Research Article

International Journal of Advanced Technology and Engineering Exploration, Vol 6(54)
ISSN (Print): 2394-5443   ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2019.650021

Improved PSO based job scheduling algorithm for resource management in grid computing

Surendra Kumar Patel[1*] and Anil Kumar Sharma[2]
Assistant Professor, Department of Information Technology, Govt. Nagarjuna Post Graduate College of Science, Raipur (C.G.), India[1]
Assistant Professor, Department of Information Technology, Govt. P.G. College, Kawardha (C.G.), India[2]

Received: 24-February-2019; Revised: 27-May-2019; Accepted: 29-May-2019

Abstract

Scheduling jobs to resources in grid computing is a complicated task due to the dynamic nature of resources. An efficient job scheduling algorithm is required to reduce the total time and cost of job execution and improve load balancing among resources in the network. The main problems in managing resources are hardware and software failures, jobs management downtime, etc. To solve this, the PSO is introduced. The PSO algorithm is based on a simplified model of social behaviour exhibited by the buzzing behaviour of insects, birds and fish. In this research, we propose a new algorithm for job scheduling, called improved particle swarm optimization (IPSO). The proposed algorithm generates a velocity vector that is used to point out that the direction of swarm movement and particle position are updated. Therefore, it refines and improves the efficiency of the execution, the research capacity of the global research, the accuracy of the solution and guarantees the load balancing of the programming of the grid activities. Consequently, the proposed work has been simulated with the help of the OptorSim simulator and it has been shown that our proposed algorithm provides an effective solution for planning the resources over grid scheduling network.

Keywords

Grid computing, Job scheduling, Computational grid, PSO, IPSO, Resource management, OptorSim.

1.Introduction

Recent research on computer technology has followed the emergence of a replacement computing paradigm called Grid computing. A resource within the grid computing environment are some things that are necessary to prevent the associated operation, such as a processor used for processing. The adjustment of the resources of the computational grid is responsible for the innovation of resources and the assignment of an activity to a particular resource. In general, it is easy to obtain information on the ability to process data from the available resource [1, 2]. The current IT industry is operating with very high amounts of data using additional processing power and high data storage volumes. Grid computing is proposed as an effective resource management for the organization, since it involves the use of resources from different spaces, from different owners and with the different individual performance of jobs [3, 4].

Resource supervision and task scheduling are very important and complex problems in the grid computing environment. To manage these resources, work planning and load balancing techniques are needed that are responsible for the efficient use of network resources, reduction of work timeout and intelligent access latency. After a thorough investigation of an existing grid involving a large number of CPU clusters, we observed that grid programming decisions can significantly improve the computational time if the features of current usage patterns are included [5, 6].

It can be argued that the scheduling of the tasks of the current network and the allocation of resource management techniques are excessively focused on low-level technological aspects and, based on them, it is difficult to conceive how a global model can contain and explain all these little details. To overcome these problems, the new load balancing scheduling algorithm is extremely advantageous both for user interaction and system administration.

*Author for correspondence

Therefore, the main objective of this research is to develop a technique that uses jobs with the PSO method to reduce jobs response time and increase calculation performance. Based on the challenges discussed, the objectives of this research are as follows:

- To reduce the job completion time of an application by appropriately allocating the jobs to the processors.
- To realize the full potential of grid computing, grid middleware desires to support various services such as uniform access, economic computation, job scheduling, composition and resource management.
- To solve the problem of job activities in the grid computing environment, the proposed method of improved PSO algorithms is used in a way that produces a better concert and provides an optimized result.

This paper consists of five sections which are organized as follows: section 1 describes the introductory part of the investigation. Section 2 presents the study of literature and research work in the field of related subjects. Section 3 presents the research methodology used in this research and the OptorSim simulator is used to support the administration of resources and other capabilities, such as job allocation, job planning and balancing, etc. Section 4 presents the analysis of the results and section 5 concludes the research with a summary of the main results, the discussion of future research directions, final comments and describes the possibilities for future work.

2.Related work

The resources of the network are dispersed in a heterogeneous way in a large geographical area in which numerous and different types of users connect simultaneously. Since the nature of the resources is dynamic, the administration of these resources and the users adequately, the tasks must be planned precisely and the analogous resources required must be assigned accordingly.

There are several heuristic algorithms designed for static programming of activities in a network environment to minimize the training of researchers [7, 8]. Because of the vigorous environment and heterogeneous resources of the network, the most well-known scheduling algorithms are not suitable for bulky systems. The exploitation of resources aims to amplify the time occupied by resources. In this research paper, our proposed algorithm aims to focus on resources that are not used correctly. Towards the algorithm identifies resources that are overloaded and tries to contribute the load to other less loaded resources or inactive resources. Subsequently, the algorithm eliminates the task of a deeply loaded resource, refines the work and assigns the resources loaded without haste. Therefore, the parallelism in the use of resources increases considerably and the purpose can be reduced. Hence, the load on resources can also be balanced [8, 9].

The authors have achieved the result that the grid is a dynamic system, which always changes over time due to a number of factors, such as ease of use of new resources, system errors / resources, new requests, completion of an activity in execution, etc. When one of these situations occurs, it is necessary to reprogram the work or reassign it for execution [10].

The algorithm review makes a decision on the amount of resources (such as time and storage space) needed to run them for resource management [11]. The authors suggested that it is necessary to reduce communication times, processing times and increase the use of resources in the case of light programming or small jobs [12].

There have been several research works on the problem of network activity planning, and the different algorithms have their advantages and disadvantages, but they still need further analysis and research to improve the performance of the computer network scheduling algorithm [13]. The authors developed a new hybrid approach algorithm. The proposed hybrid heuristic model included the PSO algorithm and the SA algorithm. Performed static assignment of activities in a way designed to minimize the cost. It occurred in the cost of execution and in the cost of communication when it was executed on different machines [14−16]. Recently, in [17] a real-time device scheduling system with the aim of reducing the cost of electricity while maintaining user comfort and developing a new device control algorithm, using the networks of Petri dishes that interact with users have been introduced with the devices in order of priority based on user comfort. The authors explained a plan to disseminate mobile average resources for virtualized cloud information services [18].

Alkayal et al. [19] have developed a particle swarm optimization algorithm based on multiple activities. Which one is based on a new classification strategy? The main work of this research was that the activities were scheduled in virtual machines to minimize waiting times and maximize system performance. To overcome the difficulties, Dordaie and Navimipour [20] analyzed the optimization of hybrid particle swarm with the combination of the slope ascent algorithm for planning activities in cloud environments. This method has been programmed correctly. Although, the hybrid algorithm takes the maximum time to complete the task. Similarly, Verma and Kaushal [21] have improved the swarming of multi-target hybrid particles for scientific workflow scheduling.

The optimization of the swarm of particles in combination with genetic operators and the distribution of Cauchy has updated the positions within HPSO + GA. This was applied for the effective movement of the particles in the fuzzy processing time [22].

After analyzing the literature of the resource scheduling algorithm, we applied IPSO, whose goal was to minimize average working time, in general, the number of job replication and efficient use of networks.

3. Methodology

3.1 Simulation tool and environment

In this research, the OptorSim simulator [13] is used to simulate the resource scheduling algorithm proposed for resource management.

Several tools are available to simulate resource scheduling algorithms to manage resources in Grid computing environments. In this research, we use OptorSim. We evaluated the algorithm by simulating different working ranges between 100 and 2000 in our experiments.

3.1.1OptorSim

OptorSim is a grid data simulation tool based on a composite modelling prototype, which allows the user to simulate the behaviour of a wide range of distributed networks for a given application, within the different calculation and load scenarios. The OptorSim architecture shown in *Figure 1*.

This architecture is based on the data management components of the European transmission grid in which the calculation and storage resources are represented by the calculation elements (CE) which is called computing element and by the memory elements (SE) which is storage element, respectively, organized in network sites [15]. The CEs performs the work of processing the data files, which are stored in the roles of SEs and users, since it sends the jobs to the grid based on the chosen sending pattern, and therefore the Resource Broker (RB) programs the jobs in the network sites.

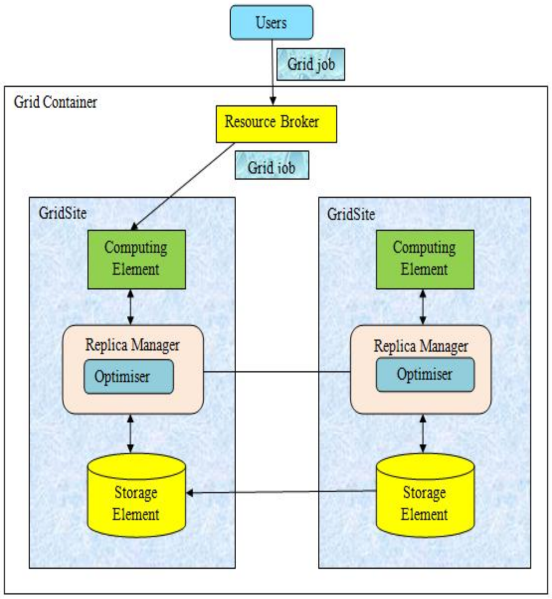


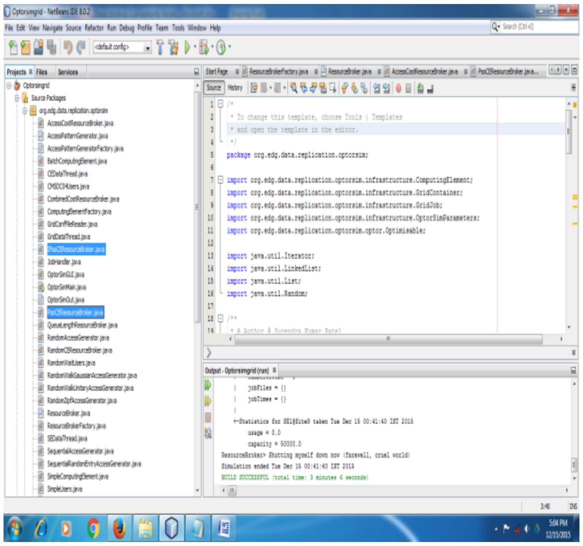
Figure 1 OptorSim architecture [16]

When a job is processed in a CE, it will go through its list of files to be processed based on the chosen access pattern of jobs and its nature. If a file is required that is not present on the execution site, it must be replicated or read remotely [16].

3.1.2 Design of the simulation environment screen

In this research, we used two configuration files to check different OptorSim entries; the first is the grid configuration file that specifies the topology of the grid and the content of each site (the number of SEs and CEs) per user. The other is the job configuration file that contains information for the simulated jobs.
Therefore, we see the optimization as an ongoing activity, which takes place in two moments during the useful life of a job.

- In the first phase of an optimization was done when choosing the calculation element in which the work must be performed.
- In the second phase, the optimal selection of dynamic replication is achieved during the time of execution of a job; in this phase, the creation of replicas can be activated by the optimization algorithm.


Optorsimgrid - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Search (Ctrl+I)
Projects Files Services
Optorsimgrid
Source Packages
org.edg.data.replication.optorsim
AccessCostResourceBroker.java
AccessPatternGenerator.java
AccessPatternGeneratorFactory.java
BatchComputingElement.java
CEDataThread.java
CMSDC04Users.java
CombinedCostResourceBroker.java
ComputingElementFactory.java
GridConfFileReader.java
GridDataThread.java
IPsoCEResourceBroker.java
JobHandler.java
OptorSimGUI.java
OptorSimMain.java
OptorSimOut.java
PsoCEResourceBroker.java
QueueLengthResourceBroker.java
RandomAccessGenerator.java
RandomCEResourceBroker.java
RandomWaitUsers.java
RandomWalkGaussianAccessGenerator.java
RandomWalkUnitaryAccessGenerator.java
RandomZipfAccessGenerator.java
ResourceBroker.java
ResourceBrokerFactory.java
SEDataThread.java
SequentialAccessGenerator.java
SequentialRandomEntryAccessGenerator.java
SimpleComputingElement.java
SimpleUsers.java
Start Page
ResourceBrokerFactory.java
ResourceBroker.java
AccessCostResourceBroker.java
PsoCEResourceBroker.java...
Source History
Output - Optorsimgrid (run)
3:40 INS
5:04 PM
12/15/2015

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package org.edg.data.replication.optorsim;

import org.edg.data.replication.optorsim.infrastructure.ComputingElement;
import org.edg.data.replication.optorsim.infrastructure.GridContainer;
import org.edg.data.replication.optorsim.infrastructure.GridJob;
import org.edg.data.replication.optorsim.infrastructure.OptorSimParameters;
import org.edg.data.replication.optorsim.optor.Optimisable;

import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
import java.util.Random;

/**
 * @ Author @ Surendra Kumar Patel
```

```
    |    jobFiles = {}
    |    jobTimes = {}
    |
    +-Statistics for SE1@Site8 taken Tue Dec 15 00:41:40 IST 2015
         usage = 0.0
         capacity = 50000.0
ResourceBroker> Shutting myself down now (farewell, cruel world)
Simulation ended Tue Dec 15 00:41:40 IST 2015
BUILD SUCCESSFUL (total time: 3 minutes 6 seconds)
```

Figure 2 Simulator environment screen layout

3.2. Parameters used

The performance estimation of the proposed algorithm has been done using the following metrics:

- **Mean Job Time-** For executing a job the mean job time is required which is defined as average time.
- **Number of Replication-** The replication means that it's promoted increased fault tolerance, low bandwidth consumption, high data availability and improved scalability of the system.
- **Effective Network usage-** It indicates networks that local files or remote file used on the network.

Mean job time, number of replications and effective network usage are the critical parameters that affect resource management in an efficient manner.

4. Result and discussion

Improved PSO based Job scheduling performs the task by using different metrics. Experimental data had shown the comparison of standard PSO and improved PSO approaches.

4.1. Performance evaluation and analysis of the algorithm

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

4.2.Proposed improved particle swarm optimization (IPSO) algorithm

4.2.1. Scenario- 1 for PSO and IPSO

The simulation parameter values used for PSO and IPSO scheduling algorithm are shown in *Table 1*

Table 1 Grid simulation parameters for PSO and IPSO 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	100
Hope count	50

In this experiment 100 to 2000 jobs are submitted as the user input using scenario-1 performance parameter *Table 2*.

Table 2 Mean job time vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-1

No. of Jobs	**Mean job time**	
	Standard PSO	**Improved PSO based job scheduling**
100	6544	5890
200	9804	9725
300	17317	16001
400	23043	18592
500	23504	19831
1000	35557	27702
1500	44234	33196
2000	45310	36515

The simulation result shows that the mean job time obtained by PSO for 100 jobs is 6544 milliseconds whereas the total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 5890milliseconds. The mean job time obtained by PSO for 200 job is 9804 milliseconds whereas the Total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 9725 milliseconds.

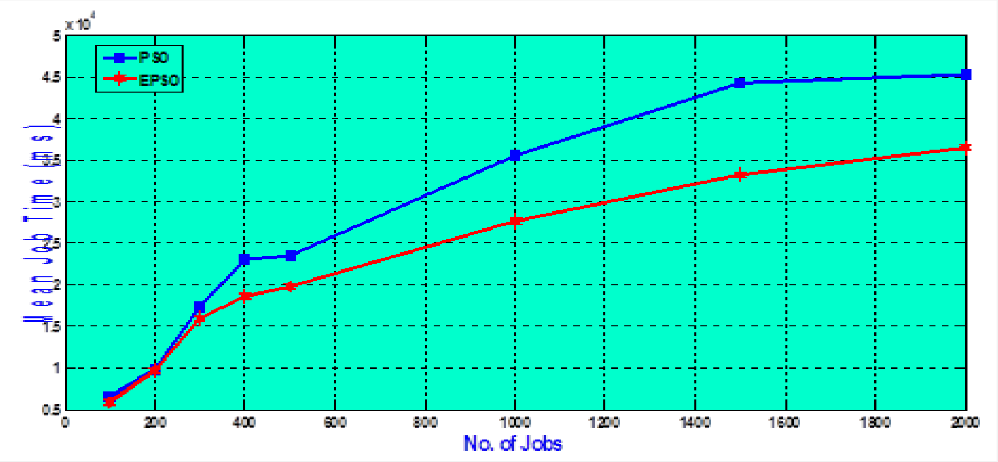


Figure 3 Mean job time vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-1

On the basis of these experimental results we can easily conclude that the proposed IPSO scheduling algorithm is efficient than the existing PSO.

Table 3 Number of replications vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-1

No. of jobs	**Number of replications**	
	Standard PSO	**Improved PSO based job scheduling**
100	238	130
200	399	333
300	349	276
400	596	377
500	439	354
1000	1105	1042
1500	1823	1760
2000	2316	2213

It gives 18.44% performance average improvement over the existing algorithm.

From *Table 3* and *Figure 4* we conclude that the number of replications of all jobs on grid for PSO and IPSO in resource management. Thus, the method of IPSO performs better function than the existing PSO.

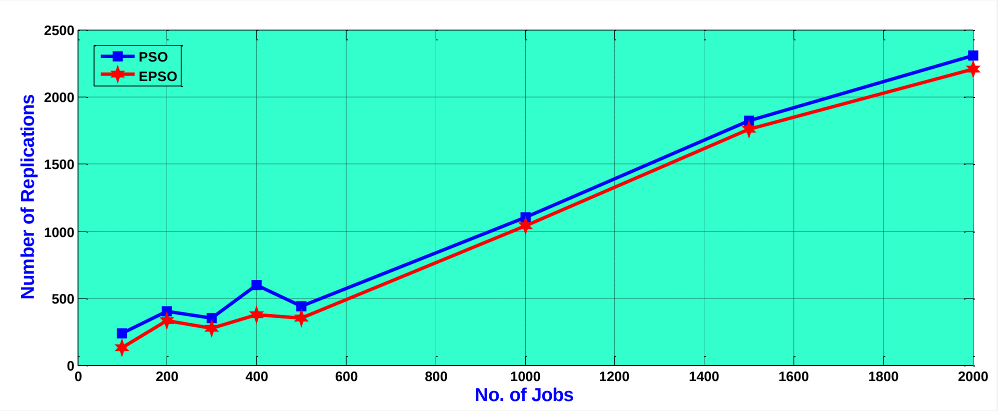


Figure 4 Number of replications vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario

In *Table 4* and *Figure 5* shown that effective network usage of all jobs on grid for PSO and improve particle swarm optimization (IPSO) in resource management.

Thus, the method of IPSO performs better result than the existing PSO.

Table 4 Effective network usage vs. Jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-1

No. of jobs	Effective network usage	
	STANDARD PSO	**IMPROVED PSO BASED JOB SCHEDULING**
100	0.7989225	0.7601586
200	0.7934066	0.7457895
300	0.77240025	0.7227543
400	0.74765416	0.69526255
500	0.7637902	0.7104087
1000	0.81484824	0.7502435
1500	0.8045658	0.7327863
2000	0.82947034	0.71544332

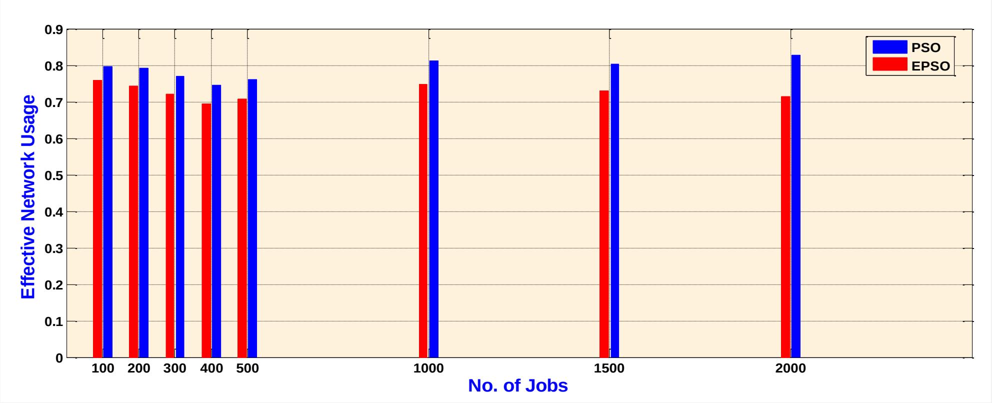


Figure 5 Effective network usage vs. Jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-1

4.2.1.1 Results analysis of scenario-1 for PSO and IPSO

Table 2 to 4 and *Figure 3 to 6* shows the result of various experiments conducted for 100 to 2000 jobs which are submitted as the user input using scenario-1 performance parameter table. The simulation result show that the mean job time obtained by PSO for 100 jobs is 6544 milliseconds whereas the total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 5890 milliseconds. Number of replications obtained by PSO for 100 jobs are 238 whereas the number of replications obtained by the proposed IPSO algorithm for the same number of jobs is 130. Effective network usage obtained by PSO for 100 jobs is 0.7989225 whereas the effective network usage obtained by the proposed IPSO algorithm for the same number of jobs is 0.7601586.

4.3. Scenario- 2 for PSO and IPSO

Table 5 Grid simulation parameters for PSO and IPSO of scenario-2

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

Hence it gives improvement of 18.44% in Mean Job Time, 10.74% in the number of replications, and 7.78% in effective network usages. Hence, the proposed IPSO scheduling algorithm is more efficient than the existing PSO.

The simulation result shows that the mean job time obtained by PSO for 100 jobs is 7401 milliseconds whereas the total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 5914 milliseconds.

Table 6 Mean job time vs. Jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	Mean Job Time	
	Standard PSO	**Improved PSO Based Job Scheduling**
100	7401	5914
200	14938	10689
300	17897	13249
400	25324	20206
500	27312	19108

No. of jobs	Mean Job Time	
	Standard PSO	Improved PSO Based Job Scheduling
1000	39710	33736
1500	42974	34838
2000	42974	35838

Mean job time obtained by PSO for 200 jobs is 14938 milliseconds whereas the total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 10689 milliseconds.

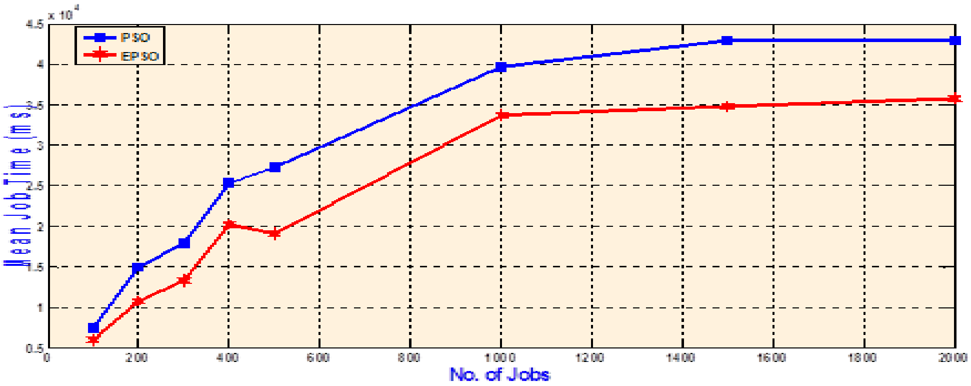


Figure 6 Mean job time vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

Similarly, for jobs, 300, 400, … 2000, are shown in *Table 5*. Thus, the method of IPSO performs better function than the existing PSO.

From *Table 6* and *Figure 6* we conclude that the number of replications of all jobs on the grid for PSO and IPSO in resource management. Thus, the method of IPSO performs better function than the existing PSO.

From *Table 3* and *Figure 6* we conducted experiments for 100 to 2000 jobs which are submitted as the user input using scenario-2 parameter table.

Table 7 Number of replications vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	Number of replications	
	Standard PSO	Improved PSO based job scheduling
100	243	216
200	240	134
300	416	378
400	498	429
500	519	326
1000	1028	975
1500	1958	1803
2000	1754	1402

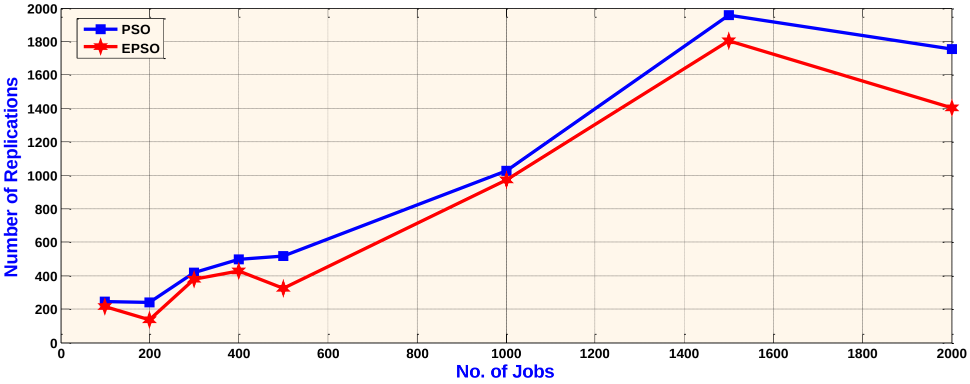


Figure 7 Number of replications vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

The simulation result shows that the effective network usage obtained by IPSO for 100 jobs is 0.7935593 whereas it is obtained by the proposed IPSO with a digital pheromone algorithm for the same number of jobs is 0.7825357.

Table 8 Effective network usage vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

No. of jobs	**Effective network usage**	
	Standard PSO	**Improved PSO based job scheduling**
100	0.7935593	0.7825357
200	0.78409893	0.7529928
300	0.77331854	0.7386923
400	0.7588567	0.7147091
500	0.7588567	0.6947091
1000	0.8054105	0.7097865
1500	0.8580156	0.7514796
2000	0.8980245	0.7683347

The effective network usage obtained by IPSO for 200 jobs is 0.78409893 whereas it is obtained by the proposed IPSO for the same number of jobs is 0.7529928. Similarly, for jobs, 300, 400, … 2000, are shown in table 4.30.

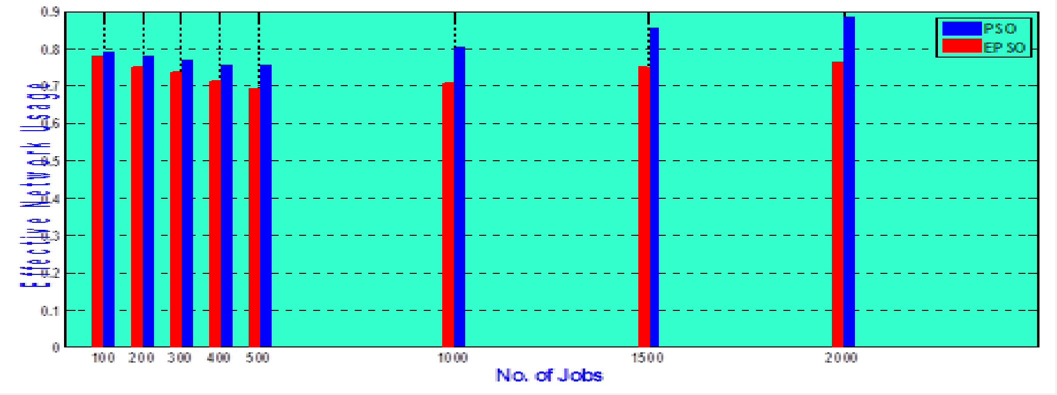


Figure 8 Effective network usage vs. jobs for PSO and IPSO when no. of jobs from 100 to 2000 of scenario-2

On the basis of these experimental results we can easily conclude that the proposed IPSO scheduling algorithm is more efficient than the existing PSO. It gives 7.78% performance improvement over the existing algorithm.

4.3.1Results analysis of scenario-2 for PSO and IPSO

Table 7 to 8 and *Figure 5 to 8* shows the result of various experiments conducted 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 obtained by PSO for 100 jobs is 7401 milliseconds whereas the total mean job time obtained by the proposed IPSO algorithm for the same number of jobs is 5914 milliseconds. Number of replications obtained by PSO for 100 jobs are 243 whereas the number of replications obtained by the proposed IPSO algorithm for the same number of jobs is 216. Effective network usage obtained by PSO for 100 jobs is 0.7935593 whereas it is obtained by the proposed IPSO algorithm for the same number of jobs is 0.7825357.

Hence it gives improvement of 20.57% in mean job time, 14.99% in the number of replications, 13.95% in the total number of local file access, 12.35% in the total number of remote file accesses and 8.03% in effective network usages.

Hence, the proposed IPSO scheduling algorithm is more efficient than the existing PSO.

5. Conclusion and future work

In this research, we proposed a resource management algorithm that manages resources. To manage these resource jobs, the programming and resource allocation algorithm was evaluated. In this work, we improved the particle swarm optimization algorithm for scheduled work and the allocation was developed and evaluated. The proposed algorithm takes into account the heterogeneity of the computational resources of the network and solves the problem of the single point of error suffered by many current policies. Therefore, the results of the summary of the average improvement in research work are shown in *Table 9.*

Table 9 Average improvement summary of proposed algorithm PSO and IPSO with various scenarios

Parameters	**Scenario-1**	**Scenario-2**
Mean job time	18.44%	20.57%
Number of replications	10.74%	14.99%
Effective network usage	7.78%	8.03%

The results of simulated experiments shown that the improved particle swarm optimization algorithm is able to get the better schedule than existing algorithms. Moreover, proposed scheduling algorithms have shown improvement in mean jobs time, no. of replications, and effective network usages. Some additional methods and algorithm to be found in future with security issues for present resource management technique.

This research paper addressed the innovative ideas for researcher to an optimized scheduling algorithm using critical parameters to reduce mean job time; increase resource utilization, decrease job replication and increase fault-tolerant but several issues still require investigating.

The most obvious is as follows:

- Security becomes a significant challenge in open distribute systems. Security problems haven't been self-addressed during this analysis of research.
- One attainable resolution is to leverage existing analysis in grid security, like Globus security infrastructure. Future work could extend this research and mix it with Globus protocols for security.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Foster I, Kesselman C. The grid: blueprint for a future computing infrastructure. Morgan Kaufmann Publishers Inc.; 1999.

[2] Zhang L, Chen Y, Sun R, Jing S, Yang B. A task scheduling algorithm based on PSO for grid computing. International Journal of Computational Intelligence Research. 2008; 4(1):37-43.

[3] Garg SK, Buyya R, Siegel HJ. Time and cost trade-off management for scheduling parallel applications on utility grids. Future Generation Computer Systems. 2010; 26(8):1344-55.

[4] Patel SK, Sharma AK. Grid computing: status of technology in current perspective. International Journal of Software & Hardware Research in Engineering. 2014; 2(6).

[5] Patel SK, Sharma AK. Optimization of dynamic resource scheduling algorithm in grid computing environment. International Journal of Computer Sciences and Engineering. 2018; 6(3): 20-26.

[6] Singh H, Youssef A. Mapping and scheduling heterogeneous task graphs using genetic algorithms. In heterogeneous computing workshop 1996 (pp. 86-97).

[7] He X, Sun X, Von Laszewski G. QoS guided min-min heuristic for grid task scheduling. Journal of Computer Science and Technology. 2003; 18(4):442-51.

[8] Chauhan SS, Joshi RC. QoS guided heuristic algorithms for grid task scheduling. International Journal of Computer Applications. 2010; 2(9):24-31.

[9] Dong F, Luo J, Gao L, Ge L. A grid task scheduling algorithm based on QoS priority grouping. In international conference on grid and cooperative computing 2006 (pp. 58-61). IEEE.

[10] Izakian H, Abraham A, Ladani BT. An auction method for resource allocation in computational grids. Future Generation Computer Systems. 2010; 26(2):228-35.

[11] Patel SK, Sharma AK. Design and implementation of an efficient resource sharing algorithm for grid computing. International journal of software & hardware research in engineering. 2014; 2(5).

[12] Kaur S, Kaur S. Survey of resource and grouping based job scheduling algorithm in grid computing. International Journal of Computer Science and Mobile Computing. 2013; 2(5):214-8.

[13] Patel SK, Sharma AK, Seetha A. Implementing job scheduling to optimize computational tasks in grid computing using PSO. International Journal of Computer Applications. 2015:20-4.

[14] Sivanandam SN, Deepa SN. Introduction to genetic algorithm. Springer ; 2007.

[15] Patel SK. Design and development of a new technique including policies for resource sharing management in computational grid System. 2017.

[16] Bell WH, Cameron DG, Millar AP, Capozza L, Stockinger K, Zini F. Optorsim: a grid simulator for studying dynamic data replication strategies. The International Journal of High Performance Computing Applications. 2003; 17(4):403-16.

[17] Özkan HA. Appliance based control for home power management systems. Energy. 2016; 114:693-707.

[18] Priya V, Babu CN. Moving average fuzzy resource scheduling for virtualized cloud data services. Computer Standards & Interfaces. 2017; 50:251-7.

[19] Alkayal ES, Jennings NR, Abulkhair MF. Efficient task scheduling multi-objective particle swarm optimization in cloud computing. In conference on local computer networks workshops 2016 (pp. 17-24). IEEE.

[20] Dordaie N, Navimipour NJ. A hybrid particle swarm optimization and hill climbing algorithm for task scheduling in the cloud environments. ICT Express. 2017.

[21] Verma A, Kaushal S. A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. Parallel Computing. 2017; 62:1-19.

[22] Jamrus T, Chien CF, Gen M, Sethanan K. Hybrid particle swarm optimization combined with genetic operators for flexible job-shop scheduling under uncertain processing time for semiconductor manufacturing. IEEE Transactions on Semiconductor Manufacturing. 2017; 31(1):32-41.

Dr. Surendra Kumar Patel is currently working as Assistant Professor in the Department of Information Technology, Govt. N.P.G. College of Science, Raipur, India. He has published more than 15 research papers and 01 reference book. His main research work focuses on Grid Computing and Cloud Computing. He has more than 10 years of teaching experience and 6 years research experience.
Email: surendrapatelit2004@gmail.com

Dr. Anil Kumar Sharma is currently working as Assistant Professor in the Department of Information Technology, at APSGMNS Govt. P.G. College Kawardha, India. He has published more than 15 research papers. His main research work focuses on Sensor Network He has more than 10 years of teaching experience and 5 years research experience.