**Improving the reliability of bioinformatics software with unit testing**

Jacqueline Young

M.S. Student

Department of Bioinformatics and Genomics

University of North Carolina at Charlotte

Farnaz Fouladi, Ph.D.

Postdoctoral Research Fellow

Department of Bioinformatics and Genomics

University of North Carolina at Charlotte

Anthony Fodor, Ph.D.

Professor

Department of Bioinformatics and Genomics

University of North Carolina at Charlotte

There is a lack of software engineering practices in bioinformatics when compared to commercial software development. When established best practices for quality control measures, such as unit testing, are not implemented, the reliability of bioinformatics software is uncertain. Therefore, findings based upon insufficiently tested software should be considered insufficiently validated. There is a general consensus among software engineers that in order to create high quality tests, the software already needs to be structurally testable. Additionally, there is research that supports the number of methods executed by a JUnit test is strongly related to that test uncovering a defect. Here, we strive to increase the reliability of an existing bioinformatics Java application, called SequenceVariant. First, we refactored the entire code base of SequenceVariant using the separation of concerns design principle in order to create a testable, modularized code library. Next, we collectively utilized JUnit and Mockito frameworks, within an Eclipse integrated development environment (IDE), to create unit tests and simulate external dependencies, respectively. Finally, we utilized EclEmma to quantify the distribution of code coverage from unit tests across the entire SequenceVariant code base. We achieved 71.2%, 58.2%, and 55.0% code coverage from 54 new unit tests for coverage metrics types/classes, methods, and lines, respectively. Our efforts of refactoring SequenceVariant into a modular code base and then creating unit tests that covered core transformational and computational functions improved the reliability of SequenceVariant. For future work, additional unit tests could be created to cover other workflow functions of SequenceVariant.