5- Anubhav's Cheerleaders- Aiden Burgess (abur970)

Team members: Aiden Burgess, Anubhav Khanna, Charles Paterson, Seif Younes, Shih-Hao Chen, Sreeniketh Raghavan

1 Implementation

The implementation started by Sam and I creating packages and skeleton classes derived from our pseudocode and UML diagrams. This allowed the parallelisation of the next major tasks to complete milestone 1: parsing the command line input, reading the dot file, executing the scheduling algorithm, and finally writing the result out as a file.

The first technical choice that was decided was to use JavaFX for our visualisation as our team had previous experience with this library. After Sam researched and tested scheduling algorithms, we decided to use DFS branch and bound as it was faster in all randomly generated test cases.

The current implementation uses the DFS branch and bound algorithm, with only one bound function. Further work is required on visualisation, multithreading, and optimization of the algorithm.

One of the challenges we faced initially was finding out how to structure an application with optional visualisation and setting up the tests. Currently, a challenge our team is facing is how to introduce scenebuilder and JFoenix to aid the creation of the visualisation, as we are using different OS's. Another challenge is finding a bounding function that is more efficient and effective than our current implementation.

My contribution to the team includes attending meetings, creating skeleton code with Sam, developing the command line parser alongside tests for that parser and reviewing the pull request for dot file reading and writing.

2 Development Process

Development of the process has been hosted on Github. Each feature is assigned to a team member and a branch is created specifically for this feature. For branches to be merged into master, the pull request must be reviewed by at least one member who did not develop that branch.

Communication has taken place in person frequently during meetings and with updates/planning online. Everyone participates in the discussions, so all opinions are heard. We have adapted to the current situation by utilising Zoom for video calls. Currently decision making has been done ad-hoc, with no leader.

There haven’t been any major conflicts, but many smaller discussions online and in meetings. These were resolved through presenting reasons for the decision until both parties agreed. In the future we may need a leader to decide.

The tools and technologies used to achieve milestone 1 are: Java, Git, Trello, Messenger and Google Drive. Our documentation is hosted mainly on Google Drive, with Javadoc comments in the code.

3 Reflection on development process

By laying out the flow of the code, we were able to parallelise four different tasks at the same time, allowing our team to be very efficient. Each team member has had a task with a deadline throughout the development process.

Overall, I think the development process has worked well, given that the milestone 1 implementation was complete on the 13th of August, 6 days before the deadline. This implementation includes an optimal algorithm along with a test suite for each class. This success can be attributed to the accountability of each team member to show and talk about what they have done at meetings.

To improve our process, I believe we should have a team leader to manage the meetings and encourage communication, as this has been ad-hoc so far.

The team has been very cohesive, as we have rapport and trust from knowing each other before starting this project together.