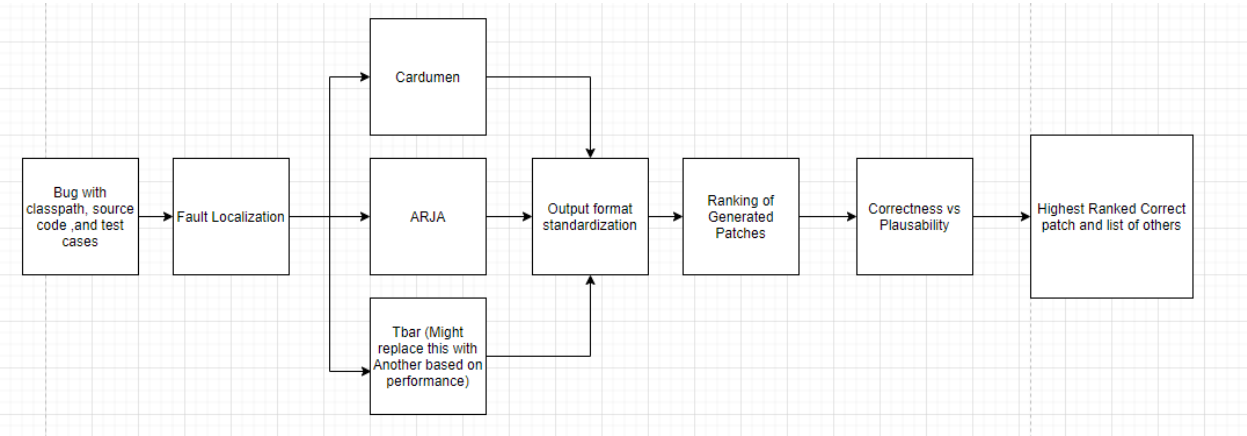
Progress Report - 1

- **Approach**



Right now we have decided on the above structure for the project with each bug being passed into the system first having a common fault localization using GZoltar and Ochiai followed by the bug being passed through Cardumen , ARJA and TBar. We will be doing some level of mixing and matching and could replace any of the methods selected above with jGenProd, jKali, Nopol or SimFix depending on the results of our experiments.

Once the output is generated by each of the tools then we will convert the outputs to a standard model which will then be passed to the ranking mechanism that we will create. The ranking mechanism will be dependent on metrics which we are currently deciding on based on research from the papers we have studied.

Once the ranking of patches is done we will then pass the patches through a classifier to determine if the patch is correct vs plausible and generating one final patch which will be presented to the user (For now we will also be generating a supplementary list with all the patches and corresponding ranks and classification)

- **Novelty**

With our approach we hope to be able to generate both a wider range of patches to bugs and we will create a unique ranking and classification methodology on top of these approaches which will help determine the best possible fix to help the user save time with respect to finding a fix and reducing time spent by developers going through patches which do not make sense

- **Value to User Community**

The developers who use our tool would be able to easily get a fix for the bugs they have in their system and would need to spend less time parsing through the generated fixes which

are already classified as correct and ranked to give them what we believe would be the best fix thus saving them time and effort

- **Dataset(s)**

We will be using Defects4J as the base dataset of our tool , It is a well defined Java based bug benchmark containing the necessary ingredients for bug fixing tools including source code, the needed libraries , the test cases , etc.

We also plan to run the comparison subjects on Defects4J and compare results, we will probably stick to a subset of the bugs within Defects4J using charts and a few other specific subsections

- **Comparison Subjects**

We will be running different tools like Nopol, SimFix as well as all of the approaches we used but individually on Defects4J and evaluating the number of fixes generated by each approach versus the approach that we have chosen.

We will evaluate our approach based on:

1. For our comparison we will be noting down the number of unique patch fixes generated by each system in the comparison list and how they perform
2. We will also be trying to determine the amount of time taken for each approach to fix a particular bug.
3. For each tool we will also be going through a subset of the patches generated to determine if a particular patch is correct and if it can be used in a real world scenario .
4. We will also capture the amount of time it takes to go through an incorrect patch to understand the amount of time saved by our classification.
5. We would also like to take into consideration the ranking for a subset of patches and how well the ranking methodology performs in setting up users to go through the best possible patch

- **Distribution**

We will be creating a github repository with the final code placed on to the repo and a separate compilation of results. We were thinking of making the repositories publicly accessible to allow anyone to access them and also provide links to the codes for the techniques we will be using in the readme

Team Member Roles

1. George

   Will be picking up the experiment of the different tools to run together for the best results and also the isolating of a single fault localization methodology for the tools in one common methodology without using new approaches for each one
2. Mavis

Will be responsible for looking into the ranking methodology and patch generation output of each tool and propose/devise how we would approach ranking the patches generated.

3. Yin

   Will be responsible for identifying the patch correctness vs plausibility. Learning from this paper called Identifying Patch Correctness in Test-Based Program Repair and finding out how we are going to construct a classifier that would generate the most ‘valuable’ patch to the user.