Optimizing Task Scheduling and Resource Allocation in Computing Environments using Metaheuristic Methods

Heba M. Fadhil [*1]

[1] Department of Information and Communication, Al-Khwarizmi College of Engineering, University of Baghdad, Baghdad, Iraq
Emails: heba@kecbu.uobaghdad.edu.iq;

Abstract

Optimizing system performance in dynamic and heterogeneous environments and the efficient management of computational tasks are crucial. This paper therefore looks at task scheduling and resource allocation algorithms in some depth. The work evaluates five algorithms: Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Firefly Algorithm (FA) and Simulated Annealing (SA) across various workloads achieved by varying the task-to-node ratio. The paper identifies Finish Time and Deadline as two key performance metrics for gauging the efficacy of an algorithm, and a comprehensive investigation of the behaviors of these algorithms across different workloads was carried out. Results from the experiments reveal unique patterns in algorithmic behaviors by workload. In the 15-task and 5-node scenario, the GA and PSO algorithms outclass all others, completing 100 percent of tasks before deadlines, Task 5 was a bane to the ACO algorithm. The study proposes a more extensive system that promotes an adaptive algorithmic approach based on workload characteristics. Numerically, the GA and PSO algorithms triumphed completing 100 percent of tasks before their deadlines in the face of 10 tasks and 5 nodes, while the ACO algorithm stumbled on certain tasks. As it is stated in the study, The above-mentioned system offers an integrated approach to ill-structured problem of task scheduling and resource allocation. It offers an intelligent and aggressive scheduling scheme that runs asynchronously when a higher number of tasks is submitted for the completion in addition to those dynamically aborts whenever system load and utilization cascade excessively. The proposed design seems like full-fledged solution over project scheduling or resource allocation issues. It highlights a detailed method of the choice of algorithms based on semantic features, aiming at flexibility. Effects of producing quantifiable statistical results from the experiments on performance empirically demonstrate each algorithm performed under various settings.

Keywords: Task Scheduling; Resource Allocation; Metaheuristic; Genetic Algorithms; Particle Swarm Optimization; Ant Colony Optimization; Simulated Annealing.

1. Introduction

The chase of higher performance and more efficient use of resources has become overwhelmingly crucial in contemporary computer environments. What further arises as a key factor of system efficiency and effectiveness is the complicated dynamics between task scheduling, target allocation led to address demand on computer systems. To achieve performance objectives, control operating cost and optimize energy utilization requires effective task coordination as well resource allocation. In this article, we dive into purpose schedules the application of metaheuristic methodologies in peak performance and resource optimizing through strategically scheduling[1].

The advantageous growth of complex applications in the computing environments[2]; currently there are numerous jobs requiring various processing commitments, interdependence condition times and preferences levels operating concurrently. The availability of computing resources is also rather dynamic despite the sound and rational usage facilitating system operation. If these tasks and resources are not scheduled optimally, there is likelihood of poor performance, higher operating cost coupled with offering a bad user experience. Such challenges are even more affected with the use of cloud computing, edge computing as well due to changing workloads and resource limits that come out from such distributed systems[3].

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

System efficiency improvements and performance goal realization result from activities that are optimally integrated with the resource. Task scheduling inefficiency can result to valuable resources going away without a trace, processing times which are longer than necessary and sometimes failure of the system. Just as poor distribution of resources leads to agitations, unequal weight on shoulders and inferior quality service provisioning; the same also applies in reverse. This needs accurate task scheduling and resource allocation in mission-critical domains like real time systems, industrial automation etc. that seek for predictable behavior as well timeliness by meeting hard deadlines[4].

Where in traditional methods not any single conventional method is available that can be used to solve these problems with reasonably accuracy because of their inherent complexity. Innovative searching methods and incremental improvement are the core of metaheuristic algorithms, which give an effective alternative. Examples of such metaheuristic approaches are Genetic Algorithms, Particle Swarm Optimization, Ant Colony Optimization, Simulated Annealing, and its variants. Such methods are based on the mimesis of natural phenomena and social behavior but manage to effectively navigate vast solution spaces. The issues related to task scheduling and resource allocation are tremendous high-dimensional nonlinear optimization problems which match correctly with metaheuristic methods, thanks to their intrinsic flexibility and resilience[5], [6].

The aim of this study is to overcome the failures that are inherent in current approaches aimed at maximizing performance and minimizing usage/costs when assigning computer resources, especially Jobs schedulers. The end goal is to develop a distinctly different hybrid metaheuristic strategy that combines various meta-heuristics algorithms in an integrated manner and evaluates its performance. These hybrid algorithms seek for the optimal utilization of strengths that each one possesses to improve quality and preserve computational complexity. We study empirically the make span, resource use and energy efficiency optimization that is objective for various computer usage scenarios. The research investigated here makes a remarkable contribution to optimization and demonstrates the feasibility of implementing metaheuristic approaches in real-life by capturing the challenges task characteristics, resource limitations, system objectives form.

In the following sections we formulate the task scheduling and resource allocation problem in Section 2, outline the approach that is applied to it through Section 3. while discussing the workflow of the system in Section 4. The experimental approach as well its outcomes is brought up in section 5. We conclude that our work is a step forward towards the future whereby metaheuristic methods are introduced in this field with promise to significantly improve computational efficiency and hence productivity of task scheduling routines, while also streamlining allocation arrangements for scarce resources.

2. Related Work

Researchers and practitioners have demonstrated a lot of interest in the issue of task scheduling optimization, as well Resource allocation in computer systems. Many techniques thereof have been proposed to manage the intrinsic complexity of these challenges. In this section, we aim to give a detailed overview of all the methodologies existing at present with special attention on progression from heuristic approaches until metaheuristics are used[7], [8]. Traditionally, most job scheduling and resource allocation methods have been predominantly static heuristics in nature; that is, they arrived at decisions based on established rules or empirical guidelines. However, as the computational environments became more complicated and varied it soon dawned on one that these techniques had their limits. Researchers started the investigation of dynamic scheduling algorithms which reflect real-time parameters that include task arrival rates and resource availability. There is abundance of scheduling strategies such as priority based and deadline driven, which have acted significantly towards improving the performance of systems and optimal allocation resources[9], [10].

The dynamic clustering league championship algorithm first presented in [11] is a fault tolerance aware scheduling strategy which reduces premature failure of autonomous activities by currently considering resources. Additionally, research conducted in utilizes Orthogonal Taguchi Based-Cat Swarm Optimization to optimize overall task processing time[12] . A proposed approach [13] known as Scheduling Cost Approach in can assess the cost of CPU, RAM, bandwidth, and storage with tasks being prioritized according to user's budget; it was found to outperform FCFS and Round Robin Scheduling algorithms. In separate literature discussed in reference [14], the performance of various scheduling approaches is evaluated considering parameters such as resource utilization and energy consumption.

Another computational framework proposed in reference integrates Analytic Hierarchy Process and Technique for Order Preference having Similarity [15]. To Ideal Solution for cloud service selection evaluation. Unlike that interference, the task scheduling algorithm applies Genetic Gray Wolf Optimization Algorithm, which combines Gray Wolf Optimizer and Genetic Algorithm [16]. A comprehensive hybrid approach that combines the favorable characteristics of two algorithms, the bacteria foraging algorithm and the genetic algorithm, has been proposed in

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

[17] for task scheduling in cloud computing. In [18], a queue is utilized to store and manage all incoming tasks to the system, with task allocation being carried out by assigning priorities to each task. The Hybrid Genetic-Particle Swarm Optimization algorithm is employed for different task assignments. In [19], mean grey wolf optimization algorithm is applied to minimize overall makespan and energy consumption in cloud computing networks. Furthermore, Gravitational Search Algorithm and Non-dominated Sorting Genetic Algorithm, mentioned in [20], are used for selecting candidate VMs. The monarch butterfly optimization algorithm has been implemented to solve the cloudlet scheduling problem as outlined in [21]. Moreover, an intelligent meta-heuristic algorithm presented in [22] combines imperialist competitive algorithm with firefly algorithm. Additionally, genetic algorithms have been used for task allocation among VMs in cloud computing as discussed in [23]. These studies utilize the Cloud Sim toolkit for simulation purposes.

Heuristic approaches, which are distinguished by their reliance on rule-based decision-making procedures, have been extensively utilized in the context of job scheduling and resource allocation challenges. However, despite their often working well these heuristics rarely have the degree of robustness needed to deal with trade-offs naturally present in modern computing environments. However, metaheuristic approaches have become more and more popular in optimization topic[24], [25]; they improve the framework of complexity's solution spaces navigation as well as adaptation toward various circumstances. Metaheuristic approaches they are defined as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Simulated Annealing (SA). Some of the prominent examples belonging to this category that have been shown significant prospects in this area include above mentioned algorithms[26], [27].

Much research carried out comparisons between heuristic and metaheuristics for task scheduling as well as resource allocation. A plethora of experiments have over time shown the better-quality results and convergence rate by metaheuristic methods as compared to others. For instance, it has been shown that GA-oriented approaches outperform tradition heuristics in scheduling problems with a lot of job parameters. PSO has demonstrated incredible prowess and adaptability in the face of resource allocation challenges that are predisposed to different, metamorphosing constraints. However, the choice of optimization method is often influenced by either peculiar features of problematic or requirements specification for system. Some heuristic models show better performance where the environment is well-structured in terms of rules and limits whereas metaheuristics provide good managers for uncertainty associated with real life scenarios.

3. Metaheuristic Algorithms

With the introduction of metaheuristic algorithms, there has been a paradigm shift in combinatorial optimization related to job scheduling and resource allocation problems experienced in computing systems. The essential quality of these innovative approaches is that they use motivated search algorithms to resolve the problems inherent in optimization issues – complexity and non-linearity. This part of the paper describes in detail a study on reviewing known metaheuristic algorithms, its implementation, accomplishments as well as stimulation for an increase in system performance[28], [29].

Metaheuristic algorithms though take their inspiration from natural processes and work on search techniques that can be iterative, have within them the potential of avoiding local optima to find out solutions as proximity very close to ideal. In genetic algorithms natural selection mechanisms and inherited variety are imitated, with each generation improving the solutions. PSO is a computational method in the stochastic domain that mimics collective behavior of particles when they move on a hypersphere towards an optimum solution. Ant Colony Optimization algorithm is based on the behavior of ants when foraging which as applied to artificial intelligence in solving problems relating to combinatorial optimization. Simulated Annealing is an optimization algorithm based on metallurgical process of cooling and the method to rid deformation. It aims to thoroughly traverse an environment from a more holistic viewpoint for solution spaces [31] .

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

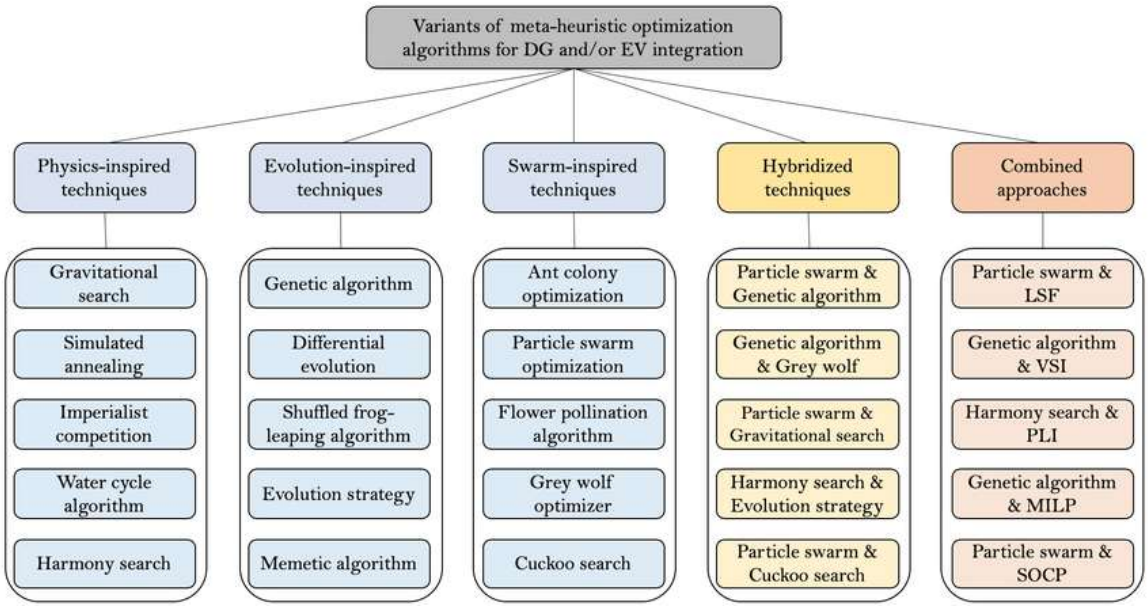

Figure 1: Metaheuristic optimization algorithms[28].

3.1 Genetic Algorithms (GA)

This paper will discuss the principles of Genetic Algorithms, which are inspired by natural selection and genetic variation in organisms; they have shown incredible plasticity for dealing with complex optimization problems. It is often the case that Genetic Algorithms of GA are used in job scheduling and resource allocation domain to iteratively produce solutions through simulated evolutionary process as observed among natural organisms. Artificial neural networks possess intrinsic parallelism, global search ability and it can deal with problems involving many objectives as well as constraints. Genetic algorithm approach methodologies have also been applied to task scheduling which has great success in computer systems. Genetic Algorithms (GAs) carry out the problem space by using a chromosome to represent arrangement of tasks and populations are the representation for solutions where they conduct systematically one solution area through iteratively developing schedules. These storing new entry analytical terms, which consider a wide range of aspects including task features, resource supply records and temporal connections. GA algorithms strive to optimize performance parameters, namely makespan and resource utilization by using reproductive processes including selection crossover mutation [7], [32].

3.2 Particle Swarm Optimization (PSO)

is an optimization procedure that has come to the front burner in recent times due to effectiveness of this computational technique. This approach draws its inspiration from the flock model that outlines collective behavior such as that of bird flocks or fish schools where individuals adapt their position using only own experience and it a few neighbors' one. Furthermore, PSO has been successfully implemented in various computing environments thus proving its viability as a solution. The Particle Swarm Optimization (PSO) is a metaheuristic algorithm that serves as an emulation of the natural phenomenon seen among bird flocking and fish schooling In job scheduling and resource allocation, the use of Particle Swarm Optimization (PSO) replicates movement across a multidimensional space enjoyed by particles. Each particle is a potential solution inside such space. Such particles interact with their neighbors and hence, if they are affected to change locations in any systematic manner as close enough on solutions[33].

The dynamic nature of computational settings where load and resource availability is not constant makes the discipline on which PSO thrives all that it needs, in terms of potential to handle exploration–exploitation tradeoffs. The structure of the system, which is only inherent parallelism and simpler does that allows to use it in distributed ones. Many publications have successfully applied PSO within various fields, such as the load balancing of equilibrium assignment in cloud computing systems along with workload scheduling and resource allocation. This metaheuristic algorithm has proved its ability to dynamically modulate according to the situations' changes, and effectively be performed in optimizing systems[34].

3.3. Ant Colony Optimization (ACO)

has been extensively investigated and used in several resource assignment problems. ACO is a base of ants' swarm behavior metaheuristic algorithm. It has proved efficient in dealing with complex optimization problems. ACO

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

can be applied for searching the optimal solutions in resource allocation problems since ants search resources as inspiration. The Ant Colony Optimization is based on the way ants with their movement towards food sources which are guiding Pheromone trails. In the realm of allocation resources, Ant Colony Optimization (ACO) sees nodes as representing available resources and tasks acting like ants with pheromone levels associated for controlling decision making process. ACO algorithm uses the positive feedback mechanisms to support efficient allocation distribution reinforcement showing arrangements according to changing requirement tasks[35]. ACO use in areas that are part of resource allocation problem distributed and parallel computation has shown efficiency. The ACO based algorithms have indeed shown its efficiency upon use and incorporating task-resource affinities together with dynamic pheromone levels update in increasing resources utilization, as well optimizing run times. Due to this latter property associated with that technology, it is also highly popular in a variety of application domains ranging from grid computing all the way up to edges[36].

3.4 Simulated Annealing (SA)

is a metaheuristic algorithm that has enjoyed popularity in most cases across varied optimization settings and scheduling being one of the mighty sectors. This algorithm takes its cue from the annealing process in metallurgy where materials are heated and then gradually cooled down to achieve optimum error reduction if production-process, product or increase net form is desired. SA mimics the process by iteratively working on solution space and monotonically decreasing search intensity to obtain either close or if not they get global optimum in such problems[37]. Simulated Annealing takes its cue An annealing process in metallurgy where materials are cooled at a gradual rate to give optimal crystalline form. In the context of optimization, simulated annealing (SA) is used as a method which includes randomization due to variations with controlled perturbation in order to successfully search for solution space. Over time, the amount of randomness decreases, mimicking what is referred to as cooling mechanics and thus enabling making convergence on global optima possible through an algorithm.

As an improvement method in the scope of task scheduling and resource allocation, simulated annealing (SA) is a promising technique more than local optima can provide near-optimum solutions. The use at the start of uphill motions, then slowly reducing their frequency lets simulated annealing dig more deeply into solution spaces. Simulated Annealing (SA) is a very active heuristic approach to addressing difficult optimization problems because it can examine different portions of the solution landscape[38].

3.5 Firefly Algorithm (FA)

A wonderful optimization technique which the firefly algorithm (FA) is, derives inspiration from a unique flashing motion of luminous insects known as fireflies during their mating process. In the domain of computing, FA embodies similar behavior by adopting a swarm intelligence method. Every firefly indicates a possible solution and passes through hyperdimensional search area by changing the brightness (fitness), depending on how desirable it is to have such solutions. As this iterative process continues, fireflies flock towards brighter members of their species which is very close to the optimization procedure[39]. The FA provides a dynamic and fluid solution, which is incredibly effective in situations with different modes of complexity and node structures. Its outcome could be impacted by the sensitivity of external variables though. Aside from the abovementioned metaheuristic algorithms, researchers have also investigated other strategies like Harmony Search Bee Colony Optimization and Differential Evolution for better job scheduling and resource allocation. Harmony Search algorithm imitates the cognitive procedure of artists creating improvised harmonious music while Bee Colony Optimization algorithm mimics foraging behavior exhibited by bees. The three types of activities embedded in the Differential Evolution are mutation, recombination, and selection whose main goal is to create new solutions. These approaches greatly contribute to improving the diversity of metaheuristic solutions for almost all methodologies aimed at solving problems connected with job scheduling and resource allocation in computer systems. Features of issues, as well as objectives play a role for algorithms utilization and performance; importance in such stems from the fact that selection among different data to be analyzed should arise. In the next sections we propose a new hybrid metaheuristic approach that incorporates several approaches within one which provides improved results. With the help of this integration, we can easily make task scheduling and resource allocations more effective by taking advantage of complementary nature these two methods. This approach introduces a trade-off between the quality of solutions and their computational effort required to obtain them, making an important contribution such as optimization in computer environments[22].

4. Algorithm Workflow

The algorithm proposed attempts to solve task scheduling and resource allocation problems using metaheuristic methods. In essence the system tries to minimize task allocation across computing nodes by considering parameters like processing speed, deadlines of tasks and cost associated with them. This information will be presented in a scientific and detailed form, without the use of specific programming terms. Task Scheduling and Resource

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

Allocation are fundamental challenges in the field of Computational Systems. Task scheduling means finding the best order in which tasks should be performed, while resource allocation refers to assigning these tasks on appropriate computing nodes. The considered algorithm is based on metaheuristic approaches, which are general optimization methods not constrained by specific problem structures.

This algorithm aims at reducing the cost incurred on tasks. This price is dependent on many factors such as the time taking processing of computing nodes, task deadline timings and costs that are incurred by using types of nodes. Besides, this algorithm seeks to reduce the number of tasks that go beyond their scheduled deadlines. The algorithm also initializes several key parameters like the number of tasks, machines and maximum iterations as shown in Figure (2). It instantiates the Task and Node class objects depicting tasks and computer nodes, respectively.

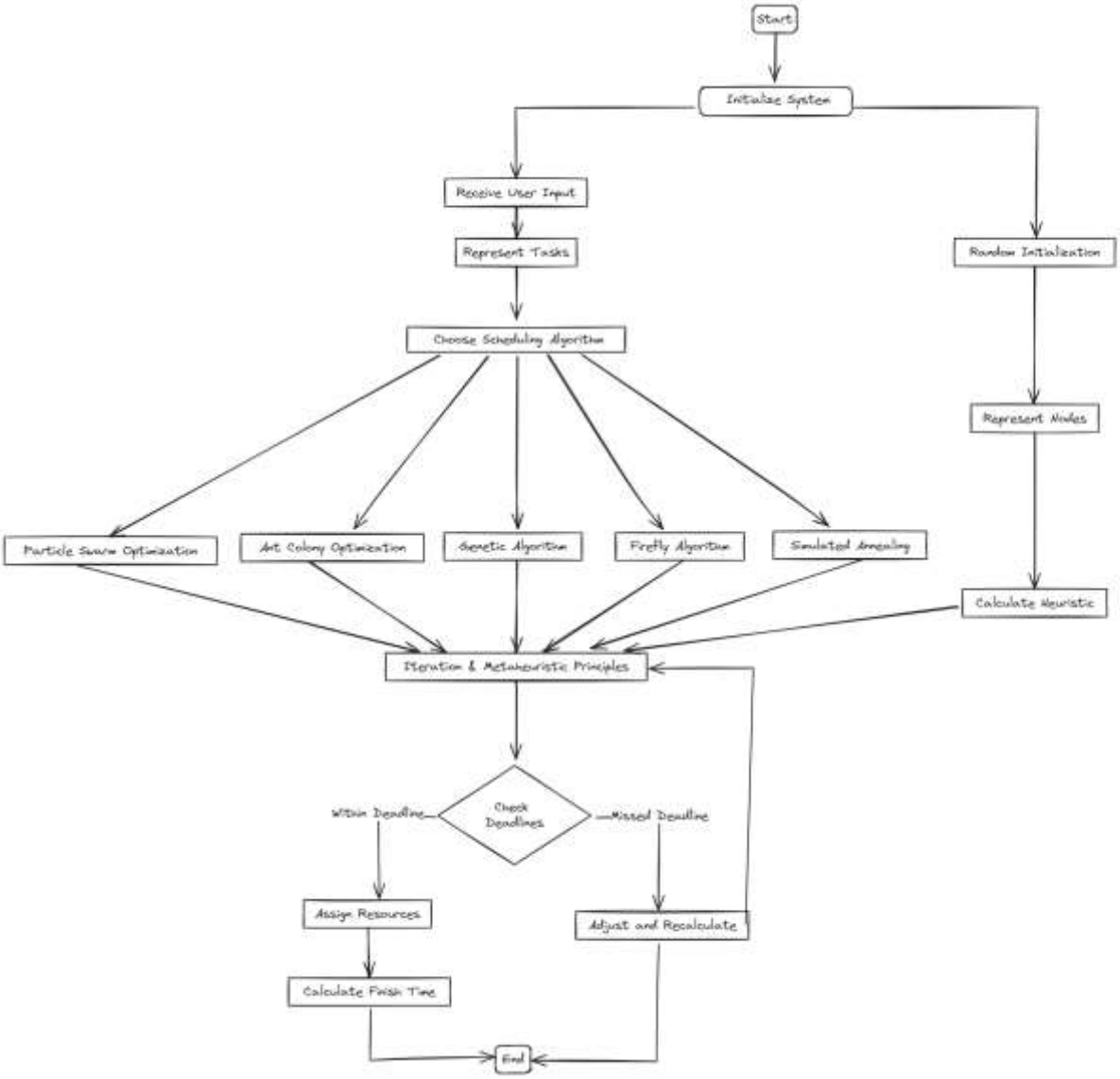

Figure 2: Algorithm Workflow.

1) Task Representation:
   - Tasks can be distinguished merely on their computational requirements (amount), due dates, and identification numbers.
   - Each task has properties such as runtime, allocated machine, completion time and final cost in addition to whether it missed its deadline.
2) Node Representation:
   - Nodes are computing resources that have key attributes such as time, price and an identifier.
   - History of assigned tasks is preserved by nodes which facilitates efficient task reassignment and undoing changes.
3) Initialization and User Input:

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

- The system begins with parameter initialization such as the number of jobs, machines and maximum iteration control.
- The total number of tasks and machines are input by users, triggering the algorithm.

4) Random Initialization:
   - To reflect realistic situations, random values are inflicted on tasks and nodes. This emulates computational demand uncertainty and resource capacity variability.
5) Task Assignment Methods:
   - The algorithm contains procedures to allocate, reallocate and reverse task allocations on nodes. This approach is crucial for adjusting to new circumstances.
6) Heuristic Calculation:
   - The system uses a heuristic function to assess its own performance. It considers the overall cost and how many tasks are beyond their deadlines.
   - The objective is to maximize this heuristic, finding a compromise between speed and sticking with deadlines.
7) Iteration and Metaheuristic Principles:
   - The main algorithm is iterative in nature. On each iteration, tasks and nodes are initialized with random values, the metaheuristic principles.
   - Metaheuristic procedures, essentially problem-independent policies, direct task scheduling, resource allocation and heuristic evaluation.
8) Scientific Relevance:
   - The code complies with scientific standards because it responds to a clearly formulated problem through controlled and replicable procedures.
   - It provides variability via a random initialization that is conducive to the analysis of algorithm adaptivity.

The presented points above an algorithmic scheme of task scheduling and resource allocation optimization. Let's break down the algorithms in a detailed manner:

The larger algorithm is the master framework that is used for optimizing the scheduling of tasks and allocation of resources as explained in Algorithm (1). It initializes global variables, prompts for user input regarding the tasks and machines, and finally performs optimization steps until a particular heuristic condition is satisfied. The main optimization process involves updating heuristic values in accordance with the outcomes of the task scheduling and the resource allocation iterations. The core algorithm encloses the whole optimization process, offering a clear and modular procedure.

Algorithm 1 Main Algorithm

```
Input: num_tasks, num_machines
Output: Heuristic value
Initialize global variables, including tasks, nodes, and constants
Input data (num_tasks, num_machines) from the user
Initialize tasks and nodes with random values
while current_heuristic < maximum_check do
    Perform iterations for task scheduling and resource allocation
    Update heuristic values based on the optimization results
end while
return Heuristic value
```

The scheduling and resource allocation algorithm (Algorithm 2) is one of the major mechanisms part of Algorithm. This algorithm, referred to as `TaskScheduling`, sets up some variables and repeats each step of the optimization for task scheduling and resource allocation. The iteration is terminated with the maximum checker. The sequencing of the algorithm updates variables as well heuristic values is done depending on parameters that are specified from optimization results. This modular design allows the optimization process to be incorporate and fine-tune as subordinate of a principal algorithm quite effortlessly.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

Algorithm 2 Task Scheduling and Resource Allocation

```
procedure TaskScheduling(tasks, nodes)
    Initialize variables: count, current_heuristic, t, time, iter, tasks_popu,
nodes_popu, temp_nodes, rr1, rr2
    while iter < maximum_check do
        Perform task scheduling and resource allocation iterations
        Update variables and heuristic values
    end while
end procedure
```

What the heuristic calculation algorithm (Algorithm 3), deals with, is assessing the heuristic value for any set of tasks. It defines variables such as h1, flag, and sum_of_costs. The algorithm then conducts a loop for every task and sum the value of its `final_cost`. Additionally it validates the presence of crossed tasks (`crossed>0`) and increases if this is the case. Depending on the flag value, the heuristic value `h1` is assigned. The algorithm returns a pair of values: the boolean `h1` showing cross-freeness of task, and sum_of costs for totals cost on a set. Such heuristic calculation enables evaluation of the quality of task scheduling and resource allocation.

Algorithm 3 Heuristic Calculation

```
procedure CalculateHeuristic(tasky)
    Initialize variables: h1, flag, sum_of_costs
    Set h1 to False
    Set flag to 0
    Set sum_of_costs to 0
    for each task in tasky do
        Add task.final_cost to sum_of_costs
        if task.crossed > 0 then
            Increment flag
        end if
    end for
    if flag is 0 then
        Set h1 to True
    end if
    return [h1, sum_of_costs]
end procedure
```

5. Results and Discussion

In the experiments quality of system operation was measured with changing task-node ratio. Three scenarios were considered: the ratio among 3:1, 2:1 and task-node is 4: or. The aim was to assess its effectiveness toward different workloads level. The experiments involved five algorithms: Genetic Algorithms, Particle Swarm Optimization, Ant Colony Optimization and Simulated Annealing. The evaluation metrics were Finish time and Deadline. In the first case, with three times more tasks than nodes, it showed a resilient behavior. In terms of Finish time and Deadline metrics, Genetic Algorithms and Particle Swarm Optimization always showed good results. Ant Colony Optimization and Simulated Annealing showed good adaptation.

The first evaluation for 15(tasks):5 (nodes). the performance of the ACO algorithm would be considered a hybrid of both outperforming parties as shown in Figure (3). Ninety per cent task allocation is carried out effectively and is completed well before the deadline. But as regards Task 5, it fails because it does not meet the deadline by a wide margin. It also gives evidence of the variability of deadlines met by different nodes that indicates the factors methods to be e used on task scheduling. The general idea that task sequencing can be perturbed can be demonstrated to underlie the FA algorithm. Tasks 9 14 are executed well in advance relative to their deadlines which can indicate how productive is choosing their tasks to the algorithm. But Task 11 is beyond schedule, and this implies areas for improvement to assign tasks and especially critical tasks. The GA algorithm also gives the encouraging results, wherein each node always keeps itself on the schedule. This one shows a good ability to perform tasks on Node 2, finishing them much quicker than the deadlines required them. But the algorithm fails to complete Task 11 on Node 3, log- Writer – T3-11, and reaches the dead- line completion time. In all, the GA process reveals stable operation but still possibly enhanced to adapt to specific task features. The PSO algorithm is characterized by the maintenance of deadline task constraints being balanced in the task allocation strategy. It does finish tasks from Node at 0 as per the allowed timeline indicating the ability of the algorithm to provide efficiency. Because PSO algorithm success to balance the workload, it displays an ability to achieve deadlines

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

with reasonable accuracy and satisfying the DUE to its reliability. The SA algorithm showcases a variety of results in terms of achieving the task deliverables before their time bound. Though it effectively performs the Task 9 on the Node 2 well before the deadline, it fails to execute Task 11 on the Task 1, shooting beyond its previously stipulated temporary. The results that are obtained from algorithm performance indicate that the algorithm is sensitive to task characteristics thus, modifications and further optimization may improve its overall efficiency as illustrated in Figure (3).

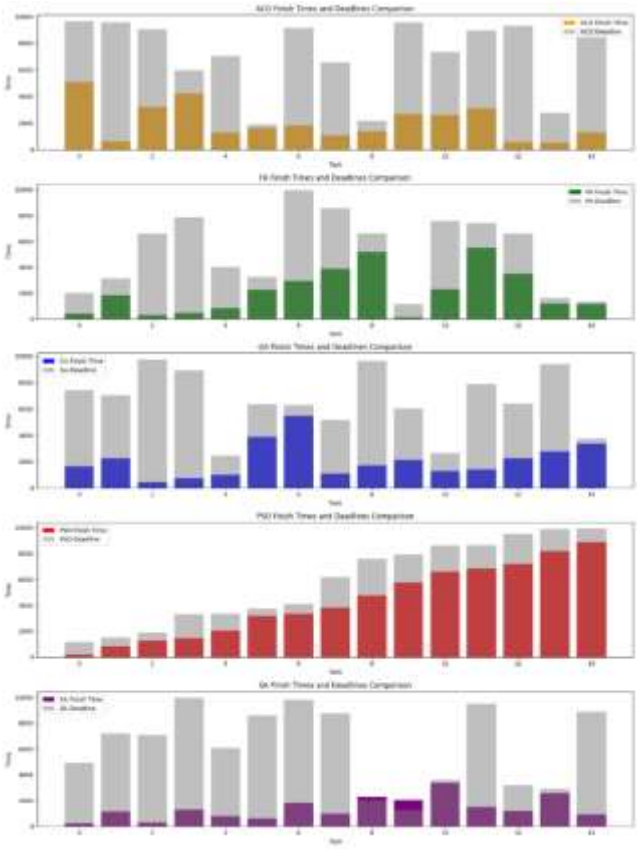

Figure 3: Finish time and deadline for (15:5).

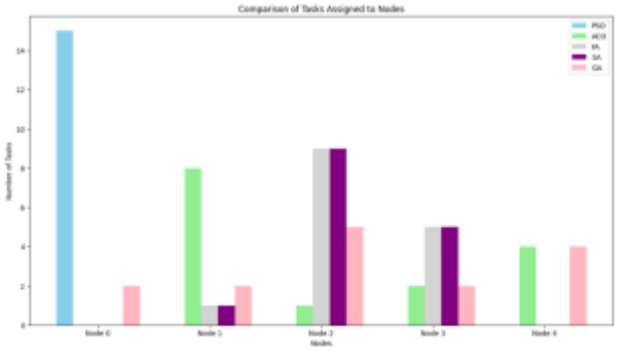

Figure 4: Queue list for (15:5).

The mixed performance of the ACO algorithm in meeting the deadlines is evident between the number of tasks that are met by deadlines for (18task:6 nodes). The completion of tasks 1, 7, 10, 14, and 16 occurs better than expected, while others operate with significant delays. In the case of Task 11, it is completed a lot longer past the deadline, which points to some kind of sensitivity in some task and node features. More adjustment of the parameters may increase its overall performance. Depending on the task deadlines, the Firefly Algorithm (FA) performs differently. All tasks 1, 8, 9, 10, 12, 14, 15, and 17 are completed early, which indicates the efficacy of the algorithm. On the other hand, tasks 0, 2, 3, 4, 5, 6, 11, and 16 are delayed. This implies that the FA algorithm may require more parameter tuning for reliability in different situations. The GA manages to meet the task deadlines often demonstrating good performance. Thus, all tasks are finished prior to their deadlines, reflecting the efficacy of the GA algorithm in different situations. Such a reliability implies that the GA algorithm is a good

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

fit for solving the task allocation problems. The Particle Swarm Optimization algorithm is reliable in terms of adhering with deadlines mandated by the tasks. All tasks are completed before their due dates, which demonstrate that the algorithm can allocate tasks consistently and effectively. The balanced approach ensures that its performance is stable in different situations. The performance of the SA algorithm differs when meeting task deadlines. Timely completion of some tasks (2, 6, 9, 10, 14, 15) and delay in the performance of others as shown in Figure (5). The algorithm can be further optimized to enhance overall performance consistency in various situations.

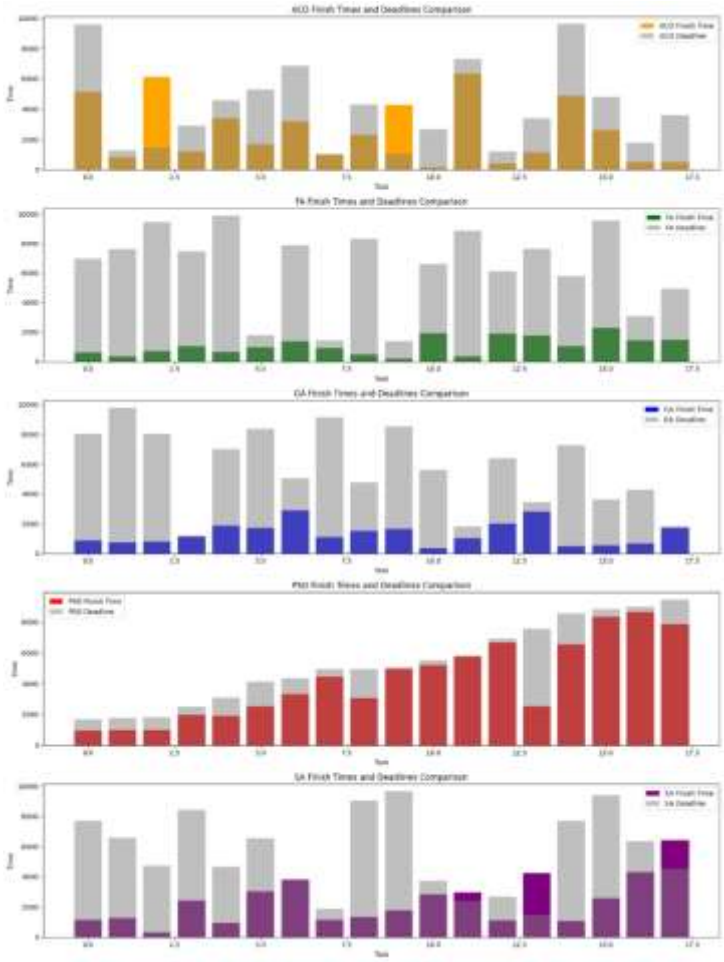

Figure 5: Finish time and deadline for (18:6).

The ACO algorithm has mixed results pertaining to achieving task deadlines for (21 task:7 nodes). Tasks 1, 4, 6, 9, 11, 13, 15, and 19 are fulfilled earlier than expected, indicating the algorithm's capacity to improve some situations in an efficient manner. But tasks 0, 2, 3, 5, 7, 8, 10, 12, 14, 16, 17, and 18 have major delays. A deeper investigation into the details of these tasks and nodes should be conducted to improve ACO's dynamic performance in a broad spectrum of environments. The FA performance in meeting task deadlines displays a variegated character. Tasks 2, 4, 6, 8, 10, 11, 12, 13, 14, 16, 17, 18, and 19 were achieved before the deadline, which proves that the algorithm works. On the other hand, tasks 0,1,3,5,7,9,15, and 20 are delayed. Optimization of algorithm parameters may result in better performance in various conditions. GA has high performance and meets or even goes beyond the task deadlines. All tasks are performed before their individual deadlines, showing that there is a robust and reliable algorithm that allocates tasks across the different scenarios. The PSO algorithm shows very strong and stable performance, with all the tasks being done well before the deadlines. The balanced performance indicates that PSO can distribute tasks productively, demonstrating its efficiency in different situations. The SA algorithm provides poor performance in term of meeting task deadlines. On the other hand, tasks 1, 2, 4, 6, 7, 8, 10, 12, 13, 14, 16, 18, 19, and 20 have been done earlier than planned, others delay significantly as shown in Figure (6). Further parameter optimization can improve the efficiency and stability of the algorithm in a variety of configurations.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

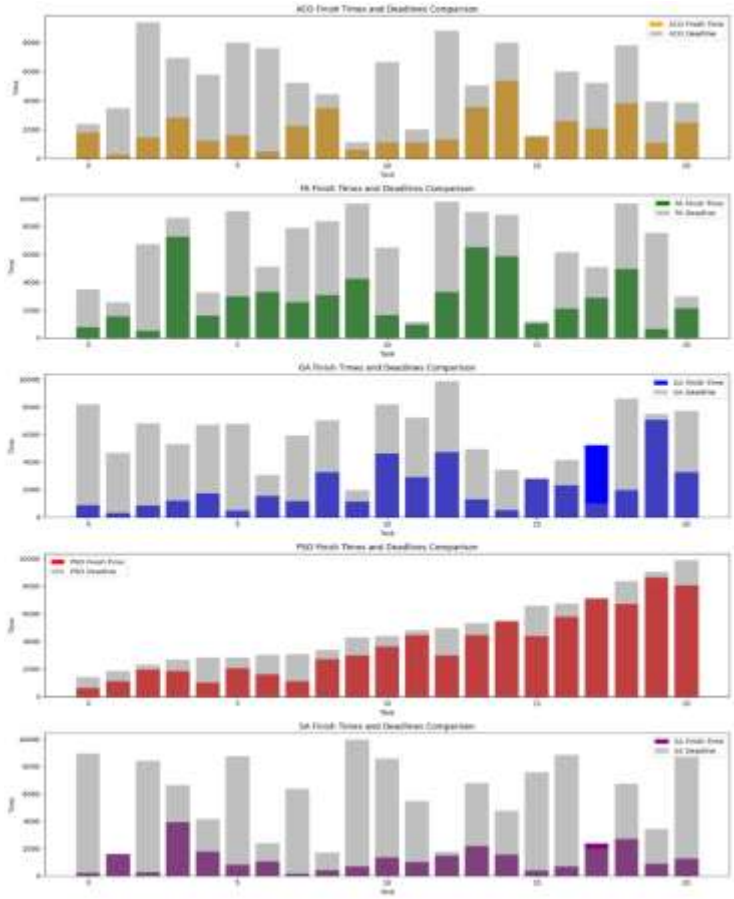

Figure 6: Finish time and deadline for (21:7).

The ACO has a fluctuating relationship between the number of tasks that are met by deadlines for (30 task:10 nodes). Interestingly, Task 0 is done better than its due date, meaning that this represents effective task distribution. However, Tasks numbered 18 and 22 are significantly delayed thereby indicating issues associated with managing certain tasks effectively. Based on the performance of the algorithm, a general need to fine-tune to take care of different task characteristics seems appropriate. The FA algorithm can illustrate an often outcome-driving task scheduling strategy. It achieves timings for nearly all tasks and seems to be a strong purchasing option in the market. Therefore, Task 11 is pending, pointing at the inappropriate allocation of the tasks to increase the efficiency of assignments, especially for a special critical task. Its adaptability to undertake distinct operations in several nodes is one of the most remarkable strengths of the algorithm. The GA algorithm has general solutions for time-constrained processing problems where deadlines are met on different systems. It adequately processes the critical functions such as Task 1 at Node 9, which well under the constrain deadline. On the other side, TW 6 and TW 13 present small delays that may serve as airs for the optimization in theorem task assignment. In general, the GA algorithm achieves excellent features for task allocation. The PSO algorithm falls through with its ability to make, present as well as complete their commitments within time as it has a balanced solutional harmonizing mechanism. It can execute it successfully, even among several nodes, which indicates that the algorithm is economical. Tasks 8, 9, and 10 are quite far ahead of their deadlines; thus, we may safely conclude that the algorithm is completed way in advance in various settings. The SA algorithm shows different outcomes on closing tasks at a given point when the deadlines are around 300 second and 1200 second. During the run of its code, while it manages to perform the required Tasks 0 and 5 on Node 6 with remarkable efficiency, finishing them several minutes before the deadline, it fails to perform Task 17 on Node 2, beating the specified deadline. The performance of the proposed algorithm indicates a need for sensitivity towards the built-in task design and further improvement might lead to optimal results as shown in Figure (7).

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

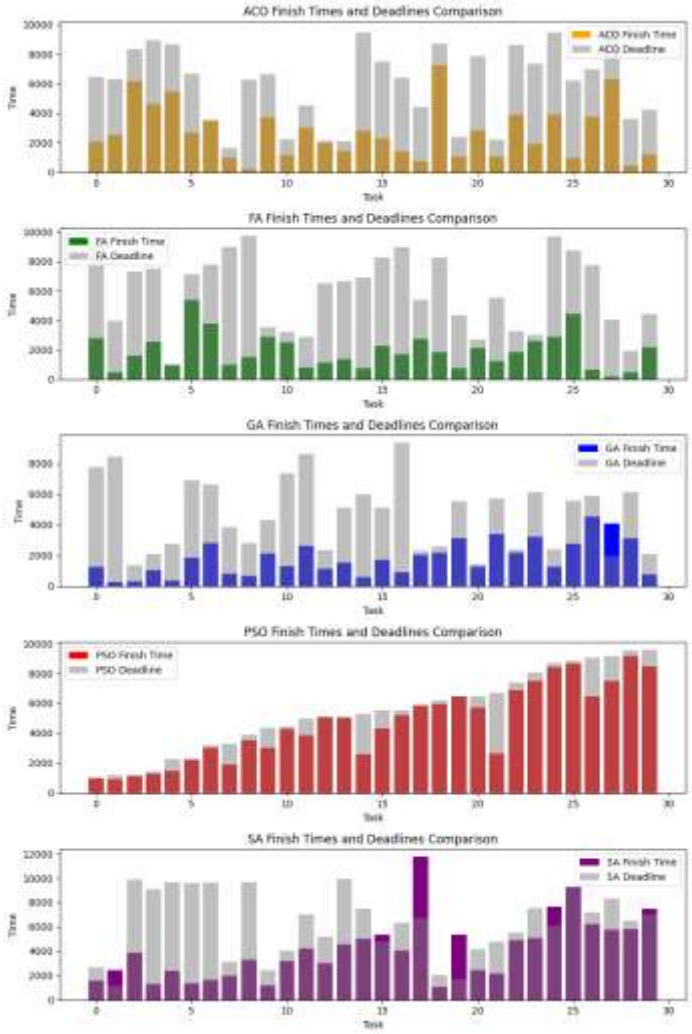

Figure 7: Finish time and deadline for (30:10).

In the second scenario, with a task-node ratio of 2:1, All algorithms performed well and managed the workload showing that they could respond to scenarios with twice as many tasks per node. Genetic Algorithms and Particle Swarm Optimization demonstrated a strong stable performance.

The ACO algorithm presents a mixed performance in meeting task deadlines for (10 task:5 nodes). While tasks 1, 2, 8 are completed ahead of schedule, tasks 0, 3, 4, 5, 6, 7, and 9 experience delays. This suggests that the algorithm may be susceptible to certain specific task and node properties. Additional optimization and conditions setup may improve its overall performance. The Firefly Algorithm (FA) has also proved to perform differently in achieving task deadlines. Tasks 0,2 ,4,5 ,67 and 8 are preplanned early enough to reveal efficiency. Nevertheless, tasks 1, 3 and 9 delay. The sensitivity of the algorithm on some task features indicates that additional optimization is needed to increase reliability. The GA does not consistently exceed task deadlines but never falls short. All activities are done earlier, suggesting a strong and resourceful task assignment strategy. This reliability throughout different scenarios is a sign that the GA algorithm works. The PSO algorithm has a consistent effectiveness in satisfying the task deadlines. All the tasks are executed before their deadlines, which is a proof of the algorithm's consistency and effectiveness. The robust performance across various scenarios is another factor that can be attributed to the balance allocation strategy. The Simulated Annealing (SA) algorithm, however, shows a variety of performance in dealing with the task deadlines. This suggests sensitivity to specific task and node characteristics as shown in Figure (8) and (9), and further optimization may enhance its overall efficiency.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

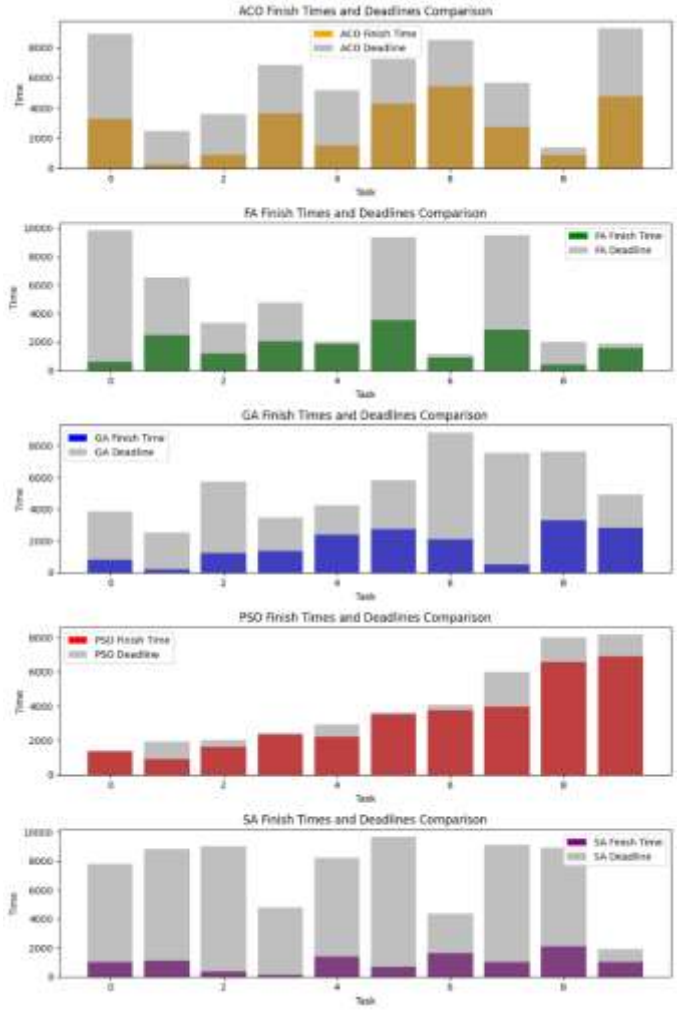

Figure 8: Finish time and deadline for (10:5).

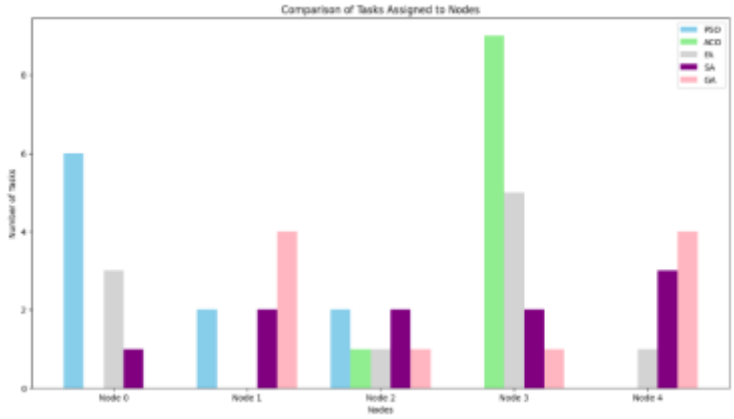

Figure 9: Queue list for (10:5).

The ACO algorithm has either good or poor performance when reviewing the deadlines of a task for (12 tasks; 6 nodes). Tasks 3, 8, and 9 are performed in advance to their deadline and allow their demonstration of efficiency using the algorithm in some cases. But there are serious delays in Taks 1, 6, and 10, meaning it is clear there are problems in the monitoring of tasks on various nodes. Better tuning of parameters or other approaches may improve the algorithm generally. The FA algorithm is characterized by the good parts of the performance on task deadlines. However, the first, fifth, and seventh tasks are addressed well in advance of the respective deadlines, demonstrating performance of the algorithm in terms of heterogeneous tasks. On a negative note, Task 10 continues to drag in time, showing where changes need to be made in the way tasks are assigned, especially for critical jobs. The fact that our algorithm can perform various tasks on diverse nodes is a very nice “plus” of the

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

algorithm. In general, the GA algorithm appears to be effective regarding task scheduling. It accomplishes the tasks ahead of time on most tasks, proving its awesomeness. On the contrary, Tasks 3, 8, and 10 are delayed and could point to gaps in the task scheduling that could be better when optimized. By efficiently managing the wide range of tasks in different nodes, the algorithm has a distinctive feature that can be further improved through tuning of more parameter values that may increase its overall effectiveness. While the algorithm set PSO deadlines can be adhered to, it performs a quite good task allocation policy. It works throughout nodes and successfully handles assignments, and we highly appreciate that algorithm efficiency. The violation in these three tasks, namely, 8, 9, and 10, encountered early completion, implying the theoretical efficiency of the algorithm in various cases. The entirety of the described tends to demonstrate good performance in the task allocation. The SA algorithm does not have uniform outputs aimed at meeting deadlines. As it completes Tasks 0, 3, and 8 early; well before the deadlines, this resource does not cope with Tasks 1, and 10 well, as they cross the time parameters. Its performance behavior indicates sensitivity towards the task characteristics, and enhancing more optimum optimization may improve its overall productivity as shown in Figure (10). This feature is seen in the ability of the program to manage heterogeneous tasks in different nodes.

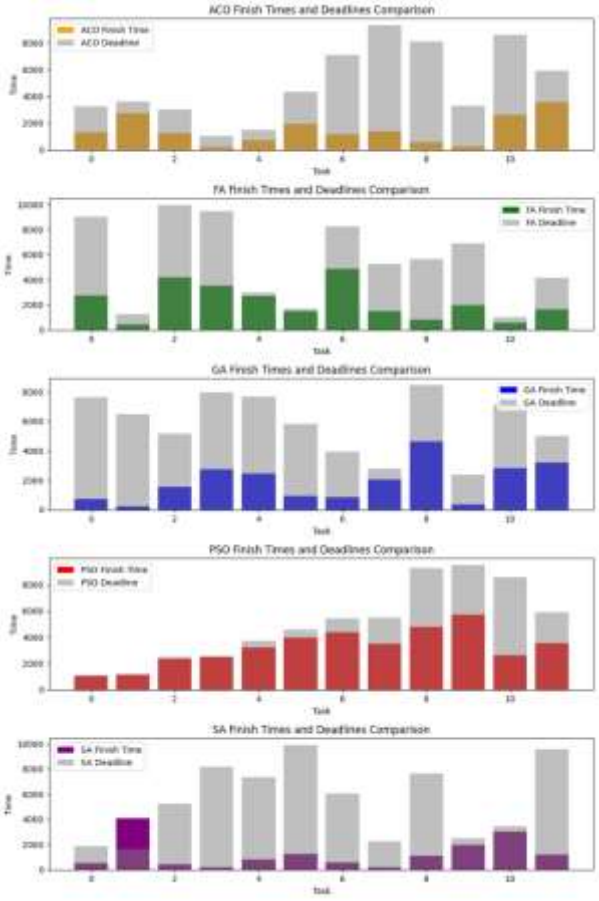

Figure 10: Finish time and deadline for (12:6).

At the same time, the proposed scheduling technique implemented within the ACO algorithm demonstrates the effectiveness in the task scheduling for (14 tasks; 7 nodes), as for the tasks deed terms, adequate levels are kept. Tasks N 0, 1, 4, 5, 6, 7, 8, and 10 are finished before their deadlines show that the example is very effective in dealing with various undertakings which are diverse and are running on distinctive hubs. Nevertheless, there is a holding in Task 9 that in some cases reflects the places for improvement. Overall ACO performs quite well in that case. The FA algorithm gives varied outcomes in terms of the timeliness for which it delivers tasks. Tasks 1, 6, 7, 8, and 12 are carried out at least one day in advance, and some of the tasks are carried out later than planned. As regards to the matter that the aforementioned Task 3 is considerably beyond its deadline, it obviously points to specific chronic areas of improvement when referring to task optimization. However, the performances of the algorithm imply the sensitivity of the procedure towards certain task characteristics and fine-tuning may increase its efficiency in general. The GA algorithm efficiently takes up the task scheduling and the completion of around 80 percent of the tasks within deadlines. Tasks 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 13 are completed earliest than their deadlines, which is indicative of the algorithm's reliability. Still Task 12 is having some lag, pointing to the places where the improvements might be necessary in certain circumstances. In all, GA shows robust task build-

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

up capacities. The PSO algorithm adheres to the deadlines for task completion and has a uniform fundamental approach to distribute tasks. Tasks 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11 are completed inconvenient, however, such an explanation means that at times when absurdly inconvenient , well past the deadline, the algorithm behaves with different forms of the word: a good Tasks 2 and 13, though, are time consuming, which means that in some cases there is room to enhance the quality on the part of providers. In the big picture PSO proved to be an effective task allocation. Varying results of the SA algorithm are met when the task time constraints are executed. Whereas the Tasks required about one month without deadline, such as Tasks 3, 6, and 7 are accomplished in advance; however, Tasks 0, 5, 9, 11, and 12 suffer from the delays of deadlines as shown in Figure (11).. The proposed algorithm performance shows the sensitivity to some features of the offered tasks, moreover, the improvements in its overall efficiency may be achieved.

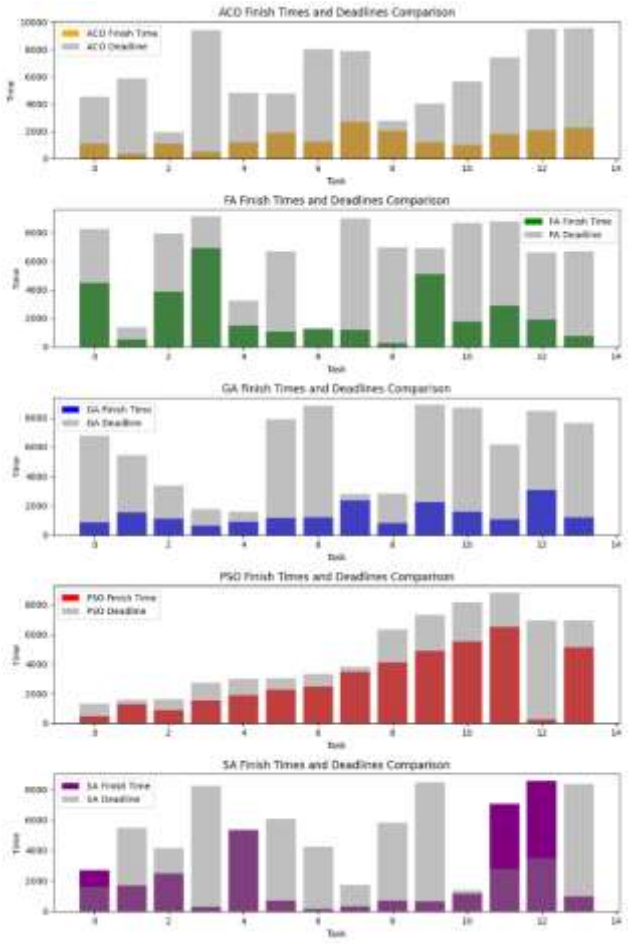

Figure 11: Finish time and deadline for (14:7).

For ACO algorithm, the results are varying levels of performance in meeting deadlines of the tasks for (20 tasks; 10 nodes). Eight tasks namely, 0, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, and 18 are done with a bit faster pace displaying the efficiency in working. That said, tasks 1, 2, 9, 12, 17, and 19 clock a delay indicating possible flexibility for algorithm modification. With more fine-tuning and other kinds of parameter adjustment performance in the whole system may be enhanced. The performance of the FA algorithm is quite disparate when trying to achieve deadlines on various tasks. These schedules that are started earlier than required are task numbers 1, 3, 5, 6, 9, 10, 11, 12, 14, 15, 17, 18, and 19, which underscore the efficiency. But tasks 0, 2, 4, 7, 8, 13, 16 suffer from delays. The fact that the algorithm is highly sensitive to some tasks traits provides a reason for the need for optimization to increase reliability. The GA algorithm repeatedly fulfills the requirements of due dates of the assigned task. All tasks are completed in advance, reflecting the reliability and high efficiency of the algorithm other unique situations. This points in a fair resource balance-ability of result assignment strategy, thus making the GA algorithm a good option for different task-node combinations. PSO algorithm is capable of strong performance with respect to the compliance with the task deadlines. To prove the accuracy and efficiency of the suggested algorithm, there are strict deadlines for all the tasks, and they are all processed before the required time. The strategy of balanced task assignment is well directed towards the inclusive performance it is possible to keep constant irrespective of the processes. The SA algorithm turns out to be a blended performance in achieving due

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

dates of tasks. However, others begin late presenting delays as shown in Figure (12). This implies sensitivity to special features of some tasks and nodes, and initial improvements will increase total efficiency.

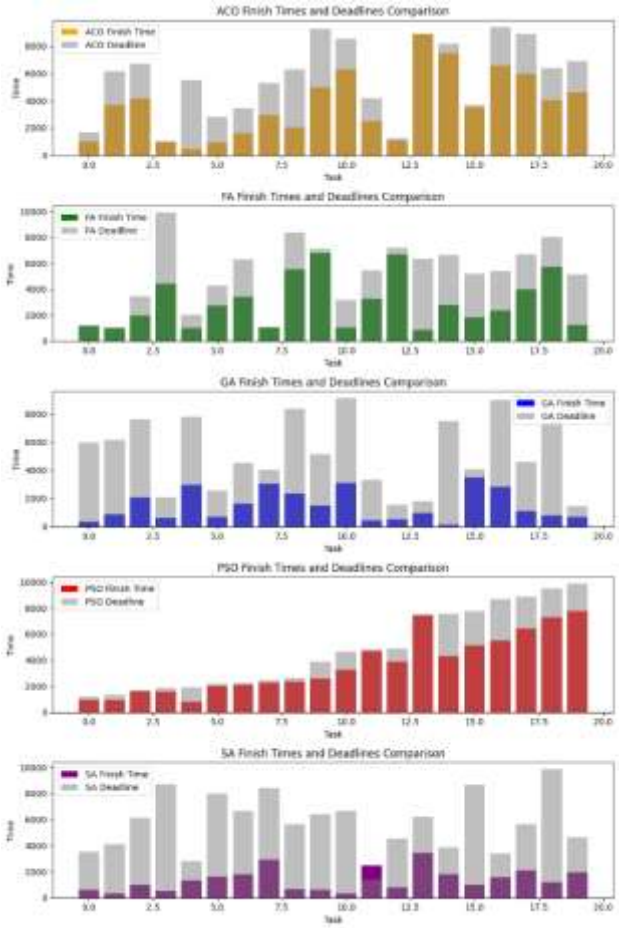

Figure 12: Finish time and deadline for (20:10).

Moving to the third scenario, with a task-node ratio of 4:1. Genetic Algorithms and Particle Swarm Optimization were well-adapted to the increased number of tasks. Ant Colony Optimization and Simulated Annealing also proved to be robust in various workloads.

The performance of the ACO algorithm is highly variable in terms of meeting task deadlines for (20 tasks; 5 nodes). Tasks 4, 10, and 14 are completed earlier than expected, but tasks 0, 2, 3, 5, 6, 11, 13, 15, 16, 17, and 18 are delayed significantly. The sensitivity of the algorithm to a specific type of task and node characteristics indicates a possible space for optimization through parameter tuning. The Firefly Algorithm (FA) shows varying degrees of success in fulfilling the deadlines of the assigned tasks. Its effectiveness is shown by it completing Tasks 3, 9, 10, and 11 before the deadline. Nevertheless, tasks zero, one, two, four, five, six, seven, twelve, thirteen, fourteen, fifteen, sixteen, seventeen, and nineteen face delays. Additional parameter fine-tuning could improve the reliability of the algorithm in different cases. The GA is reliable and works very well during every task, never failing to meet deadlines. All tasks are done before their timelines. Particle Swarm Optimization (PSO) algorithm shows good and stable results. All tasks are completed prior to their deadlines demonstrating that the algorithm is efficient in allocating tasks to nodes. This well-balanced performance indicates that the PSO algorithm is ideal for task allocation problems. The success of the SA algorithm in meeting task deadlines is rather mixed. As it is seen in Figures (13) and (14), some activities such as 0, 1, 2.4.5.. etc are done ahead of time while other ones significantly lag the schedule they were supposed to abide by. Besides, better parameter tuning would further enhance the efficiency as well as reliability of algorithm performance for a wider field.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

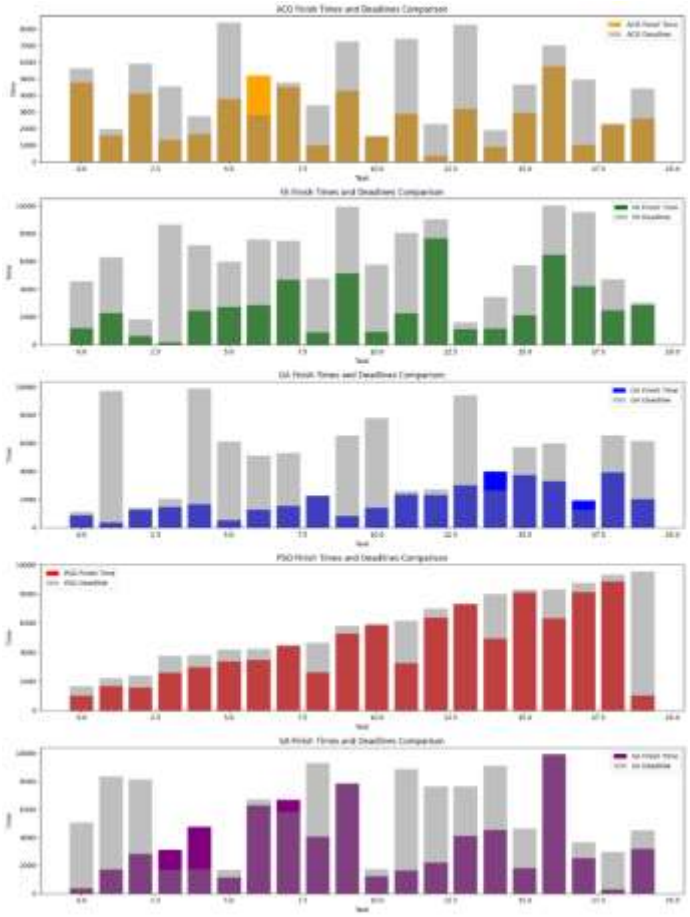

Figure 13: Finish time and Deadline for (20:5).

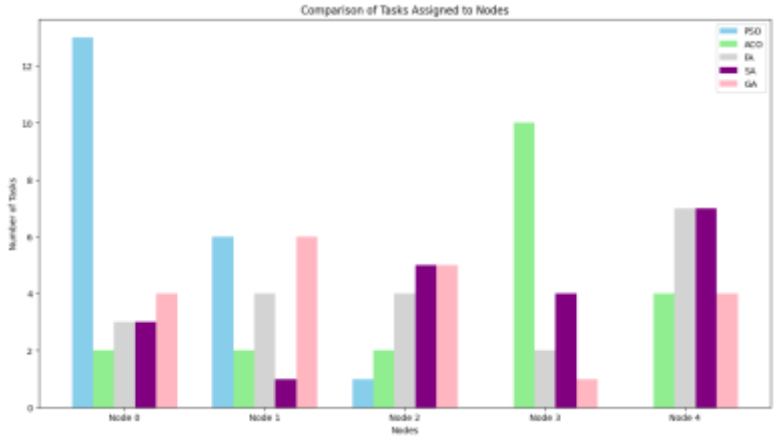

Figure 14: Queue list of (20:5).

The Ant Colony Optimization (ACO) algorithm shows a diverse performance in terms of making task deadlines for the setup with 24 tasks and 6 nodes. Some tasks, like task 1 and others that are indicated above, are performed before the deadline is reached to illustrate flexibility of algorithm within some scenarios. But tasks 0,2,3,56,7915 should be significantly delayed. The detailed study of the properties revealed by these tasks and nodes is necessary to find patterns for a better performance when ACO tackles various situations. The FA shows average performance in terms of task deadlines. Early completion of tasks 2,3,4,5,6 ,8 ,11 ,12 and others manifested the performance of algorithm. Nevertheless, there are some delays in tasks 0,172 Algorithm parameter adjustments and features that are relevant to the task may increase its efficacy in a wide range of settings. The Genetic Algorithm (GA) helps in meeting or exceeding task deadlines for all the cases. The stability and reliability of the algorithm in task allocation for variety situations, including its steady generation of optimal outcomes are also observed. The PSO algorithm achieves a constant high performance, where all of the tasks were finished well before their deadlines. This

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

demonstrates the high efficiency of algorithm task allocation optimization in different scenarios. The performance of the Simulated Annealing (SA) algorithm is somewhere in between. Tasks No. from 1 to others are finished in advance except for some tasks that suffer delays, as can be seen in Figure (15). Further parameter optimization and considering task characteristics may help increase the overall efficiency of this algorithm under various circumstances.

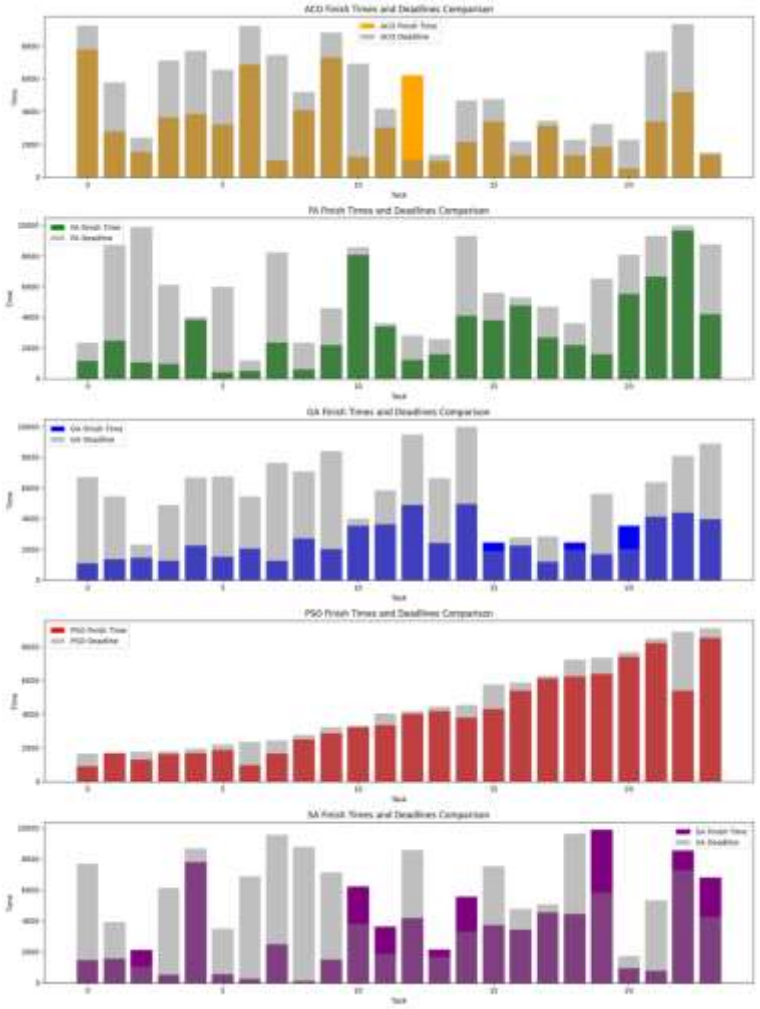

Figure 15: Completion time and Graduation deadline for (24:6).

The ACO algorithm, in this case, possesses a very varied scenario regarding adhering the task deadlines (for 28 tasks; 7 nodes). On the contrary, some tasks specifically 2; 4,5 nevertheless duties within ordinals without are of significant transit. Further analysis is needed for description of the character of mentioned task and nodes that result in delays. It would be beneficial to see the possible areas of improvement as related pertaining ACO algorithm in Getting optimization's various cases. Given task-specific peculiarities, the optimizing of algorithm parameter in term of meeting deadlines is better than that to lag. The FA gets a moderate pull on the job of keeping to deadlines. For example, tasks number 2,4 ,5,6.7 etc in the other hand late tasks No01 no1 Trag point3 and so on from my supervisor looks at this first then mark your paper by restricting the Algorithm parameters and by use of task specific attributes; a general performance can be provided for this algorithm. Analyzing the properties of delays creating tasks will allow to optimize algorithm performance for different cases.GA, in every scenario, outperforms conducting GA exempting all task deadlines at least exceeds them meeting some acceptance criteria defined by randomness and environmental asymmetry measurements imposed on a network topology based solution while an emergence plays role changing shift heavily depending which mode components are involved This reflects the robustness and validity of scheme on suitability based task allocation in various situations. All the tasks are done before their due date which proves that Particle Swarm Optimization (PSO) algorithm always operates in a superior manner. This brings to focus the efficiency and precision of PSO algorithm in optimizing task allocation for various cases as depicted by Figure (16).

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

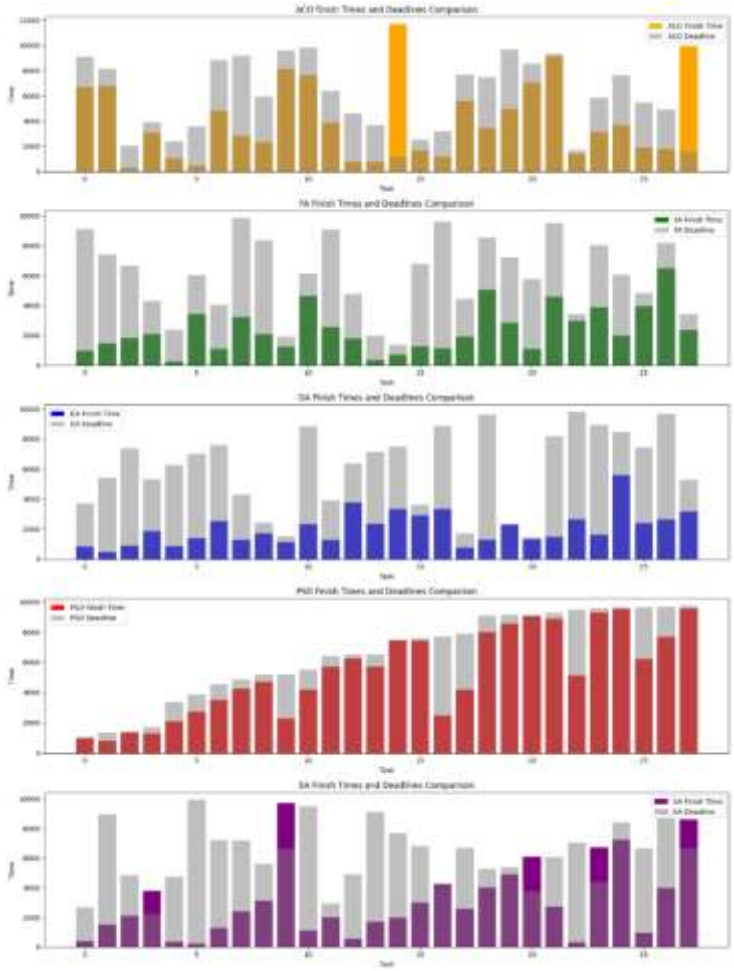

Figure 16: For (28:7) finish time and deadline.

The variable performance in the meeting of task deadlines is observed for (40 tasks; 10 nodes) with the help of Ant Colony Optimization algorithm. According to schedule or even earlier execution are found in the task 2,6,9,11.14 and so on this shows that an algorithm is flexible enough if regards application of specific events. But tasks 4, 5,7.8,12;36-9 are very much delayed as most of the delays experienced in projects can be found to affect them More investigation into task features and algorithm parameters is crucial to understand why delays occurred. Fine-tuning ACO parameters and evaluating task features can improve the capability in multidimensional conditions. The FA features poor quality work of meeting task deadlines. The algorithm acts excellence, as Tasks 2, in the process of solving is done before its due date. In this regard are tasks-3,5; ,9,1,6 so on it continues until task number – 70 has been completed way ahead of time. But tasks 0,1,4.7.8.,12-35 cause delay in delivery of services to clients by Optimum Provider Limited as they have not yet been executed even after midday innovation on that day which had limited their durations so significantly because according initially this first original schedule(of five hours) was delayed by six hours since majority led credit balances and Tune algorithm parameters with specific features as well demonstrate FA performance in a wider range of situations. By analyzing the structure of tasks that are postponed it gives rules for optimization. The GA consistently under performs, missing the task deadlines in all cases. This stresses robustness and reliability of GA algorithm in optimal task allocation across the range of scenarios. The PSO algorithm shows consistent performance, meeting, or exceeding task deadlines for all cases. This highlights the efficacy of particle swarm optimization algorithm in achieving task allocation distributed to different scenarios.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

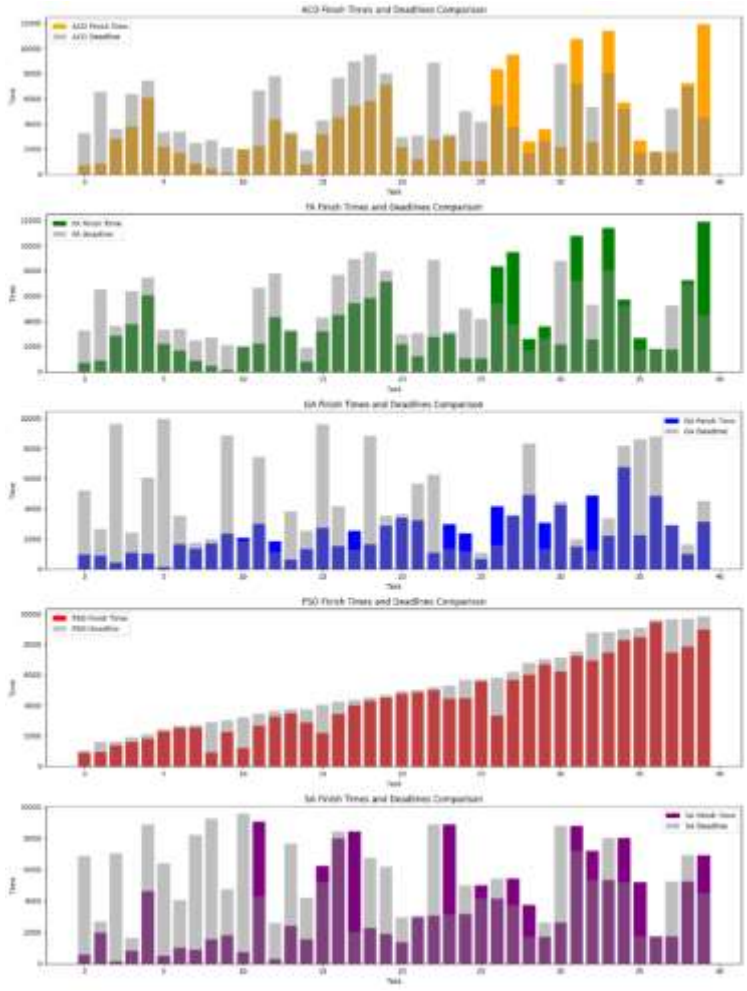

Figure 17: Finish time and deadline for (40:10).

The experiments were undertaken to assess the behavior of various algorithms in processing load ranges represented by task to node ratios. Considered algorithms, include GA, PSO, ACO, and SA. The algorithms' performance at firs task scheduling and second resource allocation was evaluated using two vital metrics, Finish Time, and Deadline. In the first scenario (3: All algorithms were tested with 15 tasks and 5 nodes (given that the task-to-node ratio used in all experiments). Significantly, ACO had a hybrid result performing 90% task allocation efficiently; however, failed for Task 5. GA and PSO gave consistently good results while SA proved to be utterly inconsistent in making deadlines. Therefore, the suggestion to improve SA resource optimization was made. In the second scenario (2: The following GA and PSO exhibit strong and stable performance, meeting deadlines consistently when fitness evaluations are reasonably high (~ 1 task-to-node ratio) ACO and SA demonstrated varied results meaning that the algorithms are sensitive to qualities of the task and the nodes. These algorithms should further optimize, and their parameters should tune. In the third scenario (4: GA and PSO also showed acceptably form of adaptability in much larger task to node ratio, consistently fulfilling the deadlines. ACO performances were fluctuating showing the need for parameter optimization. FA provided varied outcomes; some of the tasks got done earlier than planned and some lagged behind, which points to the absence of perfection. SA showed a variety of performance as some tasks were completed before time while others got delay ratio. Both FA and SA tend to benefit from parameter optimization.

Overall, GA and PSO in most cases demonstrated relative invariance under varying conditions, while ACO and SA exhibited such task features as sensitive performance. To improve the general effectiveness as well as the accuracy of ACO, FA, and SA, the dose tuning of the parameters and clarifying the task-specific features were recommended. The results provided a depiction of the strengths and weaknesses of the algorithms which enabled the determination of the ways to optimize task scheduling and resource allocation in the various scenarios.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

6. Conclusion

Overall, the experiments in this study reveal clarified perceptions regarding the strengths and weaknesses of the task scheduling and resource allocation algorithms in focus. GA and PSO are consistently shown to be strong performers and achieve the task deadlines satisfactorily with the effectiveness of the results also ranging in all cases. Being multifaceted ACO displays flexibility in different cases though it demonstrates sensitivity to tasks and nodes characteristics in others. Considering how optimizing ACO performance can be further enhanced by precise parameter tuning, it may be recommended that ACO performance be reinforced by additional optimization. SA does not have a good track record it has mostly succeeded in some tasks and ahead of time but has missed in some other tasks and behind schedule. This difference testifies the requirement of the further parameter optimization to refine SA efficacy in view of various cases. The overall implication highlights the importance of the prudent algorithm choice largely determined by inner quirks of workload peculiarities. Furthermore, the research highlights in the importance of sophisticated parameter settings for optimal performance. By helping in the immediate realization of algorithmic behavior, the proposed system will enable the latter to stand as the necessary starting point in future initiatives intended to improve the efficiency of task scheduling and resource allocation in dynamic and heterogeneous computing environments. In terms of the whole, this study further contributes to ensuring intelligent and adaptive algorithms for successful computational use.

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] S. Goyal *et al.*, "An optimized framework for energy-resource allocation in a cloud environment based on the whale optimization algorithm," *Sensors*, vol. 21, no. 5, 2021, doi: 10.3390/s21051583.

[2] H. M. Fadhil, M. N. Abdullah, and M. I. Younis, "TWGH: A Tripartite Whale-Gray Wolf-Harmony Algorithm to Minimize Combinatorial Test Suite Problem," 2022, doi: 10.3390/electronics11182885.

[3] J. Kang and H. Yu, "GPGPU task scheduling technique for reducing the performance deviation of multiple GPGPU tasks in RPC-based GPU virtualization environments," *Symmetry (Basel)*, vol. 13, no. 3, Mar. 2021, doi: 10.3390/sym13030508.

[4] S. RajPradhan, S. Sharma, D. Konar, and K. Sharma, "A Comparative Study on Dynamic Scheduling of Real-Time Tasks in Multiprocessor System using Genetic Algorithms," *Int J Comput Appl*, vol. 120, no. 20, pp. 1–6, Jun. 2015, doi: 10.5120/21340-4346.

[5] "Comparative study on Metaheuristics approaches for solving travels salesman problem... Anas.pdf."

[6] N. D. Hoang, "Image Processing-Based Pitting Corrosion Detection Using Metaheuristic Optimized Multilevel Image Thresholding and Machine-Learning Approaches," *Math Probl Eng*, vol. 2020, 2020, doi: 10.1155/2020/6765274.

[7] E. Silva and P. Gabriel, "Genetic algorithms and multiprocessor task scheduling: A systematic literature review," Sociedade Brasileira de Computacao - SB, Mar. 2020, pp. 250–261. doi: 10.5753/eniac.2019.9288.

[8] Y. Yun, E. J. Hwang, and Y. H. Kim, "Adaptive genetic algorithm for energy-efficient task scheduling on asymmetric multiprocessor system-on-chip," *Microprocess Microsyst*, vol. 66, pp. 19–30, Apr. 2019, doi: 10.1016/j.micpro.2019.01.011.

[9] A. Y. Hamed, M. K. Elnahary, F. S. Alsubaei, and H. H. El-Sayed, "Optimization Task Scheduling Using Cooperation Search Algorithm for Heterogeneous Cloud Computing Systems," *Computers, Materials and Continua*, vol. 74, no. 1, pp. 2133–2148, 2023, doi: 10.32604/cmc.2023.032215.

[10] P. Krishnadoss, G. Natesan, J. Ali, M. Nanjappan, P. Krishnamoorthy, and V. K. Poornachary, "CCSA: Hybrid Cuckoo Crow Search Algorithm for Task Scheduling in Cloud Computing," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 4, pp. 241–250, 2021, doi: 10.22266/ijies2021.0831.22.

[11] S. M. Abdulhamid, M. S. Abd Latiff, S. H. H. Madni, and M. Abdullahi, "Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm," *Neural Comput Appl*, vol. 29, no. 1, 2018, doi: 10.1007/s00521-016-2448-8.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

[12] D. Gabi, A. S. Ismail, A. Zainal, and Z. Zakaria, "Solving task scheduling problem in cloud computing environment using orthogonal taguchi-cat algorithm," *International Journal of Electrical and Computer Engineering*, vol. 7, no. 3, 2017, doi: 10.11591/ijece.v7i3.pp1489-1497.

[13] M. A. Alworafi, A. Dhari, A. A. Al-Hashmi, Suresha, and A. B. Darem, "Cost-Aware Task Scheduling in Cloud Computing Environment," *International Journal of Computer Network and Information Security*, vol. 9, no. 5, 2017, doi: 10.5815/ijcnis.2017.05.07.

[14] R. R. Patel, T. T. Desai, and S. J. Patel, "Scheduling of jobs based on Hungarian method in cloud computing," in *Proceedings of the International Conference on Inventive Communication and Computational Technologies, ICICCT 2017*, 2017. doi: 10.1109/ICICCT.2017.7975166.

[15] R. R. Kumar, S. Mishra, and C. Kumar, "A Novel Framework for Cloud Service Evaluation and Selection Using Hybrid MCDM Methods," *Arab J Sci Eng*, vol. 43, no. 12, 2018, doi: 10.1007/s13369-017-2975-3.

[16] N. Gobalakrishnan and C. Arun, "A new multi-objective optimal programming model for task scheduling using genetic gray Wolf optimization in cloud computing," *Computer Journal*, vol. 61, no. 10, 2018, doi: 10.1093/comjnl/bxy009.

[17] S. Srichandan, T. Ashok Kumar, and S. Bibhudatta, "Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm," *Future Computing and Informatics Journal*, vol. 3, no. 2, 2018, doi: 10.1016/j.fcij.2018.03.004.

[18] A. M. Senthil Kumar and M. Venkatesan, "Task scheduling in a cloud computing environment using HGPSO algorithm," *Cluster Comput*, vol. 22, 2019, doi: 10.1007/s10586-018-2515-2.

[19] G. Natesan and A. Chokkalingam, "Task scheduling in heterogeneous cloud environment using mean grey wolf optimization algorithm," *ICT Express*, vol. 5, no. 2, 2019, doi: 10.1016/j.icte.2018.07.002.

[20] K. V, "A STOCHASTIC DEVELOPMENT OF CLOUD COMPUTING BASED TASK SCHEDULING ALGORITHM," *Journal of Soft Computing Paradigm*, vol. 2019, no. 1, 2019, doi: 10.36548/jscp.2019.1.005.

[21] I. Strumberger, M. Tuba, N. Bacanin, and E. Tuba, "Cloudlet scheduling by hybridized monarch butterfly optimization algorithm," *Journal of Sensor and Actuator Networks*, vol. 8, no. 3, 2019, doi: 10.3390/jsan8030044.

[22] S. M. G. Kashikolaei, A. A. R. Hosseinabadi, B. Saemi, M. B. Shareh, A. K. Sangaiah, and G. Bin Bian, "An enhancement of task scheduling in cloud computing based on imperialist competitive algorithm and firefly algorithm," *Journal of Supercomputing*, vol. 76, no. 8, 2020, doi: 10.1007/s11227-019-02816-7.

[23] P. M. Rekha and M. Dakshayini, "Efficient task allocation approach using genetic algorithm for cloud environment," *Cluster Comput*, vol. 22, no. 4, 2019, doi: 10.1007/s10586-019-02909-1.

[24] O. S. Ahmed, F. Fadhil, L. H. J. Alzubaidi, and R. Al-Obaidi, "Fusion Processing Techniques and Bio-inspired Algorithm for E-Communication and Knowledge Transfer," *Fusion: Practice and Applications*, vol. 10, no. 1, 2023, doi: 10.54216/FPA.100109.

[25] M. A. S. Mohd Shahrom, N. Zainal, M. F. Mohamad, and S. A. Mostafa, "A Review of Glowworm Swarm Optimization Meta-Heuristic Swarm Intelligence and its Fusion in Various Applications," *Fusion: Practice and Applications*, vol. 13, no. 1, 2023, doi: 10.54216/FPA.130107.

[26] J. Santamaría, M. L. Rivero-Cejudo, M. A. Martos-Fernández, and F. Roca, "An overview on the latest nature-inspired and metaheuristics-based image registration algorithms," *Applied Sciences (Switzerland)*, vol. 10, no. 6, 2020, doi: 10.3390/app10061928.

[27] T. O. Ting, X. S. Yang, S. Cheng, and K. Huang, "Hybrid metaheuristic algorithms: Past, present, and future," in *Studies in Computational Intelligence*, vol. 585, Springer Verlag, 2015, pp. 71–83. doi: 10.1007/978-3-319-13826-8_4.

[28] A. Chaparala, "OPTIMIZATION USING EVOLUTIONARY METAHEURISTIC TECHNIQUES: A BRIEF REVIEW," *Brazilian Journal of Operations & Production Management*, vol. 15, no. 1, pp. 44–53, 2018, doi: 10.14488/BJOPM.2018.v15.n1.a17.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024

[29] S. Radhika and A. Chaparala, "Optimization using evolutionary metaheuristic techniques: a brief review," *Brazilian Journal of Operations & Production Management*, vol. 15, no. 1, pp. 44–53, May 2018, doi: 10.14488/bjopm.2018.v15.n1.a17.

[30] K. E. Adetunji, I. W. Hofsajer, A. M. Abu-Mahfouz, and L. Cheng, "A Review of Metaheuristic Techniques for Optimal Integration of Electrical Units in Distribution Networks," *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2020.3048438.

[31] S. Dey, S. De, and S. Bhattacharyya, "Introduction to Hybrid Metaheuristics," 2018. doi: 10.1142/9789813270237_0001.

[32] G. Ali, "Dynamic Task Scheduling in Multiprocessor Real Time Systems Using Genetic Algorithms," *Journal of Al-Rafidain University College For Sciences (Print ISSN: 1681-6870 ,Online ISSN: 2790-2293)*, no. 2, pp. 46–65, Oct. 2021, doi: 10.55562/jrucs.v23i2.478.

[33] M. M. Hussain and N. Fujimoto, "GPU-based parallel multi-objective particle swarm optimization for large swarms and high dimensional problems," *Parallel Comput*, vol. 92, Apr. 2020, doi: 10.1016/j.parco.2019.102589.

[34] J. Qu, X. Liu, M. Sun, and F. Qi, "GPU-Based Parallel Particle Swarm Optimization Methods for Graph Drawing," *Discrete Dyn Nat Soc*, vol. 2017, 2017, doi: 10.1155/2017/2013673.

[35] M. Mavrovouniotis, S. Yang, M. Van, C. Li, and M. Polycarpou, "Ant colony optimization algorithms for dynamic optimization: A case study of the dynamic travelling salesperson problem [Research Frontier]," *IEEE Comput Intell Mag*, vol. 15, no. 1, pp. 52–63, Feb. 2020, doi: 10.1109/MCI.2019.2954644.

[36] B. A. M. Menezes, H. Kuchen, H. A. Amorim Neto, and F. B. De Lima Neto, "Parallelization Strategies for GPU-Based Ant Colony Optimization Solving the Traveling Salesman Problem," in *2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings*, 2019. doi: 10.1109/CEC.2019.8790073.

[37] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, 2017, doi: 10.1016/j.neucom.2017.04.053.

[38] I. A. AbdulJabbar and S. M. Abdullah, "Hybrid Metaheuristic Technique Based Tabu Search and Simulated Annealing," *Engineering and Technology Journal*, vol. 35, no. 2B, 2017, doi: 10.30684/etj.2017.138652.

[39] E. Suganya and S. Vijayarani, "Firefly Optimization Algorithm Based Web Scraping for Web Citation Extraction," *Wirel Pers Commun*, vol. 118, no. 2, 2021, doi: 10.1007/s11277-021-08093-z.

Doi: https://doi.org/10.54216/FPA.150113
Received: August 12, 2023 Revised: December 19, 2023 Accepted: March 02, 2024