**Review of MapReduce**

# **Summary**

MapReduce is a programming model and an associated implementation for processing and generating large data sets, which hides the details of parallelization, fault- tolerance, locality optimization, and load balancing with large cluster of commodity machines and highly scalability.

**Programming Model:**

①Users create *Map* function and *Reduce* function

②*Map* function uses the original data inputted by the users to produce a set of intermediate key/value pairs. Then the MapReduce library put all the values associated with the same key together.

③*Reduce* accepts this intermediate key and the set of values for that key. Then it merges these values to form one output value or zero.

**Contributions**

① Their main contributions are the simple and powerful interface that enables automatic parallelization and distribution of large-scale computations, combined with an implementation of this interface that achieves high performance on large clusters of commodity PCs.

② A large variety of problems are easily expressible as MapReduce computations.

**Comments**

MapReduce is an efficient and economical way to solve large-scale parallel computing problems. This is also very friendly for users to change their functions which meet their requirements.

This model separates control flow and data flow so that the master and task workers are all replaceable, in this case, it has highly fault tolerance.

**Limitation**: MapReduce uses one master to assign map tasks and reduce tasks. When the master fails, the recovery is not very convenient. The recovery needs the clients to retry. One potential solution is to use a shadow master to replace failed master automatically.