

Energy efficient clustering and Job scheduling in Mobile Grid Computing Environment

S.Kavitha Bharathi, M.Dhavamani

Abstract: In the mobile grid environment, job scheduling is the main constituents to resolve the overload of mobile devices. The job scheduling should be energy efficient in the mobile grid so that energy efficient clustering and job scheduling are presented in this paper. Prior to the job scheduling, the grid server selects the Cluster Heads (CHs) based on the maximum capacity of Resource Providers (RPs) or mobile devices in the network. Then the clusters are formed by joining the other RPs with a low latency communication link to the corresponding CHs. After the formation of clusters, the CH schedules the received jobs which are to be processed to the suitable RPs in the network. For this efficient job scheduling, Oppositional Particle Swarm Optimization (OPSO) algorithm is presented. Optimization process of PSO algorithm is enhanced by integrating the Opposite Based Learning (OBL) method. Simulation results show that the performance of the proposed approach is evaluated in terms of energy efficiency and overhead.

Index Terms: Mobile grid environment, cluster formation, job scheduling, Oppositional Particle Swarm Optimization, Opposite Based Learning.

I. INTRODUCTION

Grid computing includes the accumulation of network-connected computers to form a large-scale, distributed system for synchronized computing and resource sharing. By spreading computing workload across this distributed scheme, grid consumers can take benefit of massive computational, storage, and bandwidth resources that would then only be accessible to traditional multiprocessor supercomputers [1-3]. The grid is already being effectively utilized in numerous scientific implementations where huge amounts of information have to be administered and/or stored. Such demanding implementations have fashioned, justified and diffused the notion of grid computing, amongst others, in the scientific community. On another aspect, the availability of wirelessly linked mobile devices has grown significantly in recent years, generating a massive collective unexploited pool for resource utilization. Thus, the extrapolation of the resource aggregation model of grid computing to mobile computing devices seems only natural. Such a scheme that usages computing resources of networked mobile devices (with or without fixed computers) is labeled "mobile grid". Having been created from a group of mobile devices, a mobile grid would permit the networked devices to achieve a precise mission that may be beyond an individual device's computing

or communication capacity [4, 5].

While the field of mobile computing is growing speedily, there are still a large number of trials that these devices face that remain open issues to be solved [6]. If we scrape the surface a little, it may appear that the unification of mobile wireless user devices with high-performance grid computing might not be that simple. Afterward all, grid calculating to date has used multiprocessors and PCs as the computing nodes within its matrix. User computing devices like smartphones are characteristically limited by abridged CPU, memory, secondary storage, and bandwidth capabilities, (as associated with desktop computers) augmented power consumption sensitivity, augmented heterogeneity, unpredictable long periods of complete disconnectivity, unreliable, low bandwidth and high latency communication links; and very dynamic network layout due to devices entering and leaving in a very unpredictable method [7, 8].

One of the main challenges in a computational grid is how to efficiently map jobs, also called jobs or applications, to grid resources and hence utilize geographically distributed computers which are connected through heterogeneous environments in an efficient, reliable and secure manner. This mapping is called job scheduling in grid computing. Similar to job scheduling in traditional computing systems, this mapping is known to be an NP-complete problem [4]. However, it is more complicated in grid computing due to its complex, dynamic nature, a high degree of job and resource heterogeneity, problem size, and other factors such as existing local schedulers and policies. For energy efficient job scheduling, the following contributions are presented in this paper.

Initially, the resource providers (RPs) or mobile devices in the network are grouped as a number of clusters. RP with the maximum capacity is selected as the cluster head (CH). This CH joints the other RPs with low latency communication link and forms a cluster.

The selected CH schedules job to the suitable RPs using the proposed Oppositional Particle Swarm Optimization (OPSO) algorithm.

This proposed approach is simulated in the Network Simulator (NS2).

The performance of the proposed approach is compared with that of existing algorithms PSO and GA based job scheduling.

The performance of the proposed OPSO based job scheduling is evaluated in terms of energy efficiency and overhead.

Rest of this paper organized as follows. Section 2 relates our projected work with the



Revised Manuscript Received on March 25, 2019.

S.Kavitha Bharathi, Assistant Professor, Department of Computer Applications.

M.Dhavamani, Assistant Professor (Sr.G), Department of Mathematics Kongu Engineering College, Perundurai – 638 060.

previous work. The basic of PSO algorithm is described in section 3. Section 4 presents the energy efficient clustering and job scheduling in a mobile grid environment. Results of our projected method are deliberated in segment 5. This paper is determined with section 6.

II. RELATED WORKS

In this segment, we relate our projected work with some previous works that have been done in the same domain. Du Li-juan and Cao Zhi-wen [12] projected Layered architecture for the mobile grid. They implemented their projected technique on the basis of the concept of the overlay network. Their projected Mobility-Aware Routing Mechanism (MARM) maintained on mobility on the overlay network layer. Mobility-aware routing mechanism was united distributed hash table technology with the notion of mobile IP. In this technique, mobile nodes never contributed in the routing procedure but they updated their location data dynamically to the DHT network that was collected of agent nodes.

Li Chunlin and Li Layuan [13] presented exploiting composition of mobile devices for maximizing user QoS under energy constraints in the mobile grid. Their projected Mobile device service composition procedure included two portions that are mobile device resource allocation through a device resource market and mobile device service provisioning over device service market. With the help of market mechanisms, interactions were performed amongst mobile device service agent, mobile grid user agent and mobile device resource agent. They augmented the benefits of mobile device resource agent, mobile device service agent and mobile grid user agent with the help of the Mobile device service composition optimization. They utilized utility optimization to formulate the problem of services composition of mobile devices.

AlirezaSouri and NimaJafari Navimipour [14] offered an adopted type of resource discovery method to address multi-attribute and range queries. They decoupled resource discovery behavior model to information gathering, discovery, and control behavior. With the help of a Binary Decision Diagram (BDD) based formal verification method simplified the mapping procedure between three behaviors. They extracted the anticipated properties of resource discovery method from control behavior in the form of CTL and LTL temporal logic formulas, and verify the properties in information gathering and discovery behaviors lengthily with the help of this formal method.

Yingwei Jinet al [15] projected a hierarchical routing model with help of mobile agents. Using this projected technique, they composed and updated the dynamic vicissitudes of routing data in grid schemes. Their projected optimal solution demonstrated that it made inference on the known traffic data and approaches to an unbiased distribution. Their technique was scalable, fault tolerant and relies entirely on local data.

Li Chunlin and Li Layuan [16] projected an economics-based distributed negotiation structure amongst mobile devices in the mobile grid. In their model, energy negotiation and transactions within buyer devices and seller

devices were positioned. They achieved Dynamic apportionment of energy resources in the mobile grid via online transactions within markets. The author utilized optimization algorithms to upsurge predefined utility functions at the time of the transactions within buyers and sellers. Their projected algorithm disintegrated mobile grid system optimization issue into a series of two sub-issues.

Wei-Chang Yeh and Shang-Chia Wei [17] offered an economic-based resource allocation model to originate the service reliability of Grid-computing from cellular automata Monte-Carlo simulation (CA-MCS) for the service level arrangement. They assessed the total rental-time cost of Grid resources by virtual payment valuation for the free rider issue. They rehabilitated the Grid scheme into the multi-state unreliable network in advance regarding the probability of the mission completion. They economized total rental-time cost and guaranteed the Grid-computing service being dependable with the help of their projected binary-code Genetic Algorithm (bGA) and an integer-code Particle Swarm Optimization (iPSO).

III. BACKGROUND OF THE RESEARCH

A. Basic of Particle Swarm Optimization (PSO) algorithm

PSO algorithm is developed by Eberhart and Kennedy in 1995. This algorithm is inspired by the behavior of bird flocking. This is an evolutionary algorithm based on population and initialized with a population of random solutions. The initialized solutions of this algorithm are called particles. Each particle is initialized with its random position and velocity. Selection of optimal path using this algorithm is described as follows:

The candidate solutions or particles are initialized with a d dimensional vector. In this approach, attacker free paths are considered as the candidate solutions i.e., each solution represents the path between source and destination. It can be initialized as follows,

$$x_k(t) = \{x_{k1}(t), x_{k2}(t), \dots, x_{kd}(t)\} \quad (1)$$

Where, $x_{kd}(t)$ represents the position of the kth particle in the dth dimension vector. The population of this algorithm at time 't' is considered as follows,

$$x(t) = \{x_1(t), x_2(t), \dots, x_m(t)\} \quad (2)$$

After initializing the candidate solutions, the fitness value of each solution is evaluated based on the condition of maximal or minimal. Then the solution is updated based on its position and velocity vector. Until finding the optimal solution, each solution is updated using equations (3) and (4). In each iteration, local or personal best values denoted as Pbest and global best value denoted as Gbest are estimated.



$$V_{kd}(t+1) = w * V_{kd}(t) + \left(P_{best_{kd}}(t) - X_{kd}(t) \right) c_1 r_1 + \left(G_{best_{kd}}(t) - X_{kd}(t) \right) c_2 r_2 \quad (3)$$

$$X_{kd}(t+1) = X_{kd}(t) + V_{kd}(t+1) \quad (4)$$

Where, $X_{kd}(t)$ and $V_{kd}(t)$ represent the position and velocity of the k th particle in d th dimensional space at iteration t . c_1 and c_2 represent the acceleration coefficients which equal to 2. r_1 and r_2 represent the random variables in the range $[0, 1]$. w is inertia weight which is used to control the searching process. This inertia weight value is decreasing while increasing the iteration. This can be calculated as follows,

$$w = w_{\max} i_{\max} - \frac{w_{\max} i_{\max} - w_{\min} i_{\max}}{t_{\max} i_{\max}} \times t \quad (5)$$

Where, $w_{\max} i_{\max}$ and $w_{\min} i_{\max}$ represent the maximum and minimum inertia weight respectively. $t_{\max} i_{\max}$ represents the maximum number of iterations. $P_{best_{kd}}(t)$ and $G_{best_{kd}}(t)$ represent the best position of the particle k and best position of the group at iteration t . If the fitness of the k th particle ($X_{kd}(t+1)$) is better than that of previous $P_{best_{kd}}(t)$, then the particle is considered as new $P_{best_{kd}}(t+1)$. Otherwise, the particle $X_{kd}(t)$ is considered as new $P_{best_{kd}}(t+1)$. Besides, if the fitness of the k th particle ($X_{kd}(t+1)$) is better than that of previous $G_{best_{kd}}(t)$, then the particle is considered as new $G_{best_{kd}}(t+1)$. Otherwise, the particle $X_{kd}(t)$ is considered as new $G_{best_{kd}}(t+1)$.

$$P_{best_{kd}}(t+1) = \begin{cases} X_{kd}(t) & \text{if } F(X_{kd}(t+1)) \text{ is better than } F(P_{best_{kd}}(t)) \\ X_{kd}(t+1) & \text{otherwise} \end{cases}$$

$$G_{best_{kd}}(t+1) = \begin{cases} X_{kd}(t) & \text{if } F(X_{kd}(t+1)) \text{ is better than } F(G_{best_{kd}}(t)) \\ X_{kd}(t+1) & \text{otherwise} \end{cases}$$

The solutions are updated until finding the optimal solution using equations (10) and (11). Once the optimal solution is attained, the algorithm will be terminated.

IV. ENERGY EFFICIENT CLUSTERING AND JOB SCHEDULING

A. Overview

In this paper, energy efficient clustering and job scheduling techniques are presented to improve the performance of the mobile grid computing system. As shown in figure 1, a user requests to the grid server for processing a number of jobs or tasks. The grid server holds the information of all mobile grid networks. After receiving the job_request, it verifies the available resource providers denoted as RPs (Mobile devices) and forms the clusters by selecting the cluster head (CH). Among the RPs, the RP with maximum capacity is selected as CH and the CH joins the cluster members those are in its

communication range. Then the job_request is forwarded to the selected CH. The CH schedules the jobs to the suitable heterogeneous RPs using the proposed Oppositional Particle Swarm Optimization (OPSO) based job scheduler.

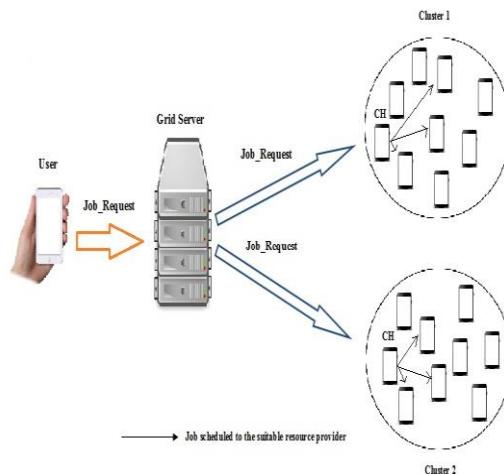


Figure 1: The proposed Job scheduling process

B. Cluster formation

In the mobile grid environment, a mobile device can play both roles such as resource provider and resource consumer. These mobile grid resources are heterogeneous i.e., they have different software and hardware configurations. If a user wishes to process N number of jobs represented as $T_N = \{T_1, T_2, \dots, T_N\}$, he/she forwards the job_request to the grid server. The grid server has the information of all mobile grids in the network. After receiving the request from the user, the grid server initiates the cluster formation among the available RPs in the network. Initially, the grid server chooses cluster head (CH) from the available RPs. Based on the capacity of the RPs, CH is selected and this capacity is calculated as follows:

$$Cap_i = (M_i * PC_i) + B_i \quad (8)$$

Where, $Capi$ denotes the capacity of i th RP, Mi denotes the amount of memory of i th RP, PCi processor count of the i th RP and Bi represents the bandwidth of i th RP.

The RP with maximum capacity [Max (Cap)] is selected as CH. The selected CH joins the nearby RPs those are communicating with the CH with low latency communication link. That means, at first, the CH forwards its current status information to the nearby RPs or mobile devices. After receiving this status information, the neighbor RPs forward the responses to the CH. Then the CH chooses the RPs with low latency communication link. By joining these RPs, the individual cluster is formed. After the formation of the cluster, the grid server forwards the job_request to the selected CHs in the network.

Algorithm 1: Procedure for cluster formation

A user forwards the job_request to the grid server which knows the information of all RPs.

The grid server calculates the capacity of each RP in the network using equation (8).

The RP with maximum capacity is selected as CH.

The CH forms a cluster by



joining the RPs with low latency communication link.

After the cluster formation, the grid server forwards the job_request to the selected CHs.

C. Oppositional Particle Swarm Optimization (OPSO) based Job Scheduling

The numbers of jobs which are to be processed are forwarded as job_request from the user to the grid server. The grid server forwards the job_request to the CHs in the network. The CH manages all non-CH members i.e., other RPs in a cluster. After receiving the job_request, the CH schedules the job to the suitable RPs in the cluster. For efficient job scheduling, Oppositional Particle Swarm Optimization (OPSO) algorithm is presented in this paper. The basic of PSO algorithm is presented in section 3. To improve the optimization process or select the global optimal solution, opposite based learning (OBL) method is integrated with the PSO algorithm. Using this method, the opposite solution for each solution is generated. This proposed OPSO algorithm schedules the N number of jobs (TN) to the M number of suitable RPs (RPM). The following phases describe the proposed OPSO algorithm.

Initialization: The candidate solutions or particles are initialized with a d dimensional vector. In this approach, the mapping between jobs or jobs and RPs is considered as the candidate solutions. It can be initialized as follows,

$$X_k(t) = \begin{Bmatrix} T_{11} & T_{21} & \dots & T_{N1} \\ T_{12} & T_{22} & \dots & T_{N2} \\ \vdots \\ T_{1M} & T_{2M} & \dots & T_{NM} \end{Bmatrix}_{kd} \quad (9)$$

Where, TNM represents the mapping between the Nth job and Mth RP in the cluster.

Oppositional solution: With the initial solutions X_k , oppositional solutions are calculated using OBL method. The oppositional solutions of X_k are calculated as follows:

$$X'_k = a_i + b_i - X_k \quad (10)$$

Where, X'_k denotes oppositional solution in dth dimensional search space, a_i denotes the lower limit of solution X_k at time t and b_i denotes the upper limit of solution X_k at time t.

Fitness: After initializing the candidate solutions, the fitness value of each solution is evaluated using the equation (11). This fitness function is defined using equations (12), (13), (15) and (16).

$$Fit_k = \min(\eta_1 \times PT_k + \eta_2 \times PC_k + \eta_3 \times E_k + \eta_4 \times RU_k) \quad (11)$$

Where, η_n represents the weight parameters in the range [0, 1]. PT_k represents the processing time of the RP at the kth solution. It can be defined as follows,

$$PT_k = \frac{1}{\max(PT) * \text{number of job}} \sum_{N=1}^{\text{Number of job}} (PT \text{ of corresponding RP} * \text{Size of the job}) \quad (12)$$

PC_k represents the processing cost of the kth solution and it can be defined as follows,

$$PC_k = \sum_{N=1}^{\text{Number of job}} \left[\frac{PT_k \times CT_k}{\text{Number of job}} \right] \quad (13)$$

Where, CT_k represents the job scheduling time to the RPi

and it can be defined as follows,

$$CT_k = \frac{1}{\max(CT) * \text{Number of job}} \sum_{N=1}^{\text{Number of job}} (CT \text{ of corresponding RP} * \text{Size of the job}) \quad (14)$$

E_k represents the energy consumption of the kth solution and it can be defined as follows,

$$E_k = \sum_{N=1}^{\text{Number of job}} (\text{size of job} \times \text{Energy of corresponding RP}) \quad (15)$$

RU_k represents the resource utilization of the kth solution and it can be defined as follows,

$$RU_k = \frac{1}{\text{Number of job}} \sum_{N=1}^{\text{Number of job}} \left[\frac{\text{Capacity of RP} - \text{Size of job}}{\text{Capacity of RP}} \right] \quad (16)$$

The solutions with minimum fitness value are considered as the optimal solutions i.e., the requested jobs are mapped to the suitable RPs. Then the best and worst fitness of the solutions or populations is sorted. Then each solution is updated until finding the optimal solution.

Updation: After calculating the fitness of the solution, the solution is updated based on its position and velocity vector. Until finding the optimal solution, each solution is updated using equations (3) and (4).

Termination: Above phases are continued until finding the optimal solution or routing path from the initialized attacker free paths. Once the optimal solution is attained, the algorithm will be terminated. Figure 2 shows the flowchart of the proposed approach.

Algorithm 2: Procedure for selecting an optimal path using PSO

Input: Mapping between TN jobs and RPM, w, c1, c2, r1, and r2.

Output: Optimal Map for job scheduling

Initialize the candidate solutions or mapping between jobs and Resource providers.

Find the oppositional solutions using OBL method using (10). Calculate fitness for each normal solution and oppositional solution using equation (11).

Sort the best and worst fitness of normal and oppositional solutions.

Update velocity and position of the solution using (3) and (4).

If

$$F(X_{k,d}(t+1)) < F(P_{best,k,d}(t))$$

$$F(X_{k,d}(t+1)) < F(G_{best,d}(t))$$

Then $P_{best,k,d}(t+1) = X_{k,d}(t)$

$$G_{best,d}(t+1) = X_{k,d}(t)$$

Else

$$P_{best,k,d}(t+1) = X_{k,d}(t+1)$$

$$G_{best,d}(t+1) = X_{k,d}(t+1)$$

End

Phases 3-8 are continued until finding the optimal or optimal mapping solution.

Then, the jobs will be processed by the corresponding RPs.



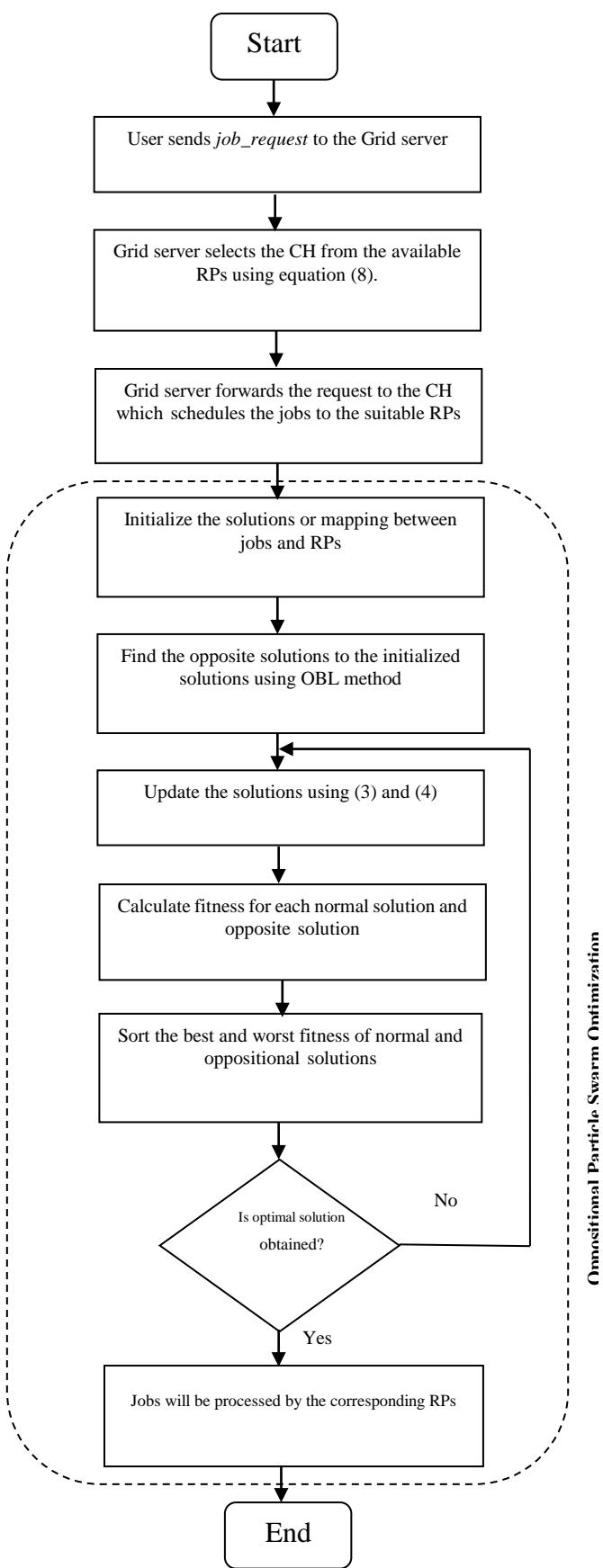


Figure 2: Flowchart of the proposed approach

V. RESULTS AND DISCUSSIONS

Our projected method is simulated with the help of the network simulator NS2. In this simulation, we have measured 100 numbers of mobile nodes and one fixed node called as

Grid server in the simulation arena of 1000×1000 m. These nodes are clustered and cluster head is nominated for each cluster by calculating the capacity of RPs. The whole simulation is accomplished within the simulation time of 50secs. Table 1 displays the simulation parameters of our projected method.

Table 1: Simulation parameters

Parameters	Value
Radio propagation model	Two Ray Ground
Area	1000×1000
MAC	802_11
Routing protocol	AODV
Simulation time	50s
Initial receiving power	0.395W
Initial transmitting power	0.660W
Initial energy	10.1J
Number of mobile nodes	100

A. Performance-based on rates

In this segment, the performance metrics of our projected method are assessed for varying rates 20, 40, 60, 80 and 100kb. Figure 3-8 display the packet delay, packet drop, energy consumption, overhead, delivery ratio and throughput of our projected method. The proposed OPSO algorithm based job scheduling method is compared with job scheduling based on existing algorithms such as Particle Swarm Optimization (PSO) and Genetic Algorithm (GA). Figure 3 and 4 display the packet delay and packet drop of our projected method correspondingly. Using our projected method, nodes in the arena are clustered and on the basis of it, cluster head has been nominated in each cluster. Resource data of each non-CH member is observed using the CH and the CH forwards the processed task to the master server. Because of that packet delay and packet drop of our projected method have been condensed. When the rate upsurges, packet delay, and packet drop also upsurge but compared to the available work those have been abridged to 43% and 30% correspondingly. Energy consumption and overhead of our projected method are shown in figure 5 and 6 correspondingly. Because of our energy efficient job scheduling technique, energy consumption of the job scheduling process based on OPSO is abridged to 20% than existing algorithms. Our projected cluster head selection decreases the overhead of OPSO based job scheduling method to 15% than the existing algorithms. Figure 7 and 8 display the delivery ratio and throughput of our anticipated method respectively. Compared to the existing algorithms, delivery ratio and throughput of OPSO algorithm have been augmented to 67% and 97% respectively.



Published By:

Blue Eyes Intelligence Engineering
& Sciences Publication

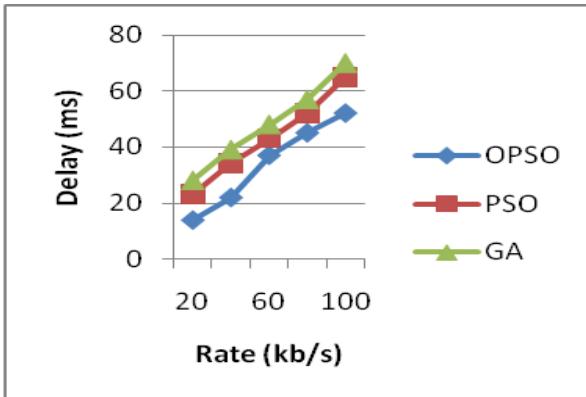


Figure 3: Rate Vs Delay

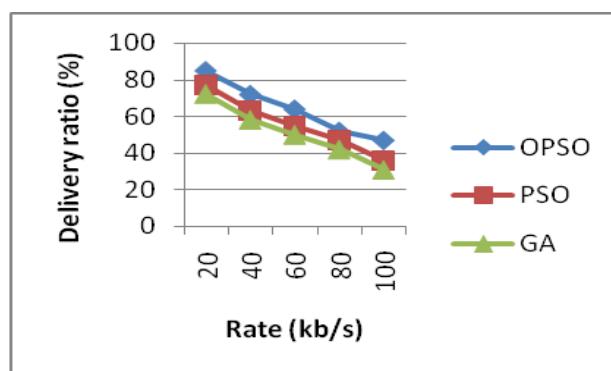


Figure 7: Rate Vs Delivery ratio

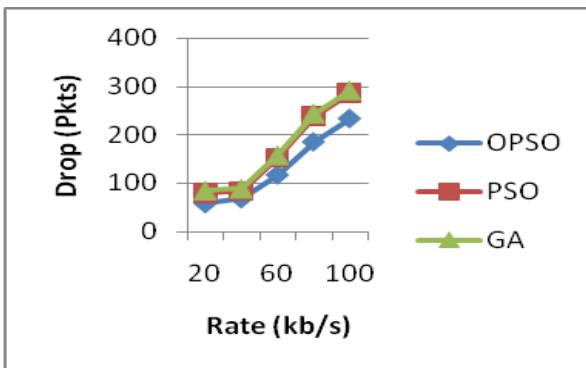


Figure 4: Rate Vs Drop

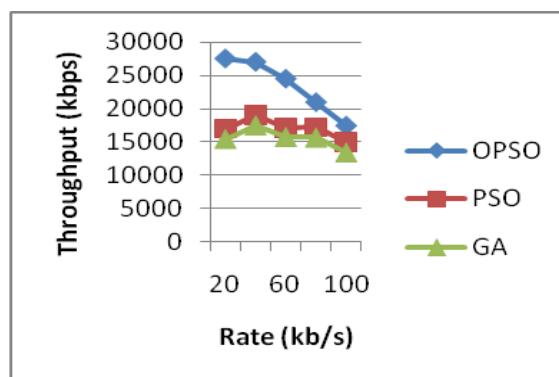


Figure 8: Rate Vs Throughput

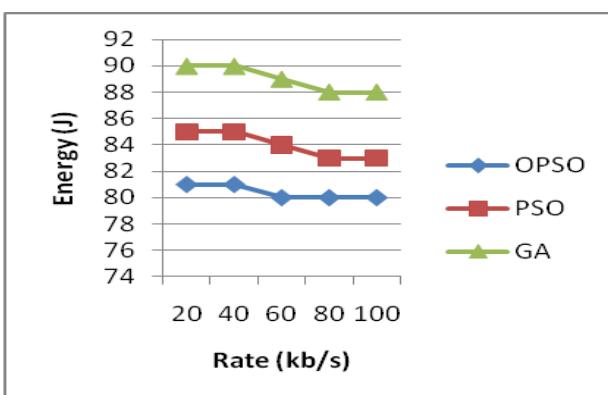


Figure 5: Rate Vs Energy consumption

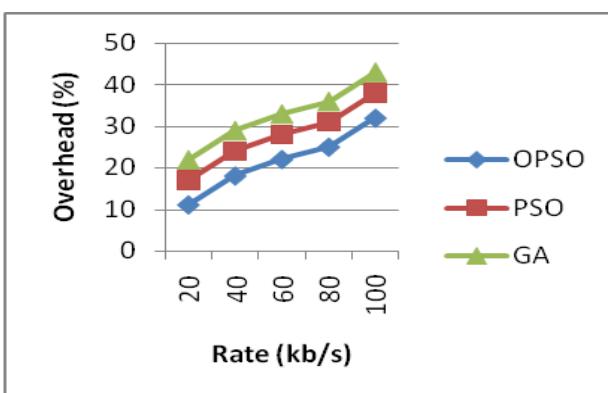


Figure 6: Rate Vs Overhead

B. Performance-based on speed

In this segment, the performance metrics of our projected method are assessed for varying speed 10, 20, 30, 40 and 50m/s. Figure 9-13display the packet delay, packet drop, energy consumption, delivery ratio and throughput of our projected method for varying speed. Packet delay and packet drop of our projected method are displayed in figure 9 and 10 correspondingly for varying speed. Compared to the existing algorithms, packet delay and packet drop of OPSO algorithm are condensed to 19% and 10% correspondingly. Figure 11displays the energy consumption of our projected method for varying speed. When the speed upsurges, energy consumption of our projected approach also upsurges. But energy consumption is abridged to 20% than that of existing algorithms. Delivery ratio and throughput of our projected method are shown in figure 12 and 13correspondingly for varying speed. Associated with the available work, delivery ratio and throughput of our projected method are augmented to 50% and 97% correspondingly.

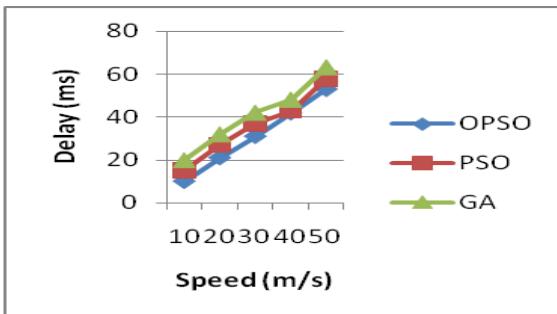


Figure 9: Speed Vs Delay

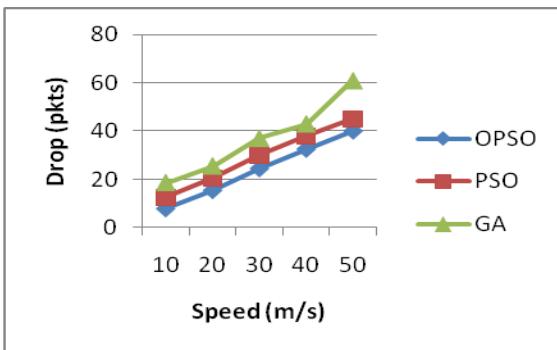


Figure 10: Speed Vs Drop

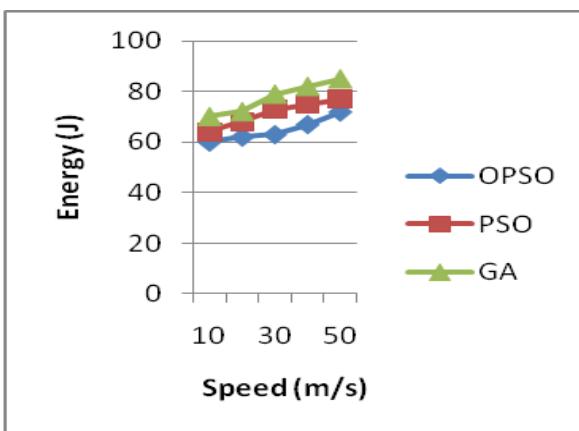


Figure 11: Speed Vs Energy consumption

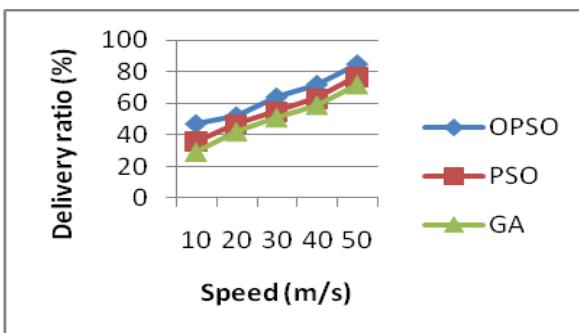


Figure 12: Speed Vs Delivery ratio

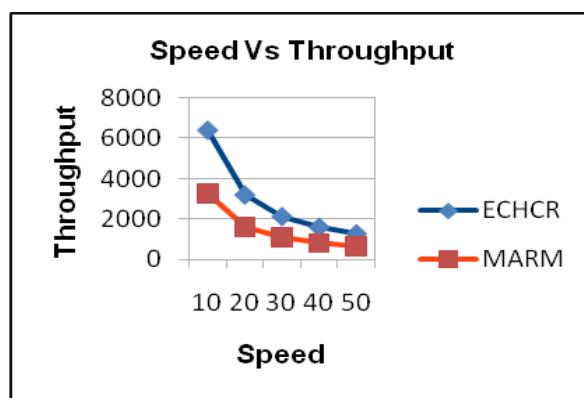


Figure 13: Speed Vs Throughput

VI. CONCLUSION

In this paper, energy efficient clustering and job scheduling have been presented for the mobile grid computing environment. This proposed approach has been implemented in the Network Simulator (NS2). The grid server in the mobile grid has selected CHs based on the maximum capacity of RPs. The selected CHs formed a number of clusters by joining the other RPs with low latency communication link. After receiving the job_request from the user, the grid server has forwarded the request to the CH. The CH has scheduled the jobs to the suitable RPs in the cluster using the proposed OPSO algorithm. The performance of the proposed job scheduling approach has been evaluated in terms of energy efficiency and overhead. Simulation results showed that the proposed OPSO based job scheduling outperforms the job scheduling based on existing algorithms such as PSO and GA.

REFERENCES

- Rosado, David G. et al. "Systematic Design Of Secure Mobile Grid Systems". *Journal of Network and Computer Applications* 34.4 (2011): 1168-1183. Web.
- Behera, Itishree and Chita RanjanTripathy. "Performance Modelling And Analysis Of Mobile Grid Computing Systems". *International Journal of Grid and Utility Computing* 5.1 (2014): 11. Web.
- Goel, Shipra. "Grid Computing Integrated With Mobile Computing Wireless Devices". *International Journal of Mobile Network Communications & Telematics* 1.2 (2011): 39-48. Web.
- Vaithiya, S. Stephen. "Resource Availability Prediction Using Semi-Markov Model In Mobile Grid Environment". *International Journal of Grid and Utility Computing* 7.4 (2016): 285. Web.
- Peyvandi, Soodeh, Rohiza Ahmad, and M. NordinZakaria. "Availability Based Scoring Model For Resource Grouping In Desktop Grid Computing". *Indian Journal of Science and Technology* 8.30 (2015): n. pag. Web.
- Pierre, Samuel. "Mobile Computing And Ubiquitous Networking: Concepts, Technologies And Challenges". *Telematics and Informatics* 18.2-3 (2001): 109-131. Web.
- Themozhi, S., A. Tamilarasi, and K. Thangavel. "Fuzzy Based Job Scheduling For Hierarchical Resource Allocation In Mobile Grid Environment". *International Journal of Soft Computing* 7.3 (2012): 97-103. Web.
- Chunlin, Li and Li Layuan. "Utility-Based Approach For Utilising Services Of Mobile Devices In Mobile Grid". *International Journal of Ad Hoc and Ubiquitous Computing* 7.4 (2011): 235. Web.
- Shrivastava, L., G. S. Tomar, and S. S. Bhadauria. "Performance

- Evaluation Of Reactive Routing In Mobile Grid Environment". International Journal of Grid and High Performance Computing 3.3 (2011): 45-53. Web.
10. C. Li and L. Li, "Energy constrained resource allocation optimization for mobile grids," Journal of Parallel and Distributed Computing, vol. 70, no. 3, pp. 245–258, 2010.
11. Y.-S. Chang and P.-C. Shih, "A resource-awareness information extraction architecture on mobile grid environment," Journal of Network and Computer Applications, vol. 33, no. 6, pp. 682–695, 2010.
12. Du, Li-juan and Zhi-wen Cao. "MARM: Mobility-Aware Routing Mechanism In Mobile Grid". 2011 7th International Conference on Wireless Communications, Networking and Mobile Computing (2011): n. pag. Web. 9 Mar. 2017.
13. Chunlin, Li and Li Layuan. "Exploiting Composition Of Mobile Devices For Maximizing User Qos Under Energy Constraints In Mobile Grid". Information Sciences 279 (2014): 654-670. Web.
14. Souri, Alireza and NimaJafari Navimipour. "Behavioral Modeling And Formal Verification Of A Resource Discovery Approach In Grid Computing". Expert Systems with Applications 41.8 (2014): 3831-3849. Web.
15. Jin, Yingwei et al. "A Mobile Agent-Based Routing Model For Grid Computing". The Journal of Supercomputing 63.2 (2011): 431-442. Web.
16. Chunlin, Li and Li Layuan. "An Economics-Based Negotiation Scheme among Mobile Devices In Mobile Grid". Computer Standards & Interfaces 33.3 (2011): 220-231. Web.
17. Yeh, Wei-Chang and Shang-Chia Wei. "Economic-Based Resource Allocation For Reliable Grid-Computing Service Based On Grid Bank". Future Generation Computer Systems 28.7 (2012): 989-1002. Web.

AUTHORS PROFILE

S.Kavitha Bharathi, Assistant Professor, Department of Computer Applications

M.Dhavamani, Assistant Professor (Sr.G),
Department of Mathematics, Kongu Engineering College, Perundurai – 638 060.