When dividing the work into chunks, each processed parallel in different machines, the runtime is dominated by the longest process in any of the computers. Overhead is incurred when independent processes have to be recombined and again processed as a whole. Datasets grow beyond the capability of a single machine if processed in parallel. Synchronization problem may arise, reliability often comes into play and can’t be sacrificed at any cost.

So it is better to overcome these factors of parallel programming by using well-structured framework like Hadoop, and help ease our computation.

Map function helps remove bad data, erroneous result and filter the dataset for further processing. Map and Reduce functions in key-value sets, with input as key and output as value. The output of the map function is processed by the MapReduce framework before being sent to the reduce function for further processing.

Just like the performance of quicksort program is depended on the split, similarly the split of the MapReduce tasks by Hadoop determines the performance of the task. Too small splits leads to overhead in maintaining them in multiple machines and in synchronising their functions in parallel.

Data locality optimization is done for efficient processing of the tasks in the node where the data is stored as bandwidth is saved. Optimal split size is the size of the block where the input data resides, as it the largest possible size that can be used.

Principle of locality is used by the map tasks as the output of the map tasks are stored in the local disks and not fed into the HDFS. This is because the output of the map task is not the final and it must be first fed to the reduce task and then the final output is obtained.