International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 04 | Apr 2019 www.irjet.net p-ISSN: 2395-0072

Code Cloning using Abstract Syntax Tree

Gunjan Chugh[1], Divya Mahajan[2], Nainika Sehgal[3], Akanksha Paul[4], Leena Budhiraja[5]

[1]*Assistant Professor, Department of Information Technology, Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi, India*
[2,3,4,5]*Student, Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi, India*

Abstract - *Today, most of the software that are being developed by the developers consists of code clones in it. Although it is sometimes necessary to meet deadlines, to save time but in the long run it always emerge as the source of bad design. It increases maintenance cost which is one of the important phases the of software development life cycle. Not only this, it sometimes becomes the source of defects in our code which results in the huge demand for resource utilization thereby increasing the cost for both development and maintenance phase of the software development the life cycle. In order to detect those clones, a number of detection techniques have been presented so far. In this paper, an approach to detect code clones from the software is presented using Abstract Syntax Tree(AST). The aim of proposing this approach is to provide a solution to these code clones by an approach that is simple yet powerful enough to easily detect the code clones present in the software.*

Key Words: **clone detection; code cloning; clone class; clone pair; code clones.**

1. INTRODUCTION

A clone is a code segment that has been created through duplication of another piece of code [4]. Code cloning is the practice of duplicating existing source code for use elsewhere within a software system [7]. It is a process of replicating code blocks by doing copy-and-paste in order to save time and meet deadlines. Doing copy and paste proves to be a better solution in the short term to meet the deadlines but in the long run it results in the huge amount of cost that the organization have to spent in order to maintain the software not only this but also it results in the increase in amount of resource utilization. If faults found in one code block, then the entire cloned blocks need modification and it becomes more difficult tasks to maintain if the system becomes big.

For detecting these code clone's numerous techniques have been proposed. Text based techniques are the earliest and provide the easiest way of clone detection. In these techniques, code is compared line by line in the form of simple strings.[9] Then token based technique was proposed which is also the primary step for abstract syntax tree approach that is presented in this paper. In this techniques, the code is first transformed into tokens before comparing. In abstract syntax tree approach, lexical analyzer is used to transform the code into tokens then source code is transformed into AST using the required language parser and then code clones are detected by finding similar sub trees from the AST. [8] This method determines exact tree matches; a number of adjustments are needed to detect equivalent statement sequences, commutative operands, and nearly exact matches. We additionally suggest that clone detection could also be useful in producing more structured code, and in reverse engineering to discover domain concepts and their implementation [6].

2. RELATED TERMINOLOGY

Clone Pair

Two code segments form a clone pair, if they are related to each other by an equivalence relation [3]. An equivalence relation holds all reflexive, symmetric and transitive relations. A clone try is outlined as a try of matching code segments.

Clone Class

Clone class is defined as a set of code segments with similar code portions. Each code segment in a clone class forms a clone pair with other code segments of that class.

Classifications of Class

Code segments can be identical in two ways. Either they can be identical on the basis of their program text or they can be functionally identical. They are classified as follows: -

i. On the basis of Program text

On the basis of Program text these clones are classified in three ways namely: -

1) **Type-1 Clones**

If a code segment is copied as it is with some minor amendments in whitespaces, layout and comments then it comes under type-1 or exact clones [5]. In figure 2.1, code segment 2 is an exact copy of code segment 1.

Code1:-	Code 2:-

int s=0, p=1, n=5;
while (n > 0)
{
s= s+n;
p=p* n;
n= n-1;
}
	int s=0, p=1, n=5;
	while (n > 0)
	{
	s= s+n;
	p=p* n;
	n= n-1;
	}

Fig-2.1 Exact Clones

2) **Type-2 Clones**

If a code segment is copied with some amendments in name of the variables, functions, types and identifiers as shown in figure 2.1 then it comes under type-2 or renamed clones.

Code1:-	Code 2:-

int s=0, p=1, n=5;
while (n > 0)
{
s= s+n;
p=p* n;
n= n-1;
}
	int a=0, b=1, c=5;
	while (c > 0)
	{
	a=a+c;
	b=b*c;
	c=c-1;
	}

Fig-2.2 Renamed Clones

3) **Type-3 Clones**

If a code segment is copied with some changes like insertion or deletion of statements along with change in name of variables, functions and type, then it comes under type-3 or near miss clones.

Code 2:-	Code 3:-

int a=0, b=1, c=5;
while (c > 0)
{
a=a+c;
b=b*c;
c=c-1;
}
	int a=0, b=1, c=5,s=0;
	while (c > 0)
	{
	a=a+c;
	b=b*c;
	s= a+b;
	c=c-1;
	}

Fig -2.3: Near- miss Clones

ii. On the basis of Functional Similarity

4) On the basis of Functional Similarity, clones are classified as follows: -

If two code segments perform the same functionality but they are having different syntax, then they are said to be type-4 or semantic clones [5]. These clones are the most difficult to detect.

Code 1:-	Code 2:-

int num1=2, num2=6, i, prod=0;
for(i=1; i<= num2; i++)
{
prod+= num1;
}
	int num1=2, num2=6, prod;
	prod= num1 * num2;

Fig -2.4: Semantic Clones

3. OCCURRENCE OF CODE CLONES

Software clones appear for many reasons:
1.Code reuse by copying existing codes.
2.Programming styles of the programmers.
3.Instantiation of definitional computations.
4. Failure to identify/use abstract data types.
5. Enhancement in the performance.
6. Shortage of time or other resources. [8]

4. METHODOLOGY OF PROPOSED WORK

4.1. Clone Detection Using ASTs

To find the code fragments which yields similar result or are similar in syntax are the main problem in clone detection. So for this, first the program or the file in which we are trying to find clones is fragmented into parts before comparison. Then, it has to be determined as impossible, two arbitrary program fragments halting under the same circumstance is not determined. Hence, it is impossible theoretically to finalize that they compute identical results. [8]

There are some steps in the process of clone detection: -

i. The code is first parsed and then an AST is produced for it using the lexical analyzer for parsing.

ii. Then algorithms are applied to find clones.

a. The purpose of the basic algorithm, which is the first algorithm, is to detect sub –tree clones.

b. The second algorithm is sequence detection algorithm. This is helpful in the detection of statement and in the declaration of sequence clones.

c. The third algorithm attempts to generalize combinations of other clones and looks for more complex near miss-clones.

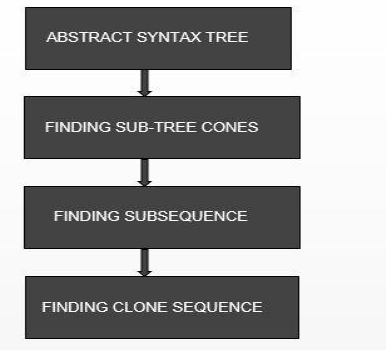

Fig 4.1: Methodology

4.2. Finding sub-tree clones

For finding sub-tree clones compare every subtree to every other sub-tree for equality. Though it seems easy in theory but in practice there are many problems associated with it like: near-miss clone detection, sub-clones, and scale. Near misses can be handled by comparing trees for similarity rather than exact equality whereas the scale problem is harder.

In order to handle this problem associated with practical implementation of finding sub-cones we generally partition the sets of comparisons by categorizing sub-trees with hash values. The approach is based on the tree matching technique for building DAGs for expressions in compiler construction [2]. This allows the easy detection of actual subtree clones. But this approach works well only when we are trying to find exact clones but when we are trying to locate near-miss clones hashing on complete subtrees fails because for a good hash function, it must include all elements of the tree, thus sorts trees with minor differences into different buckets.

This problem can be solved by choosing an artificially bad hash function. This bad hash function should be characterized in such a way that the main properties one wants to find on near-miss clones that are preserved. A hash function that ignores small subtrees is a good choice. [9]

4.3. Finding clone sequences

Finding clone sequences means to detect statement sequence clones in ASTs. And for that we are using the Basic algorithm as a foundation. Sequences of subtrees appear in AST as a consequence of the occurrence in the dialect grammar of rules encapsulating sequences of zero or more syntactic constructs. These sequence rules are typically expressed by the use of left or right recursion on production rules [2].

4.4. Generalization

This method consists of visiting the parents of the already-detected clones and check if the parent is a near miss clone too. An advantage of this method is that any near miss clones must be assembled from some set of exact sub clones, and therefore no near miss clones will be missed. [9]

5. BENEFITS OF CODE CLONING

Detection and removal of code clones promises decreased software maintenance costs of possibly the same magnitude. There are variety of benefits provided by code cloning detection and they are as follows: -

- **Reduced Probability of Defects** :-Detection of the code clones in the source code may also reduce the probability of bug propagation in the system.
- **Low Resource Requirement**: -Code cloning detection results in reducing the system size due to which compilation time as well as memory requirements for the system also get reduced. It may also result in less expensive software and hardware upgrades than the one having code clones in it. [9]
- **Reduced Maintenance Work and Cost: -**Code cloning may result in reduced maintenance effort because during the maintenance phase if an error or bug is found in one code fragment, then all its corresponding clones should be find out to detect the same error or bug. This makes maintenance a complex and time consuming task. Code cloning multiplies the effort required during maintenance. Hence detecting code clones helps to reduce the maintenance work and cost.
- **Reduced Chances of Bad Design: -**Code duplication also causes unfavourable effects on the system's design. It results in poor abstraction and raises difficulty in reusing the code in future projects. [2]
- **Inconsistent Updates:** -Code cloning may result in inconsistent updates because if there is a need to modify a piece of code, one needs to modify all clone segments of that piece of code. This can be avoided by code clone detection and removal.
- **Helps in Reducing Code Size :-**If the detected code clones are replaced by function calls to a generic code segment performing the same functionality as that of the code clone, then it results in reducing the complexity and size of software system. It also improves maintainability and readability of code. [1]
- **Better Understanding of Problem :-**If working of a cloned segment is apprehended, one is able to understand the working of all duplicate code segments of the cloned segment.
- **Helps in Reducing Code Size :-**If the detected code clones are replaced by function calls to a generic code segment performing the same functionality as that of the code clone, then it results in reducing the complexity and size of software system. It also improves maintainability and readability of code.

- **Discovering Domain: -**Clone detection is not only helpful in producing more structured code but also in discovering domain concepts and their idiomatic implementations. [8]

6. CONCLUSION

The code clone detection is an issue in the software system which decrease the software's comprehensibility as well as maintainability. Therefore, its analysis and detection is necessary for improving the quality, maintenance and design of the software system. In this paper, discussion in terms of attributes based on code clone detection is presented. The clone detection method is implemented using abstract syntax trees (ASTs), which for finding exact and near miss clones for arbitrary fragments in the source code [2]. Since detection done in the program structure. clones can be factored in the source using standard transformational methods. The approach is based on variations of methods for compiler to find common subtree on elimination using hashing and then we find subsequence. The method is straightforward to implement using parsing technology.

REFERENCES

[1] Lee, Y. J., Lim, J. S., Ji, J. H., Cho, H. G., & Woo, G. (2012). Plagiarism detection among source codes using adaptive methods. *KSII Transactions on Internet and Information Systems (TIIS)*, *6*(6), 1627-1648.

[2] Baxter, I. D., Yahin, A., Moura, L., Sant'Anna, M., & Bier, L. (1998, November). Clone detection using abstract syntax trees. In *Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)* (pp. 368-377). IEEE.

[3] Morshed, M., Rahman, M., & Ahmed, S. U. (2012). A literature review of code clone analysis to improve software maintenance process. *arXiv preprint arXiv:1205.5615*.

[4] Jiang, Z. M., & Hassan, A. E. (2007, September). A framework for studying clones in large software systems. In Seventh IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM 2007) (pp. 203-212). IEEE.

[5] Roy, C. K., & Cordy, J. R. (2007). A survey on software clone detection research. *Queen's School of Computing TR*, *541*(115), 64-68.

[6] Baxter, I. D., Yahin, A., Moura, L., Sant'Anna, M., & Bier, L. (1998, November). Clone detection using abstract syntax trees. In *Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)* (pp. 368-377). IEEE.

[7] Kapser, C. (2009). Toward an understanding of software code cloning as a development practice.

[8] Ijptjournal.org. (2019). *Code Cloning Detection using Abstract Syntax Tree*. [online] Available at: http://www.ijpttjournal.org/volume-9/IJPTT-V9P407.pdf [Accessed 17 Apr. 2019].

[9] Baxter, I., Yahin, A., Moura, L. and Bier, L. (2019). Code Cloning Using Abstract Syntax Tree. *www.eecs.yorku.c*, [online] p.2. Available at: http://www.eecs.yorku.c [Accessed 17 Apr. 2019].

BIOGRAPHIES

Gunjan Chugh, she is currently pursuing PhD with specialization in Computer Science from Delhi Technological University, New Delhi. She received her M. Tech degree in Computer Science from Banasthali University, Rajasthan in 2013 & is currently working in Department of Information Technology, at Dr. Akhilesh Das Gupta Institute of Technology and Management, New Delhi. Her research interest includes Artificial Intelligence, Machine Learning and Information Security.

Divya Mahajan, she is currently pursuing Bachelor of Technology with specialization in Information Technology from Guru Gobind Singh Indraprastha University, New Delhi. Her research interest lies in software engineering, data mining and Machine learning.

Nainika Sehgal, she is currently a student who is pursuing Btech in Information Technology from IP university. She has searched and made a research paper on Code detection using an abstract syntax tree. The main motive was to study about clones and different techniques and how to detect it. Her career objective is to continuously enhance her knowledge, skills and experience by getting involved in challenging work environment and utilize them for personal and organizational growth to the best of her ability.

Akanksha Paul, currently she is a student in Guru Gobind Singh Indraprastha University, New Delhi pursuing her B. Tech in Information Technology. She made a research paper on Code Cloning for better understanding of clones and code reuse.

Leena Budhiraja, currently she is pursuing B. Tech in Guru Gobind Singh Indraprastha University, New Delhi in Information Technology. Her objective of life is to pursue a challenging career in life and enhance her knowledge.