**Search Algorithm Parallelization**

Seth Hanusik, Raeshawn Bart, and Robert Deal

**Background**

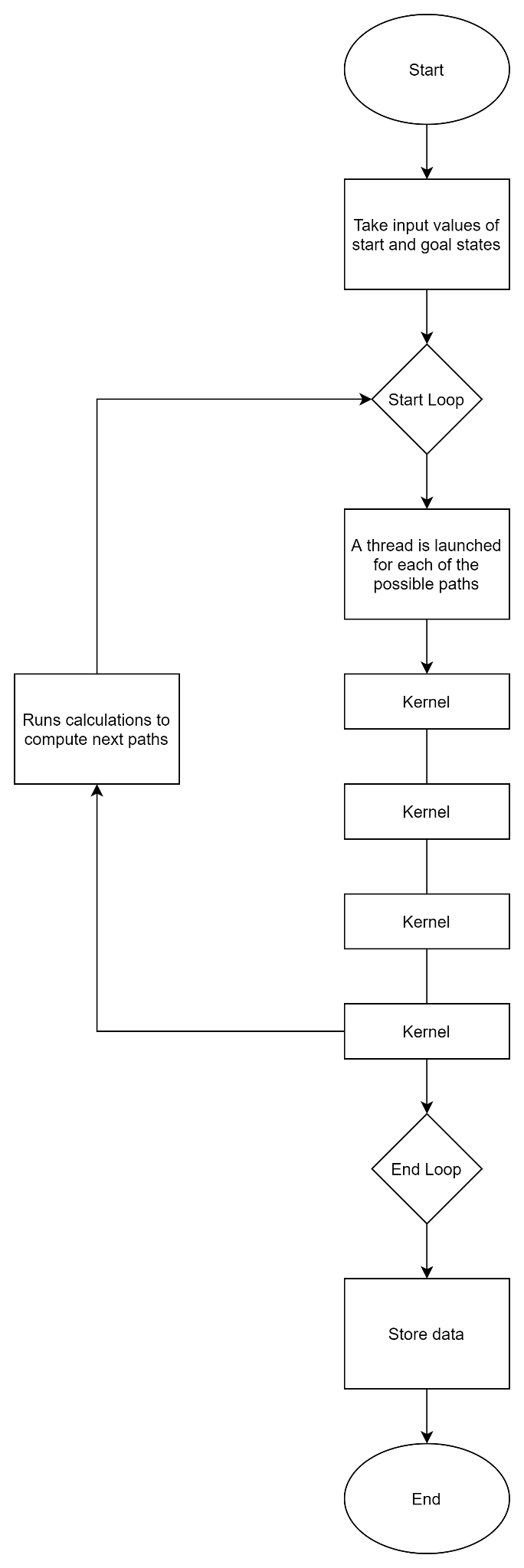
The reasoning behind the choice of this topic is that I want to parallelize a search algorithm to be able to the overall yielded improvement that can be created through different kinds of processors and numerous threads. I remembered back to my AI class and how I had to write code that terminated the program if it exceeded a certain time, so I want to be able to run those algorithms in a significantly faster time.

**Application Level Objectives**

I want to see an improvement on the time it takes to complete complex search algorithms specifically A\* and Breadth First Search. At the end, the benefit would be that rather than taking an extended period of time as the amount of paths possible for the algorithm to take expands, we will be able to write the algorithm so each thread takes the calculations for the next step of the path resulting in a significant time decrease in solving.

**Design Overview**

The parallelization here would be allowing each thread to take a possible path of the algorithm to do its calculations rather than being only able to do one at a time as on a CPU.



**Performance Goals and Validation**

We don’t have a specific number yet for the improvement we should see when parallelizing it, but there will be a program in C++ that runs serially, and we will use the same inputs in that program in the parallel code and compare the length in time that it takes the algorithm to complete. The bottlenecks could come in where the number of possibilities of paths exceeds the number of threads that we have running. Our testing procedure would just be to compare the execution time of the serial and parallel programs with the same inputs.

**Schedule and Division of Work**

The first week will be the planning stage where we ultimately decide how the parallelization will be done on the serial program. The next milestone will be implementation of the first instance of parallelization by the end of the second week. The third week will be dedicated to making improvements on our first instance. Week four will be for putting the report together and any final modifications we might need. Each team member will work on the implementations equally, giving each person their own algorithm instance to implement. For the report, we will each take our part of the implementation and describe that and we will each write part of the other sections of the report.