# Lightweight MapReduce Framework -- Project Checkpoint

15-418/618 Parallel Computer Architecture and Programming

Zihao He (zihaohe), Shiming Zhuang (szhuang)

Website: https://git.hzh0512.com/p/mapreduce

## Updated Schedule

- Week 4 (Until 11/25): (1) Zihao: further test the static version. (2) Shiming: test on more optimized work distribution designs.

- Week 5 (Until 12/2): (1) Both: optimize the framework and read papers about machine learning implementations on MapReduce framework. (2) Both: implement some mahine learning algorithms.

- Week 6 (Until 12/9): Both: test on the performance of ML algorithms.

- Week 7 (Until 12/16) (12/15 Final Report): Both: write the final report.

## Completed Work

We are following the proposed schedule. We have implemented an asynchronous network communication library based on libevent and the first version of static scheduler can be run on laptops and GHC machines.

## Goals and Deliverables

Our work is compatible with the schedule in the proposal. We have reached the 75% goal.

Up to now, our work for deliverables are focus on basic word counting task. We tested the function of mapper and reducer based on word count tasks. Since the functions work properly in our test cases, we have confidence that it is applicable to more difficult works, such as the machine learning algorithms described in our proposal.

## Poster Session

In the poster session, we plan to show the performance of our project in different applications.

We plan to implement different machine learning algorithms in our framework, and demonstrate the performance of our implementation with comparison to single-threaded implementation. In addition, we can build and deploy other parallel computing framework and compare their performance.

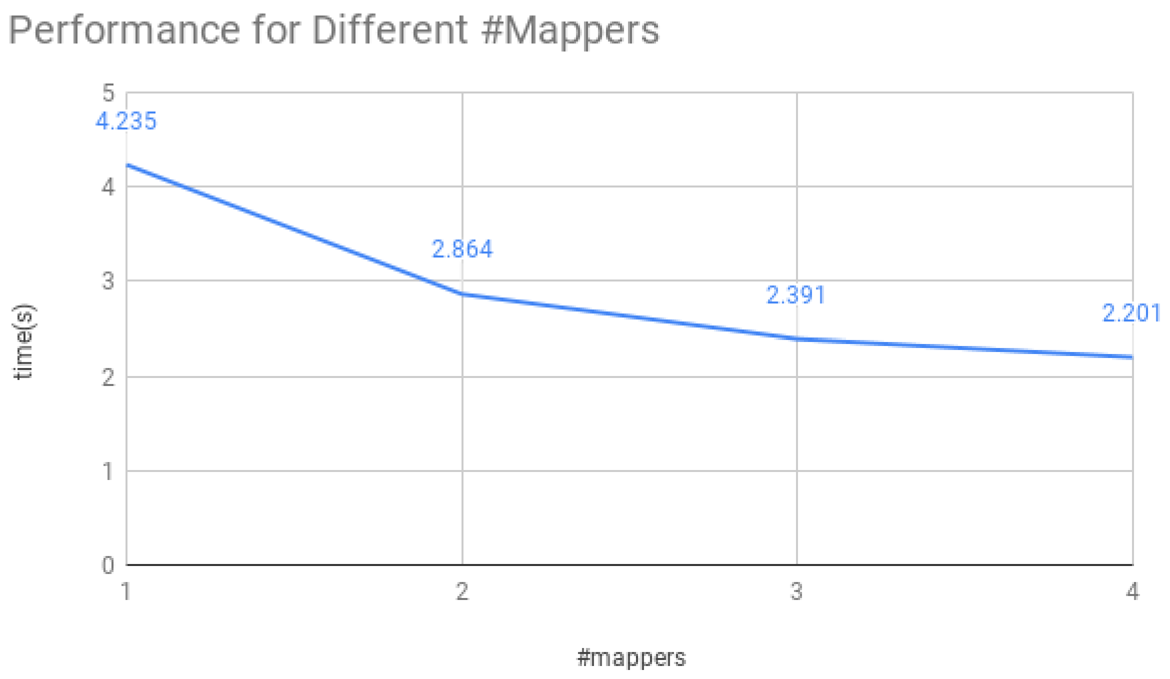
## Preliminary Result

The following result are tested on a localhost with a quad-core Intel i5-8267u CPU.

### Different Number of Mappers

The number of reducers is 2, and the number of tasks is 12.

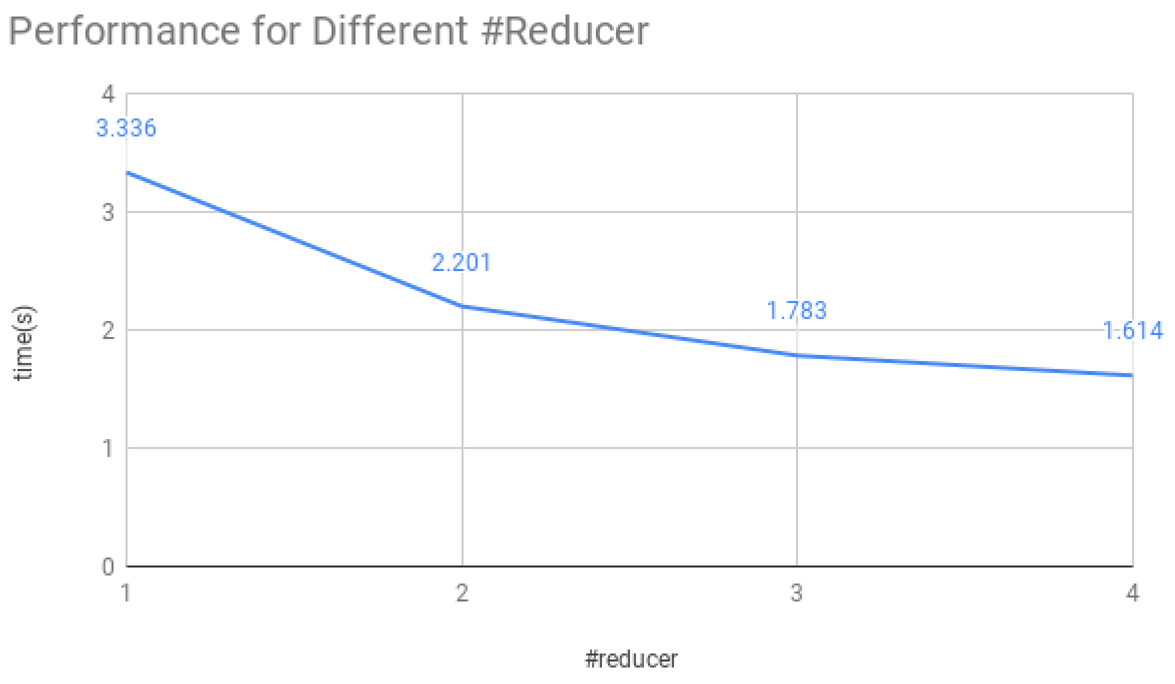
| #mapper | time(s) |
|---|---|
| 1 | 4.235 |
| 2 | 2.864 |
| 3 | 2.391 |
| 4 | 2.201 |



### Different Number of Reducers

The number of mappers is 4, and the number of tasks is 12.

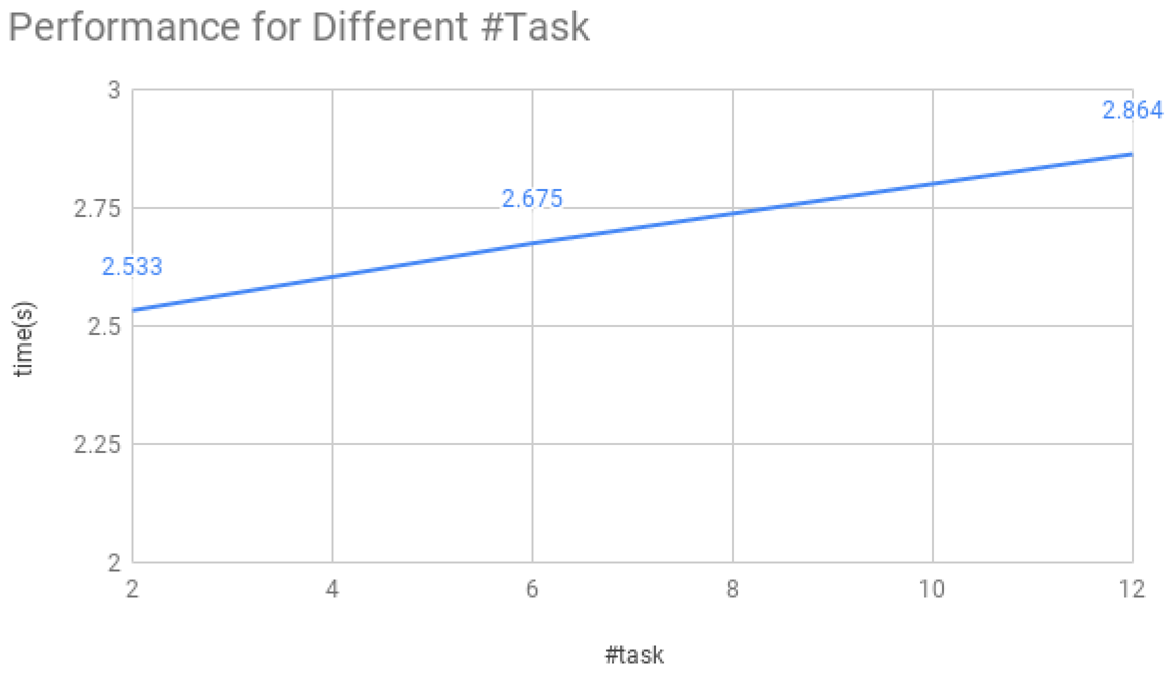
| #reducer | time(s) |
|---|---|
| 1 | 3.336 |
| 2 | 2.201 |
| 3 | 1.783 |
| 4 | 1.614 |



## Different Number of Tasks

The number of mappers is 2, and the number of mappers is 2.

| #mapper | time(s) |
|---|---|
| 2 | 2.533 |
| 6 | 2.675 |
| 12 | 2.864 |



## Issues

### Dynamic Work Scheduler

The original version of our implementation is a static work scheduler. This implementation simplifies the work for master, but it is prone to meet workload imbalance issues. We need to implement a dynamic scheduler and consider the assign strategy properly to avoid workload imbalance.

### Reduce File I/O

In our map-reduce runtime, it is possible that a mapper and a reducer sit on the same machine. In this way, there would be unnecessary file I/O for data produced by the mapper on current node. It is a potential optimization for less file I/O and tranfering the data in memory.