## **Detecting Duplicate Contributions in Pull-Based Model Combining Textual and Change Similarities (DSCI.1)**

**Phase 2**

The pull-based development model is an integral part of many large open-source software (OSS) projects. On this model, programmers all over the world may branch a program’s existing code, alter it to fix some issue, and then submit a pull request (PR) to have their modified code merged into the official build of the software. PRs are met with a process of review and revision which ultimately results in a decision to accept the changes in the project or to reject them. Volunteer submissions to an OSS project contribute greatly to its rapid development. According to studies, communication and coordination between contributors have always been the main challenges to open source software (OSS) communities [1].

The open-source model has proven extremely useful, but its decentralized nature can lead to significant redundancy of effort when multiple PRs are submitted to address the same issue, as they often are in practice. In the absence of rapid detection of these duplicates, multiple groups will work on reviewing and improving solutions to the same problem when only one of these solutions can be implemented. Thus, early detection of PRs that duplicate each other’s purpose would be ideal but is rendered difficult by the size and decentralized nature of OSS projects. Prior studies indicate that duplication is among the most common reasons for rejecting PRs [2,3], which means wasted effort from PR duplication is commonplace. Currently, there are no tools for automatically detecting duplicate PRs for GitHub, and the established way of finding duplicate PRs is manual identification by teams of reviewers but this cannot be the most efficient or the best way to do it: popular projects receive too many PRs for duplicates to be reliably spotted. An effective method for automatic detection of duplicate PRs would thus have great utility.

Some work [4] has been done on the detection of duplicate PRs based on the textual similarity between the descriptions that form part of a PR. But individuals often use different words to state the same thing even without the cultural and linguistic differences that may exist between programmers on international OSS projects. We thus seek to follow the work of Li Z., Yu Y, Wang T *et al*. [5] and improve the measure of similarity used to find likely duplicates by incorporating another source of similarity information: similarity between the actual changes to project code involved in PRs. Finally, we aim to combine the similarity score for both textual and change similarity and rank them according to their similarity and likeness of duplication.

**References**

[1] Herbsleb J D, Mockus A. An empirical study of speed and communication in globally distributed software development. IEEE Transactions on Software Engineering, 2003, 29(6): 481-494. DOI: 10.1109/TSE.2003.1205177.

[2] Gousios G, Pinzger M, van Deursen A. An exploratory study of the pull-based software development model. In *Proc. the 36th International Conference on Software Engineering*, May 2014, pp.345-355. DOI: 10.1145/2568225.2568260.

[3] Steinmacher I, Pinto G, Wiese I S, Gerosa M A. Almost there: A study on quasi-contributors in open-source software projects. In *Proc. the 40th International Conference on Software Engineering*, May 2018, pp.256-266. DOI: 10.1145/3180155.3180208.

[4] Li Z, Yin G, Yu Y, Wang T, Wang H. Detecting duplicate pull-requests in GitHub. In Proc. *the 9th Asia-Pacific Symposium on Internetware*, September 2017, Article No. 20. DOI: 10.1145/3131704.3131725.

[5] Li Z, Yu Y, Wang T, Yin G, Wang H. Detecting Duplicate Contributions in Pull-Based Model Combining Textual and Change Similarities. In the Journal of Computer Science and Technology, January 2021, 36(1): 191-206. DOI: 10.1007/s11390-020-9935-1.