Technium Vol. 4, Issue 5 pp.77-82 (2022)
ISSN: 2668-778X
www.techniumscience.com

Token-based code clone detector for Dart language

Yasir Mohammed Khazaal[1], Asma'a Y. Hammo[2]

[1]*Software Department, College of Computer Science and Mathematics, Mosul, Iraq
[2]*Software Department, College of Computer Science and Mathematics, Mosul, Iraq
[1]yasirmk@uomosul.edu.iq [2]asmahammo@uomosul.edu.iq

Abstract

Reusing the code in multiple locations is called in software engineering Code Clone. Detection of a clone is important to reduce the maintenance time and size of code. There are four techniques to detect the code clone: text-based, token-based, AST(Abstract Syntax Tree) based, and PDG(Program Dependence Graph) based.
In this paper, we propose and implement the token-based method for code clone detection in the Dart language. The proposed method is evaluated by injecting clones to Dart programs. Results show that the proposed method detects all the injected codes.

Keyword

Clone detection, Dart language, token-based method.

1- Introduction

A code clone is a code fragment that appears in multiple locations in the source file[4]. This clone makes the maintenance of software difficult, takes more time also leads to the spread of bugs in all locations that copied the fragment to it[2]. This clone can occur in the level of file, class, method, or number of lines. There are four categories of clones [9].

1- The first category where the clone is identical to the original fragment, changing only in comment or whitespace[1,6,2,8].
2- The second category includes the clone as in the first category plus changes in identifiers names (variable names, methods names, and identifiers numbers value).[1,2,8]
3- The third category in addition first and second categories changes code fragments by adding or deleting or modifying code fragments.[5,2,8]
4- The fourth category takes the functionality of the code fragment and rewrites it in different syntaxes (copy the function of the code fragment).[6,8]

Researchers propose multiple code clone detection methods (text-based, token-based, AST(Abstract Syntax Tree) based, and PDG(Program Dependence Graph) based) that are different in the precision and time have taken to find the clone[11]. Code clone detection is very important to other software engineering stages [3].

Dart language appears in 2011, It is developed by Google and used for developing fast applications for many platforms (Mobile applications, Web Applications, and Desktop applications).[10]

This research aims to propose the methodology and implementation of Code Clone Detector for Dart language that is based on the token.

Section 2 contains the methodology of the proposed method. Section 3 contains evaluation of the proposed method. Section 4 contains conclusion and future work.

2- Methodology

The source code files written in Dart language under testing are read and pass the Preprocessing stage. After this it's tokenized into tokens then compare them to find the code clone between files. Figure -1 shows the overall method.

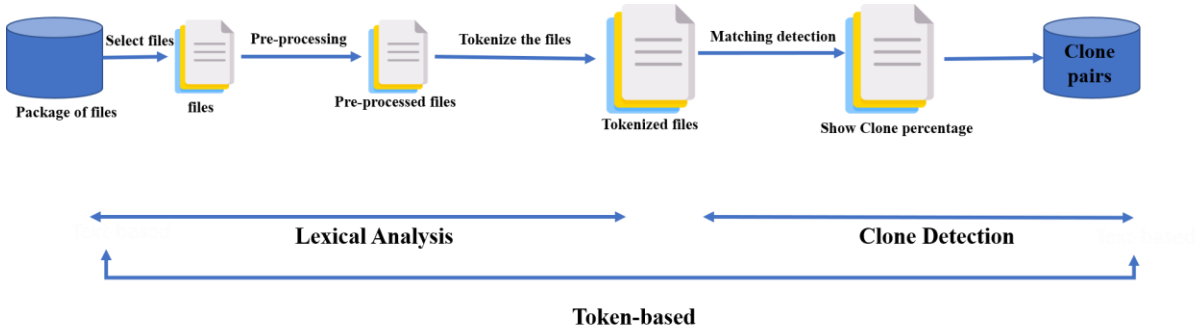


Figure 1 Overall method

2-1 preprocessing

In this step, all comments, whitespace, and import of libraries are removed from the source code[3,5,7]. In Dart language, the Comment can be one line by adding the (//)symbol to the line or multiple lines comment by adding the (/*) symbol at the beginning of the comment and the (*/) symbol at end of the comment. For import libraries in the Dart language, The ("import") keyword is used to enable user to use these libraries[10]. All this with whitespace will be removed in this step. This step is shared between two types of clones (first Category and second Category).

2-2 Lexical analysis(tokenizer)

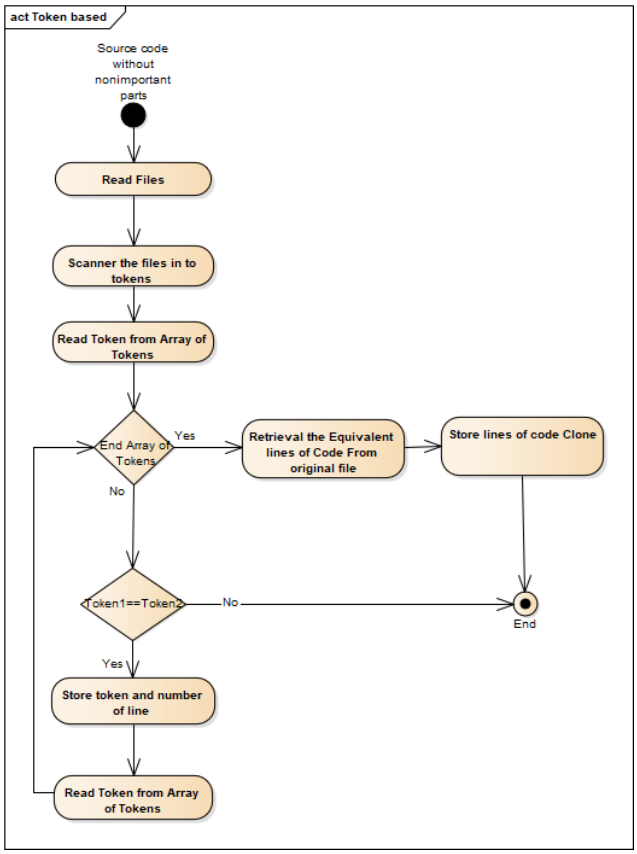
Lexical analysis is the first stage of a compiler that takes the source code after preprocessing and breaks each line into a set of tokens each token has information (token, token type, and line number). This information will be used to find the clone between two files. For First Category, this token will be used without any normalization to find the similarity between the two files.

For Second Category, after getting tokens, all the tokens that are type identifiers (variables, methods name, numbers) are replaced by a special token we use ("#VOD"). This replacement makes code fragments that renamed identifiers similar to each other.

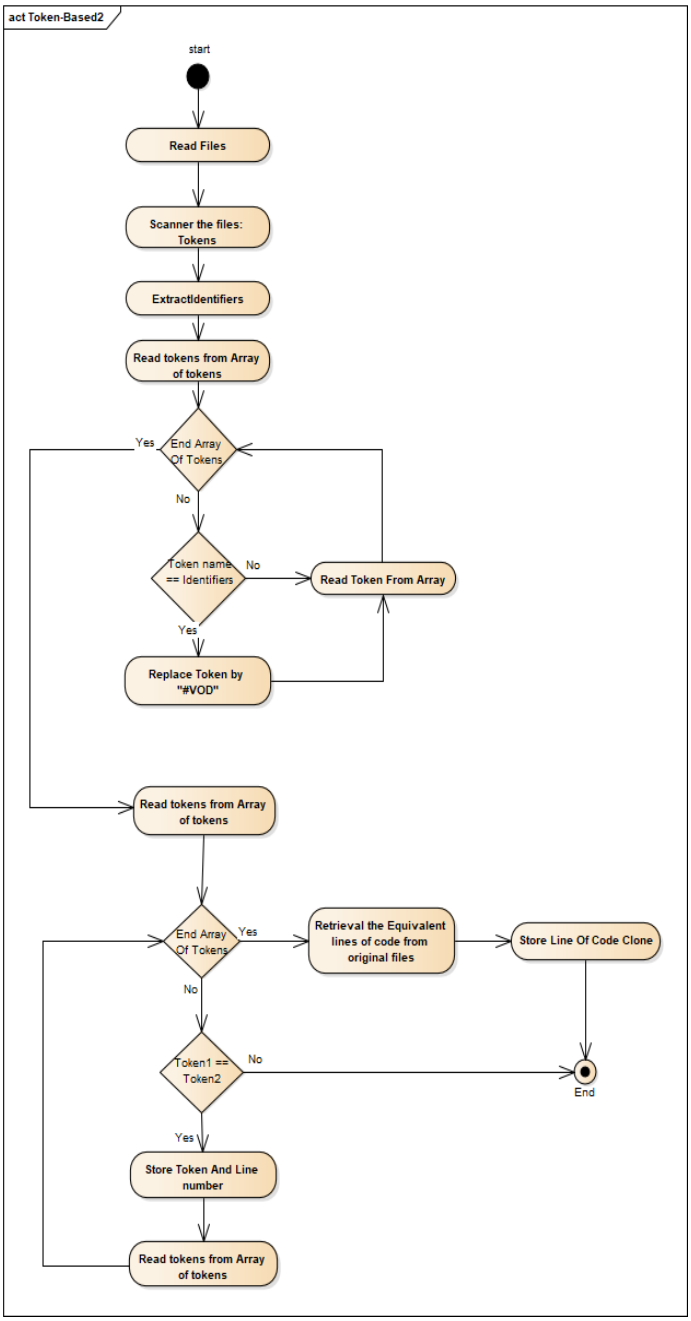
2-3 Matching detection

From all the previous steps the tokens are compared and detected. The clone pair then is converted into line numbers on the source files and then compute the clone percentage between two files and then display results.

Figure 2 shows the Activity diagram for the proposed Token-based method a- the first category b - The second category.



a-First Category



b-Second Category

Figure 2 the Activity diagram for The proposed Token-based method

The sequence diagram shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes involved and the sequence of messages exchanged between the processes needed to carry out the functionality. Figure 3 shows the sequence diagram for the proposed method.

Technium Vol. 4, Issue 5 pp.77-82 (2022)
ISSN: 2668-778X
www.techniumscience.com

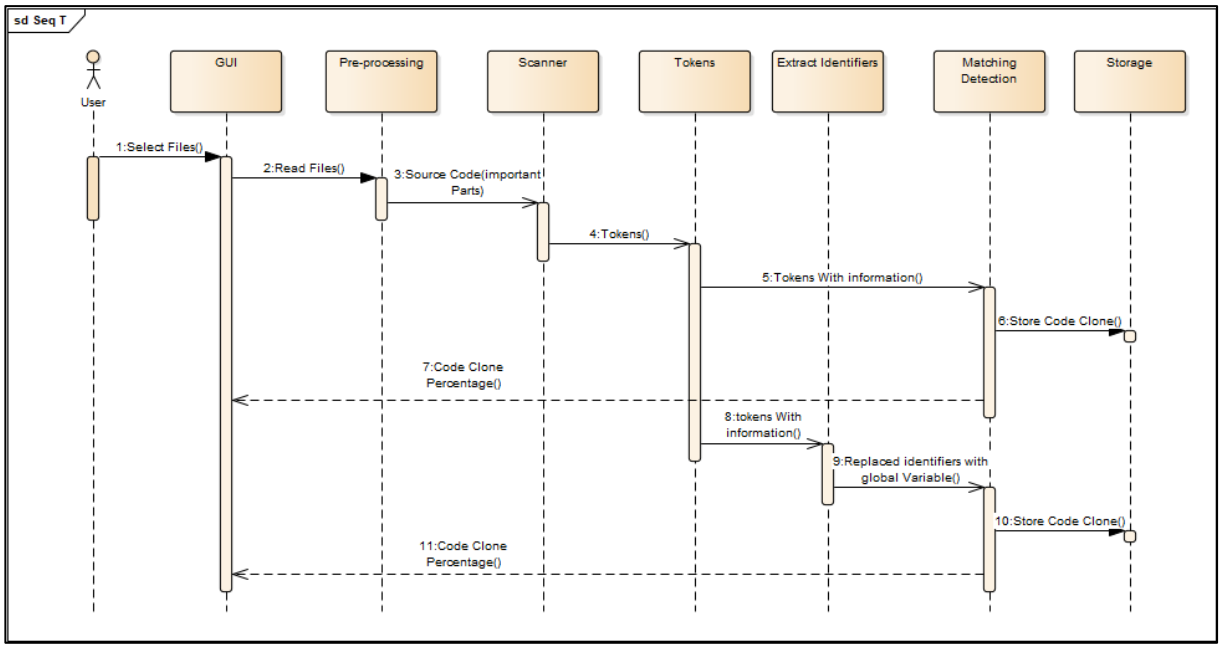


Figure 3 Sequence diagram for proposed Token-based method.

3. Evaluating The Proposed Method

To evaluate the method token-based for detection clones of first and second categories. The following steps are used :

1. obtaining Source code files written in Dart language from Github.
2. injecting these files with clones of code
3. Editing files as shown in table 1.

Table 1 shows the types of edits applied to clones

Edit	Category
Add whitespace.	First Category
delete whitespace.	First Category
Change in between token (/* */) comments	First Category
Change at the end of line (//) comments.	First Category
Add import library	First Category
Delete import library	First Category
Systematic renaming of an identifier.	Second Category
Arbitrary renaming of an identifier	Second Category
Change in value of a numeric	Second Category

After executing of the proposed Code Clone Detector, all the injected clones are retraveled without a false position. The Recall for these clones and editing is shown in table 2.

Table 2 Recall for Token-based method

Copies Number / Recall	First Category		Second Category	
	inject	Find	inject	Find
	25	25	25	25
Token-based	%100		%100	

The detector is tested on files more than one and measure the time required for each one (ms). The average time for the token-based is shown in Figure 3 for each category. The second category takes more time because it needs additional time to search and replace all identifiers with a special token.

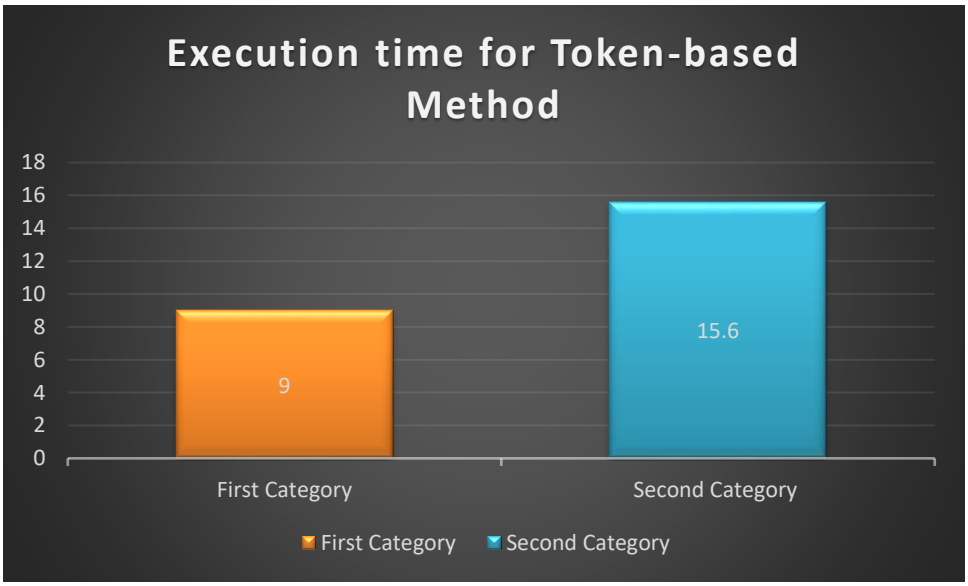


Figure 4 Execution time for The proposed Token-based method

4. Conclusion and future work

The paper proposed a token-based code clone detector. It detects clones in programs written in Dart language. The implemented detector is tested by injecting clones to Dart programs obtained from Github by codes. It detects all the clones.

In the future, the same method can be used for other languages such as Java, C++, …. Etc. Also make the method work on the function level and detect the third category.

References

[1]. L. LI, H. FENG, W. ZHUANG, N. MENG, AND B. RYDER: Cclearner: A deep learning-based clone detection approach.In: Proc. IEEE Int. Conf. Softw. Maintenance Evol. (ICSME), Sep. 2017, pp. 249–260.

[2]. Y. SEMURA, N. YOSHIDA, E. CHOI, AND K. INOUE : CCFinderSW: Clone detection tool with flexible multilingual tokenization.In: Proc. 24th Asia–Pacific Softw. Eng. Conf. (APSEC), 2017, pp. 654–659.

[3]. P. WANG, J. SVAJLENKO, Y. WU, Y. XU, AND C. K. ROY: CCAligner: A token-based large-gap clone detector.In: Proc. 40th Int. Conf. Softw. Eng., 2018, pp. 1066–1077.
[4]. Y. GOLUBEV, V. POLETANSKY, N. POVAROV, AND T. BRYKSIN: Multi-threshold token-based code clone detection.IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2021, pp. 496-500, doi: 10.1109/SANER50967.2021.00053.
[5]. T. KAMIYA, S. KUSUMOTO, AND K. INOUE: Ccfinder: a multilinguistic token-based code clone detection system for large scale source code. Software Engineering, IEEE Transactions on, vol.28, no.7, pp.654–670, (Jul2002).
[6]. P.CHARLAND, B. FUNG AND M.R. FARHADI: Clone Search for malicious code correlation. 2014, Retrieved from https://dmas.lab.mcgill.ca/fung/pub/CFF12ist.pdf .
[7]. A. SHENEAMER AND J. KALITA: Code clone detection using coarse and finegrained hybrid approaches. In: Proc. IEEE 7th Int. Conf. Intell. Comput. Inf. Syst. (ICICIS), Dec. 2015, pp. 472–480.
[8]. J. SVAJLENKO AND C. K. ROY:The Mutation and Injection Framework: Evaluating Clone Detection Tools with Mutation Analysis.In: IEEE Transactions on Software Engineering, vol. 47, no. 5, pp. 1060-1087, doi: 10.1109/TSE.2019.2912962. (2021)
[9]. M .KAUR , S. KAUR, AND B. SOHAL :Review on Software Cloning and Clone Detection. International Journal of Control Theory and Applications, pp. 463-472. (2016)
[10]. Dart Programming Language Specification 6th edition draft Version 2.15-dev'',(2022).
[11]. C.K ROY AND J. R. CORDY : A Survey on Software Clone Detection Research.Technical Report No. 2007-541, School of Computing Queen's University at Kingston Ontario, Canada. *arXiv:2107.04712 [cs.SE]*. (2007).