Assignment 5: Multiprocessor Map-Reduce

1. **Names of group members.**

   Hua Yang, 128-00-2637

   Erik Kamp, 132-00-4838

2. **Detailed instructions for using the program, including any preprocessing and postprocessing steps that must be taken. Make sure that you submit only the program source, headers, and makefile. A user should do nothing more than run *make* to compile the program. Make sure your instructions clearly specify the command name and usage and provide an example.**

   The program will only accept the following command:

   ./mapred –a [wordcount, sort] –i threads –m num_maps –r num_reduces infile output

3. **Architectural description of the framework, including where you had to employ synchronization primitives.**

   - For this project, we only wrote the multi-threaded version, using the p_thread library.
   - For splitting the files equal to x number of mappers, we use the provided split.sh file.
   - The structure that we used to store the mappers and reducers output is an ut_hash.

- Depending on the specified m number of mappers and r number of reducers, we create m threads of mappers and r threads of reducers.

4. **Tests that were run.**

   The tests we did utilized the files from Project Gutenberg.

5. **Description of difficulties and problems that you encountered.**

   - We initially had a problem with forks, where we did not wait for the child processes to finish.
   - The threads were tricky because of the race conditions.
   - We had trouble coming up with a data structure to store the mapped outputs, turned out that ut_hash was the best solution.

6. **If you implemented both threads and processes, an evaluation of the performance difference between the two.**

   We only did threads, since we're in a group less than 3.