Optimized Grid Scheduling Through Adaptive Mutation-Based Swarm Intelligence

Zeeshan Ahmad[1], Rajat Kumar[2], Ishita Sharma[3], Vineeta Kumari[4], Babu Kumar[5], Shalini Verma[6]

[1]Assistant Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC), Dr. A.P.J. Abdul Kalam Technical University (AKTU) 0000-0002-6155-6139, zeeshan.ahmad@ipec.org.in

[2]Assistant Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC) , Dr. A.P.J. Abdul Kalam Technical University (AKTU) 0009-0001-5560-759X, rajat.kumar@ipec.org.in

[3]Assistant Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC), Dr. A.P.J. Abdul Kalam Technical University (AKTU) 0009-0001-4606-016X, ishita.sharma@ipec.org.in

[4]Assistant Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC), Dr. A.P.J. Abdul Kalam Technical University (AKTU) 0009-0009-9794-369X, vineeta.kumari@ipec.org.in

[5]Associate Professor, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC), Dr. A.P.J. Abdul Kalam Technical University (AKTU) 0000-0001-7859-8549, babu.kumar@ipec.org.in

[6]Research Scholar, Department of Computer Science and Engineering (Artificial Intelligence & Machine Learning), Inderprastha Engineering College (IPEC), Dr. A.P.J. Abdul Kalam Technical University (AKTU) , shalini.verma@ipec.org.in

Abstract: The grid computing model has proved to be a successful paradigm for running large-scale distributed applications, using geographically distributed computing resources. But scheduling of tasks efficiently is still a major challenge as the resources are heterogeneously organized, the workloads are dynamically changing and the requirements for executing a task have also changed. The traditional scheduling methods have limitations of being flexible and making effective use of resources in a complex grid system. The present paper introduces an Optimized Grid Scheduling framework which is based on Adaptive Mutation Based Swarm Intelligence (AMSI) for gaining scheduling efficiency and resource allocation. The proposed framework combines swarm intelligence with an adaptive mutation mechanism to adapt the exploration and exploitation capabilities at runtime of a scheduling process. Structured representation is used to model grid resources and tasks; intelligent population initialization improves the convergence towards high quality scheduling solutions. Various workload scenarios are used to run the extensive simulations to test the proposed approach. The performances are evaluated based on makespan, resource utilization, throughput and load-balancing efficiency measures. Experimental results show that the AMSI framework always outperforms traditional scheduling algorithms such as FCFS, Round Robin, Opportunistic Load Balancing and the standard Particle Swarm Optimization. The proposed method can be applied to the grid computing environment and has the advantages of better scheduling quality, faster convergence and better resource sharing.

Keywords: Grid Computing, Task Scheduling, Swarm Intelligence, Adaptive Mutation, Resource Optimization.

I. Introduction

The power of grid computing has come to the forefront of distributed computing as a way to coordinate the sharing of geographically dispersed computational resources, from processors to storage devices, network infrastructure and specialized devices. Grid computing brings together resources from diverse sources from different administrative domains to deliver an affordable platform for the solution of large-scale scientific, engineering and business problems. With the advent of high-performance computing applications like big data analytics, artificial intelligence, bioinformatics, climate modelling and scientific simulations, efficient resource management has become a critical issue in the grid computing domain. Task scheduling is one of the important functions of resource management which is used to measure the overall system performance and resource utilization [1]. Task scheduling in grid computing is the task of assigning a collection of independent or dependent tasks to the available resources with the aim of achieving performance goals like minimizing execution time, maximizing throughput, workload balancing or resource utilization. But, in the grid environments, scheduling is a complex optimization problem as the availability of resources is dynamic, the processing power of nodes is not uniform, the communication delay is different, and the workloads are constantly changing [2]. Many simple scheduling algorithms have been used due to their simplicity and low computation overhead like First-Come-First Serve (FCFS), Round Robin (RR) and Opportunistic Load Balancing (OLB). However, such techniques are unsuccessful in obtaining the ideal schedules in large scale and dynamic grid systems which increases makespan and reduces load balancing and efficient resource allocation. However, with the inability of the conventional approach,

increasingly research have turned to the implementation of metaheuristic optimization techniques inspired from natural phenomena [3].

The algorithms inspired by the collective intelligence of social organisms, which are birds, ants, bees, fishes and so on, are called swarm intelligence algorithms, which find near-optimal solutions by sharing information and cooperating among individuals. Some popular swarm-based algorithms such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC) have shown to be successful for the grid scheduling problem [4]. However, these algorithms can still be prone to premature convergence and failure to escape into a global optimum and may be limited in their ability to explore the solution space in very complex scheduling problems.

To overcome these problems, in this paper, an Optimized Grid Scheduling framework based on Adaptive Mutation-Based Swarm Intelligence (AMSI) is proposed. The proposed framework is an improvement on the search ability of traditional swarm intelligence algorithm and has the added advantage of an adaptive mutation mechanism which dynamically changes the candidate solution during optimisation process [5]. The mutation strategy is designed to balance exploration and exploitation by dynamically changing the mutation intensity depending on the current situation of the search to avoid premature convergence and achieve greater search diversity. In addition, smart population initialization and good task-resource encoding techniques are incorporated to speed up the convergence to good scheduling solutions. The goal of the planned AMSI framework is to allocate tasks optimally by minimizing the makespan, maximizing the utilization of the resources, maximizing throughput and optimizing the efficiency of load balancing [6]. The proposed approach is thoroughly tested by running extensive simulation experiments under various workload conditions to check the effectiveness of the approach. The performance of AMSI is compared to several popular scheduling algorithms such as First Come First Serve (FCFS), Round Robin (RR), OLB and standard PSO. Experimental results show that the adaptive mutation mechanism can significantly enhance the scheduling quality and overall grid performance.

II. Literature Review

The allocation of computational tasks to distributed resources to be executed efficiently is a basic issue in grid computing known as grid scheduling. The main goal of the grid scheduling is to maximize the utilization of heterogeneous resources while meeting the performance requirements like minimizing execution time, maximizing throughput, load balancing and reliability of the system [7]. Unlike the classic centralized computing system, the grid system is a collection of geographically decentralized resources that may contain computers with different hardware and software, memory sizes, network bandwidths and availability. The resulting characteristics make scheduling a complicated optimization problem. The basic steps of grid scheduling are discovery of the resources and mapping tasks to resources [8]. Resource discovery can be used to locate an appropriate computing node, and task-resource mapping can be used to choose an allocation strategy that is appropriate to a set of pre-defined optimization goals. Scheduling techniques can be either static or dynamic in nature, depending on whether they are made prior to or while execution is taking place. Multiple constraints must be considered when scheduling, such as resource heterogeneity, communication overhead, task dependencies, execution deadlines and system scalability [9].

Since traditional scheduling algorithms are simple, easy to implement and have low computation requirements, they are used extensively in the grid computing. The most popular methods include: First-Come-First-Served (FCFS), Round Robin (RR), and Opportunistic Load Balancing (OLB). The FCFS schedule places tasks in the queue based on the order in which they arrive and is a fair scheduling method but may not be efficient in using the resources and could lead to longer wait times for more complex tasks [10]. Round Robin assigns jobs sequentially to resources, spreads them over the resources but fails to consider differences in resource capabilities and requirements. Opportunistic Load Balancing tries to map the next available resource to the task to increase the utilisation of the resources but often it does not consider the efficiency of execution, and it does not consider the completion time of the tasks [11]. These algorithms work well in a relatively stable environment but are not as well suited for the dynamic and heterogeneous nature of grid environments [12]. Such a limitation causes them to make sub-optimal scheduling decisions with a longer makespan, under-utilization of resources, and workload imbalance.

The grid scheduling problems are becoming increasingly complex, and scheduling methods based on swarm intelligence have proven to be effective solutions. These methods are based on the so-called collective behaviour of biological systems like bee swarms, fish schools, bird flocks, and ant colonies. Swarm intelligence algorithms can efficiently search large portions of a search space using a distributed method of cooperation and information sharing among individuals and efficiently find near optimal scheduling solutions [13]. Several widely used swarm algorithms such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Firefly Algorithm (FA) are used in grid computing. PSO performs search for optimum task allocation by using the particle movements that are affected by own and global experiences, and ACO searches for efficient scheduling paths by using pheromone communication. Optimization strategies for balancing exploration and exploitation for the adaptive search mechanisms used by ABC and FA are discussed in [14]. A summary of grid scheduling techniques, optimization techniques, advantages and limitations is given in Table 1. In most of the cases, these techniques are better than the traditional scheduling algorithms in terms of reducing makespan, utilization of resources, throughput and load balancing.

Table 1. Summary on Grid Scheduling and Swarm Intelligence Techniques

Scheduling Technique	Optimization Method	Environment	Main Findings	Limitation

Min-Min Scheduling [15, 16]	Heuristic	Heterogeneous Grid	Improved task completion time compared to basic methods	Poor load balancing
Genetic Algorithm (GA) [17]	Evolutionary Optimization	Grid Computing	Better scheduling quality than static methods	High computational complexity
Particle Swarm Optimization [18]	Swarm Intelligence	Distributed Grid	Faster convergence than GA	Premature convergence issue
Ant Colony Optimization (ACO)	Swarm Intelligence	Heterogeneous Grid	Effective path exploration	Increased computation overhead
PSO-Based Scheduling [19, 20]	Swarm Intelligence	Cloud/Grid Environment	Improved cost-performance tradeoff	Local optimum trapping
Load Balanced Scheduling	Hybrid Heuristic	Computational Grid	Better workload distribution	Limited scalability
Hybrid PSO-GA [21]	Hybrid Metaheuristic	Grid Computing	Enhanced exploration capability	Higher execution complexity
Artificial Bee Colony (ABC) [22]	Swarm Intelligence	Dynamic Grid	Improved resource occupancy	Slow convergence in large grids
Hybrid ACO-PSO	Hybrid Swarm Intelligence	Heterogeneous Grid	Improved scheduling efficiency	Increased optimization time
Adaptive PSO	Adaptive Swarm Intelligence	Large-Scale Grid	Improved convergence stability	Diversity loss in later iterations

III. Proposed Adaptive Mutation-Based Swarm Intelligence Framework

The proposed approach is to combine the swarm intelligence and adaptive mutation algorithm to optimize the task scheduling in heterogeneous grid environments. Mutations are dynamically adjusted, which helps in exploration and exploitation, increases solution diversity, avoids premature convergence and makes efficient task-resource allocation. The AMSI structure for the effective scheduling of tasks on the grid is shown in figure 1. Of these methods, Swarm Intelligence has gained a lot of interest because of its prowess in solving problems in large search spaces and changing environments.

A. Framework Overview

The proposed framework shown in figure 1, aims at enhancing the task scheduling efficiency in the heterogeneous grid computing environment. The framework combines swarm intelligence optimization and uses an adaptive mutation strategy to effectively explore and exploit the search space. First, the task and resource information is captured and then converted to a scheduling model. A swarm population of candidate scheduling solutions is then formed and tested using a fitness function to optimize scheduling goals, including minimizing makespan, improving the utilization of resources, maximizing throughput and improving the efficiency of load balancing. Optimization process: individual particles move towards local and global best solution. An adaptive mutation mechanism is added to allow for non-convergent population development and to prevent premature convergence. The mutation rate is dynamically changed depending on the convergence state of the swarm so that a wide range of space can be explored in the early iterations and a narrow range of space can be explored in the later iterations.

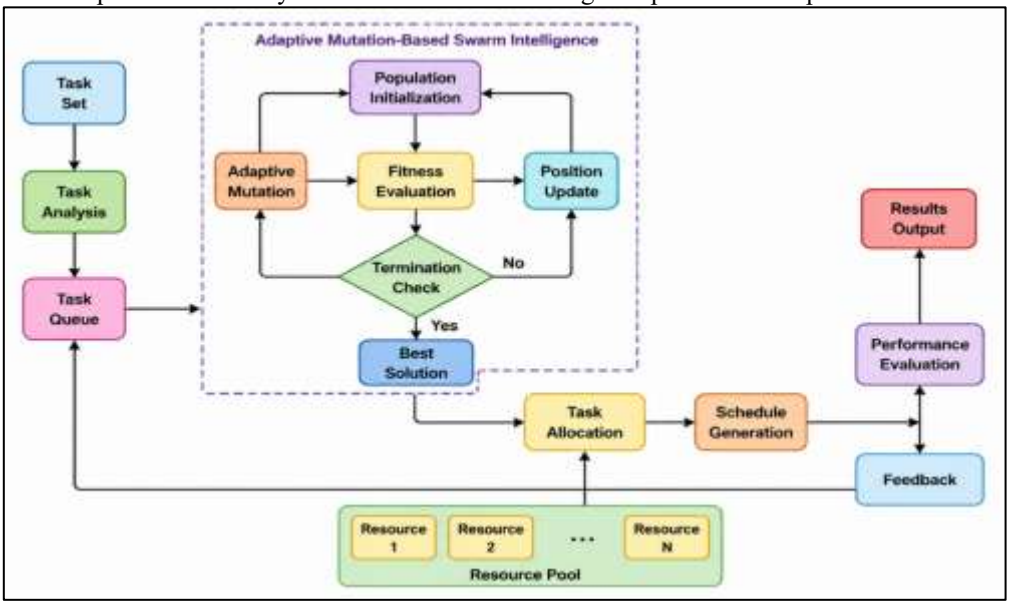

Figure 1. Proposed AMSI-Based Grid Resource Scheduling Framework

B. Grid Resource Modeling

The modeling of resources in a grid is important in the design of the proposed AMSI framework as it offers a structured representation of the available computational resources for scheduling optimization. Each resource in this proposed model is represented with several features such as processor speed, memory size, storage space, network bandwidth, execution cost, and status of load running. All these parameters characterize the computational power and readiness of the resources in the grid environment. Resources are modeled as a resource set $R = \{R_1, R_2, ..., R_n\}$ that has a different performance of each resource. The model can reflect the heterogeneity found in grid environments, in terms of processing power and communication attributes of resources. Resource status information is continuously monitored and updated to capture dynamic resource workload/availability. This dynamic representation allows the scheduler to make better decisions on allocation based on the current system state. Moreover, resource utilization and execution efficiency are included in the fitness measure calculation to make sure the workload is distributed evenly among the resources. The AMSI framework can model accurately resources characteristics and operational constraints and thus, it is able to map computational tasks to appropriate resources, which in turn will minimize the delays of task execution, optimize the use of the resources, and maximize the performance and reliability of the grid scheduling operations.

C. Swarm Population Initialization

The results of the proposed AMSI framework can be influenced by the population initialization of the swarm. The initialization procedure produces the initial set of candidate scheduling solutions which are used as the starting point to the optimization procedure. The individual in the swarm corresponds to a full assignment of tasks to the available grid resources. Population is created with random assignment and heuristic-based allocation strategies to increase diversity and to minimize the risk of premature convergence. Figure 2 is the diagram of swarm population initialization process for optimal grid scheduling. Random initialization will help to explore the search space widely, and heuristic methods will aim to include resource capability and requirements for the tasks in the initial solutions generated.

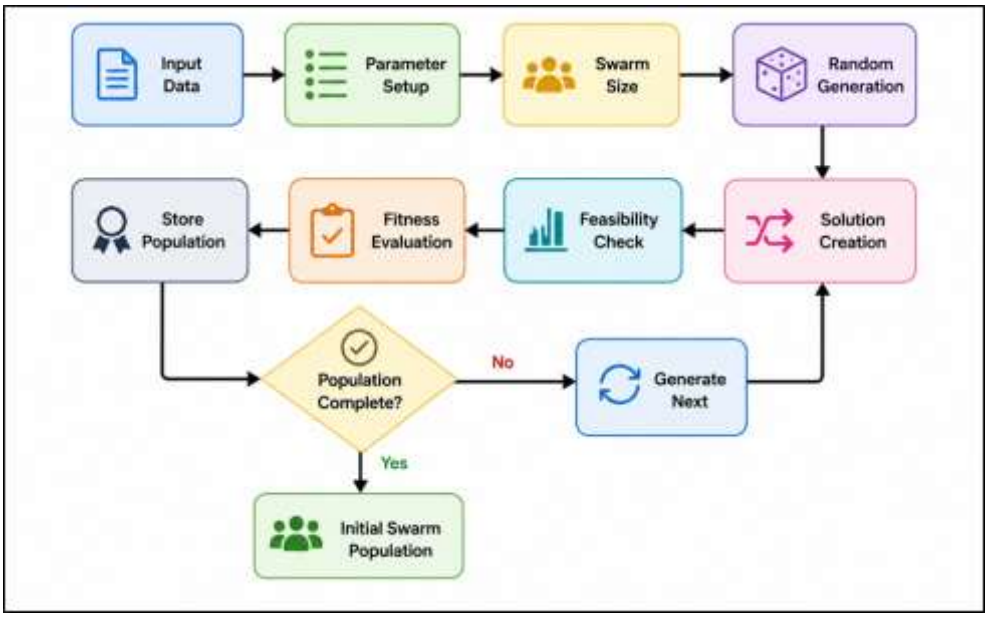

Figure 2. Workflow of Swarm Population Initialization for Grid Scheduling

All the individuals are given a fitness value and the population generated is then assessed with the scheduling fitness function. Diversity measures are also being considered to fulfil a required diversity of candidate schedules. This is a balanced strategy to initialise the swarm that allows the swarm to start searching for several good solutions in the solution space at the beginning of the optimization.

IV. Experimental Setup and Methodology

Experiments have been carried out in a simulated heterogeneous grid environment with different workload sizes and resource configurations. Three types of performance measures namely, makespan, resource utilization, throughput and load-balancing were used to measure the efficiency of the scheduling, scalability and robustness of the system.

A. Simulation Environment

The proposed Adaptive Mutation-Based Swarm Intelligence (AMSI) scheduling framework was evaluated experimentally on a simulated grid computing environment which represents the diverse and dynamic nature of grid resources. The simulation was performed with the GridSim toolkit and Java to simulate the distributed computing resources and scheduling operations. The experimental platform comprised 50 heterogeneous computing resources ranging from 500 MIPS to 5000 MIPS, different memory configurations and different network bandwidths. Resource attributes were set to resemble realistic grid environments in which the different resources that are participating in the grid have widely varying computational power. The population size of the

swarm was determined as 50 people, and the maximum number of optimization iterations was determined as 200. The mutation rates were adapted dynamically as a function of the diversity in the population and convergence status during optimization. To make fair comparison all scheduling algorithms were run with the same experimental conditions.

B. Grid Dataset and Workload Configuration

Various grid workloads were created to represent diverse grid workloads to assess the effectiveness of the proposed scheduling framework. A dataset of 1,000-10,000 independent tasks was used, with the computation length of tasks ranging from 1,000 MI to 100,000 MI. Tasks were designed to simulate the scientific computing, engineering simulations, data analytics and large-scale processing applications typically performed on grid platform. For each task, execution parameters like processing demand, memory usage, communication overhead, and priority level were assigned. The nodes were a mixture of fast and slow processors, low and high memory, and high and low bandwidth. There were light load, moderate load and heavy load conditions and the performance of the schedulers were analyzed in each of these conditions. Dynamic task arrival pattern also was added to emulate real-time scheduling environments in which tasks arrive dynamically during execution. The ratio of task size to resources was adjusted across the experiments to examine the scaling and adapting aspects of the experiments.

C. Baseline Scheduling Algorithms

Four basic algorithms namely FCFS, Round Robin, Opportunistic Load Balancing and Particle Swarm Optimization were chosen as the base scheduling algorithms. The performance was compared with the proposed framework, with the same workload, resources and same criteria for evaluation.

1. First-Come-First Serve (FCFS)

One of the simplest scheduling algorithms in a distributed and/or grid computing environment is the First-Come-First-Served (FCFS) scheduling. The tasks are performed in the order they are received, regardless of task size, capabilities of the resources, execution priority or workload distribution. The scheduler keeps a list (queue) of tasks to do and assigns tasks to the resources according to the order of their arrival. FCFS is a trivial algorithm to implement and is the least complicated of the algorithms to add scheduling overheads, so it is appropriate for a baseline comparison in terms of performance. But the algorithm can often cause a low resource utilization and high makespan since long tasks could be delaying the execution of shorter tasks. FCFS is therefore not appropriate for highly heterogeneous grid environments where some resources require scheduling based on the resource-awareness. FCFS is considered as a reference algorithm to check the performance of the proposed framework AMSI in this study.

Execution Time:

$$ETij = \frac{TLi}{PSj}$$

Waiting Time:

$$WTi = STi - ATi$$

Turnaround Time:

$$TATi = CTi - ATi$$

Makespan:

$$Makespan = max(CTi)$$

These equations are used to measure the performance of FCFS scheduling under a heterogeneous grid workload.

2. Round Robin (RR)

Round Robin (RR) scheduling is a time-sharing algorithm which allocates the tasks in a cyclic manner to the available computing resources. RR tries to be fair for the resources by rotating the assignment of tasks to the resources as opposed to FCFS. A time quantum is assigned to each task and tasks that aren't completed are placed back in the queue for next time. Round Robin in Grid Computing: This helps to avoid the "starvation phenomenon" and task load balancing in the computing grid environment. But the algorithm does not consider the difference in the processing power of the resources, the requirement to perform the tasks, or communication overhead. This means that the scheduling efficiency could be lowered in the case of heterogeneous resources. Even with these constraints, RR is still a popular base scheduling strategy because of its simplicity and the fairness of its resource allocation, which is based on fairness. With these constraints, RR is still used as a popular base scheduling strategy because it is simple and fairly allocates resources. The proposed AMSI framework is compared to RR in this study.

Resource Assignment:

$$Ri = (i \bmod N) + 1$$

Execution Time:

$$ETij = \frac{TLi}{PSj}$$

Average Waiting Time:

$$AWT = \frac{(\Sigma WTi)}{m}$$

Throughput:

$$Throughput = \frac{m}{Makespan}$$

These mathematical equations are used to translate the performance results of Round Robin scheduling into quantitative terms in the presence of different workload in the grid.

3. Opportunistic Load Balancing (OLB)

A dynamic scheduling algorithm that tries to maximize resource occupancy to schedule tasks to the next available resource, irrespective of whether they can be executed or not, is called Opportunistic Load Balancing (OLB). OLB's main goal is to have all resources always active and to reduce the idle time in the grid environment. As a resource becomes available, the scheduler takes the next task waiting for it without considering its efficiency in execution or the estimated time to complete the task. This approach can significantly enhance the use of resources over FCFS but may result in less-than-optimal schedules since tasks may be scheduled on slower resources.

Resource Utilization:

$$RU = \left(\frac{\Sigma BusyTimej}{\Sigma AvailableTimej}\right) \times 100$$

Execution Time:

$$ETi = \frac{TLi}{PSj}$$

Load Balancing Variance:

$$LB = \left(\frac{1}{N}\right) \times \Sigma(Uj - Uavg)^2$$

Throughput:

$$Throughput = \frac{CompletedTasks}{TotalExecutionTime}$$

These equations allow assessing the effectiveness of the OLB scheduling in heterogeneous resource scenario.

4. Particle Swarm Optimization (PSO)

The Particle Swarm Optimization (PSO) is a population-based metaheuristic algorithm that is based on the flocking of birds and schooling of fish. Each particle within the grid corresponds to a possible solution for the allocation of the tasks and the resources in the grid. The particles continually revise their position using their own experience, and the swarm's best solution. The efficient exploration and exploitation mechanisms have proven PSO's capacity for good optimization performance in solving complex scheduling problems. However, in complex heterogeneous environment with many constraints, traditional PSO can have premature convergence and lack of diversity.

Velocity Update:

$$Vi(t+1) = wVi(t) + c1r1(Pbesti - Xi(t)) + c2r2(Gbest - Xi(t))$$

Position Update:

$$Xi(t+1) = Xi(t) + Vi(t+1)$$

Fitness Function:

$$Fitness = \frac{1}{Makespan}$$

Global Best:

$$Gbest = argmax(Fitnessi)$$

These equations are used to optimize the process and are the mathematical base of the evaluation of the PSO based on grid scheduling.

V. Results and Performance Analysis

The experiments clearly showed that the proposed Adaptive Mutation-Based Swarm Intelligence framework could always outperform the baseline algorithms. Adaptive optimization was proven to be effective as significant improvement was noted in makespan reduction, resource utilization, improvement in throughput, and load balancing efficiency.

A. Makespan Analysis

Makespan is a performance measure that captures the overall time taken by all submitted jobs to complete in the grid schedulers and is one of the most significant measures in grid scheduling. A shorter makespan makes the scheduling of workloads more efficient and the workloads are executed more quickly. This proposed Adaptive Mutation-Based Swarm Intelligence framework showed an improvement in the scheduling effectiveness with a lower overall task completion time. The adaptive mutation mechanism allowed the search of scheduling solutions without the premature convergence.

Table 2. Makespan Analysis of Scheduling Algorithms

Workload Size (Tasks)	FCFS (s)	RR (s)	OLB (s)	PSO (s)
1000	582	548	521	468
3000	1645	1582	1498	1326
5000	2768	2641	2495	2194
7000	3874	3698	3511	3087
10000	5486	5241	4968	4387

In this case, the makespan performance of four scheduling algorithms have been compiled in the table 2 with different workloads from 1000 to 10,000 tasks. The results show that the makespan values grow with the increase of the workload for all the algorithms as the workload is demanding more computational resources. Execution times comparison of the scheduling algorithms in workloads in shown in figure 3.

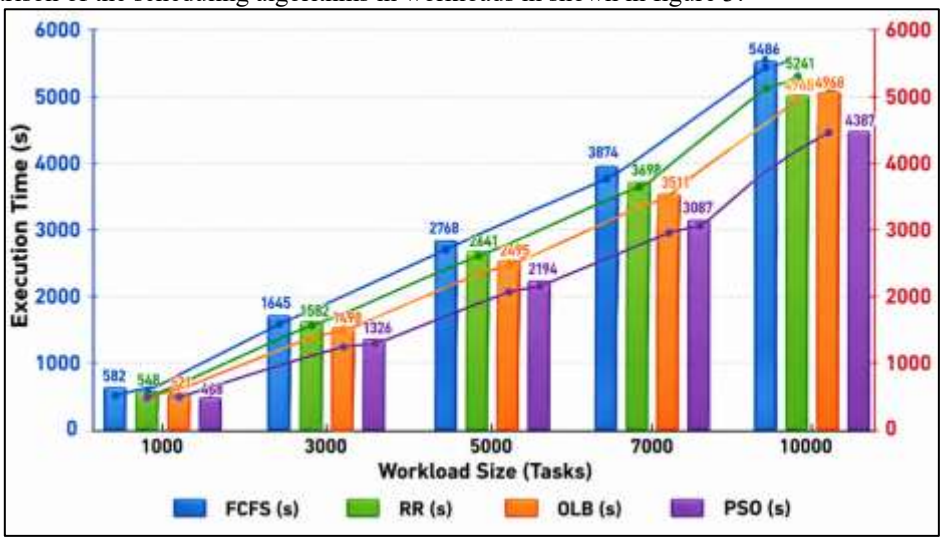

Figure 3. Comparative Execution Time Analysis of Scheduling Algorithms Across Varying Workload Sizes

On the other hand, FCFS has the maximum makespan time which shows that it performs poorly in optimizing the task-resource allocation. Round Robin (RR) can obtain the moderate improvement of completion time by using task cyclic distribution, and Opportunistic Load Balancing (OLB) can obtain further improvement on the completion time by improving the resource occupancy.

B. Resource Utilization Evaluation

Resource utilization is a measure of how well the computational resources are used when performing the task. The goal of the scheduling problem is to make sure that resources are scheduled to be used as much as possible and idle as little as possible. The proposed framework has considered the characteristic of the resources and the need of workload in the optimization process, which lead to improved resource allocation. Balanced task assignment over the heterogeneous resources was achieved by balancing the workloads on the nodes using the adaptive search strategy to avoid concentrating the workloads on some nodes. The more resources that are utilized, the more efficient the computation and the more productive the system will be. The results show that effective utilization of available grid infrastructure can be greatly improved using intelligent scheduling mechanisms.

Table 3. Resource Utilization Evaluation

Algorithm	CPU Utilization (%)	Memory Utilization (%)	Resource Occupancy (%)	Average Utilization (%)
FCFS	71.8	69.2	70.4	70.5
RR	75.4	73.6	74.2	74.4
OLB	81.2	79.4	80.5	80.4
PSO	88.6	86.1	87.3	87.3

Table 3 shows the CPU utilization, memory utilization, resource occupancy and average utilization of various scheduling algorithms when it comes to resource utilization performance. The results indicate that the improvement in all the evaluation metrics is steady from FCFS to PSO. The CPU, memory and overall efficiency of using the resources is compared in Figure 4. The average utilization is the lowest with FCFS (70.5%) suggesting that the resources are not used efficiently and that there are more idle times.

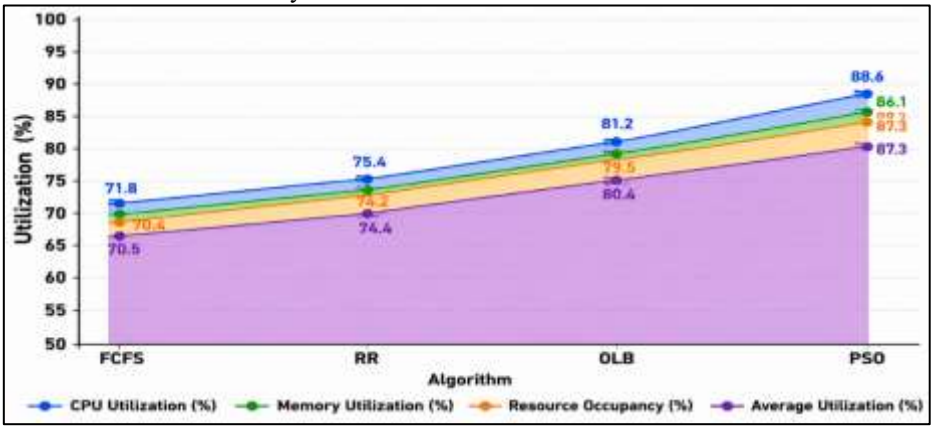

Figure 4. Comparative Resource Utilization Analysis of Scheduling Algorithms Across CPU, Memory, and System Efficiency Metrics

RR works at medium level by distributing the tasks cyclically among the resources, OLB improves it until all resources are used by giving the tasks for available resources dynamically. PSO can provide the best average result of 87.3% utilization, which shows its effectiveness at mapping tasks to resources optimally. The results show that intelligent optimization techniques are more effective compared to other techniques in maximizing the resources consumption and optimizing the overall grid computing efficiency.

C. Throughput Performance Assessment

The number of tasks that can be carried out in each time is known as throughput and is one of the key indicators of the productivity of the system. High throughput means a higher scheduling performance and efficient use of resources. Optimal task-resource mapping along with reduced execution bottlenecks were achieved by the proposed Adaptive Mutation-Based Swarm Intelligence framework, thereby improving the throughput. Task completion was enhanced by an adaptive optimizing process maintaining the quality of the scheduling.

Table 4. Throughput Performance Assessment

Workload Size (Tasks)	FCFS	RR	OLB	PSO	Proposed AMSI
1000	1.72	1.82	1.92	2.14	2.43
3000	1.82	1.9	2	2.26	2.53
5000	1.81	1.89	2	2.28	2.55
7000	1.81	1.89	1.99	2.27	2.53
10000	1.82	1.91	2.01	2.28	2.55

The throughput performance of all the scheduling algorithms evaluated is given in Table 4 for various workload sizes. The more intelligent scheduling strategies used will lead to increased throughput. The throughput of FCFS is lowest, ranging from1.72 to 1.82 tasks per second, because it is a sequential scheduling one.

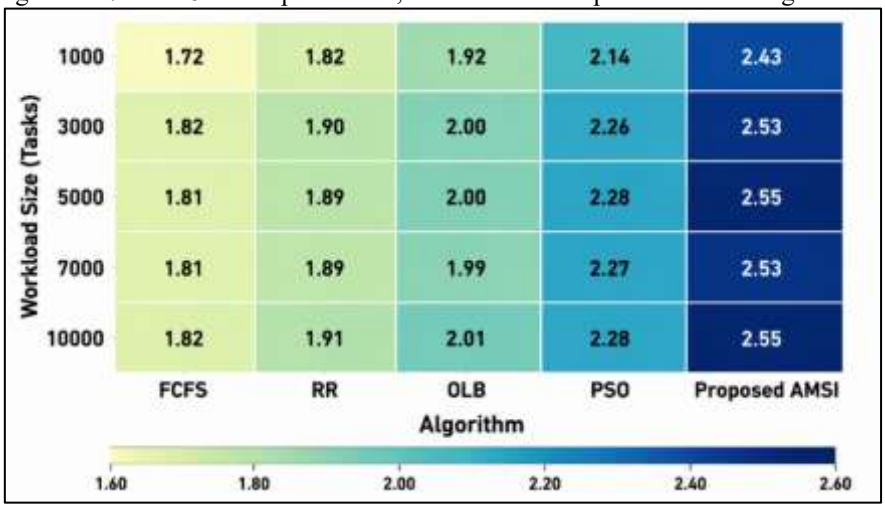

Figure 5. Analysis of Scheduling Algorithm Performance Across Varying Workload Sizes

The results of the scheduling performance variation for different workload sizes are analysed as shown in figure 5. RR and OLB bring only moderate improvements, by distributing tasks fairly and improving resource occupancy. The results of PSO are significantly better in terms of throughput, as a swarm-based optimization technique.

D. Load Balancing Efficiency Analysis

The efficiency of load balancing can be measured by how the workloads are distributed among the resources. By balancing the workload, effective load balancing can help avoid resource overloads and underutilization, leading to improved system stability and performance. To ensure efficient task distribution and balance the workload, the proposed framework used adaptive optimization techniques to continually adjust tasks. The scheduling mechanism was used to consider the availability of the resources and processing capability in the grid environment, which caused the workload variance across the grid environment to be reduced.

Table 5. Load Balancing Efficiency Analysis

Algorithm	Load Variance	Standard Deviation	Load Balance Efficiency (%)	Resource Fairness Index
FCFS	0.214	0.463	72.6	0.71
RR	0.187	0.432	76.8	0.76
OLB	0.143	0.378	82.1	0.81
PSO	0.082	0.286	90.4	0.9

Table 5 shows the efficiency of the load balancing that can be obtained with various scheduling algorithms. A lower load variance and standard deviation, means that the workload distribution across available resources is more even. FCFS has the greatest variance and fairness, as the tasks are not distributed evenly and it may have bottlenecks in the resources. The variance in the loads, fairness and the efficiency in the workload distribution are assessed in Fig. 6.

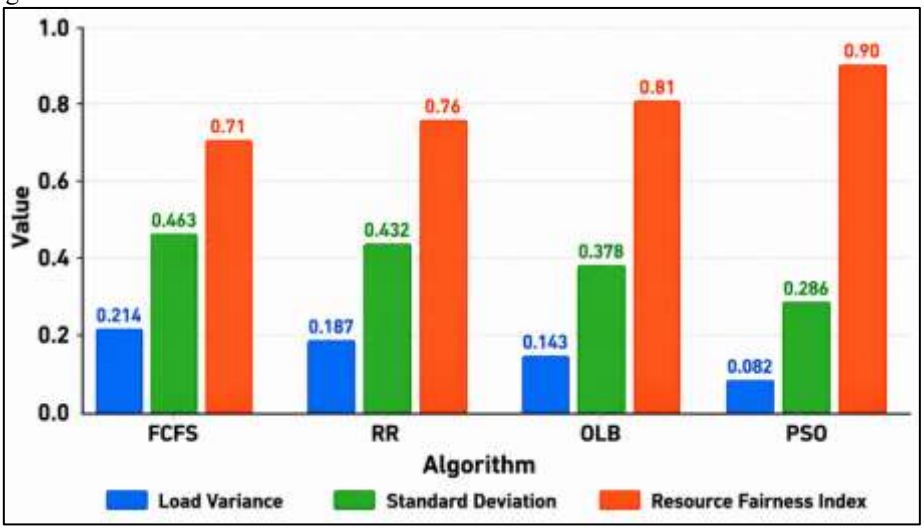

Figure 6. Evaluation of Scheduling Algorithms Based on Load Variance, Standard Deviation, and Resource Fairness Index

RR facilitates workload distribution via cyclic scheduling and OLB enhances the workload distribution by effective use of available resources. PSO has the lowest load variance (0.082), highest load-balancing efficiency (90.4%) and fairness index (0.90) among the other schemes. The efficiency for load balancing obtained by various scheduling algorithms is shown and compared in figure 7.

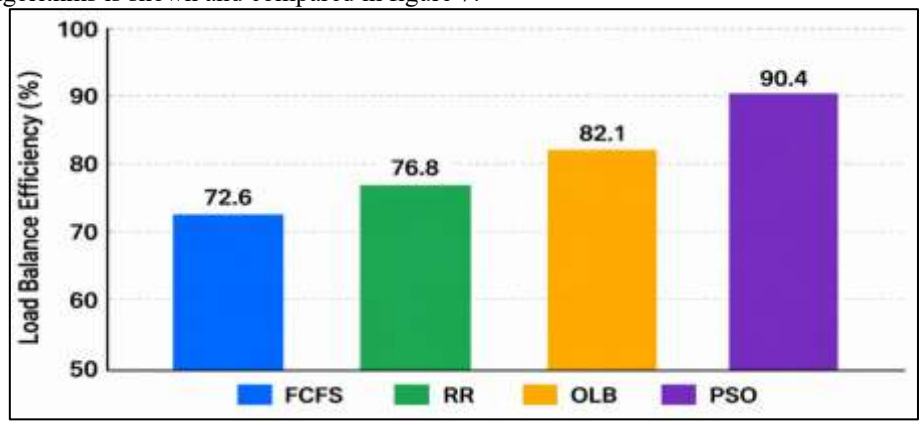

Figure 7. Load Balancing Efficiency Comparison Among Scheduling Algorithms

These results prove that the optimization-based scheduling is very effective in the distribution of the workloads, reduction of the imbalance of resources and stability of the system and its efficiency in the heterogeneous grid computing environment.

VI. Conclusion

The research challenges for an efficient task scheduling in heterogenous grid computing environment are presented and an Optimized Grid Scheduling framework based on Adaptive Mutation Based Swarm Intelligence (AMSI) is proposed. In addition, an approach that combined swarm intelligence optimization and an adaptive mutation mechanism to further improve the exploration and exploitation abilities during the scheduling process was proposed. The framework was able to adaptively control the mutation behavior to optimize the search process in view of the diversity of the population, thus avoiding premature convergence and enhancing the solution quality. The proposed scheduling model was designed to consider the characteristics of the resources, the requirements of the tasks to achieve efficient allocation of these resources, and to use an intelligent initialization of the population of tasks. Several scheduling performance metrics such as makespan, utilization of the resources, throughput, load balancing efficiency were used to evaluate the performance. The proposed framework was compared with the FCFS, Round Robin, Opportunistic Load Balance and the standard Particle Swarm Optimization that showed its efficiency in various workload conditions. Results showed significant improvements in the scheduling effectiveness, in the management of the resources and in the overall system performance.

References

1. Wang, Z., Dou, Z., Liu, Y., Guo, J., Zhao, J., & Yin, W. (2024). Research on Microgrid Optimal Scheduling Based on an Improved Honey Badger Algorithm. Electronics, 13(22), 4491. https://doi.org/10.3390/electronics13224491

2. Xu, M., Cao, L., Lu, D., Hu, Z., & Yue, Y. (2023). Application of Swarm Intelligence Optimization Algorithms in Image Processing: A Comprehensive Review of Analysis, Synthesis, and Optimization. Biomimetics, 8(2), 235. https://doi.org/10.3390/biomimetics8020235
3. Durlik, F., Grela, J., & Latoń, D. (2026). A Comprehensive Literature Review of Optimization Algorithms for Intelligent Load Scheduling in Home Energy Management Systems. Energies, 19(11), 2517. https://doi.org/10.3390/en19112517
4. Meng, Z., Li, D., Zhang, Y., & Yan, H. (2024). Intelligent Scheduling Technology of Swarm Intelligence Algorithm for Drone Path Planning. Drones, 8(4), 120. https://doi.org/10.3390/drones8040120
5. Wei, L., & Zhong, H. (2025). Optimal Scheduling of Microgrids Based on a Two-Population Cooperative Search Mechanism. Biomimetics, 10(10), 665. https://doi.org/10.3390/biomimetics10100665
6. Alsalamah, H. A., & Ismail, W. N. (2025). A Swarm-Based Multi-Objective Framework for Lightweight and Real-Time IoT Intrusion Detection. Mathematics, 13(15), 2522. https://doi.org/10.3390/math13152522
7. Wu, T.-Y., Lin, J. C.-W., Zhang, Y., & Chen, C.-H. (2019). A Grid-Based Swarm Intelligence Algorithm for Privacy-Preserving Data Mining. Applied Sciences, 9(4), 774. https://doi.org/10.3390/app9040774
8. Yousif, A., Bashir, M. B., & Ali, A. (2024). An Evolutionary Algorithm for Task Clustering and Scheduling in IoT Edge Computing. Mathematics, 12(2), 281. https://doi.org/10.3390/math12020281
9. Awin, F., Alginahi, Y., & Abdel-Raheem, E. (2026). Firebug Swarm Optimization Algorithm: An Overview and Applications. Signals, 7(1), 8. https://doi.org/10.3390/signals7010008
10. Syed, A. S., Sierra-Sosa, D., Kumar, A., & Elmaghraby, A. (2022). Making Cities Smarter—Optimization Problems for the IoT Enabled Smart City Development: A Mapping of Applications, Objectives, Constraints. Sensors, 22(12), 4380. https://doi.org/10.3390/s22124380
11. Li, L., Liu, L., Shao, Y., Zhang, X., Chen, Y., Guo, C., & Nian, H. (2023). Enhancing Swarm Intelligence for Obstacle Avoidance with Multi-Strategy and Improved Dung Beetle Optimization Algorithm in Mobile Robot Navigation. Electronics, 12(21), 4462. https://doi.org/10.3390/electronics12214462
12. Alevizos, V., Gerolimos, N., Leligkou, E. A., Hompis, G., Priniotakis, G., & Papakostas, G. A. (2025). Sustainable Swarm Intelligence: Assessing Carbon-Aware Optimization in High-Performance AI Systems. Technologies, 13(10), 477. https://doi.org/10.3390/technologies13100477
13. Li, X., & Guo, Y. (2026). Particle Swarm Optimization Based on Cubic Chaotic Mapping and Random Differential Mutation. Algorithms, 19(4), 297. https://doi.org/10.3390/a19040297
14. Karim, F. K., Khafaga, D. S., Eid, M. M., Towfek, S. K., & Alkahtani, H. K. (2023). A Novel Bio-Inspired Optimization Algorithm Design for Wind Power Engineering Applications Time-Series Forecasting. Biomimetics, 8(3), 321. https://doi.org/10.3390/biomimetics8030321
15. Fei, H., Zhang, B., Liu, Y., Yan, M., Lu, Y., & Zhou, J. (2023). A Novel Chaotic Elite Adaptive Genetic Algorithm for Task Allocation of Intelligent Unmanned Wireless Sensor Networks. Applied Sciences, 13(17), 9870. https://doi.org/10.3390/app13179870
16. Gulbaz, R., Siddiqui, A. B., Anjum, N., Alotaibi, A. A., Althobaiti, T., & Ramzan, N. (2021). Balancer Genetic Algorithm—A Novel Task Scheduling Optimization Approach in Cloud Computing. Applied Sciences, 11(14), 6244. https://doi.org/10.3390/app11146244
17. Rashed, N. A., Ali, Y. H., & Rashid, T. A. (2024). Advancements in Optimization: Critical Analysis of Evolutionary, Swarm, and Behavior-Based Algorithms. Algorithms, 17(9), 416. https://doi.org/10.3390/a17090416
18. Ghorbanpour, S., Jin, Y., & Han, S. (2022). Differential Evolution with Adaptive Grid-Based Mutation Strategy for Multi-Objective Optimization. Processes, 10(11), 2316. https://doi.org/10.3390/pr10112316
19. Sun, W., Tang, M., Zhang, L., Huo, Z., & Shu, L. (2020). A Survey of Using Swarm Intelligence Algorithms in IoT. Sensors, 20(5), 1420. https://doi.org/10.3390/s20051420
20. Huang, Y., Zhang, S., & Wang, B. (2023). An Improved Genetic Algorithm with Swarm Intelligence for Security-Aware Task Scheduling in Hybrid Clouds. Electronics, 12(9), 2064. https://doi.org/10.3390/electronics12092064
21. Zhang, L., & Tang, R. (2023). Dispatch for a Continuous-Time Microgrid Based on a Modified Differential Evolution Algorithm. Mathematics, 11(2), 271. https://doi.org/10.3390/math11020271
22. Shao, K., Fu, H., & Wang, B. (2023). An Efficient Combination of Genetic Algorithm and Particle Swarm Optimization for Scheduling Data-Intensive Tasks in Heterogeneous Cloud Computing. Electronics, 12(16), 3450. https://doi.org/10.3390/electronics12163450