CS 6704: CTTC 2

Sourabh Shetty

Identifying Features in Forks

For popular repositories that have a lot of active forks I do think that this is an excellent idea. However, while I do think the research is interesting, I do question whether all the forks really need to be visualized. A lot of people tend to simply fork interesting repositories, many not even intending to do any work on them, and even a lot of those that do intend to work on it or extend some functionality frequently do not end up doing any work on it. For an extreme example, a friend recently showed me a repository they had forked simply because they found the name amusing.

On a similar note, on popular repositories, a lot of the forked repositories contain changes that are not particularly interesting or worth looking into. And many developers make forks and even work on them, without intending to merge the changed code back into the main codebase. Assuming that a satisfactory license is involved, the fork often becomes an independent, albeit derived, project.

In such cases visualizing these repositories isn't very meaningful and a simple counter does a better job at conveying the same information.

Enlightened Debugging*

I liked the thoroughness of the work done in this paper. The example they provided was very detailed and the depiction of each iteration was also very informative. I've read papers that can be descriptive, but without a good example it can make it hard to read and understand.

That being said I wasn't sure I liked the way the test cases were selected. For evaluation, only 27 real faults were considered, and the other 1780 faults that were considered were entirely mutation faults that were generated simply to increase the number of faults. While that in itself is not a bad thing when it comes to testing, I don't believe it truly simulates how faults are found "in the wild". I think perhaps they could have used more real faults to evaluate the technique.

The technique also depends on feedback from the user. This depends on the competence and focus of the user, which might not always be at the same level across the board, which could affect performance.

That being said, I do think that the technique itself is a good idea and that with further work, perhaps even using some sort machine learning techniques, this could be very promising.