International Journal of Open Source Software and Processes
Volume 13 • Issue 1

SBHDetector: A Fuzzy-Based Hybrid Approach to Detect Renaming and Shifting Between Versions

Ritu Garg, Indira Gandhi Delhi Technical University for Women, India*
Rakesh Kumar Singh, Indira Gandhi Delhi Technical University for Women, India
https://orcid.org/0000-0001-8729-2293

ABSTRACT

Mining software repositories in a collaborative environment during software evolution or maintenance faces challenges due to creation of larger than necessary slices or unnecessary splitting of revision history and detection of edge level changes. Due to these limitations, GIT and Diff and Merge Tools do not accurately detect the similarities and changes between versions due to renaming or shifting. Detection of these similarities accurately helps to detect code clones and change patterns that improves understandability, knowledge transfer, and tracking changes. Therefore, the authors proposed fuzzy-based hybrid technique to detect the similarities/changes between versions considering RS by enriching the revision history at three granularities: file, class, and method level. Thirty percent more entities have been found similar/change by deriving classification model with f-score and ROC area more than 0.985 and 0.994 respectively for all applications. Hence, the proposed technique improves productivity, reusability, and maintainability with respect to VCA.

KEYWORDS

Changes, Diff and Merge Tools, Maintenance, Refactoring, Repositories, Revision History, Similarity, Software Evolution

INTRODUCTION

Mining Software Repositories (MSR) is widely used during the software evolution and maintenance phase for Version-Controlled Applications (VCA). During the evolution of code base with time, Software Repositories (SR) records the historical information in respect of versions/revisions for VCA (Agrawal et al., 2020). Analysis of these SR for different versions helps in identification of code clones (Roy et al., 2007) and change patterns that further enhances understandability, knowledge transfer and tracking similarities/changes in collaborative work. GIT, Open Source Distributed Version Control System (OSDVCS) is the most frequently used tool for collaborative work (GIT, n.d.). There are three challenges that restrict these MSR approaches in using SR-

DOI: 10.4018/IJOSSP.300752

*Corresponding Author

- **Low performance for larger than necessary slices of Revision History:** SR incorporates the changes in artifacts of software code shared among multiple developers. Usually, these changes are pushed/pulled using the diff and merge techniques to/from remote repository that may clutter the SR. These changes are recorded at coarse granularity (at file level) resulting in larger than necessary slices of Revision History (RH) (Zhu et al., 2020). Example- In order to extract changes at finer granularities (class and method level), it requires preprocessing for History Slicing using developer scripts that may be cumbersome and error-prone (Higo et al., 2020).
- **Restructuring may lead to unnecessary splitting of RH esp. where similarity index is low between the entities:** The sequence of changes may involve major or minor refactoring along with recording of change-type such as renaming, moving or splitting but restricted with limitations of default similarity (Higo et al., 2020). Example- If similarity percentage of an entity before and after revision falls below 50% then GIT accept the change as creation of entity in Successive Version (SV) and removal of entity from preceding Version (PV). Due to this, it leads to unnecessary splitting of RH for that entity which makes the entity un-trackable for previous versions.
- **Textual similarity lacks in identification of changes at edge level:** GIT is content addressable (Git-Internals-Git-Objects, n.d.), thus offers textual similarity in software code. However, existing approaches treats software code as graphs where changes may occur both at node and edge level (Störrle, 2015). Due to this, change in any reference to a particular object or hierarchy of interrelated object or change in number of calls to an object or dependencies etc. are not recorded in RH.

Therefore, changes in artifacts are missing at finer granularities and edge level leading to incomplete RH (Tang et al., 2022). Moreover, inaccuracy arises from the splitting of RH that breaks the link for changes in all PV by treating them as new entities with fresh RH. Due to this incomplete or inaccurate RH, it is very difficult to track these entities for MSR approaches (Agrawal et al., 2020) making it unreliable and time-consuming (Grund et al., n.d.). In order to overcome these challenges, the authors proposed SBHDetector, a technique with fuzzy based hybrid approach for the detection of similarities/changes between the versions. It classifies the similarities/changes in coarse & finer granularities even at edge level accounting both renaming and shifting of entities when empirically validated using eight Subject Systems (SS). The classification model has been built with high precision and recall in all the cases to identify similarities/changes where Random Forest outperforms in majority SS. Thus, it proves the generalizability of proposed approach and helps to improve productivity, reusability and maintainability with respect to VCA.

Comparing proposed technique with Understand provides 30% more similarities/changes at entity level where Understand fails to capture renaming and shifting. Whereas comparing it with GIT reports increase in number of similar/changed entities at file level by identifying those entities also, where GIT splits the RH due to less similarity, and preserve the linking in RH with PV. Therefore, the proposed technique provides better accuracy and completeness in RH for tracking of similarity/change analysis among entities during software evolution. Improvement in tracking of entities further improves the process of change management and origin analysis.

The organization for rest of the paper is as under where section 2 discusses the related work and motivation for the fuzzy based hybrid approach. Section 3 represents the fuzzy based hybrid technique along with the block diagram used in SBHDetector. Section 4 deals with results and discussion corresponding to the evaluation of proposed technique. It follows section 5 as risks to validity for proposed approach and section 6 briefs the conclusion and future work.

RELATED WORK AND MOTIVATION

Different researches focused on identification of the similarities or changes at file, method and class level for MSR approaches. The authors targeted techniques that uses Diff & Merge operation in