Journal homepage: http://www.journalijar.com
Journal DOI: 10.21474/IJAR01

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

Improving Efficiency in MapReduce by Optimization Algorithm.

S.Valarmathi[1] and S.Hemalatha[2].

1. P.G Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology Coimbatore, India
2. Associate Professor, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology Coimbatore, India.

Manuscript Info

Manuscript History:

Received: 14 April 2016
Final Accepted: 19 May 2016
Published Online: June 2016

Key words:
Map Reduce, Optimization Algorithm, Parallel Hybrid-Particle Swam Optimization.

***Corresponding Author**

S.Valarmathi.

Abstract

In this Work, task-level scheduling algorithms with respect to resource selection and deadline constraints in heterogeneous Map Reduce environment is consider. However, with the advance of computing technologies and the ever-growth of diverse requirements of end-users, a heterogeneous set of resources that take advantages of different network accelerators, machine architectures, and storage hierarchies allow clouds to be more beneficial to the deployments of the Map- Reduce framework for various applications. The heterogeneity is manifested in the through employment of the optimization algorithms in the resource selection process PSO is selected as the auto optimization strategy for Hybrid resources scheduling algorithm (PH-PSO). There are mainly two types of PSO distinguished by different updating rules for calculating the positions and velocities of particles which are task and resource in the map reduce framework. Hyper parameter selection is a kind of continuous optimization problem in the large scale and iterative applications, and feature selection is a kind of binary optimization problem.

Introduction:-

Industries of today, reliant increasingly on large scale data analytics for their business decisions of day to day life. This reliant on large scale data decision motivated the development of MapReduce. MapReduce is a model of program for data-intensive computation which becomes popular in recent years. MapReduce is an implementation on cluster for processing and producing huge data sets with a parallel, distributed algorithm. Together MapReduce is a framework for problems to be parallelizable process across large datasets with the help of huge number of computers (nodes), collectively known as a cluster or a grid. In MapReduce the jobs is partitioned as small tasks like map and reduce tasks and execute parallel among large number of machines (nodes).

In todays companies, the MapReduce frameworks performance and efficiency have became critical to their success. The map and reduce tasks is collectively called as job which is scheduled concurrently on multiple machines which reduces the running time of a job. The job scheduler is the central component of MapReduce system. Job completion time is minimized by spanning the jobs of map and reduces tasks, which is done by job scheduler. The key-value block is give as input to map tasks where the key-value pair is stored in file system of distributed environment, the map tasks performs a user-specified map function and the output will be the intermediate key-value pairs. Subsequently, the reduce task collects and apply the reduce function of user-specified on a collected key-value pairs for final output. From task to task and from job to job the resource consumptions run time varies. Several recent studies have reported that production workloads often have diverse utilization profiles and performance requirements [2], [3]. Hadoop is the most famous frame for implementing the MapReduce. The Hadoop cluster consists of commodity machines of large number where one node acts as a master and other nodes as slaves. The resource manager is runs on a master node which is responsible for task scheduling on the slave nodes. The local node manager runs on slave node which is responsible for allocating and launching resources for each task. For this above process the Java Virtual Machine(JVM) is launched by the task tracker. The JVM executes the map or reduce function.

System architecture design:-

In this section, present the architecture of the system. Then, illustrate the work flow of the system in detail.

A. System Work Flow:-

The dataset is the main part in the Map Reduce. The Map Reduce works entirely on the base of the dataset information. That dataset is described as the workload in this system. The dataset is described as task or job. Once the job arrives into the system, the jobs characteristic are analyzed for the further usage. The characteristic of the jobs are nothing but the information or details about jobs. That information is stored in a centralized location. The centralized location is named as Resource Information Centre. In this part only the optimization algorithm is applied. The PSO algorithm is used as the optimization algorithm. The optimization algorithm is used to allocate the job to the best resource. The allocation of resource for the job is based on the jobs characteristic which has been analyzed before. This allocation helps in the fast analysis of data which improves the overall performance. This flow is show in the Fig 3.1.

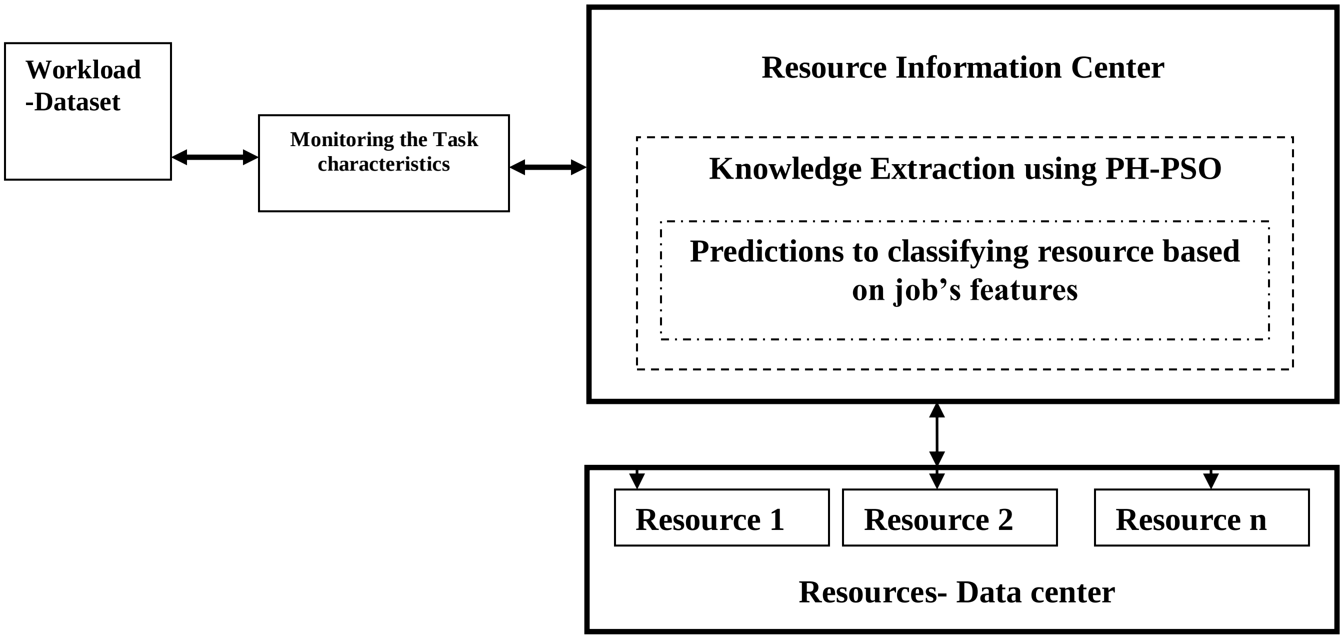

Workload -Dataset
Monitoring the Task characteristics
Resource Information Center
Knowledge Extraction using PH-PSO
Predictions to classifying resource based on job's features
Resource 1
Resource 2
Resource n
Resources- Data center

Fig 3.1: Flow of Framework

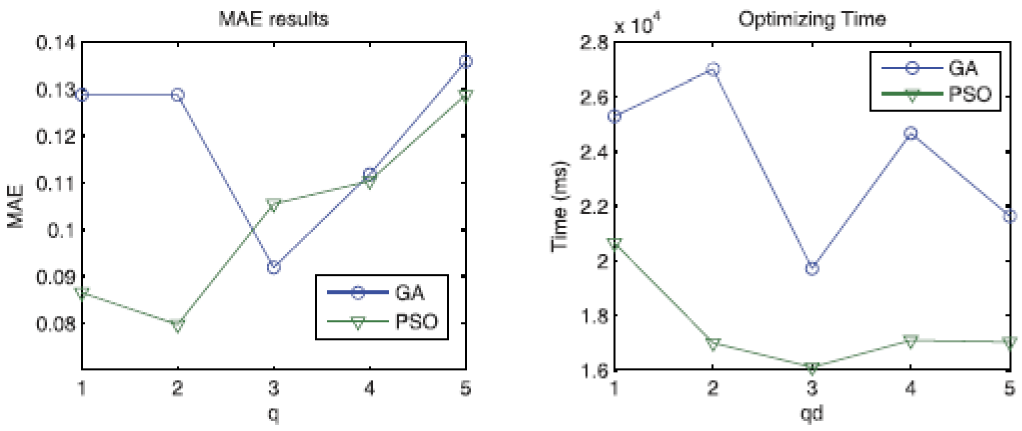
Comparative study on modeling algorithm:-

Genetic algorithm (GA) and particle swarm optimization (PSO) are two typical machine learning strategies in the category of evolutionary computation. These two methods can be employed to optimize the prediction model, for the expectation of achieving higher performance. GA was proposed by John Holland and his students in 1975 [15], inspired by the theory of natural selection and evolution. GA uses a set of chromosomes to represent solutions. The chromosomes from one population are taken and used to form a new population which is called offspring. The chromosomes with better fitness will have more chances for reproduction, and consequently, the new population will be better than the old one.

The PSO was proposed by Kennedy and Eberhart [12], inspired by social behavior of nature system, such as bird flocking or fish schooling. The system initializes a population of random particles and searches a multidimensional solution space for optima by updating particle generations. Each particle moves based on the direction of local best solution discovered by itself, and global best solution shared by the swarm.

This study aims at comparing the optimization performance of GA and PSO by simulation. We concentrate on hyperparameter selection using host load data set. Parameters are initialized with values that are commonly used: acceleration constants c1 and c2 are selected according to [12], decreasing inertia weight w linearly with time as proposed in [13], and changing SVR's hyperparameters C; "; _ exponentially during optimization [16]. The initialized parameters of GA and PSO and optimized hyperparameters of SVRs are given in Tables 4.1 and 4.2. If the input feature number is too small, we can't tell the difference of optimizing time between the two, so we set it to 10.

MAE was used to measure the accuracy of the optimized model, and optimizing time was recorded to measure its efficiency. From Figs. 4.1a and 4.2b we can find that PSO achieves lower error than GA in most of the q values considered, and has less optimizing time. Based on such comparative results, PSO is selected by our system as optimization strategy for prediction models.



(a) (b)

Fig 4.1 Comparison between GA and PSO (a) Accuracy (b) Efficiency

parameter	$q=1$	$q=2$	$q=3$	$q=4$	$q=5$
feature number of samples (m)	10	10	10	10	10
population	10	10	10	10	10
mutation rate	0.2	0.2	0.2	0.2	0.2
crossover rate	0.9	0.9	0.9	0.9	0.9
interval of C	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
interval of ε	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
interval of γ	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
C	0.25	256.0	0.25	0.125	4.0
ε	128.0	0.125	0.0078	0.0625	0.125
γ	1024.0	0.03125	0.25	64.0	8.0

Table 4.1Parameter of GA

parameter	$q = 1$	$q = 2$	$q = 3$	$q = 4$	$q = 5$
feature number of samples (m)	10	10	10	10	10
population	10	10	10	10	10
Inertia weight (w)	$1.4 \rightarrow 0.5$	$1.4 \rightarrow 0.5$	$1.4 \rightarrow 0.5$	$1.4 \rightarrow 0.5$	$1.4 \rightarrow 0.5$
Acceleration constants	2	2	2	2	2
interval of C	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
interval of ε	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
interval of γ	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$	$[2^{-10}, 2^{10}]$
C	9.0889	9.7656e-4	1.6678	0.9805	1024.0
ε	1024.0	1024.0	0.1041	0.1101	266.4968
γ	9.7656e-4	0.8051	0.0414	1.5194	9.7656e-4

Table 4.2 Parameter of PSO

Proposed optimization algorithm:-

There are mainly two types of PSO distinguished by different updating rules for calculating the positions and velocities of particles: continuous version [12], [13] and binary version [14]. Hyperparameter selection is a kind of continuous optimization problem, and feature selection is a kind of binary optimization problem. Concerning our optimization problem definition, this study proposes a parallel optimization algorithm which hybridizes continuous PSO and binary PSO together, namely PH-PSO.

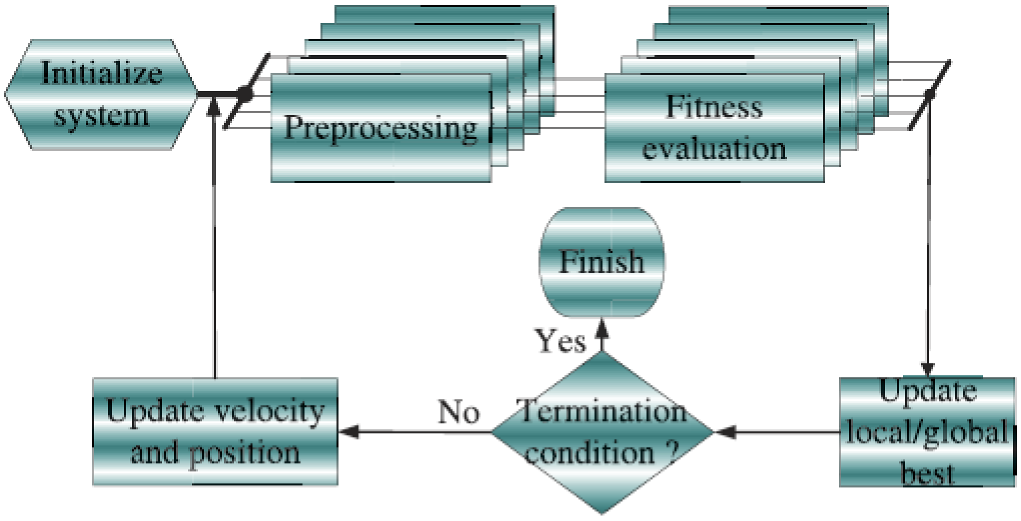


Fig 5.1 Flow chart of PSO.

Flow of Algorithm:-

No.of Particles – No .of Tasks
Search Space – No. of Resources
Each task (particle) treated as point in N dimensional Space
Each particle organizes it's the best solution based on fitness function is treated as p best
Any particle organized in the particular space based on fitness function is treated a g best
Particle is iterated to attain the pbest location and g best locations by Random weight assignment at each iteration – Position Modification

$$\mathbf{V_i^{k+1} = wV_i^k + c_1\, rand_1(\ldots) \times (pbest_i - s_i^k) + c_2\, rand_2(\ldots) \times (gbest - s_i^k)} \text{ -----(1)}$$

Where v_i^k : velocity of agent i at iteration k,
c_j : weighting factor,
w:weightingfunction,
rand : uniformly distributed random number between 0 and 1,
s_i^k : current position of agent i at iteration k,

$pbest_i$: pbest of agent i,

gbest: gbest of the group

$$w = wMax-[(wMax-wMin) \times iter]/maxIter \quad ------- \quad (2)$$

where wMax= initial weight,

wMin = final weight,

maxIter = maximum iteration number,

iter = current iteration number.

$$s_i^{k+1} = s_i^k + V_i^{k+}1 \quad ---(3)$$

A large inertia weight (*w*) facilitates a global search while a small inertia weight facilitates a local search.

By linearly decreasing the inertia weight from a relatively large value to a small value through the course of the PSO run gives the best PSO performance compared with fixed inertia weight settings

Evaluation on proposed system:-

A. Data Preprocessing:-

The data preprocessing is the phase that need to be done before scheduling the resources and task. Hadoop framework is been established with complex distributed systems composed of resources and Tasks. Resource information extraction process has been build to monitor and predict Hadoop resource state information for our system architecture. Task users do not need traversal of all the nodes or hadoop expertise to get information. We design a uniform and friendly interface component for accessing the information. Computing system architecture maintains a service container for taking jobs; such container should be reused for seamless fusion between a environment and task or workload. Executing jobs is the fundamental function of a Hadoop system, we deploy resource sensors on computing nodes since it's inevitable, they also run and sleep dynamically to reduce overhead, while we deploy other components out of computing nodes to avoid extra overhead.

B. Genetic Algorithm

This algorithm is used to schedule the resource and task. The steps involved in this algorithm as follows,

(a) Fix a prediction model of machine learning algorithm, and set its default hyper-parameters. Separate the sample set into three parts: training set, validation set and test set.

(b) Feed the learning algorithm with a sample of training set, repeat it one by one until all samples are used. For some algorithms, the training procedure runs only once; for others, iterations are needed.

(c) Feed the trained model with all samples of validation set, record the errors between true data and predicted ones.

(d) Fix an optimization algorithm, which evolves the hyper-parameters of prediction model for better fitness (performance).

(e) When a termination condition is met, optimized prediction model is achieved, and then tested using test set.

C. Modeling the hybrid PSO in the map reduce framework

In this module, define combined criteria for fitness evaluation of the resource for task, and propose a Parallel Hybrid Particle Swarm Optimization (PH-PSO) algorithm for resource prediction through map reduce framework. PH-PSO takes both hyper-parameter selection and feature selection under consideration, thus is expected to enhance the accuracy and efficiency of data centres. There are mainly two types of PSO distinguished by different updating rules for calculating the positions and velocities of particles: continuous version and binary version .Hyper-parameter selection is a kind of continuous optimization problem, and feature selection is a kind of binary optimization problem. Concerning the optimization problem definition, this module proposes a parallel optimization algorithm which hybridizes continuous PSO and binary PSO together, namely PH-PSO. A system is initialized with a population of random particles (task) and searches a multi-dimensional solution space (resource) for optima by updating particle generations. Each particle calculates its own velocity and updates its position in each iteration until the termination condition is met.

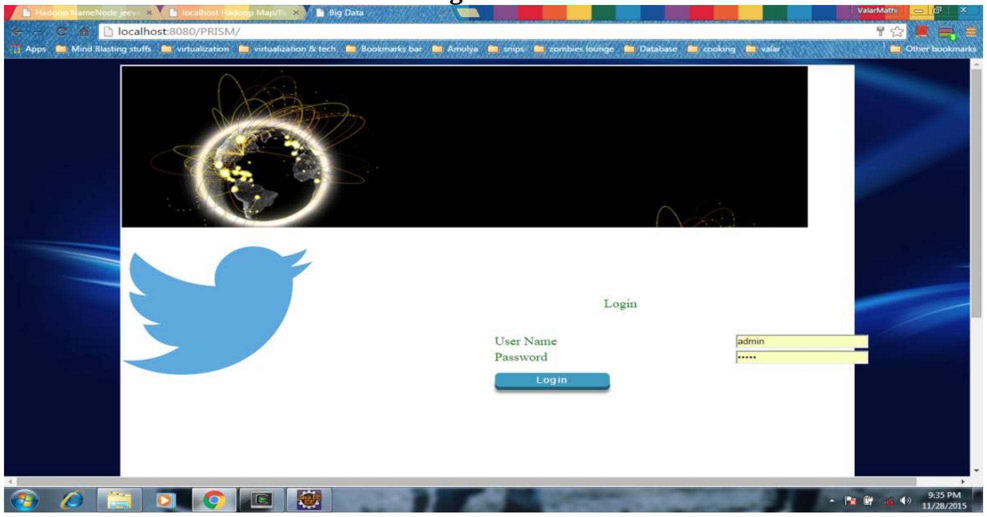
D. Modeling the resource Selection mechanism based on PSO

In this module, design is to model the monitoring system to map phase. The process indicates certain operations (i.e. I/O operation) and calculates the running performance of resource as monitoring data, such as latency and bandwidth; estimate the CPU and memory usage to evaluate their overhead. Sampling frequency is set to once per minute.

Prediction service controls the overall prediction procedure through reduce phase, and evaluation services are used for evaluating model fitness in parallel. The number of evaluation services used for fitness evaluation is equal to the number of particles in PH-PSO algorithm. All the tests are implemented through dynamic collaboration of system services. High accuracy and efficiency is the primary design goal of prediction subsystem. Present the prediction and optimization results of bandwidth. In *q*-step-ahead prediction, *q* = 1, 2, 3, 4, 5 are considered. In model optimization, implemented four different strategies including feature selection with hyper-parameter selection, feature selection without hyper-parameter selection, hyper-parameter selection without feature selection for optimization mechanism. The test data sets used are the same. Record parallel/serial CPU time for optimizing models to measure prediction accuracy. It is implied that our modeling method is suitable for both one-step-ahead and multi-step-ahead resource state predictions. The model's training time can be obviously reduced by feature selection rather than hyper parameter selection. It is clear that the optimizing time of combinational optimization is rather short by means of parallelization, namely within 3 seconds on data sets. The global best fitness of combinational optimization during each iteration was recorded to evaluate the convergence performance. A trend is obvious on host load data set that the global best fitness decreases clearly as the prediction step *q* increases. While such trend is not found on bandwidth data set, which implies that the bandwidth variation has got more noise than host load. It is also implied in these sub-figures that the combinational optimization converges during proper iterations for most of the *q* values considered.

Result:-

Login screen



admin

Fig 10.1 Login screen

Main menu page

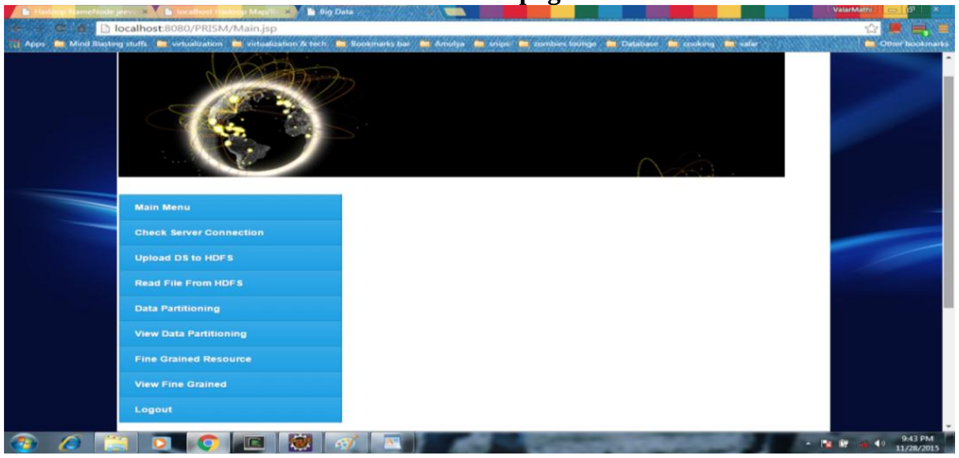


Fig 10.2 The Main Home page

Uploading the dataset into hdfs

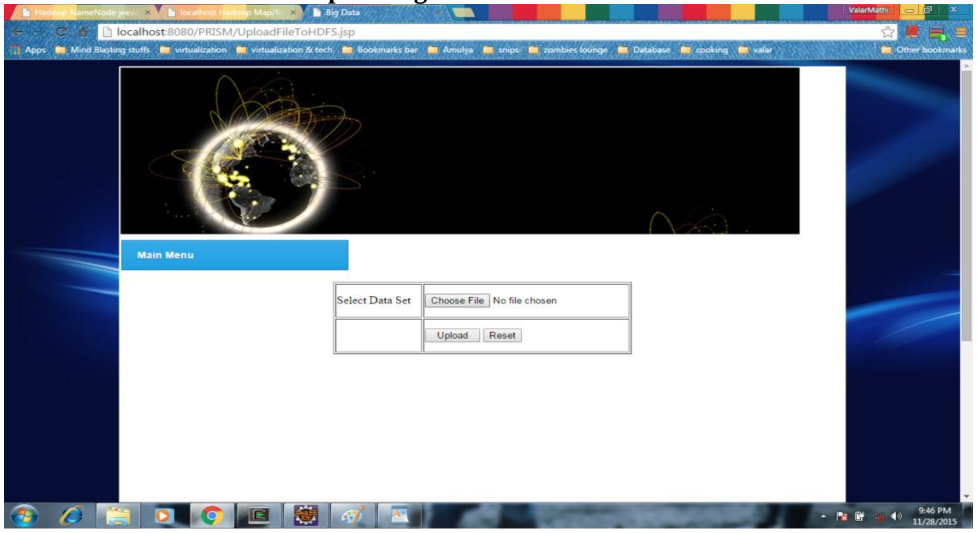
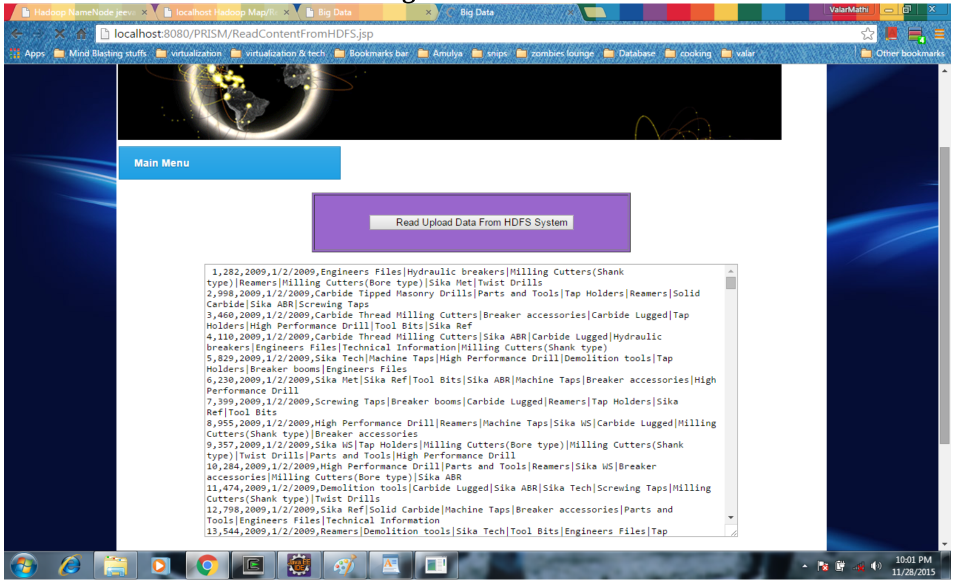


Fig 10.3 The data uploading into the HADOOP file system

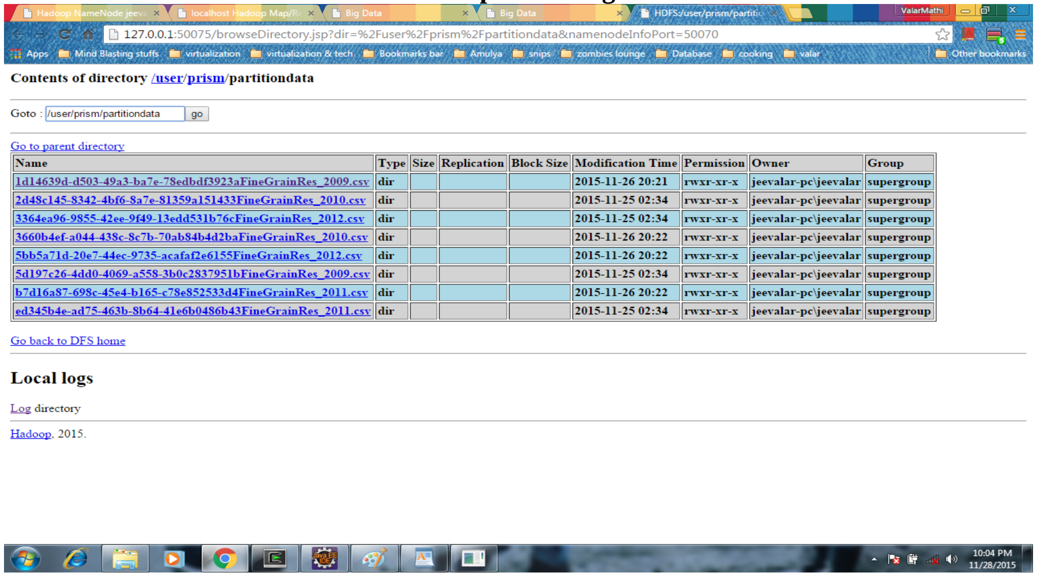
Reading the data from hdfs


localhost:8080/PRISM/ReadContentFromHDFS.jsp
Main Menu
Read Upload Data From HDFS System

```
 1,282,2009,1/2/2009,Engineers Files|Hydraulic breakers|Milling Cutters(Shank
type)|Reamers|Milling Cutters(Bore type)|Sika Met|Twist Drills
2,998,2009,1/2/2009,Carbide Tipped Masonry Drills|Parts and Tools|Tap Holders|Reamers|Solid
Carbide|Sika ABR|Screwing Taps
3,460,2009,1/2/2009,Carbide Thread Milling Cutters|Breaker accessories|Carbide Lugged|Tap
Holders|High Performance Drill|Tool Bits|Sika Ref
4,110,2009,1/2/2009,Carbide Thread Milling Cutters|Sika ABR|Carbide Lugged|Hydraulic
breakers|Engineers Files|Technical Information|Milling Cutters(Shank type)
5,829,2009,1/2/2009,Sika Tech|Machine Taps|High Performance Drill|Demolition tools|Tap
Holders|Breaker booms|Engineers Files
6,230,2009,1/2/2009,Sika Met|Sika Ref|Tool Bits|Sika ABR|Machine Taps|Breaker accessories|High
Performance Drill
7,399,2009,1/2/2009,Screwing Taps|Breaker booms|Carbide Lugged|Reamers|Tap Holders|Sika
Ref|Tool Bits
8,955,2009,1/2/2009,High Performance Drill|Reamers|Machine Taps|Sika WS|Carbide Lugged|Milling
Cutters(Shank type)|Breaker accessories
9,357,2009,1/2/2009,Sika WS|Tap Holders|Milling Cutters(Bore type)|Milling Cutters(Shank
type)|Twist Drills|Parts and Tools|High Performance Drill
10,284,2009,1/2/2009,High Performance Drill|Parts and Tools|Reamers|Sika WS|Breaker
accessories|Milling Cutters(Bore type)|Sika ABR
11,474,2009,1/2/2009,Demolition tools|Carbide Lugged|Sika ABR|Sika Tech|Screwing Taps|Milling
Cutters(Shank type)|Twist Drills
12,798,2009,1/2/2009,Sika Ref|Solid Carbide|Machine Taps|Breaker accessories|Parts and
Tools|Engineers Files|Technical Information
13,544,2009,1/2/2009,Reamers|Demolition tools|Sika Tech|Tool Bits|Engineers Files|Tap
```

Fig 10.4 Viewing the uploaded data from HDFS

Data partitioning


127.0.0.1:50075/browseDirectory.jsp?dir=%2Fuser%2Fprism%2Fpartitiondata&namenodeInfoPort=50070

Contents of directory /user/prism/partitiondata

Goto : /user/prism/partitiondata go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
1d14639d-d503-49a3-ba7e-78edbdf3923aFineGrainRes_2009.csv	dir				2015-11-26 20:21	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
2d48c145-8342-4bf6-8a7e-81359a151433FineGrainRes_2010.csv	dir				2015-11-25 02:34	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
3364ea96-9855-42ee-9f49-13edd531b76cFineGrainRes_2012.csv	dir				2015-11-25 02:34	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
3660b4ef-a044-438c-8c7b-70ab84b4d2baFineGrainRes_2010.csv	dir				2015-11-26 20:22	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
5bb5a71d-20e7-44ec-9735-acafaf2e6155FineGrainRes_2012.csv	dir				2015-11-26 20:22	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
5d197c26-4dd0-4069-a558-3b0c2837951bFineGrainRes_2009.csv	dir				2015-11-25 02:34	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
b7d16a87-698c-45e4-b165-c78e852533d4FineGrainRes_2011.csv	dir				2015-11-26 20:22	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
ed345b4e-ad75-463b-8b64-41e6b0486b43FineGrainRes_2011.csv	dir				2015-11-25 02:34	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup

Go back to DFS home

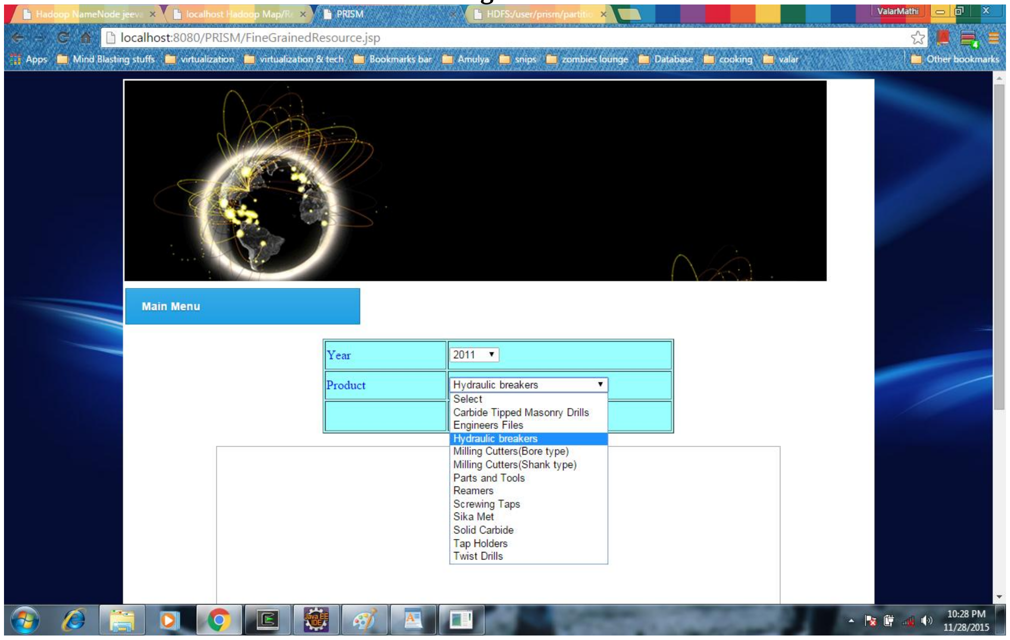
Local logs

Log directory

Hadoop, 2015.

Fig 10.5 The loaded data is partitioned

Selecting the details

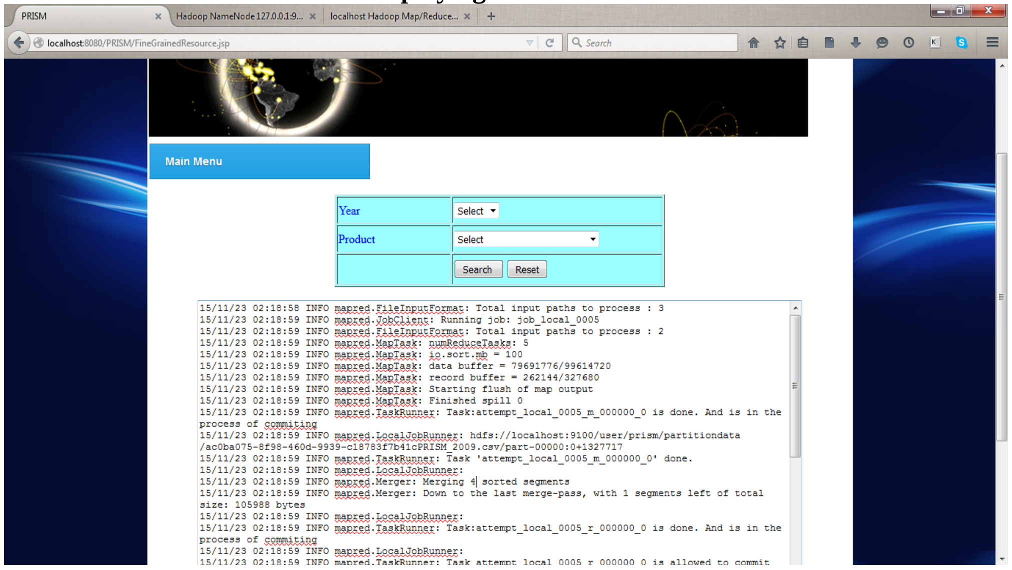


Year	2011
Product	Hydraulic breakers

Select
Carbide Tipped Masonry Drills
Engineers Files
Hydraulic breakers
Milling Cutters(Bore type)
Milling Cutters(Shank type)
Parts and Tools
Reamers
Screwing Taps
Sika Met
Solid Carbide
Tap Holders
Twist Drills

Fig 10.6 The year and the product is selected for processing

Displaying the data need

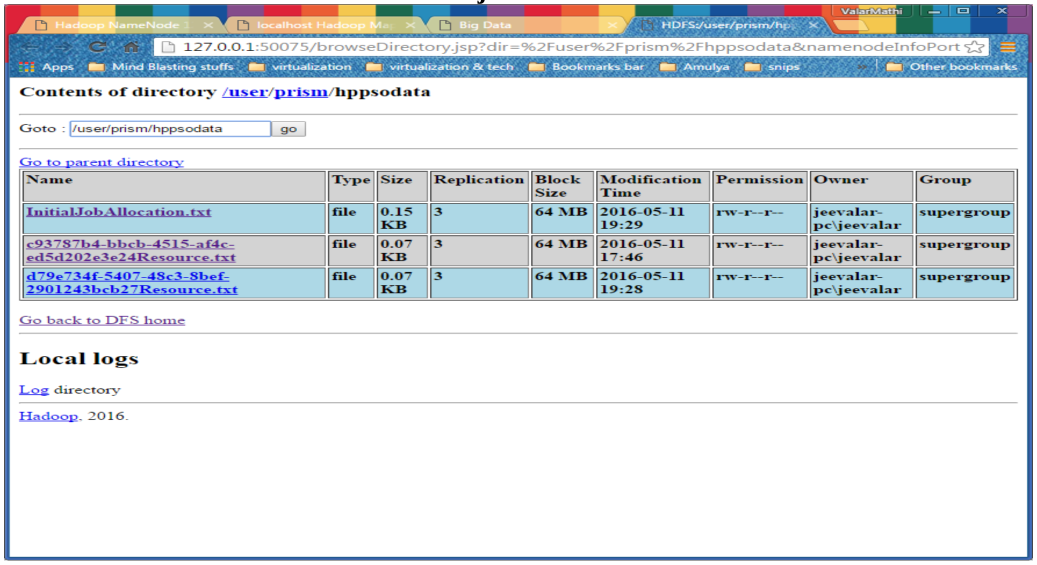

PRISM
Hadoop NameNode 127.0.0.1:9...
localhost Hadoop Map/Reduce...
localhost:8080/PRISM/FineGrainedResource.jsp
Search
Main Menu

Year	Select
Product	Select
	Search Reset

```
15/11/23 02:18:58 INFO mapred.FileInputFormat: Total input paths to process : 3
15/11/23 02:18:59 INFO mapred.JobClient: Running job: job_local_0005
15/11/23 02:18:59 INFO mapred.FileInputFormat: Total input paths to process : 2
15/11/23 02:18:59 INFO mapred.MapTask: numReduceTasks: 5
15/11/23 02:18:59 INFO mapred.MapTask: io.sort.mb = 100
15/11/23 02:18:59 INFO mapred.MapTask: data buffer = 79691776/99614720
15/11/23 02:18:59 INFO mapred.MapTask: record buffer = 262144/327680
15/11/23 02:18:59 INFO mapred.MapTask: Starting flush of map output
15/11/23 02:18:59 INFO mapred.MapTask: Finished spill 0
15/11/23 02:18:59 INFO mapred.TaskRunner: Task:attempt_local_0005_m_000000_0 is done. And is in the
process of commiting
15/11/23 02:18:59 INFO mapred.LocalJobRunner: hdfs://localhost:9100/user/prism/partitiondata
/ac0ba075-8f98-460d-9939-c18783f7b41cPRISM_2009.csv/part-00000:0+1327717
15/11/23 02:18:59 INFO mapred.TaskRunner: Task 'attempt_local_0005_m_000000_0' done.
15/11/23 02:18:59 INFO mapred.LocalJobRunner:
15/11/23 02:18:59 INFO mapred.Merger: Merging 4 sorted segments
15/11/23 02:18:59 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total
size: 105988 bytes
15/11/23 02:18:59 INFO mapred.LocalJobRunner:
15/11/23 02:18:59 INFO mapred.TaskRunner: Task:attempt_local_0005_r_000000_0 is done. And is in the
process of commiting
15/11/23 02:18:59 INFO mapred.LocalJobRunner:
15/11/23 02:18:59 INFO mapred.TaskRunner: Task attempt_local_0005_r_000000_0 is allowed to commit
```

Fig 10.7 Displaying the data for the selected details

Initial job allocation



Contents of directory /user/prism/hppsodata

Goto : /user/prism/hppsodata go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
InitialJobAllocation.txt	file	0.15 KB	3	64 MB	2016-05-11 19:29	rw-r--r--	jeevalar-pc\jeevalar	supergroup
c93787b4-bbcb-4515-af4c-ed5d202e3e24Resource.txt	file	0.07 KB	3	64 MB	2016-05-11 17:46	rw-r--r--	jeevalar-pc\jeevalar	supergroup
d79e734f-5407-48c3-8bef-2901243bcb27Resource.txt	file	0.07 KB	3	64 MB	2016-05-11 19:28	rw-r--r--	jeevalar-pc\jeevalar	supergroup

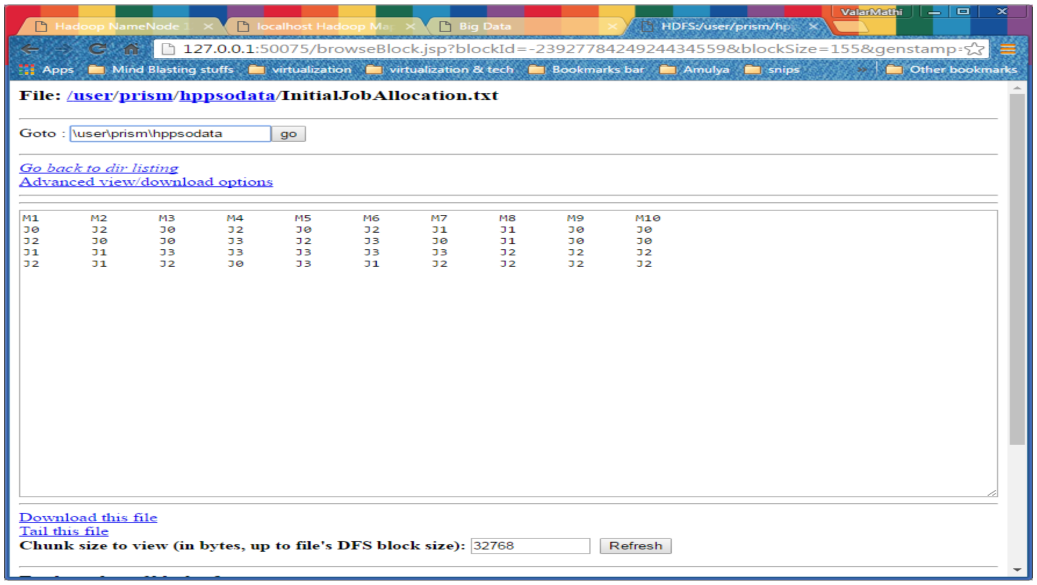
Go back to DFS home

Local logs

Log directory

Hadoop, 2016.

Fig 10.8 Initial job allocation for processing



File: /user/prism/hppsodata/InitialJobAllocation.txt

Goto : \user\prism\hppsodata go

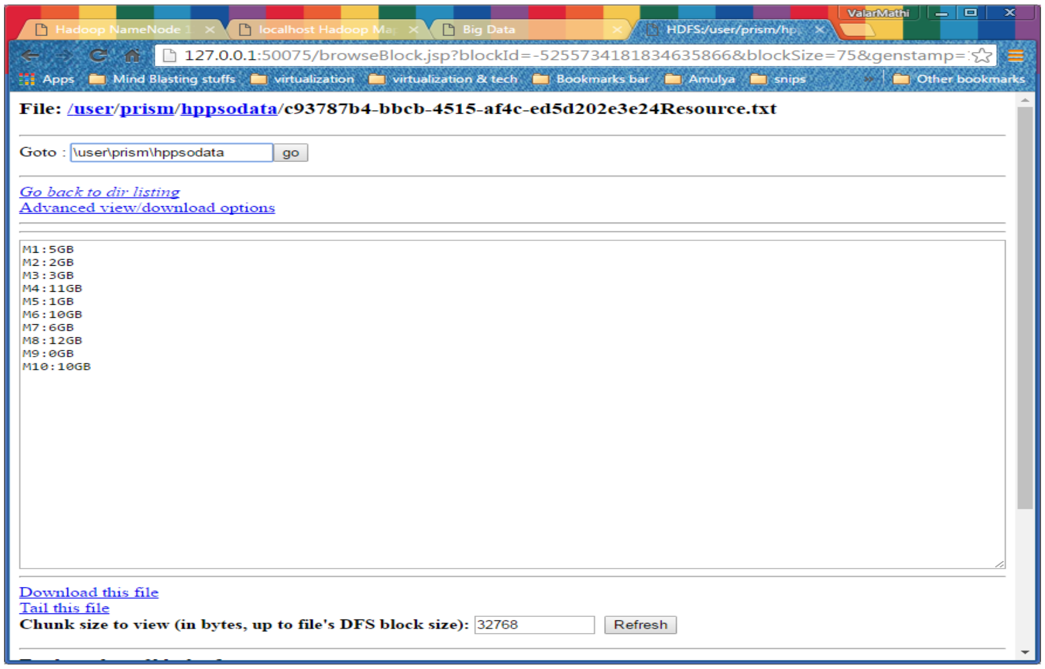
Go back to dir listing
Advanced view/download options

```
M1        M2        M3        M4        M5        M6        M7        M8        M9        M10
J0        J2        J0        J2        J0        J2        J1        J1        J0        J0
J2        J0        J0        J3        J2        J3        J0        J1        J0        J0
J1        J1        J3        J3        J3        J3        J3        J2        J2        J2
J2        J1        J2        J0        J3        J1        J2        J2        J2        J2
```

Download this file
Tail this file
Chunk size to view (in bytes, up to file's DFS block size): 32768 Refresh

Fig 10.9 Jobs allocated to the mapping nodes

Resource allocation



File: /user/prism/hppsodata/c93787b4-bbcb-4515-af4c-ed5d202e3e24Resource.txt

Goto : \user\prism\hppsodata go

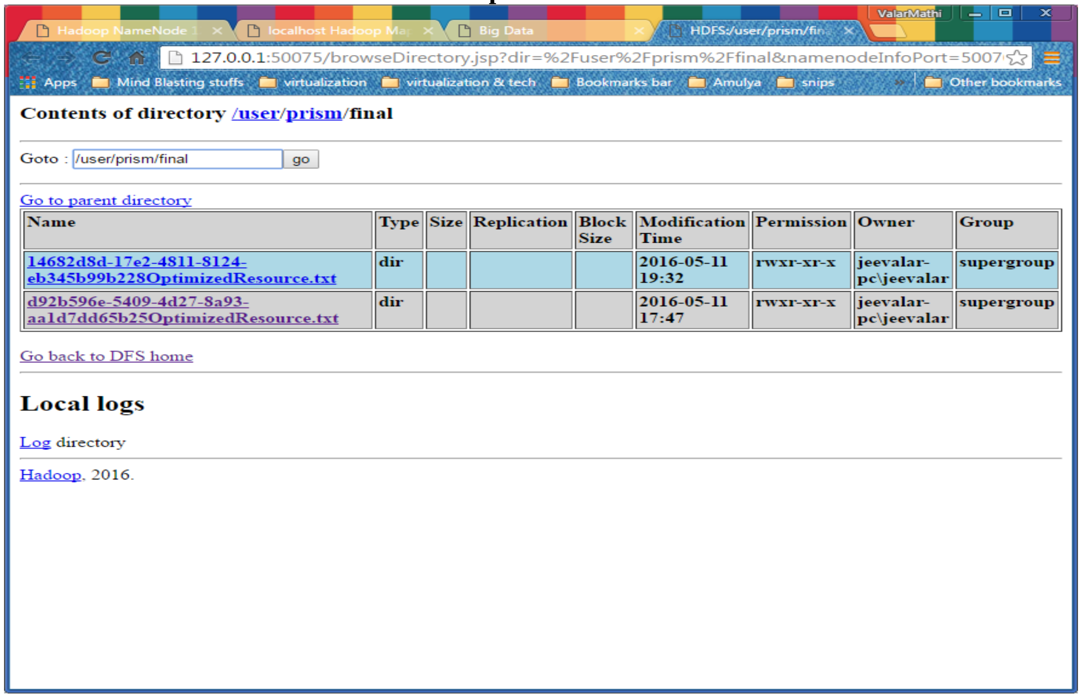
Go back to dir listing
Advanced view/download options

```
M1:5GB
M2:2GB
M3:3GB
M4:11GB
M5:1GB
M6:10GB
M7:6GB
M8:12GB
M9:0GB
M10:10GB
```

Download this file
Tail this file
Chunk size to view (in bytes, up to file's DFS block size): 32768 Refresh

Fig 10.10 Generation of the resources

Ph-pso function



127.0.0.1:50075/browseDirectory.jsp?dir=%2Fuser%2Fprism%2Ffinal&namenodeInfoPort=5007

Contents of directory /user/prism/final

Goto : /user/prism/final go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
14682d8d-17e2-4811-8124-eb345b99b228OptimizedResource.txt	dir				2016-05-11 19:32	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup
d92b596e-5409-4d27-8a93-aa1d7dd65b25OptimizedResource.txt	dir				2016-05-11 17:47	rwxr-xr-x	jeevalar-pc\jeevalar	supergroup

Go back to DFS home

Local logs

Log directory

Hadoop, 2016.

Fig 10.11 Resources with its job

Performance comparison chart

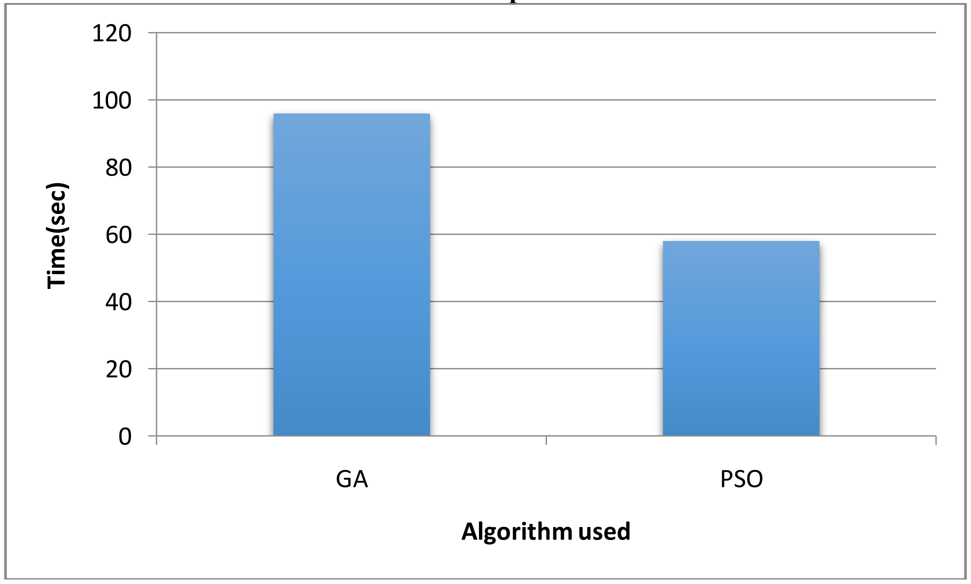


Fig 10.12: Performance chart

Conclusion:-

In this analysis, Energy reduction has been incorporated into the mapreduce implementation to improve the efficiency of the data center. The parallel scheduler for resource and task using particle swarm optimization is to manage the map and reduce tasks assignment. The Resource management is carried to manage a resource slots which reduces the consumption of energy when running the application achieves optimal schedules. Performance evaluation of the frameworks is compared with state of approaches which concludes that the framework outperforms in terms of efficiency and effectiveness.

References:-

1. Qi Zhang, Mohamed Faten Zhani, Yuke Yang, Raouf Boutaba, Fellow, and Bernard Wong, ”PRISM: Fine-Grained Resource-Aware Scheduling for mapreduce”, in IEEE TRANSACTIONS ON CLOUD COMPUTING, VOL. 3, NO. 2, APRIL/JUNE 2015, pp. 182-194.
2. R. Boutaba, L. Cheng, and Q. Zhang, “On cloud computational models and the heterogeneity challenge,” J. Internet Serv. Appl., vol. 3, no. 1, 2012, pp. 1–10.
3. M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, “Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling,” in Proc. Eur. Conf. Comput. Syst., 2010, pp. 265–278.
4. G.Sudha.Sadhasivam,N.nagaveninagaveni, Vasanth Ram Rajarathinamproposed “Design and Implementation of a Two Level Scheduler for HADOOP Data Grids” in International conference on Advances in Recent Technologies in communication and computing,source: IEEE Xplore, November 2009,pp.884-886.
5. Jianfeng Zhan, Lixin Zhang, Ninghui Sun, Lei Wang, Zhen Jia, and Chunjie Luo,”High Volume Computing: Identifying and Characterizing Throughput Oriented Workloads in Data Centers”,in arxiv: 1202.6134v2 [cs.DC] 14 jan 2103.
6. Chandrakant D. Patel, Amip J. Shah,”Cost Model for Planning, Development and Operation of a Data Center”, in Internet Systems and Storage Laboratory HP Laboratories Palo Alto, HPL-2005-107(R.1) ,June 9, 2005.
7. Jianfeng Zhan, Member, IEEE, Lei Wang, Xiaona Li, Weisong Shi, Chuliang Weng, Wenyao Zhang, and Xiutao Zang, “Cost-Aware Cooperative Resource Provisioning for Heterogeneous Workloads in Data Centers”, IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 11,NOVEMBER 2013, pp. 2155-2168.
8. Yanfeng Zhang, Qixin Gao, Lixin Gao, Fellow, and Cuirong Wang, “priter: A Distributed Framework for Prioritizing Iterative Computations”, in IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS,VOL. 24, NO. 9, SEPTEMBER 2013, pp. 1884-1893.
9. M. Livny *et al.* 1997. Mechanisms for High Throughput Computing, SPEEDUP Journal 1(1), 1997
10. R. E Bryant. 2008. Data intensive scalable computing. Retrieved January 5, 2010, from: http://www.cs.cmu.edu/bryant/presentations DISC-concept.ppt.
11. Campegiani, P. ; Lo Presti, F.; “A General Model for Virtual Machines Resources Allocation in Multi-tier Distributed Systems” published in Autonomic and Autonomous Systems, 2009. ICAS '09. Fifth International Conference on 20-25 April 2009. PP 162-167.
12. J. Kennedy and R.C. Eberhart, “Particle Swarm Optimization,” Proc. IEEE Int’l Conf. Neural Networks, pp. 1942-1948, 1995.
13. Y. Shi and R.C. Eberhart, “A Modified Particle Swarm Optimizer,” Proc. IEEE Int’l Conf. Evolutionary Computation, pp. 69-73, 2000.
14. J. Kennedy and R.C. Eberhart, “A Discrete Binary Version of the Particle Swarm Optimization,” Proc. IEEE Int’l Conf. Neural Networks, pp. 4104-4108, 1997.
15. J.H. Holland, Adaptation in Natural and Artificial Systems. MIT Press, 1975.
16. C.W. Hsu, C.C. Chang, and C.J. Lin, “A Practical Guide to Support Vector Classification,” http://www.csie.ntu.edu.tw/~cjlin/ papers/guide/guide.pdf Dept. Of Computer Science and Information Eng., Nat’l Taiwan Univ., 2003.