Triple integration optimization techniques in data Grid environment using OptorSim simulator

Abdo H. Guroob,
Dept. of Computer Science,
Mangalore University,
Mangalore, India.
abduohassan@yahoo.com

Manjaiah D.H.,
Dept. of Computer Science,
Mangalore University,
Mangalore, India.
manju@mangaloreuniversity.ac.in

***Abstract*- Data Grid Environments consist of geographically distributed resources to solve scientific problems and tasks of researchers, scientists and engineers, which are difficult to accomplish by traditional methods based on computer networks. Scheduling and replication are considered some of the most important techniques used in data grid environments, which are used to improve performance and availability to get the best throughput in the shortest possible time. Thus, some algorithms are used for these purposes. Effective scheduling working to reduce the time of implementation of tasks (makespan) of the available resources in the data grid, while replication is working to provide appropriate places or replace similar data to accelerate job execution time .**

On the other hand, there isanother technique, which is important as scheduling and replication, which can be used to reduce the time of implementation for a user request. This technique called Access Pattern, defines the order in which the files are requested for each jobto accelerate the completion of the task.

Most researchers are focusing on the scheduling, replication, or Access Pattern separately, which leads to variation in the results and gives them unsatisfactory results. The contribution of this paper is present the impact and effect of the triple integration of the three techniques to completing tasks in data grid environments by comparing the results of different algorithms available in the OptorSim simulator.

Keywords: Data Grid, Job Scheduling, Data Replication, Access Pattern, OptorSim simulator.

I. INTRODUCTION

Data Grid Environments have acquired great importance in scientific and research fields, which requirehuge computational resources; large storage capacity and speed in completing specific tasks in a minimum period time are possible for distributed data in different geographic locations.

Recently, data is increasing steadily, hence making performance and dealing with this huge amount of data very difficult. Therefore, certain techniques and algorithms have emerged, whichdeal with this data to reduce the time of implementation of tasks for different applications, which makes it possible to use them in data grid environments.

Of these techniques are Scheduling, Data Replication, and Access Pattern, which are used to send user requests to sites devoted to the implementation of these requests

Job Scheduling in data grid is done through three stages as shown in Figure (1) where the first stage begins with the discovery of available resources in the Grid, wherein the Grid environment many resources are distributed in different geographical locations.Furthermore there is the Grid Information Service (GIS), which holds all the details about these resources in the Grid that can be queried and hence know the available resources, from which the task can be executed. After this stage is to select the resources that will be dealt with in completing tasks. The last stage in the job scheduling is the process of implementation (job execution).

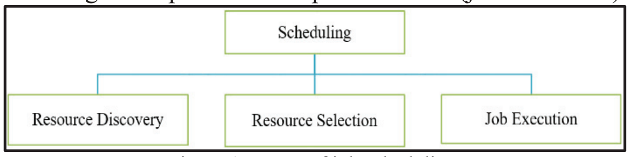


Figure 1: stages of job scheduling

In contrast, the data replication in data grid environments is the process, by which, when, and where can select the appropriate site places for getting the data and that deal with it to execute the user request in the shortest possible time.

Also the Grid Information Service provides necessary details about the storage capacity and similar copies of data in the Grid. These details send to Replica Manager. In contrast, the Replica Manager will choose the appropriate place for data replication according to the location of a requester, last using of this data, the storage capacity,and frequency of using data. Replication is important in reducing the burden on data storage centers, provide availability and fault tolerance.

However, with regards to Access Pattern, after determining computing resources that would be used to execute the task, and determining location places of the data required for implementation it Access Patternis brought the required data and sends it to the computing resources for implementation.

II. RELATED WORK

There are some works that address the challenges of scheduling and/or replication in Data Grid Environments as well as the combination between them. Ranganathan, K. and Foster, I. [7] proposed a data locality in job scheduling problem. The authors proposed their architecture based on 3 main components: External Scheduler (ES), Local Scheduler (LS) and Dataset Scheduler (DS). On this architecture, the authors developed and evaluated various data replication and scheduling strategy to study effect of the two systems. The result showed the importance of data locality in scheduling job. William H. Bell, et al [3]. Proposed architecture is the combination and improvement of data replication and job scheduling in a two-stage optimization mechanism using OptorSim and the result showed the importance of data replication and scheduling in Data Grid. Chakrabati and Tang, M et al. [8] develop the older works by integrating the scheduling and replication strategy to increase the scheduling performance.

Zomanya et al. in [13] have proposed a replica framework to minimize the data access cost. Their framework includes an application manger, an applicationanalyzer, and replica placement service. The replication manner in this paper based on data access patterns, correlation, rank, data locations, and user and application behavior. In contrast, scheduling is done based on the popular search heuristic, which is used replication information to send tasks to resources to minimizing the makespan.

In [16] Chang et al. developed the Hierarchical Cluster Scheduling algorithm (HCS) and Hierarchical Replication Strategy (HRS). Their algorithm depend on creating local copies of data to accelerate time of fetching data. The idea of the HRS algorithm is to maximize the data availability within the cluster by creating replicas. The HCS algorithm takes into account the computational capacity, data location and clusters' information as input to get the data availability within it. Chakrabarti et al. in [17] have proposed an Integrated Replication and Scheduling strategy rely on two stages, which are calculate the popularity of the files required by a set of tasks, and replication is done to facilitate ease of data access for the next set of tasks. The disadvantage in the algorithm is that the next set of jobs may or may not access the same data sets.

Dang et al. in [6] have proposed heuristics for scheduling and replication separately But they need to integrate the scheduling system with the replication to perform as a whole system together; the interaction between both is not detailed in the paper.

Lin et al. have proposed a system (Network Coordinate) called (Rigel) [4] rely on two components, which are NC calculation system that determines the user's location and NC storage infrastructure that uses to store the replica, based on Distributed Hash Table (DHT). These components for selecting the closest replica to the user, where this technique is working to reduce the latency reading. However, the repetition of NC calculation for user and replica in each request operation affects the performance of the system because of more requests by users in the environment.

III. EXPERIMENTAL ARCHITECTURE

OptorSim's architecture, which is based on the European Data Grid CMS testbed architecture, was developed by the European Data Grid project. The main components of OptorSim are Resource Broker (RB), Computing Elements (CEs), Storage Elements (SEs), Replica Manager (RM), and Replica Catalog (RC) as shown in figure 1. RB is responsible for submit jobs to the Computing Elements, CEs are responsible for running jobs of data files in the Grid site, SEs are responsible for storage of data.The Replica Manager provides file manipulation methods and interfaces between the Computing Elements and optimizers and the low-level Grid infrastructure.Replica Catalog is a catalogue of files available on the Grid. Logical File Names (strings representing a unique file) are mapped to the physical Data Files (individual instances or replicas of each file) spread around the Grid. The Replica Catalogue is not used directly but is accessed via the Replica Manager which provides methods such as copying a file around the Grid or removing a file from the Grid.

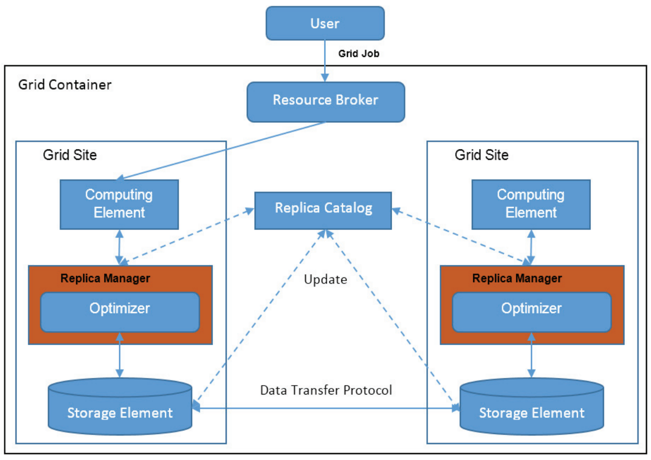


Figure 2: OptorSim architecture based on the EDG data management components.

OptorSim simulator is developed to give the best results by using different optimization techniques, whichin the architecture of the simulator that can be modified in getting the shortest possible time during the execution of jobs. Theseoptimizations techniques are:Scheduling, Replication, and Access Pattern.

A. Scheduling Mechanism

The Resource Broker runs a thread which submits jobs to the Computing Elements. Users submit jobs to the RB calling submit Job (GridJob), then the RB decides to which CE the job will be submitted according to the scheduling algorithm defined in the parameters file. This can be one of the following options:

• **RandomCEResourceBroker:**site is accessed randomly.
• **QueuelengthResourceBroker:**access to site, which has job queue, is the shortest.
• **AccessCostResourceBroker:**site has shortest time to access all files required by the job.
• **CombinedCostResourceBroker:**access cost to site in which all job in queue is shortest.

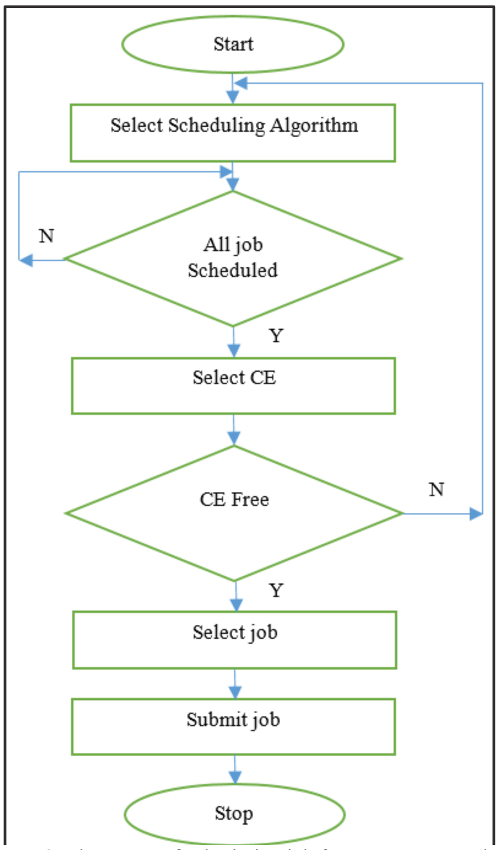


Figure 2: The stages of submission job from Resource Broker

B. Access Pattern Mechanism

The access pattern defines the order in which the files are requested for each job. **Access Pattern Generator** must be implemented and used to obtain the type of Access Pattern algorithms specified in the user parameters.
In OptorSim, there are several types of access patterns such as:
• **SequentialAccessGenerator** - The access to files in an orderly manner.
This class is the most basic Access Pattern Generator, it just hands the files back in the order they are in the configuration file, starting with the first one.
• **RandomAccessGenerator** - The access to files using a flat random distribution.
This access pattern generator selects files at random from the list in Grid Job.
• **RandomWalkUnitaryAccessGenerator** - The access to files using a unitary random walk.
This class selects a file based on a unitary random walk. This means there is an equal chance of the next file being the previous one or the next one in the list of files in the Grid Job. The starting point for the random walk is the same every time the same Grid Job is run.
• **RandomWalkGaussianAccessGenerator** - The access to files using a Gaussian random walk.
This class selects a file based on a Gaussian random walk: the sigma of which is set as half the number of files in the set. The starting point for the random walk is the same every time the same Grid Job is run.
• **RandomZipfAccessGenerator** - The access to files using a Zipf distribution.

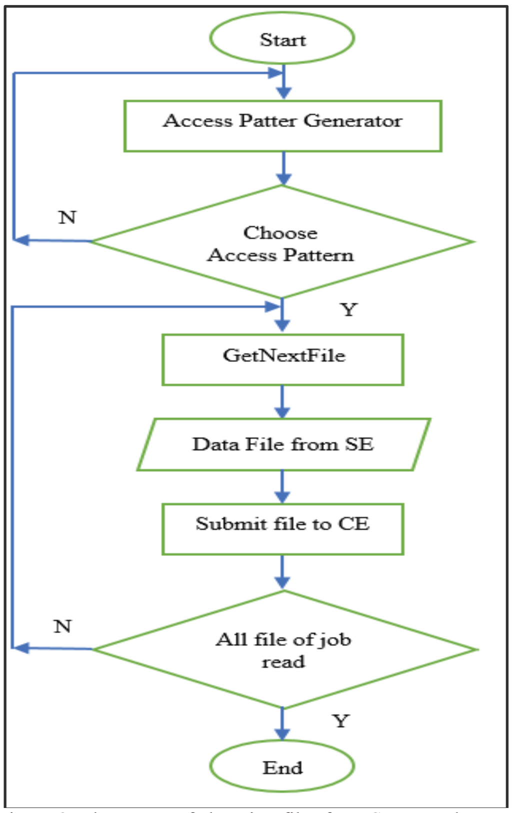


Figure 3: The stages of choosing files from Storage Element

C. Replication Mechanism

During the run time of a job, the controller of jobs (Job Handler)in OptorSim selects the replica, which can be triggered by the optimization algorithm.
Job Handler controls the flow of jobs into a Computing Element using a queueing system. It provides the methods for communication between the Resource Broker and Computing Element threads. Optimizer algorithm provides an implementation of getBestFile(), which will attempt to perform replication of files to the closest Storage Element of the Computing Element calling it (if the file is not already available there). If there is space on the close SE replication will always succeed. If there is no space the choose FileToDelete() method of the subclass is called to determine which files should be removed in order to create space. The implementation of this method should be such that it returns a null value if replication is not to take place, for example if all the files on the local SE are master files or the optimization algorithm decides it is not worthwhile to replicate.
Data Replication algorithms specified in the user parameters.
• No replication
• Least Recently Used (LRU) - always data are replicated, delete the file which has the least recent useage.
• Least Frequently Used (LFU) - always data are replicated, delete the file which has least frequently used

• Economic model (Binomial) - replicate data if economically useful, using binomial prediction function for file values.
• Economic model (Zipf) - replicate data if economically useful, using Zipf-based prediction function.

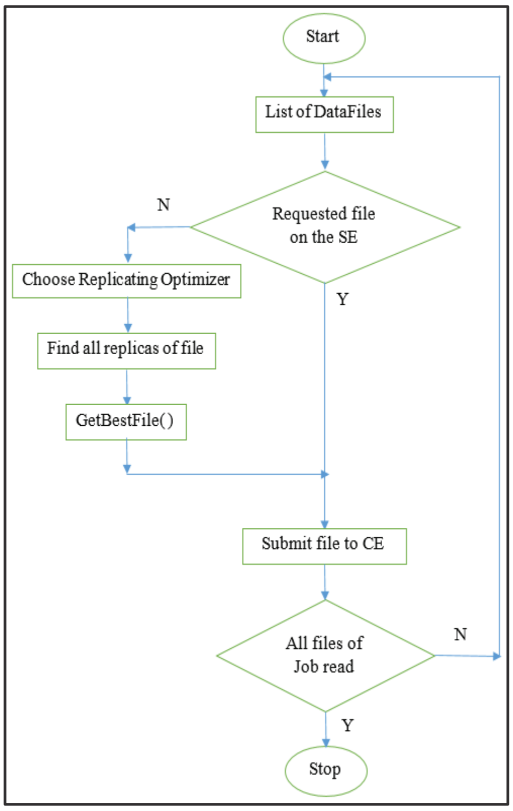


Figure 4: The stages of replica optimization

IV. SIMULATION EXPERIMENTS AND RESULTS

OptorSim contains parameters file usedin modifying and changing all available settings for executing the jobs, which Commensurate with the nature of the work to be performed.

The most important of these parameters are:

1) Grid configuration file: Contains the topology of Grid environment, locations of sites, storage capacities of each Storage Element (SE), and distributed Computing Element (CE).

The topology used in this paper is based on CMS testbed as shown in Figure (5).

Site Name	jobs it will run
All Sites	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob

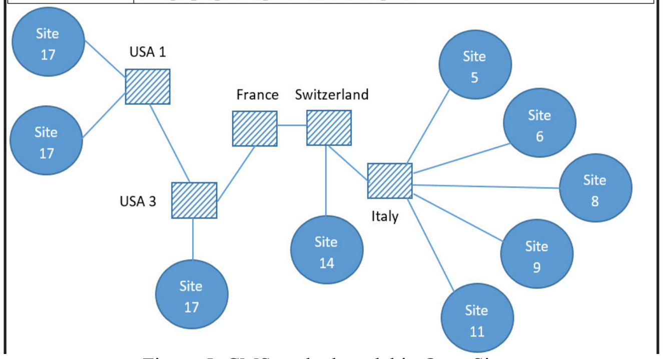


Figure 5: CMS testbed model in OptorSim

2) Job configuration file:This file contains the job table of a file name, a list of files needed, a size of each file, and CE site ID of jobs it will run as shown in table 1 andtable2 respectively.

3) Bandwidth configuration file:It contains mean available bandwidths for CMS testbed links for a file name, a different time between user and source site, and space between each site.

Table 1: Job Table

No.	Job Name	Files needed
1	jpsijob	jpsi0 jpsi1 jpsi2 jpsi3 jpsi4 jpsi5 jpsi6 jpsi7 jpsi8 jpsi9 jpsi10 jpsi11
2	highptlepjob	highptlep0 highptlep1
3	incelecjob	incelec0 incelec1 incelec2 incelec3 incelec4
4	incmuonjob	incmuon0 incmuon1 incmuon2 incmuon3 incmuon4 incmuon5 incmuon6 incmuon7 incmuon8 incmuon9 incmuon10 incmuon11 incmuon12 incmuon13
5	highptphotjob	highptphot0 highptphot1 highptphot2 highptphot3 highptphot4 highptphot5 highptphot6 highptphot7 highptphot8 highptphot9 highptphot10 highptphot11 highptphot12 highptphot13 highptphot14 highptphot15 highptphot16 highptphot17 highptphot18 highptphot19 highptphot20 highptphot21 highptphot22 highptphot23 highptphot24 highptphot25 highptphot26 highptphot27 highptphot28 highptphot29 highptphot30 highptphot31 highptphot32 highptphot33 highptphot34 highptphot35 highptphot36 highptphot37 highptphot38 highptphot39 highptphot40 highptphot41 highptphot42 highptphot43 highptphot44 highptphot45 highptphot46 highptphot47 highptphot48 highptphot49 highptphot50 highptphot51 highptphot52 highptphot53 highptphot54 highptphot55 highptphot56 highptphot57
6	Zbbbarjob	zbbbar0 zbbbar1 zbbbar2 zbbbar3 zbbbar4 zbbbar5

Table 2: Computing Element Schedule Table

No.	CE site id	Site Name	jobs it will run
1	5	Padova	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
2	6	Bari	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
3	8	Catania	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
4	9	Roma	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
5	11	Bologna	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
6	14	CERN	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
7	16	Wisconsin	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
8	17	UFL	jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob
9	19	FNAL	19 jpsijob highptlepjob incelecjob incmuonjob highptphotjob zbbbarjob

OptorSim simulator to evaluate integrated job scheduling, data replication, and Access Pattern algorithms.

This paper uses job execution time as the evaluation metrics. Job Execution Time is defined as the total time to execute all jobs, divided by the number of jobs completed.

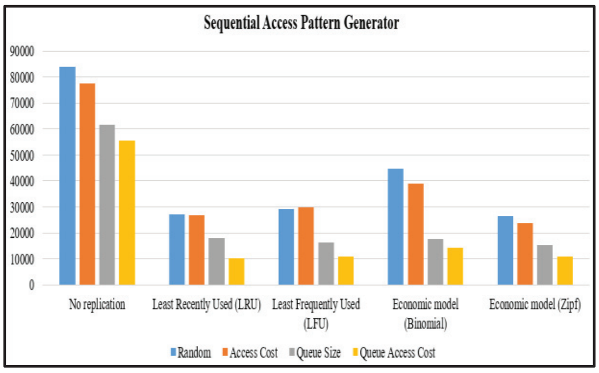
$$JobExecutionTime = \frac{TotalTime\ of\ All\ Job}{Number\ of\ Jobs}$$

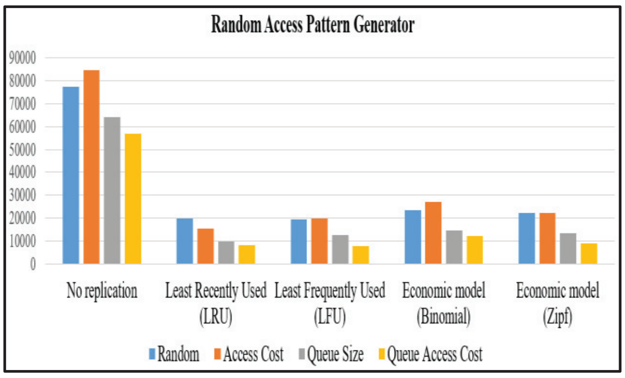
OptorSim can be run from the command line or from a GUI, and the output of simulator in statistics form are:
• Number of Jobs Remaining
• Mean Job Time of all Jobs on Grid
• Total Number of Replications
• Total Number of Local File Accesses
• Total Number of Remote File Accesses
• Percentage of CEs in Use
• Percentage of Storage Filled/Available
• Effective Network Usage.

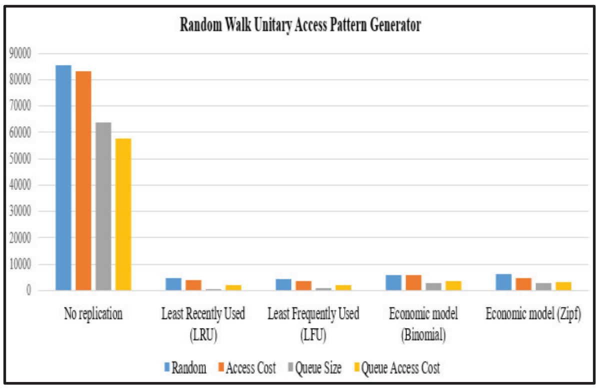
The simulation results obviously show the impact of the different algorithms, which used to get the results of the simulations. Table 3 illustrates the different parameters that set up in OptorSim simulator to evaluate the triple integrated job scheduling, data replication, and Access Pattern algorithms.

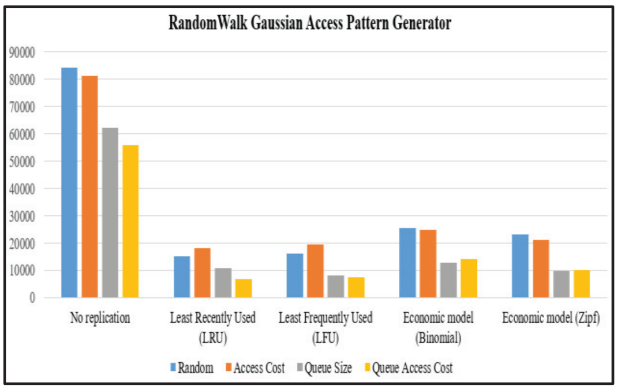
In figure 6 below shows the results of simulating more than 300 times, each algorithm 3 times. Which reveals the various results of simulation from Scheduling Algorithms, Replication Algorithms, and Access Pattern Algorithm. According to the results obtained in figure 6, where the algorithms that have the longest time in implementing jobs and the algorithms that have the shortest time in the implementation of jobs can be seen.

Where Queue size for scheduling has the shortest job execution time. Least Recently Used (LRU) for replication has the shortest job execution time. In contrast, Random Walk Unitary Access Pattern has the best result of job execution time because an equal chance of the next file being the previous one or the next one in the list of files in the Grid Job. The starting point for the random walk is the same every time the same Grid Job is run.









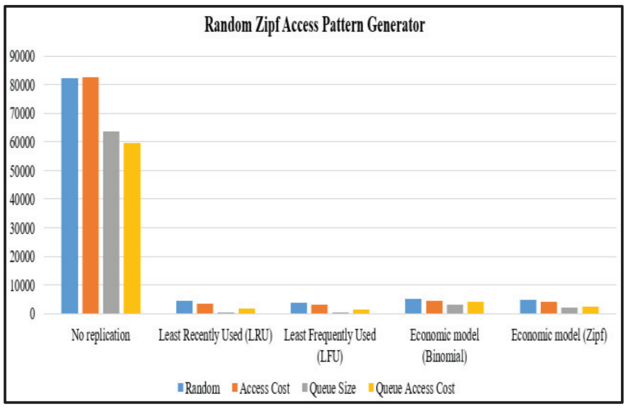


Figure 6: Mean job time for optimization algorithms with different schedulers, replication, and access pattern of 1000 jobs, CMS testbed

Table 3. Simulation Parameters

No.	Parameters	Features
1	Number of jobs	1000
2	Access History Length	1000000
3	Zipf distributed shape	0.85
4	Initial File Distribution	14
5	Job Delay (ms)	2500
6	Max. Queue Size	200
7	Auction Initial Timeout (ms)	500
8	Start Time	0.0

V. CONCLUSION

The processes of accomplishing tasks in Data Grid Environment as fast as possible are considered one of the most important dilemmas, which are faced researchers and scientists in the fields that rely on the resources of distributed environments.

So techniques and algorithms emerged to improve the performance in order to get satisfactory results as far as possible. From these techniques Scheduling, Replication, and Access Pattern, in which each one of them is working to reduce the time of implementation of tasks. In this paper is presented the impact and effect of those technologies when integrated with each other.Future work develops some algorithms and techniques that working in an integrated manner to accelerate the job execution time in real data grid environments.

ACKNOWLEDGEMENT

This research was supported by Mangalore University, Department of Computer Science, India and Hodeida University, Yemen. I would like also to thank my supervisor Professor Manjaiah D.H. for his assistance in moderating this paper and in that line improved the manuscript significantly.

REFERANCES

[1] William H. Bell, et al., “OptorSim - A Grid Simulator for Studying Dynamic Data Replication Strategies,”International Journal of High Performance Computing Applications,17(4), 2003.

[2] B. T. Huffman, R. McNulty, T. Shears, R. St. Denis and D. Waters “The CDF/D0 UK GridPP Project”, CDF Internal Note 5858, 2002.

[3] William H. Bell, et al, “Evaluation of an Economy-Based File Replication Strategy for a Data Grid,”International Workshop on Agent based Cluster and Grid Computing at CCGrid 2003, Tokyo, Japan, May 2003. IEEE Computer Society Press.

[4] Y. Lin, Y. Chen, G. Wang, B. Deng, “Rigel: A scalable and lightweight replica selection service for replicated distributed file system,” 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, CCGC, pp. 581–582, 2010.

[5] David G. Cameron, et al, “Evaluating Scheduling and Replica Optimisation Strategies in OptorSim,” 4th International Workshop on Grid Computing (Grid2003), Phoenix, Arizona, November 17, 2003. IEEE Computer Society Press.

[6] N.N Dang and S.B. Lim. (2007). Combination of replication and scheduling in data grids. International Journal of Computer Science and Network Security, 7(3):304-308.

[7] K. Ranganathan and I. Foster, “Simulation Studies of Computation and Data Scheduling Algorithms for Data Grids,” Journal of Grid Computing, Volume 1, Number 1, 2003, pp. 53-62(10).

[8] Tang, M., Lee, B., Tang, X., and Yeo, “The impact of data replication on job scheduling performance in the Data Grid,” Future Generation Computing System 22, 3 (Feb. 2006), 254-268. DOI= http://dx.doi.org/10.1016/j.future.2005.08.004.

[9] XU PengZhi, WU YongWei∗, HUANG XiaoMeng, “Optimizing write operation on replica in data grid,” SP Science China Press (SCIS), Volume 54, Issue 1, pp 1-11, January 2011.

[10] Mansouri, N., Dastghaibyfard, G.H., Mansouri, E., “Combination of data replication and scheduling algorithm for improving data availability in Data Grids,” Journal of Network and Computer Applications, vol. 36, Pages 711–722, March 2013.

[11] Abdo H. Guroob and Manjaiah D.H., “Efficient Replica Consistency Model (ERCM) for update propagation in Data Grid Environment,” IEEE International Conference on Information Communication & Embedded Systems “ICICES 2016”, 25th February 2016.

[12] J. M. Pereza, F. Garca-Carballeira, J. Carreteroa, A. Calderona, and J.Fernndeza. “Branch replication scheme: A new model for data replication in large scale data grids,” Future Generation Computer Systems, 26(1):12–20, 2010.

[13] A. Elghirani, R. Subrata, Albert Y. Zomaya. (2007). Intelligent Scheduling and Replication in Datagrids: a Synergistic Approach, CCGRID, pp.179-182, Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid’07).

[14] D. Nukarapu, B. Tang, L. Wang, and S. Lu., “Data replication in data intensive scientific applications with performance guarantee,” IEEE

Transactions on Parallel and Distributed Systems, 22(8):1299–1306, Aug. 2011.
http://doi.acm.org/10.1145/2522548.2522605.

[15] Shvachko, K., Kuang, H., Radia, S., Chansler, R, "The hadoop distributed file system," IEEE 26th Symposium on Mass Storage Systems and Technologies(MSST), pp. 1–10, May 2010.

[16] R.S. Chang, J.S. Chang, S.Y. Lin. (2007). Job scheduling and data replication on data grids, Future Generation Computer Systems 23 (7) 846-860.

[17] Chakrabarti, R. A. Dheepak, and S. Sengupta. (2004). Integration of scheduling and replication in data grids. In International Conference on High Performance Computing (HiPC).

[18] Abdo H. Guroob, Manjaiah D.H, "Data Replication in Data Grid Environment," CSI Communications: Knowledge Digest for IT Community, Volume No.39, and Issue No.8 November 2015.

[19] Abdo H. Guroob and Manjaiah D.H, "Optimization Techniques for Integrated Job Scheduling and Data Replication in Grid Environments Using OptorSim Simulator", International Journal of Latest Trends in Engineering and Technology, Special Issue SACAIM 2016, pp. 37-42, e-ISSN:2278-621X, November 2016.