Do Concern Metrics Support Code Clone Detection?

Student Lightning Paper

Alexandre Paiva
Computer Science Department
Federal University of Minas Gerais (UFMG)
Belo Horizonte, MG - Brazil
ampaiva@ufmg.br

Eduardo Figueiredo
Computer Science Department
Federal University of Minas Gerais (UFMG)
Belo Horizonte, MG - Brazil
figueiredo@dcc.ufmg.br

***Abstract* — A common decision made by a software developer is to reuse existing code as a reference. Copying existing code fragments and pasting them with or without modifications into other parts of the code is called code clone. Several methods have been used for Code Clone detection, such as software metrics, static code analysis, and text-based detection. However, concern metrics have not been used for this purpose, as indicated by a systematic literature review. Concern metrics quantify properties of concerns, such as scattering and tangling. Therefore, the question of whether concern metrics support code clone detection remains unanswered. This student lightning paper presents the results of our literature review and discuss this topic of research.**

Keywords—Code clone; concern metrics; systematic review

I. INTRODUCTION

A common decision made by a software developer when coding is to reuse existing code as reference [8]. Reasons vary and are not mutual exclusives. They include from to get an idea of how to solve a specific problem to the use something already tested [5]. Therefore, copying existing code fragments and pasting them with or without modifications into other parts of the code is called code clone. Code clone is an important area of Software Engineering research [5][8][9].

Software metrics have been traditionally used to evaluate the maintainability of software systems and to detect code smells [2][3], such as code clones [5]. In fact, several methods have been used for Code Clone detection [9], such as software metrics, static code analysis, and text-based detection. However, code clone detection is a hard task and there is still a need of methods and tools with better precision rates [5][6]. Therefore, this paper aims to motivate the use of concern metrics to help detection of code clone.

Differently from traditional metrics which quantify module properties, such as class coupling, cohesion, and size, concern metrics quantify properties of concerns realized in the source code, such as concern scattering and tangling [1]. A growing number of concern metrics have been proposed [1] and used in several empirical studies [3]. For instance, Padilha and her colleagues [3] performed an empirical study involving 54 subjects that evaluates if a set of concern metrics is useful to detect three code smells, namely Divergent Change, Shotgun Surgery, and God Class. However, as far as we know, no study so far have investigated the use of concern metrics to detect code clone.

Therefore, *the purpose of this paper is to motivate a new area of research aiming to verify whether concern metrics could be used to support code clone detection*. Before we further investigate the use of concern metrics to detect code clones, we first verify the need of new methods for this purpose by means of a systematic literature review [4]. The next section presents the protocol and reports the results of the literature review we performed. The results of this review confirm that there is no work so far that relies on the use of concern metrics to detect code clone.

II. A SYSTEMATIC REVIEW OF CODE CLONE DETECTION

Systematic literature review (SLR) is defined as a mean of evaluating and interpreting all available relevant research to a particular research question, topic area, or phenomenon [4]. It has been broadly used and accepted as a rigorous method in a variety of social science areas and, more recently, in Software Engineering.

Prior to conducting our SLR, we developed a review protocol. An SLR protocol specifies the planning procedures by describing the strategies to perform the SLR. In particular, it defines the research questions, search strategy to identify the relevant literature, inclusion and exclusion criteria, and the procedures for extracting and synthesizing data. The research question (RQ) we aim to answer in this SLR can be defined as follows.

RQ: Do concern metrics have been used or investigated for code clone detection?

In order to answer this research question, we search for relevant papers in two of the most reputable scientific databases on the Web, namely ACM Digital Library[1] and IEEE Xplore[2]. We search in the title for the words ‘code’, ‘clone’, and variations, such as ‘cloning’, ‘copy’, ‘duplicate’, ‘duplication’, and ‘similarity’. However, as clone is related to ‘chromosome’ and ‘dna’ in biology research, titles with such words were excluded. The search string we used is: (code OR

[1] http://dl.acm.org/
[2] http://ieeexplore.ieee.org/

software) and (clone OR cloning OR copy OR duplicate OR duplication OR similarity) and (not chromosome and not dna).

We select the primary studies by applying the inclusion and exclusion criteria in a stepwise fashion, as follows: (i) select studies through the search string, (ii) analyze the titles and abstracts returned in the previous step, and (iii) scan the papers returned in the previous step. The inclusion criteria are two: the paper must be related to detection of code clone and it must be written in English. The solo exclusion criterion is that papers cannot be less than 6 pages in length.

As a result of the filtering steps above, 65 papers related to code clone were found. We identify and classify the detection methods into seven categories [6][7]: Text, Token, Metric, Abstract Syntax Tree (AST), Program Dependence Graph (PDG), Count Matrix, and Smith Waterman. Figure 1 shows the classification of all methods for code clone detection cited in the literature. The last bar in this figure shows that 24 papers do not present any detection method, although they deal with code clone.

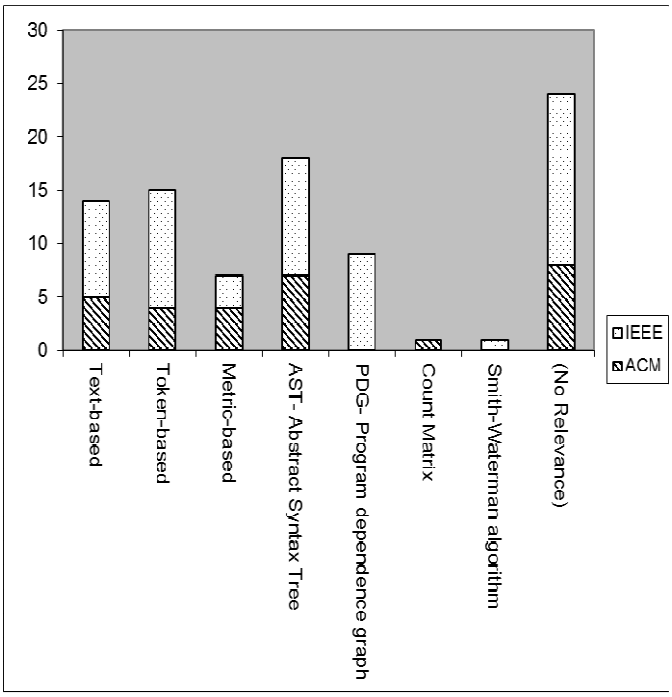

Fig. 1. EXISTING METHODS FOR CODE CLONE DETECTION.

We also observed in this study that the literature considers four types of code clone classifications: Exact, Syntactical, Modified, and Semantic. More important, we could not find in the literature any work that used concern metrics to detect code clone. This gap suggests opportunities for a new area of research. That is, since concern measurement is a relatively new research topic, we believe there is a lack of relevant work that relies on concern metrics with this aim. As future work, we aim to investigate this topic in the ongoing master degree program.

REFERENCES

[1] E. Figueiredo et al. "On the Maintainability of Aspect-Oriented Software: A Concern-Oriented Measurement Framework", European Conference on Software Maintenance and Reengineering (CSMR) 2008.

[2] M. Fowler. "Refactoring: Improving the Design of Existing Code". Addison Wesley, 1999.

[3] J. Padilha et al. “On the Effectiveness of Concern Metrics to Detect Code Smells: An Empirical Study”. International Conference on Advanced Information Systems Engineering (CAiSE), 2014.

[4] B. Kitchenham and S. Charters. “Guidelines for performing Systematic Literature Reviews in Software Engineering”, EBSE Techinical Report, EBSE-2007-01, 2007.

[5] J. Mayrand, C. Leblanc, E. Merlo. “Experiment on the Automatic Detection of Function Clones in a Software System Using Metrics”, International Conference on Software Maintenance (ICSM), 1996

[6] J. Pate, R. Tairas, N. Kraft. “Clone Evolution: a Systematic Review”, Journal of Software: Evolution and Practice, 25, p. 261–283, 2013.

[7] D. Rattan, R. Bhatia, and M. Singh. “Software Clone Detection: a Systematic Review”, Information and Software Technology, 2013.

[8] C. Roy, J. Cordy, and R. Koschke. “Comparison and Evaluation of Code Clone Detection Techniques and Tools: A Qualitative Approach”, Science of Computer Programming, 2009.

[9] T. Shippey, D. Bowes, B. Chrisianson, and T. Hall. “A Mapping Study of Software Code Cloning”, International Conference on Evaluation and Assessment in Software Engineering (EASE), 2012.

[10] C. Wohlin *et al*. "Experimentation in Software Engineering", Springer. 2012.