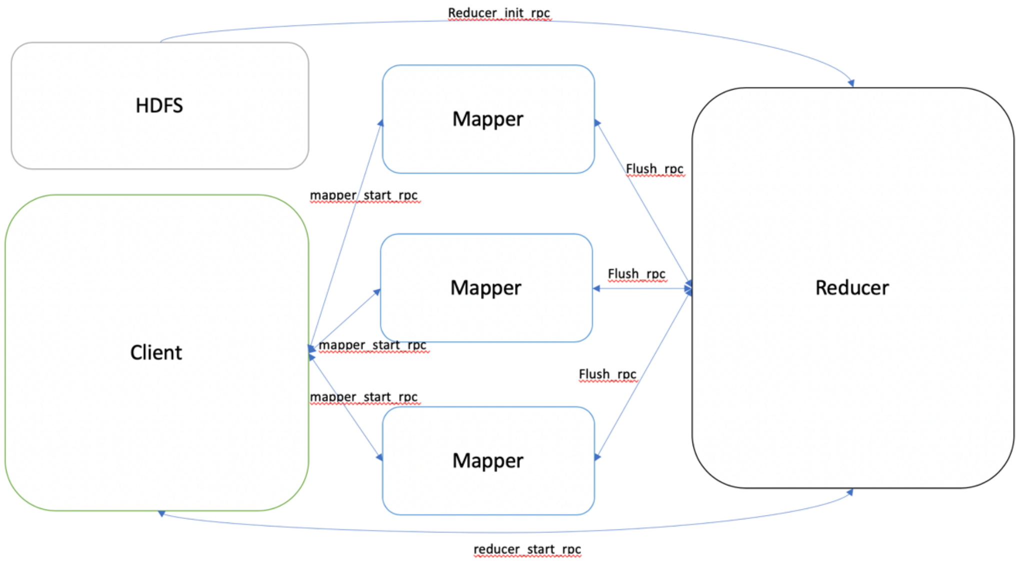
Project 1 Report

Pang-Yang Chu, Qimin Zhang

Structure



Protocols

RPC Name	Requests	Replies
mapper_start_rpc	char* application int n_threads char* n_path char * out_path char* args int id	Int success
reducer_init_rpc		int id int result
reducer_start_rpc		int id int result
flush_rpc	int id char* buffer int size	int finish

Design Decisions

1. **How do us implement executing mapper nodes and reducer node asynchronously?**
   Since our implementation of RPC calls are synchronous. we create threads in the "mr_start" API, before each thread finish, it will store the return value from rpc into map and reduce status stored in map_reduce structure. In "mr_finish" API, we used "pthread_join" to wait for all threads finish and examinate the result.
2. **How do mapper nodes signal reducer node they have finished?**
   When an argument for "flush_rpc" function, size, is set to 0, reducer will set the status of that mapper (identified with its id) to finished.
3. **How does sender/receiver synchronize with producer/comsumer?**
   We used three condition variables to implement the synchronization, which are not_full, not_empty and finished. When sender/consumer sees the buffer is empty, it waits for producer/receiver to signal not_empty. Same, When producer/receiver sees the buffer is full, it waits for sender/consumer to signal not_full. When producer/receiver finishes, it signal sender/consumer it has done.

Evaluation

We evaluate the system in the following parameter setting:

(i) number of Mapper units∈{1,4}
(ii) number of threads within a Mapper unit∈{4,16}
(iii) barrierEnable∈ {1,0}
(iv) size of per-Mapper unit buffer∈ {(10KB,100KB,1M B)};

The result is shown in README.md. We discuss our observation and conclusion here.

The result shows that the more threads we use in a single mapper machine the longer the execution time become, which sticks to our expectation. Surprisingly, we noticed that different buffer sizes and more threads run in different mapper machines does not affect the execution time a lot. We believe that internet transfer is a bottleneck, so the execution time will not change a lot by adding threads. Also, If the time caused by mapping key pairs is not very large and the buffer will not be full very often, the differences can be trivial. Last, we found out that when enable flag is set, the execution time will be a little bit shorter. We think this is because reducer does not need to alternate between threads and being block if lock is held.