computers. Hadoop has 3 replica blocks. The data replication number can increase the system stability but the unnecessary data replication raises problems as disk space.

Hence, we propose the analysis model of the data usage pattern for improvement of system performance and reducing of the wasted disk space. Our analysis model adapt MapReduce framework for distributed process the large scale data. In Map function, the master node takes the input, chops it up into smaller sub-tasks, and distributes those to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes pattern analysis sub-tasks, and passes the results back to its master node. In Reduce function, the master node then takes the results to all the sub-tasks and combines them in a way to get the output - the results to the task it was originally trying to analyze. We also conducted two types of experiments for evaluation of the proposed analysis model in HDFS.

This paper is organized as follows: Section 2 summarizes Hadoop and MapReduce programming model and Section 3 presents system architecture and the proposed data analysis algorithm. Section 4 deals with the performance analysis and simulation results and finally conclude in Section 5.

2. Related Work

2.1 Hadoop

The Hadoop is a software framework that enables distributed process of large amounts of data. It was originally built as infrastructure for the Apache Nutch web search engine project. The Hadoop has many similarities with existing Google’s GFS and MapReduce programming model. Hadoop is reliable, scalable, efficient. Hadoop maintains several copies of working data to ensure that processing can be redistributed around failed nodes. And it works on the principle of parallelization, allowing data to process in parallel to increase the processing speed. It distributes the data and processing across clusters of commonly available computers.

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware to process a large of data. **Fig. 2** shows an architecture of HDFS. A HDFS cluster has two types of node opearating in a master-worker pattern: a Namenode and a number of Datanode. A Namenode manages the file system namespace and regulates access to files by clients. Datanodes are the work horses of the filesystem. They store and retrieve blocks when they are told to (by clients or the namenode), and they report back to the namenode periodically with lists of blocks that they are storing. The Namenode makes all decisions regarding replication of blocks. It periodically receives a Heartbeat and a Blockreport from each of the Datanodes in the cluster. Receipt of a Heartbeat implies that the Datanode is functioning properly.

HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size 64MB.



**Fig. 1**. HDFS [3]

HDFS doesn’t support the mechanism for handling of data replication number automatically. The data replication number can increase the system stability but the unnecessary data replication raises problems as disk space. Hence, we propose the analysis model of the data usage pattern for improvement of system performance and reducing of the wasted disk space. The proposed mechanism improves or reduces the number of data with different situations.

2.2 MapReduce

MapReduce is a programing model and an associated implementation for porcessing and generating large data set. MapReduce consists of

2) Experimental Analysis

**Fig. 6**. Response Time

**Fig. 7**. Resources usage

**Fig. 6** shows the results of experiments with changing the virtulization types. In case of processing time, the system based on para virtulization is more faster than the system with full virtualization. As you can see **Fig. 7**, the system based on the para virtualization also occupied less amounts of resources. The similar researches[8][9] show experiemtal results that para virtualization has better than full virtualization. So we adopt the system configuration method as a para virtualization. In this paper, all experiments are conducted in the system based on para virtualization.

4.2 Data Analysis Aspects

To provide evaluation our algorithm, we executed two experiments. We use 6 physical machines and collected log data in distributed computing environment.

The first experiments are aimed at comparing common computing environment without MapReduce and distributed computing environment with MapReduce. We conducted experiments by increasing of the data amount. In second experiments, we measured process time by increasing of node numbers.

4.3 The result of the experiment

**Fig. 8** presents the results of first experiment. We compared common single node environment and distributed environment with MapReduce.

**Fig. 8**. Response Time by increasing of data numbers

As a result, the processing time in distirbuted experimrnt is faster than common single node environment. The more we increased a mount of data from 120MB to 1.23GB, the bigger the response time’s gap between two environments.

**Fig. 9**. Response Time by increasing of node numbers

**Fig. 9** presents the results of experiments in distributed environment. Regardless of the data amount, the response time is better according to increasing of node numbers. As a result, we identified the advantages of the distributed file system with MapReduce framework based on two experimental results.

1.