15-418 Parallel Computer Architecture and Programming
Final Project Checkpoint

Jordan Widjaja - Benjamin Chu
jwidjaja@andrew.cmu.edu - bcchu@andrew.cmu.edu
Tuesday, April 25th, 2017

1: (Updated) Project Schedule

- **Week of 04/17** - Our goal is to create a simple, sequential implementation of Boruvka's Algorithm (JORDAN). The initial code consisted of an in-program hard-coded graph, but we later implemented a Python script which automatically creates sample tests for various types of graphs to test on our algorithm implementation (BEN). This will be used as our baseline for establishing both correctness and naive speed.
- **Week of 04/24** - We begin to design our code to handle situations in which it will be efficient to parallelize parts of our graph. We utilize benchmarks in which we create a threshold for a (defined) dense graph beyond which it would be more time efficient to run Boruvka's algorithm in parallel on dense sections of the graph (JORDAN). Along with this, we need to know where exactly in our sequential implementation that we can parallelize independent portions of code. Recognize that finding the MST of two different sectors of a graph will not necessarily combine to an MST of the entire graph, so doing so will only create a close approximation (for larger graphs) of the MST. We will compare the results of approximating the graph to the naive, but correct, implementation (JORDAN and BEN). We use OpenMP for parallelizing code.
- **Week of 05/01** - We test our code for multiple types of dense and sparse graphs (JORDAN). Analyze the various ways of contracting the graph when analyzing dense sections to allow for more efficient running times (edge contraction or star contraction) (BEN). Find out how to utilize the software which creates graphs to analyze time efficiency versus correctness (JORDAN).
- **Week of 05/08** - Add finishing touches to writeup and organize code files. Write the final writeup and produce concrete graphical representations of our results (JORDAN and BEN).

2: Current Progress

We have been able to accomplish all of our tasks in the week of 04/17. We created a sequential implementation of Boruvka's algorithm using a Union Find data structure. Furthermore, we were able to successfully create a script which can generate larger graphs of various types into a text file and run our implementation on the file to establish basic correctness of results. Also, while not completely implemented, we have discussed critical zones by which we parallelize parts of our graph.

3: Looking Ahead

We appear to be on track in accordance to our schedule. Our final deliverable will consist of an analysis of multiple instances of graph types and a comparison to the naive sequential implementation using line graphs. We will also analyze our parallel implementation using different numbers of threads and discovering a point in which any more threads would not improve efficiency. The biggest concept behind our final results will be the tradeoff between efficiency and correctness, and at what point that it will be more important to prioritize correctness over shorter running time. Even prior to these goals, it is important to understand exactly what parts of our code will useful to parallelize without too harshly jeopardizing correctness.

4: Challenges

Much of our challenges ahead will be in the re-understanding of how to utilize OpenMP to parallelize our code, discovering the different types of graphs for more pedantic analysis.