IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-6, Issue-3 E-ISSN: 2347-2693

Optimization of Dynamic Resource Scheduling Algorithm in Grid Computing Environment

S.K. Patel[1*], A.K. Sharma[2]

[1*]Dept. of Information Technology, Govt. N.P.G. College of Science, Raipur, India|
[2] Dept. of Information Technology, A.P.S.G.M.N.S Govt. P. G. College, Kawardha, India

[*]*Corresponding Author: surendrapatelit2004@gmail.com, Tel.: +91-2263131*

Available online at: www.ijcseonline.org

Received: 23/Feb//2018, Revised: 28/Feb2018, Accepted: 21/Mar/2018, Published: 30/Mar/2018

***Abstract*—** Resource supervision and task scheduling are very important and complex problems in grid computing environment. Handle of such resources we need job scheduling and load balancing techniques which are responsible for efficient use of the grid resources, reduce job waiting time, access latency in a wise manner. After comprehensive investigation of an existing grid which involves a large number of CPU cluster, we observe that grid scheduling decisions can be significantly improved computation time if the characteristics of current usage patterns are understood. In this paper a new job scheduling algorithm, called Improved Dynamic Load Balancing (IDLB) is proposed. In the proposed algorithm the current scheduling is denoted as S* so the runtime delay is reduced by using Actual Latest Finish Time (ALFT). Finally, in this research the algorithm was simulated with the aid of OptorSim simulator and it was proved that our proposed algorithm provid an effective solution for resource management grid scheduling.

***Keywords*—** Grid Computing, Computational Grid, DLB, IDLB,Load Balance , Resource Management, Job Scheduling.

I. INTRODUCTION

Modern computer industry is operating with very large amounts of data which utilizes more processing power and high storage volumes of data. Grid computing is proposed as effective resource management to the organization, since it involves using resources from different spaces, from different ownerships and with different individual performance [1][2]. Resource management and scheduling resources in Grid computing is a complex task due to the heterogeneous and dynamic nature of the resources. Managing of resources crisis brings Grid Technology that needs algorithms and mechanisms to be redesigned for resource handling. The inspection of algorithms is the resolve the number of resources (such as time and storage) necessary to execute them [3].

Resource sharing is the essence character of grid. There are a wide range of heterogeneous and geographically distributed resources in grid. For example, there are single processor, multiprocessors, shared memory machine, distributed memory machines, workstations, etc., and they are with different capabilities and configurations, and are managed in multiple administrative domains with different policies. In practical, the resource management and scheduling in such a complex environment are confronted with a great challenge [4][5]. Job scheduling is a choice process by which application components are assigned to obtainable resources to optimize various performance metrics. The main goal of scheduling is to maximize the resource deployment and minimize processing time of the all jobs. Various research works has been done on job scheduling problem in grid, and different algorithms have their advantages and disadvantages but still further analysis and research needs to be done to improve the performance of scheduling algorithm in computational grid [6].

The research consists of five sections that are organized as follows:

Section 1 describes the introductory part of research. Section 2 presents the literature survey and the research work carried out in the field of related. Section 3 presents the research methodology used in this research and Optorsim simulator is used to support resource management and other capabilities, such as job allocation, scheduling and balancing of jobs etc. Section 4 presents result analysis and section 5 conclude the research with a summary of the main findings, discussion of future research directions, final remarks and outlines possibilities for work in future.

II. RELATED WORK

Thamarai et al. [7] proposed an algorithm that combines the advantages of three algorithms first shortest processing time, second, longest processing time, and third, earliest deadline

first. The author realises that virtualization technique is required to address the problem of resource unavailability and software mismatch. Somasundaram et al. [8] fond out the ORC scheduling in Grid environment includes the best fit followed by round robin scheduling which distributes the job among the available processors. Quan et al. [9] discussed that many developed algorithms either considered the resource characteristics or application characteristics, but the adaptive fine grained job scheduling algorithm described in considered both characteristics. AFJS monitors the resources and starts with obtaining information about the resources. In this algorithm a constraint is specified, which says that processing time of coarse grained job should not exceed the expected time. Authors suggested that there is a need to reduce the communication time, processing time and to enhance resource utilization in case of scheduling the light-weight or small jobs[9][10]. There are many applications in which consist a large number of lightweight or less processing requirement jobs. srivastava et al. [11] stated scheduling with light weight gives low performance in terms of processing time and communication time. So to achieve high performance, less processing requirement jobs are grouped before allocation of resource. This grouping algorithm integrated greedy algorithm and FCFS algorithm to recover the processing take on of fine-grained jobs. Nithiapidary et al. [12] presented a Dynamic Job Grouping-Based scheduling that maximizes the utilization of resource, and reduces the overhead time. This strategy dynamically assembles the individual Fine-grained jobs of an application into a group, and sends these coarse-grained jobs to the Grid resources.

Ang et al. [13] proposed the bandwidth aware algorithm which improved the performance by reducing the delaying factors in network environment and maximizing the utilization of grid resources. Job groups are sent based on Largest Job First (LJF) manner to the corresponding resource. Longer execution time jobs are executed concurrently with shorter execution time jobs.

A Resource Aware Scheduling Algorithm which leverages two existing task scheduling algorithms, Min–min and Max–in, is described in [16][17]. Both algorithms use an estimation of tasks completion time and resource execution time. The presented algorithm alternates the above algorithms depending on number of jobs and resources. An important feature for scheduling algorithms is to have a dynamic behavior according to real environment evolution. Such an algorithm is described in [18].

III. METHODOLOGY

3.1 Simulation Tool and Environment

In this research, the OptorSim [14] simulator is used to simulate the proposed resource scheduling algorithm for resource management. There are numerous tools available for simulating resource scheduling algorithms for managing the resources in Grid computing environments. In this research, we used OptorSim. We have evaluated the algorithm by simulating different number of jobs ranges between 100 and 2000 in our experiments.

3.1.1 OptorSim

OptorSim is a simulation tool based on a compositional modeling paradigm, which allows the user to simulate the performance behavior of a wide range of distributed systems for a given application, under different computing and network load conditions. The architecture of OptorSim which has been shown in Figure1. It is based on the European Data Grid data management components. Computing and storage resources are represented by Computing Elements (CEs) and Storage Elements (SEs) respectively, which are organized in Grid Sites [15].

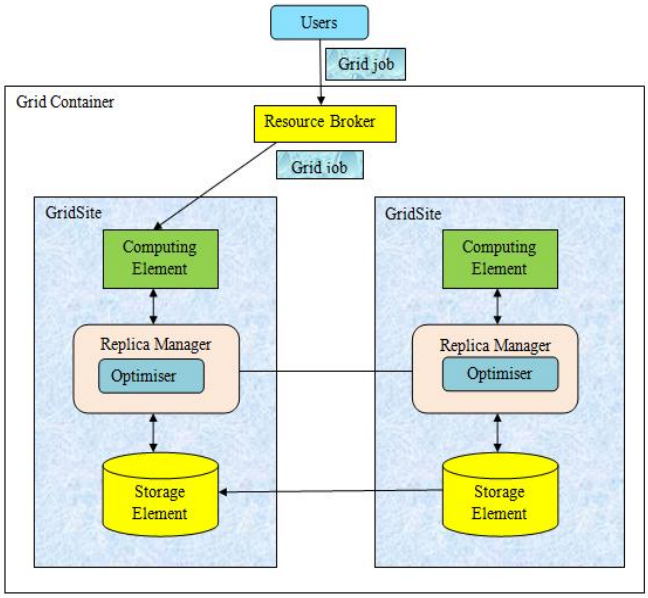

Figure 1: OptorSim Architecture [15]

3.1.2 Grid simulation Screen Layout

We have used two configuration files to control various inputs to OptorSim. One is the grid configuration file which specifies the Grid topology and the contents of each site (the number of SEs and CEs) by user. The other one is job configuration file which contains information for the simulated jobs.

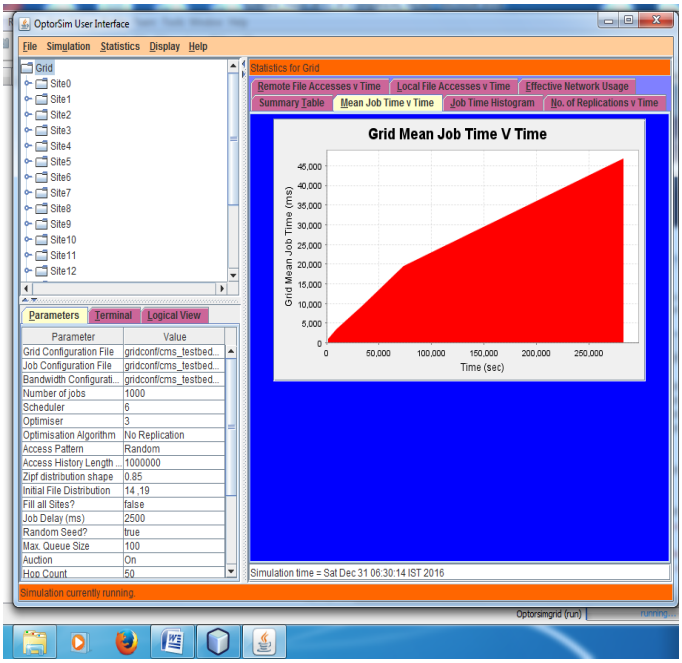

Figure 2: Simulation Scenario of OptorSim Simulator

3.2 Performance metrics

We have used following three performance parameters to simulate the results.

- Mean Job Time
- Number of Replication
- Effective Network usage

These are the critical parameters which affect the results in optimal manner.

IV. RESULTS AND DISCUSSION

4.1 Performance Evaluation and analysis of the algorithm

A number of simulation based evaluations were conducted in the research work, in order to evaluate the result of the proposed algorithm.

4.2 Proposed Improved Dynamic load Balancing (IDLB) Algorithm.

In this research, we can perform various experiments with different scenario mechanism for dynamic job scheduling.

4.2.1 Scenario-1 for DLB and IDLB

The simulation parameter values used for DLB and IDLB scheduling algorithm are shown in Table 4.1.

Table 4.1: Grid simulation parameters for DLB and IDLB of scenario-1

Number of jobs	100,200,300,400,500,1000,1500, 2000
The categories of users are	Simple
Optimizers	LruOptimiser
access pattern generators	SequentialAccessGenerator
maximum queue size	100
Hope count	50

Table 4.2: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-1

No. of jobs	Mean Job Time	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	9344	5412
200	14866	10124
300	18076	15145
400	33124	26365
500	31193	23837
1000	46917	41324
1500	44803	42889
2000	55877	51971

In this experiment 100 to 2000 jobs are submitted as the user input using scenario-1 performance parameter table. The simulation result show that the Mean Job Time (MJT) obtained by DLB for 100 jobs is 9334 milliseconds whereas the Total Mean Job Time (MJT) obtained by the proposed IDLB algorithm for the same number of jobs is 5412 milliseconds. And for Mean Job Time (MJT) obtained by DLB for 200 job is 14866 milliseconds whereas the Total Mean Job Time (MJT) obtained by the proposed IDLB algorithm for the same number of job is 10124 milliseconds

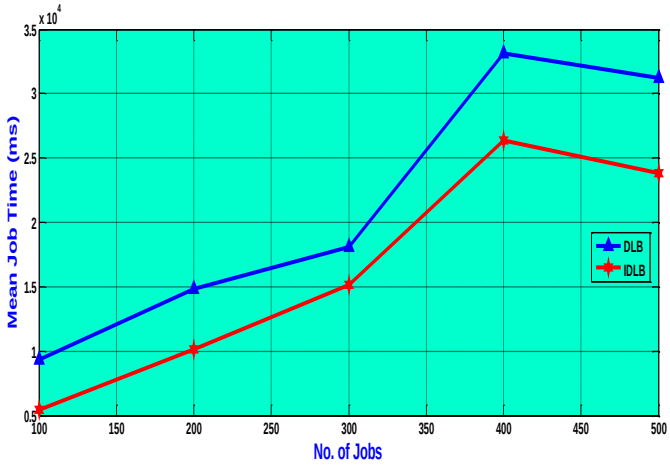

Figure 4.1: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-1

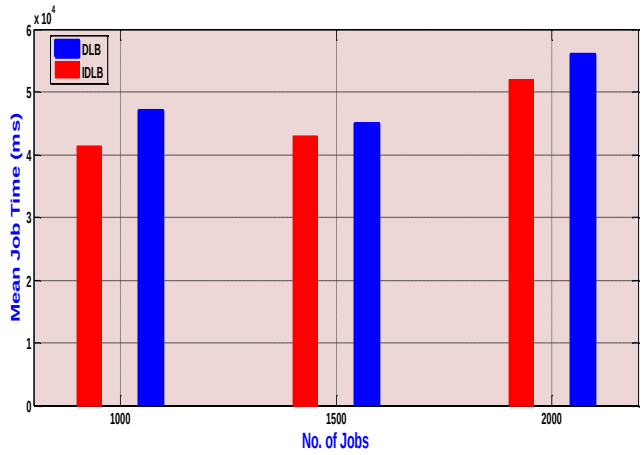

Figure 4.2: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-1

Table 4.3: Number of Replications Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-1

No. of jobs	Number of Replications	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	138	123
200	416	274
300	366	263
400	540	488
500	663	442
1000	1108	998
1500	1936	1849
2000	3037	2839

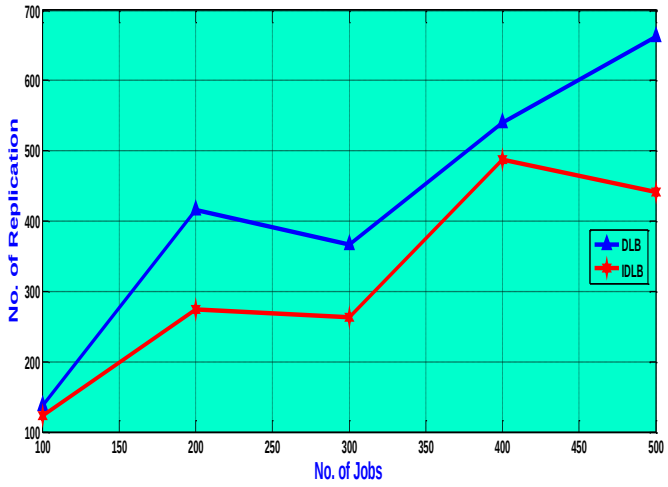

Figure 4.3: No. of replications Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-1

Similarly for jobs, 300, 400, … 2000, are shown in table 4.2. On the basis of these experimental results we can easily conclude that the proposed IDLB scheduling algorithm is efficient than the existing DLB. It gives 14.6% performance improvement over the existing algorithm.

Table 4.3 shows Number of Replications of all jobs on Grid for dynamic load balancing and improve dynamic load balancing in resource management. In this experiment 100 to 2000 jobs are submitted as the user input using scenario-1 parameter table. The simulation results show that the Number of Replications obtained by DLB for 100 jobs is 138 whereas the Number of Replications obtained by the proposed IDLB algorithm for the same number of job is 123. Similarly for jobs, 200, 300, … 2000, are shown in table 4.3. Hence, the result of the experiment clearly reveals that our proposed technique outperforms the existing techniques and provides an optimum improvement of 11.31%.

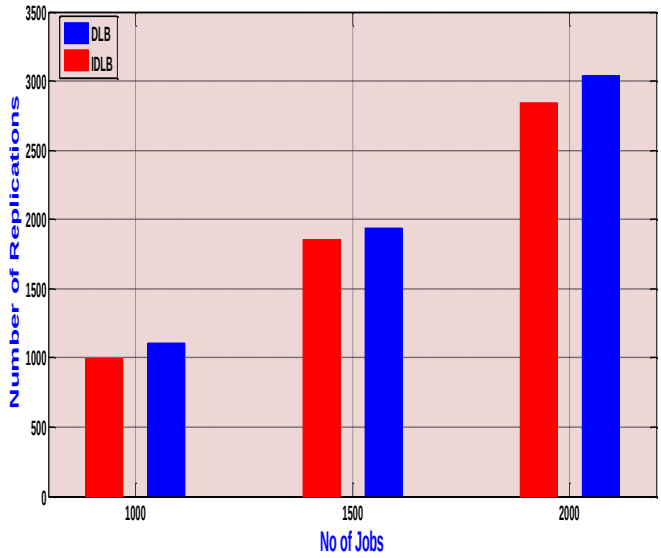

Figure 4.4: Number of Replications Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-1

In table 4.4 and Figure 4.5 to figure 4.6 shown that Effective Network Usage of all jobs on Grid for dynamic load balancing and improve dynamic load balancing in resource management. Thus the method of improve dynamic load balancing performs better result than the existing dynamic load balancing.

Table 4.4: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-1

No. of jobs	Effective Network Usage	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	0.78310346	0.7692308
200	0.7837884	0.74906967
300	0.75278625	0.70782836
400	0.77980885	0.70950417
500	0.70115485	0.60956663
1000	0.6965033	0.6102445
1500	0.67037786	0.571455
2000	0.6652069	0.5181147

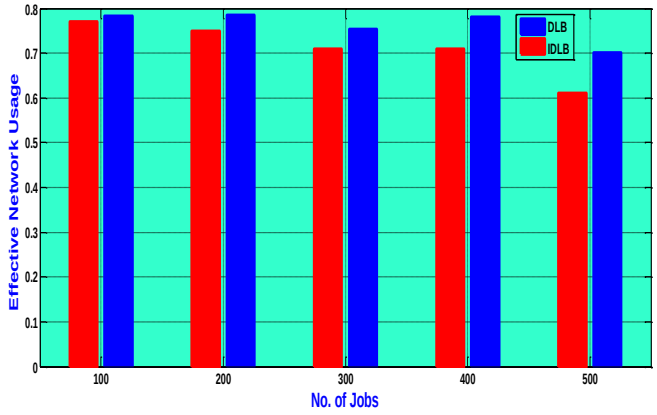

Figure 4.5: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-1

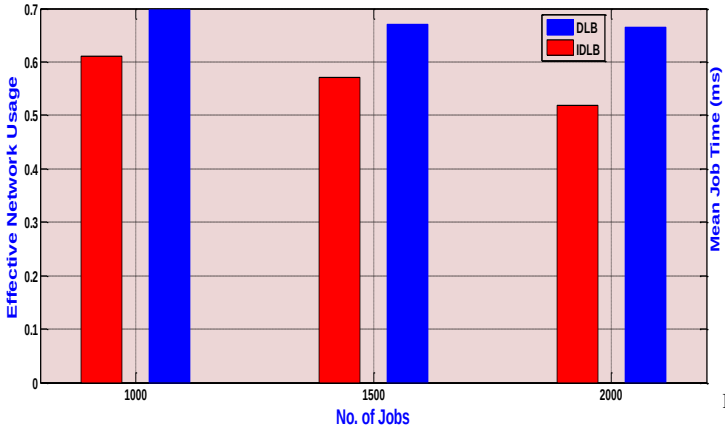

Figure 4.6: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-1

4.2.1.1 Results Analysis of Scenario-1 for DLB And IDLB

We have conducted various experiments for 100 to 2000 jobs which are submitted as the user input using scenario-1 performance parameter table. From table 4.1 to 4.4 and figure 4.1 to 4.6, The simulation result show that the Mean Job Time (MJT) obtained by DLB for 100 jobs is 9334 milliseconds whereas the Total Mean Job Time (MJT) obtained by the proposed IDLB algorithm for the same number of job is 5412 milliseconds. Number of Replications obtained by DLB for 100 jobs is 138 whereas the Number of Replications obtained by the proposed IDLB algorithm for the same number of job is 123. Effective Network Usage (ENU) obtained by DLB for 100 jobs is 0.78310346 whereas the Effective Network Usage (ENU) obtained by the proposed IDLB algorithm for the same number of job is 0.7692308. Hence improvement noted 14.6% in Mean Job Time, 11.31% in number of replication, and 10.07% in effective network usages. Hence, proposed IDLB scheduling algorithm is efficient than the existing DLB

4.2.2 Scenario- 2 for DLB and IDLB

The simulation parameter values used for both DLB and IDLB scheduling algorithm which are shown in Table 4.7.

Table 4.5 Grid simulation parameters for DLB and IDLB of scenario-2

Number of jobs	100,200,300,400,500,1000, 1500,2000
The categories of users are	Random
Optimisers	LruOptimiser
access pattern generators	SequentialAccessGenerator
maximum queue size	100
Hope count	50

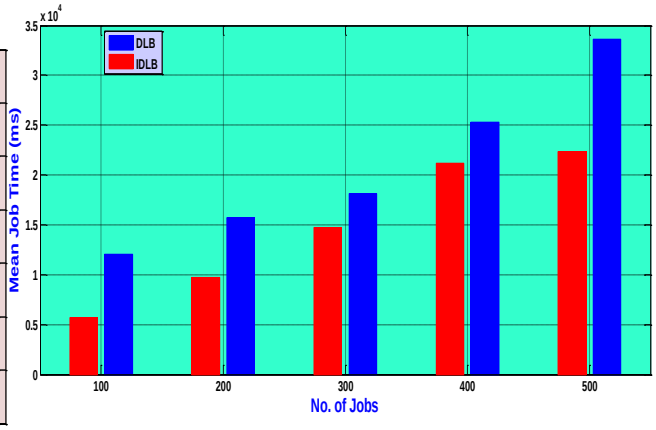

Figure 4.7: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-2

In this experiment 100 to 2000 jobs are submitted as the user input using scenario-2 parameter table. The simulation result show that the Mean Job Time (MJT) obtained by DLB for 100 jobs is 12018 milliseconds whereas the Total Mean Job Time (MJT) obtained by the proposed IDLB algorithm for the same number of jobs is 5724 milliseconds. Similarly for jobs,

200, 300, … 2000, are shown in table 4.6. On the basis of these experimental results we can easily conclude that the proposed IDLB scheduling algorithm is efficient than the existing DLB. It gives 15.15% average improvement over the existing algorithm.

Table 4.6: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	**Mean Job Time**	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	12018	5724
200	15676	9749
300	18111	14728
400	25328	21208
500	33608	22356
1000	41830	41773
1500	44736	43608
2000	45753	41989

Table 4.7 shows the Number of Replications of all jobs on Grid for dynamic load balancing and improve dynamic load balancing in resource management. In this experiment 100 to 2000 jobs are submitted as the user input using scenario-2 parameter table. The simulation results show that the Number of Replications obtained by DLB for 100 jobs is 154 whereas the Number of Replications obtained by the proposed IDLB algorithm for the same number of job is 112.

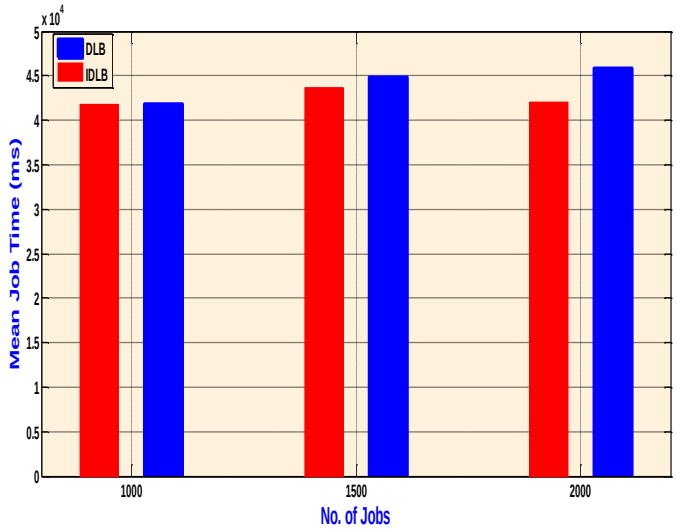

Figure 4.8: Mean Job Time Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-2

Similarly for jobs, 200, 300, … 2000, are shown in table 4.7. Hence, the result of the experiment clearly reveals that our proposed technique outperforms the existing technique and gives an optimum improvement of 14.86%.

Table 4.7: Number of Replications Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	**Number of Replications**	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	154	112
200	454	383
300	393	269
400	524	498
500	406	354
1000	1185	1010
1500	1870	1697
2000	2575	2114

Table 4.8 shows the experimental data of existing and proposed algorithm in which jobs ranges 100 to 2000 and it can be clearly seen from figure 4.9 to figure 4.10 for Effective Network Usage of all jobs on Grid for dynamic load balancing and improve dynamic load balancing in resource management. Thus the method of improve dynamic load balancing performs better result than the existing dynamic load balancing.

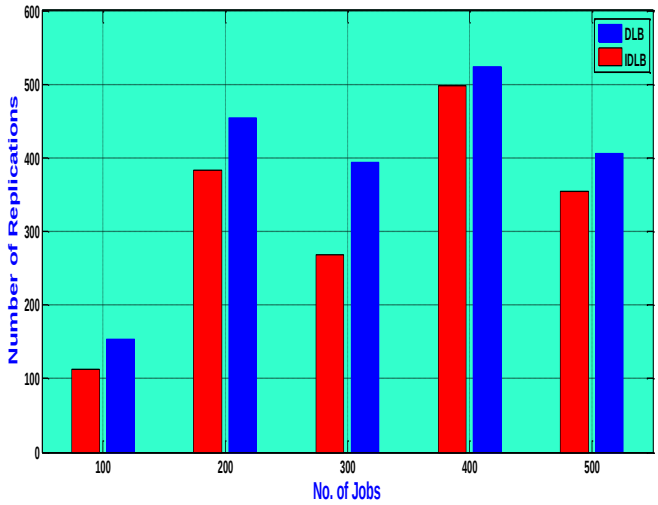

Figure 4.9: Number of Replications Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-2

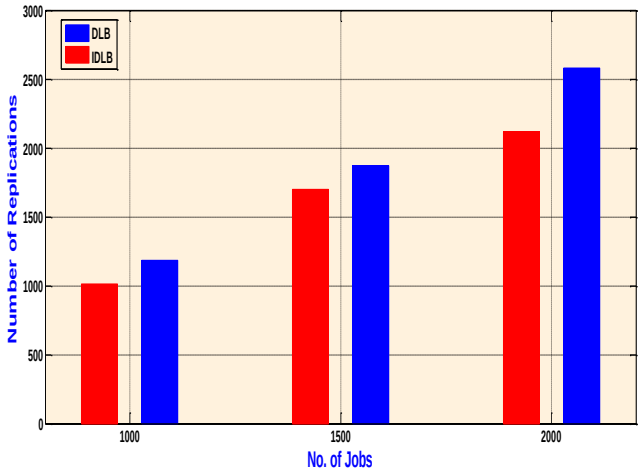

Figure 4.10: Number of Replications Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-2

Table 4.8: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	Effective Network Usage	
	Dynamic Load Balancing (DLB)	*Improved Dynamic Load Balancing (IDLB)*
100	0.77140784	0.7485761
200	0.79721814	0.74072267
300	0.7682223	0.71221496
400	0.77951903	0.71363187
500	0.76128645	0.68582406
1000	0.72801685	0.64800125
1500	0.6809057	0.5912359
2000	0.6717093	0.4814339

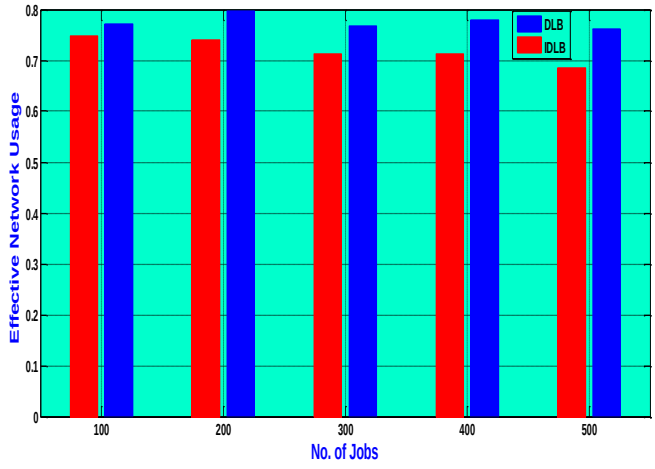

Figure 4.11: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 100 to 500 of scenario-2

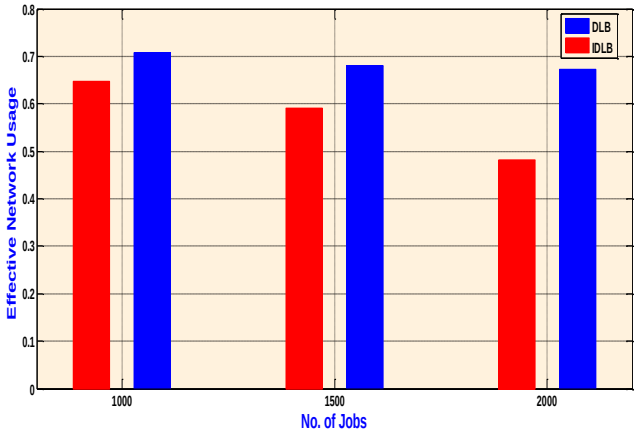

Figure 4.12: Effective Network Usage Vs Jobs for DLB and IDLB when no. of jobs from 1000 to 2000 of scenario-2

4.2.2.1 Results Analysis of scenario-2 for DLB and IDLB

Table 4.5 to 4.8 and figure 4.7 to 4.12 shown the result of conducted various experiments for 100 to 2000 jobs which are submitted as the user input using scenario-2 performance parameter table. The simulation result show that the Mean Job Time (MJT) obtained by DLB for 100 jobs is 12018 milliseconds whereas the Total Mean Job Time (MJT) obtained by the proposed IDLB algorithm for the same number of job is 5724 milliseconds. Number of Replications obtained by DLB for 100 jobs is 154 whereas the Number of Replications obtained by the proposed IDLB algorithm for the same number of job is 112. Effective Network Usage (ENU) obtained by DLB for 100 jobs is 0.77140784 whereas the Effective Network Usage (ENU) obtained by the proposed IDLB algorithm for the same number of job is 0.7485761. Hence improvement noted 15.15% in Mean Job Time, 14.86% in number of replication and 10.69% in effective network usages. Hence, proposed IDLB scheduling algorithm is efficient than the existing DLB.

V. CONCLUSION AND FUTURE SCOPE

In this research, we proposed a resource management technique which manages resources; to manage these resource jobs scheduling and resource allocation algorithm has been evaluated. In this work, we improved the dynamic load balancing scheduling algorithm for job scheduled and allocation has been developed and evaluated. The proposed algorithm takes into account the heterogeneity of the grid computational resources, and it resolves the single point of failure problem which many of the current policies suffer from. Hence, average improvement summary results of the research work are shown in table 5.1.

Table 5.1 Average improvement summary of proposed techniques with various scenarios

Parameters →	***Mean Job Time***	***Number of Replications***	***Effective Network Usage***
Scenario-1	14.6%	11.31%	10.07%
Scenario-2	15.15%	14.86%	10.69%

We clear that the method of IDLB outperforms when compare with DLB based job scheduling. Moreover, proposed scheduling algorithms have shown improvement in Mean Jobs Time, No. of Replications, Effective Network Usages. By increasing the number of jobs, as results, IDLB shown best performance over computation time. More ever some additional methods and algorithm to be find with security issues for present resource management technique.

REFERENCES

[1] Garg SK, Buyya R, Siegel HJ (2010) Time and cost tradeoff management for scheduling parallel applications on utility Grids. Future Gener Comput Syst 26:1344–1355

[2] S. K. Patel, A.K. Sharma,"*Grid Computing: Status of Technology In Current Perspective*", International Journal of Software & Hardware Research in Engineering, Vol.2, Issue.6, 2014.

[3] S. K. Patel, A.K. Sharma," *Design and Implementation of an Efficient Resource Sharing algorithm for Grid Computing*", International Journal of Software & Hardware Research in Engineering, Vol.2**,** Issue.5**,** 2014**.**

[4] Foster, and C. Kesselman. 2003,"The Grid 2: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, USA.

[5] R. Buyya, D. Abramson, and S. Venugopal. 2005, "The Grid Economy". Proceedings of the IEEE, pp. 698-714.

[6] S. K. Patel, A.K. Sharma,"Implementing job scheduling to optimize computational task in Grid Computing using PSO", International Journal of Computer application, 2015**.**

[7] Thamarai Selvi, S., Ponsy, R. K., Bhama, S., Architha, S., Kaarunya, T., Vinothini, K.,(2010). Scheduling In Virtualized Grid Environment Using Hybrid Approach, International Journal of Grid Computing & Applications (IJGCA) Vol.1, No.1.

[8] Somasundaram, K., Radhakrishnan, S., (2008). Node Allocation In Grid Computing Using Optimal Resource Constraint (ORC) Scheduling", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.6.

[9] Quan, L., Yeqing ,L.,(2009). Grouping-Based Fine- grained Job Scheduling in Grid Computing, Vol.1, pp.556- 559, IEEE First International Workshop on Education Technology and Computer Science.

[10] Kaur,S., Kaur,S.(2013). Survey of Resource and Grouping Based Job Scheduling Algorithm in Grid Computing, IJCSMC, Vol. 2, Issue. 5, pg.214 – 218

[11] Srivastava, A., Rathore, R. & Sharma, R.(2013). High Compaction Coarse Grained Job Scheduling In Grid Computing, International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR) ISSN 2249-6831 Vol. 3, Issue 2, 295-302.

[12] Nithiapidary , M.,(2005). A Dynamic Job Grouping- Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids

[13] Ang, T.F., . Ng, W.K., Ling, T.C., Por, L.Y., and Liew, C.S., (2009). A Bandwidth -Aware Job Grouping - Based Scheduling on Grid Environment. Information Technology Journal, 8: 372-377.

[14] Cameron, D. G., Schiaffino, R. C., Ferguson, J., Millar, P., Nicholson, C., Stockinger, K., and Zini, F., (2004). OptorSim v2.0 Installation and User Guide.

[15] S. K. Patel," Design And Development Of A New Technique Including Policies For Resource Sharing Management In Computational Grid System" , 2017.

[16] S. Parsa, R. Entezari-Maleki, RASA: A New Grid Task Scheduling Algorithm, JDCTA (2009) 91–99.

[17] A. Olteanu, F. Pop, C. Dobre, V. Cristea, A dynamic rescheduling algorithm for resource management in large scale dependable distributed systems, Comput. Math. Appl. 69 (9) (2012) 1409–1423.

[18] M. A. Vasile , P. Florin, " Resource- Aware Hybrid Scheduling Algorithm in heterogeneous distributed Computing", Futer Generation Computer System, 51 (2015), 61-67.

Authors Profile

Dr. Surendra Kumar Patel is currently working as Assittant Professor in the Department of Information Technology, Govt. N.P.G. College of Science, Raipur , India. He has published more than 10 research paper. His main research work focuses on Grid Computing and Cloud Computing.He has more than 10 years of teaching experince and 5 years research experience.

Dr. Anil Kumar Sharma is currently working as Assittant Professor in the Department of Information Technology, at APSGMNS Govt. P.G. College Kawardha, India. He has published more than 10 research paper. His main research work focuses on Sensor Network He has more than 10 years of teaching experince and 5 years research experience.

