Department of Computer Science,
University of Saskatchewan

July 10, 2020

The Guest Editors
Journal of Systems and Software

Re: Submitting an extended version of our earlier work

Dear Guest Editors,

We would like to submit an extended version of our earlier work on ranking some existing good clone detection tools based on their capability to predict co-change candidates during software evolution. Our earlier work titled as "**Evaluating Performance of Clone Detection Tools inDetecting Cloned Cochange Candidates**" was published in IEEE 14th International Workshop on Software Clones (IWSC), 2020.In this work, we have used six open-source software as subject systems to evaluate the performance of six clone detection tools which have been reported good in detection clone fragments from software source code by recent studies. By answering two research questions, we have been evaluated that, even though a tool which is good in detecting clone fragments from software systems may not be good in detecting cloned co-change candidates. We also investigated and reported two possible reasons for such a difference in the performance of clone detection tools while we are using them to predict co-change fragments. We extend our previous work by answering two additional research questions (RQ3, RQ4) to find a more specific reason for the variation of the performance by clone detectors in detecting co-change candidates. We have also increased the generalizability of the previous study by adding two more software systems as subject system and six more clone detection tool implementation. Therefore, our implementation has been upgraded from 6X6 to 12X8 (Clone detector X Subject Systems) in the current version of the study. In the current extended version of the study, we have shown that the performance of clone detection tools in detecting cloned co-change fragments not only dependent on the number of clone fragments detected and the line covered in the source file by those fragments but also the type of detected clone and underlying source code processing techniques also have some impact. In the following paragraphs, we describe how we extend our earlier work.

- **Adding more clone detectors in the study:**In the earlier implementation, we have applied six clone detection tools five of them (Deckard, ConQAT, iClones, NiCad, and SimCAD) have been reported very good tools for detecting Type-1, 2, 3 clones. We added one text similarity-based tool (Simian) as the sixth tool in the earlier implementation. To extend the work, we added six additional implementations of clone detection techniques by using three additional tools. New tools added are CloneWorks,

CCFinder, and Duplo. CloneWorks have been reported as a fast and flexible clone detector for large-scale near-miss clone detection experiments. CloneWorks provides options to modify it's configuration files which effects on the source code processing mechanism while detecting the clones. This is important to target specific types (1, 2, or 3) of clone by using this clone detection tool. We have applied four types of different configuration files to detect Type-3 pattern, Type-3 Token, Type-2 Blind, and Type-1 clones by using CloneWorks tool. This provided four additional sets of detected clone result. CCFinder is known as a multilinguistic token-based code clone detection system for large scale source code. Inclusion of CCFinder enriched the variation of detected clone fragments in the extended study. To make more comparison of the performance of type-1 clones in detecting co-change candidates we added Duplo in our study.

- **Adding more subject systems:** In the earlier version of the study, we added six software systems four of them are written in Java and two of them are written in C programming language. To increase the generalizability of the study we have added two more software systems written in C having diverse size and application domain from the other six systems.

- **Additional two research questions:** Our earlier implementation evaluated only two research question to find the comparison scenario of clone detectors in detecting co-change candidates and reasons behind these performance variations of those tools. We have added two more research question in the extended study to find outthe relation of the types of clone and source code processing techniques while detecting the clone as the additional reasons for these difference in performance.

- **Updating different sections in the paper to incorporate our new findings:** We have updated different sections such as Abstract, Introduction, Methodology, Experimental Result,Discussion, Conclusion and Future Works by adding our new findings from our extended research.

We believe that our extended version contains at least 60% more content than our earlier work.

Sincerely,
Md Nadim
Manishankar Mondal, and
Chanchal K. Roy