:** Many organizations measure productivity of a tester by counting several lines of code they have tested. This inspires them to copy and paste same code again to increase the length of software.
(4) **Large software structure:** The structure of large software is not easy for everyone to understand. So they try to use an example oriented system for development.

1.2. **Types of Code Clones.**

(1) **T1:** This is an exact copy, the only difference lies in white-spaces or comments.
(2) **T2:** Code wreckage is syntax wise same with a difference in variables, literals, and white-spaces.
(3) **T3:** Code wreckage which has supplementary, detached or altered statements along with the difference in variables, literals, comments and white-spaces.
(4) **T4:** Code wreckage which has dissimilar syntax but execute alike functionality.

1.3. **Metrics based Clone Detection technique.** These techniques collect a number of metrics from source code. Metrics are calculated from names, layout, expressions and control flow functions [3]. Then these metrics are compared to find clones. Generally, code if first converted to a tree or graph and then metrics are calculated.

2. LITERATURE SURVEY

According to Tajima et al. [1], a software consists of a team of a programmer. Each member writes code for an allocated segment. Because all the programmer works independently, there can be chances that few of the programmer write the same code. When the project leader integrates the whole code, the resultant code will be having a clone. This will leads to extra processing and extra storage requirements.

Rainer [2] has presented a report that focuses on software redundancy, duplication and cloning along with its types. He has discussed the root causes and negative impacts of clones. The contribution of the study is the information on how to avoid clones, evaluation of existing techniques and benchmarks for clone detector evaluations.

Rattan et al. [5] have done a comprehensive survey on the various tools and techniques available for code clone detection. They have pointed out some open issues for research in this field. The study can help users in identifying usefulness of a tool according to their requirements.

Mondal et al. [4] have conducted an experiential study using a common framework. Their idea was to implement 9 code clone methods using 4 tools on 15 systems to analyze the shock of code clone on software preservation. The problem with the study is that it is not fully implemented yet. Still the results show that cloned code is more frequently modified and not stable.

3. Existing technique

Matrix based technique derived by various researchers has put the technique for T-1 and T-2 type of CCD with high accuracy. T-3 and T-4 efficiency is not so high. The proposed matrix based technique is applied on the large dataset for identification of T-1, T-2, T-3, T-4 type of code clones.

4. Proposed technique

The proposed technique is Object oriented metrics based. This technique identifies all types of clones T-1, T-2, T-3, T-4 clones. The proposed technique checks the code clones of all three types of languages C, C++ and Java. The proposed technique first clean the source code, secondly it subdivide the source code into smaller tokens. Thirdly an ontology is prepared for the metrics generation. An ontology is having different types of keywords for example if, else, for, while, do etc. The sub divided tokes will be compared to an ontology for preparing the metrics. Finally the metrics for all the occurred keywords are generated.

5. About tool

The Identification of the code clone that exists in the code is done with the specialized tool. This tool is a Java based tool, identifies the clones into the inputted program and reports in the interface. It identifies all types of code clones like T-1, T-2, T-3 and T-4. It identifies the clones with processing in specific sequence.

- Home screen will be opened.
- It asks for the username and password for authentication purposes.
- Ask for the language for which you want to identify the code clone.
- Ask for the code to be uploaded.
- Output the metrics for the different keywords.
- Generate the graph for code clone

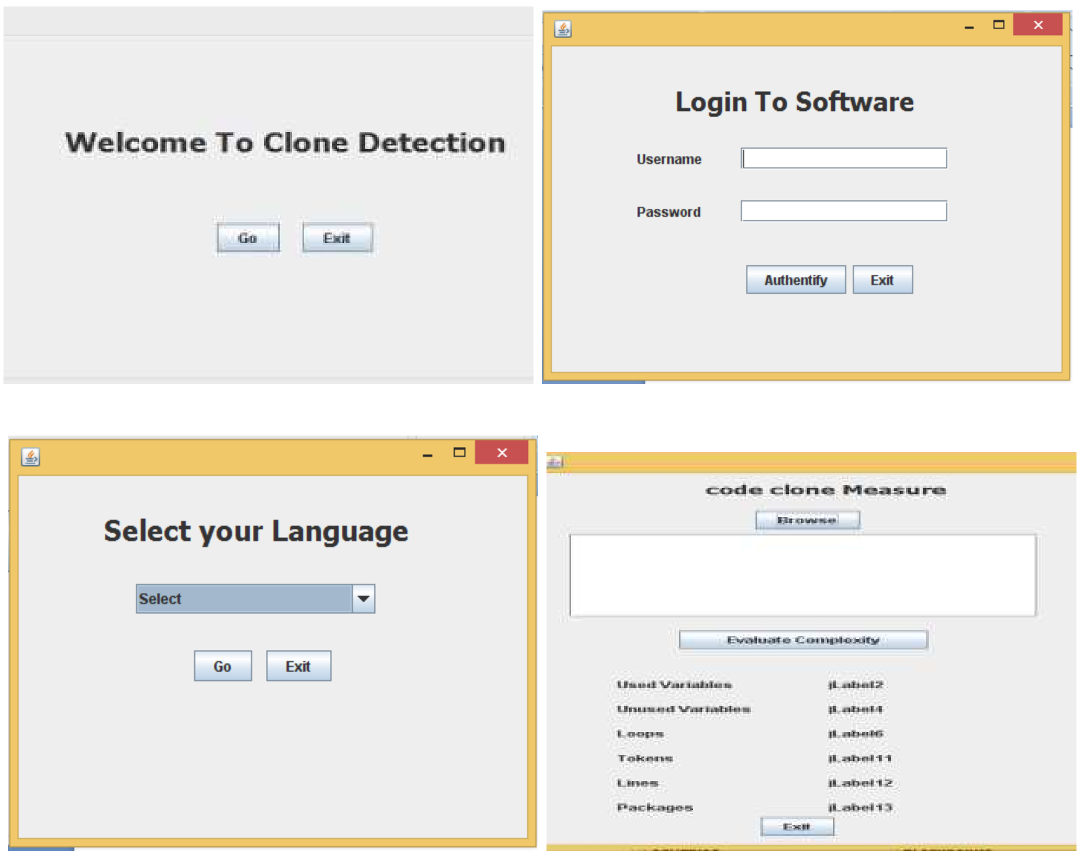

FIGURE 1. Interface page

6. Flowchart

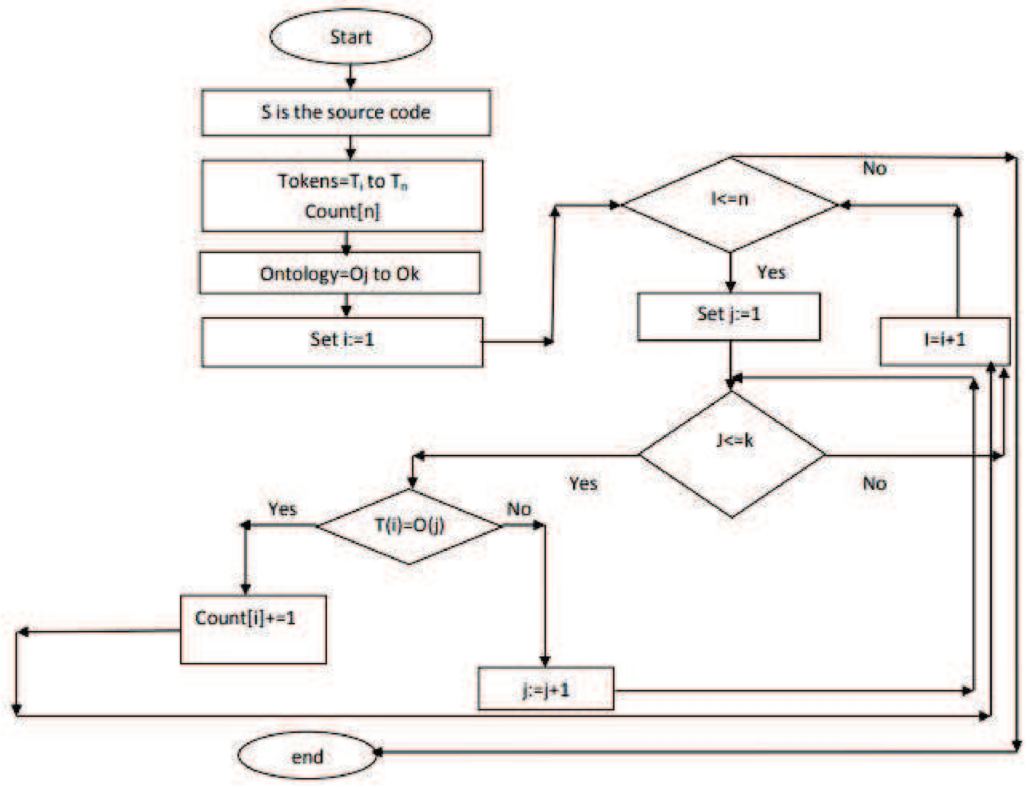

FIGURE 2. Flowchart

7. Results and discussion

7.1. **Graph for comparison.** Figure 3 shows the accuracy comparison for the existing and the proposed technique. The proposed technique performance is far better than the existing technique.

8. Conclusion

Code clone detection is the utmost important task for efficient coding. The large codes can be have different types of clones. These clones increase error probability. Error is a specific part of the code that will have a similar error in another part of the code due to duplication of the code. Removing the error in all

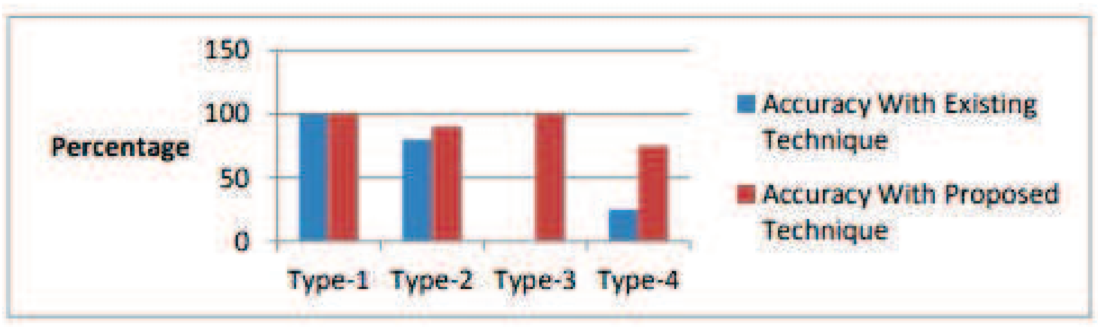

FIGURE 3. Accuracy comparison

parts of the same code will increase the cost of debugging. In the proposed approach the code clone detection is done using the matrix based technique. The whole technique is implemented using the automated Java-based tool. This tool will take input from the source program written in any language like C, C++ and Java. The performance of the proposed approach is better than the existing approach. The performance for the detection of all types of clones is higher for the proposed approach compared to the base technique. The proposed technique shows the accuracy of 100% for T-1, 90% accuracy for T-2, 100% for T-3 and 75% for T-4 clones.

9. FUTURE WORK

The proposed approach is based on object oriented metrics tested for the dataset of C, C++ and Java. The proposed technique shows the higher results compared to the existing technique. In future, the technique can be extended for other web based languages to make it a global technique.

REFERENCES

[1] R. TAJIMA, M. NAGURA, S. TAKADA: *Detecting functionally similar code within the same project,* In 2018 IEEE 12th International workshop on software clones(IWSC), (2018), 51–57.

[2] R. KOSCHKE: *Survey of research on software clones,* Dagstuhl Seminar Proceedings Schloss Dagstuhl-Leibniz-Zentrum fur Informatik, 2007.

[3] D. RATTAN, J. KAUR: *Systematic Mapping Study of Metrics Based Clone Detection Techniques,* In Proceedings of the International Conference on Advances in Information Communication Technology & Computing, (2016), 1–7.

[4] M. MONDAL, M. S. RAHMAN, R. K. SAHA, C. K. ROY, J. KRINKE, K. A. SCHNEIDER: *An Empirical Study of the Impacts of Clones in Software Maintenance,* 2011 IEEE 19th International Conference on Program Comprehension, Kingston, ON, (2011), 242–245.

[5] D. RATTAN, R. BHATIA, M. SINGH: *Software clone detection: A systematic review,* Information and Software Technology, **55**(7) (2013), 1165-1199.

[6] M. KAUR, D. RATTAN, R. BHATIA, M.SINGH: *Comparison and evaluation of clone detection tools: An experimental approach,* CSI Journal of Computing, **1**(4) (2012), 44–54.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PUNJABI UNIVERSITY
PATIALA, PUNJAB
Email address: simusharma96@gmail.com

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PUNJABI UNIVERSITY
PATIALA, PUNJAB
Email address: dhavleesh@gmail.com

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PUNJABI UNIVERSITY
PATIALA, PUNJAB
Email address: Karan_rob7@yahoo.co.in