1. Introduction

- **Project Overview**:We are going to design, build and optimize the planning algorithm for Multi-Agent Path Finding Problem (MAPF) using multi-core platforms. We will also analyze the speedup and solution costs across different high level algorithm approaches.
- **Current Status**: We have started off with some unsuccessful ideas and attempts to tackle the problems. This include trying to parallelize the low-level search algorithm (A* and BFS), which did not perform well due to the low potential for parallelism (not a lot of work can be parallelized) and high synchronization and communication overhead (need to address conflict too frequently).

  We moved on to use the priority table for planning and with heuristic to guide the priority selections. We used this method as our benchmarking method and performed extensive testing to select the right map, scenario and the right range for the number of agents as our experiments.

  After that, we tried to optimize and parallelize the algorithm. Our goal is to help increase the success rate of finding a solution and also improve the solution quality (lower the Sum of Cost). At the same time, we also tried our second approach to solve the problem, which is using batch processing based on the idea of priority planning. We tried to run A* search for agents within the same batch in parallel and resolve existing conflicts after each run. We have performed preliminary testing to verify the correctness of our new approach and are still in the process of running the experiments for parallel versions.

2. Revised Schedule and Future Work

- **Completed Tasks**:
  - We have used 2 different sequential approaches to tackle the problem.
  - We have performed experiments on the sequential approach to select the best parameters (map, scenarios and agents) for benchmarking
  - For the batch processing approach, we are working on the parallel version and trying to improve its efficiency and reduce time spent on synchronization and conflict resolving

- **Upcoming Tasks and Timeline:**
- April 16th - 19th: Finish implementation of parallel batch processing approach for improvement in time(Steven Liu) and optimizing the parallel PP algorithm to improve the solution quality and success rate (Zhifei Li)
- April 20th - 24th: Make modifications to existing batch processing to generate solution with better cost bound(Steven Liu) and run experiments for all of our implementations and produce graphs for presentation (Zhifei Li)
- April 25th - 28th: Finish benchmarking tests, summarize results, write up final report and prepare for poster session (Steven Liu and Zhifei Li). Prepare the video demo for

presentation (Zhifei Li)

3. Deliverables and Project Goals

- **Current Goals**: we think that we are able to finish all the goals we mentioned in the proposal. This includes: at least 2 different approaches (finished), parallelize those approaches (in progress), benchmark testing (in progress), video demo (will complete at the end)
- **Revised Goals**: For the benchmark testing phase, we realized that our initial proposal of goals are not sufficient. Therefore, we will investigate more deeply in the analysis section by designing lots of experiments. Currently, we will add a success rate analysis and perhaps a heuristic analysis in the experimentation section.
- **Nice to Haves**: To hedge the risk of unexpected or unsatisfactory results during the experimentation phase, we want to propose a new goal that was added as optional in the initial proposal: including a decentralized planning algorithm using MPI. The intention is that this implementation can serve as a foundation for online algorithm approaches.

4. Poster Session Plans

- We plan to show a video demo that is pipelined with our implementation and the simulation platform. This demo will include the benchmarking map and scenario that we worked on and how each robot acted using the solution we generated by the MAPF solver.
- We will also present a lot of visual graphs based on our experiments. Currently, from our previous extensive testing, we have selected two maps that are best suited for benchmarking and presentation purposes. We will present our graphs in two major ways: **how parallelism helps increase the success rate of finding the solution, and how parallelism helps improve the solution quality of the problem.**

5. Preliminary Results

- **Early Results**: Based on the benchmarking parameters we have selected, we have seen some improvements in terms of less runtime, better solution quality and success rate, which is consistent with our guesses. These preliminary results are based on the results we saw during testing our parallel versions, which is not rigorous since we are not fixing a lot of other factors, but it gives us a rough estimate of what we are expecting.

6. Issues and Concerns

- **Current Challenges**: We worried about how our second approach (Batch Priority Planning) will perform under the parallel settings. Intuitively, it might eliminate a lot of possibilities given the nature of low success rate of priority planning. Due to the rapid growth under permutations and small set of possible priority tables, we are uncertain about the success rate for the algorithm. Currently, the batch approach generates too many conflicts that require more time to fix than the benefit parallelism brings. We will try to use some algorithm or heuristics to divide and arrange work better than random assignment to decrease conflicts, but the effectiveness is unknown.