Journal of Informatics Education and Research
ISSN: 1526-4726
Vol 5 Issue 1 (2025)

Latency-Aware Task Scheduling and Resource Selection using two phase Genetic Algorithm in Federation of Clouds

Bharat Chhabra
Research Scholar,
Computer Science and Engineering Department,
MAIT, Maharaja Agrasen University
bharat.pnp@gmail.com
Dr.Pankaj Nanglia
Professor, Maharaja Agrasen University
nanglia.pankaj@gmail.com
Amit Dhiman
Consultant
HCL America, Frisco, Texas, USA
amittdhiman91@gmail.com
Dr. Neha Kishore
Associate Professor
Computer Science and Engineering Department,
MAIT, Maharaja Agrasen University, HP 174 103, India
nehakishore.garg@gmail.com

ABSTRACT
Federation of cloud is a collaborative model in which multiple cloud providers participate by sharing resource, services and data across their platform. This collaboration is aimed to create a unified system where user can benefit from multiple cloud providers while maintain the independence of each participating cloud. One of the key component for sustainability and successful operation of a federation of clouds is its efficient resource selection model. Resource selection has always been a challenge in cloud computing and is even more challenging in a Federation or multi-cloud setup. Researchers have dealt with this problem in various ways. Most of these existing algorithms in consider processor and memory needs without considering the bandwidth requirements of an application. In this paper, two-stage Latency-Aware Resource Selection (LA-RS) algorithm has been proposed to obtain a balance between various confronting objectives including Quality of Service (QoS), cost and completion time of applications. The first phase of the proposed algorithm figures out the top corresponding computing resources for the input tasks that satisfy their QoS requirements including cost and also considers network-latency in a federation or multi-cloud environment; the subsequent phase applies genetic algorithm that iteratively re-allocates the input tasks to optimize tasks execution time and cost. The comparison of proposed algorithm with existing algorithm clearly exhibits that along with considering the bandwidth of the underlying network, proposed algorithm achieves the objectives of optimal minimum execution time as well as optimal minimum cost.

1. Introduction
Federation of clouds presents an environment to consumers in which multiple cloud providers participate in order to offer services to its users (Agostinho et al. 2011). It generally involves a huge amount of data transmission. As federation of clouds presents a promise of offering a set of huge infrastructure services, platform services and application based services as on-demand service over Internet, its success largely depends on the quality of underneath network. The need of numerous data exchanges among physical or virtual resources in data-intensive scientific workflow applications also necessitates an efficient task-scheduling and resource selection scheme that carefully optimizes various objectives of scheduling and meets deadline constraints

(Malawski et al. 2012; Nayak and Tripathy 2018), optimizes cost (Moschakis and Karatza 2012) and conserves energy etc. Since, applications scheduled without considering the available bandwidth may impede performance execution, resulting in wastage of resources. Also a scheduling algorithm that does not consider bandwidth as a parameter may schedule an application to a cloud provider through a channel having low availability of bandwidth while another channel with better bandwidth may remain unused.

Many Researchers have investigated the problem of scheduling the tasks and resource allocation in clouds and federation of clouds from various perspectives such as execution cost, makespan, energy, reliability and fault tolerance etc. Few researchers have developed the algorithms with only one objective in focus whereas a bunch of them have considered bi-objective algorithms based on meta-heuristic approaches (Heba 2024; Mezmaz et al. 2011).

In the recent past, many strategies and algorithms have been proposed by researchers (Heba 2024; Khan and Ahmad 2009; Kofahi et al. 2019) to optimize the task scheduling and resource selection in a federation of clouds. But unfortunately, most of these algorithms are centred on allocation of processor and memory to various applications and ignore the important factor of bandwidth requirements. The contribution of this paper is a multi-objective resource selection algorithm that takes into account the bandwidth of the links connecting different providers' domains in order to avoid quality of service degradation. The rest of the paper is organized as follows. Section 2 represents the related work in this domain. Section 3 involves formulation of problem and section 4 proposes the latency-aware resource selection algorithm. Section 5 and 6 experimentally evaluate the proposed algorithm, results received and discussion based upon it. Section 7 presents a conclusion.

2. Related Work

A heuristic based critical greedy algorithm was proposed for IaaS clouds to achieve minimum end to end delay besides keeping the cost under user specified constraint in paper (Malawski et al. 2012).

A novel Min-Min algorithm is proposed (Huankai Chen et al. 2013) to schedule such applications on clouds that involve considerable network communication. The proposed algorithm is capable of adapting to the change of network transmission speed autonomously. Authors (Mezmaz et al. 2011) developed bi-objective dynamic level scheduling algorithm and bi-objective genetic algorithm for a federation of clouds. Both of these algorithms can trade off execution time against the reliability of the application.

Moreover, a cooperative game theory based solution has been proposed (Khan and Ahmad 2009) for task scheduling in computational grids. Their algorithm minimizes the energy consumption and makespan while meeting the deadline constraints. (Mezmaz et al. 2011) proposed a parallel bi-objective genetic algorithm for cloud computing system that focused on minimizing the energy consumption and makespan. The algorithm was based on cooperative island farmer-worker model and reported a non-dominated Pareto set of scheduling solutions.

Additionally a variety of eight heuristics for energy consumption aware task scheduling in large-scale distributed systems (Lindberg et al. 2012) has been presented. Six of these heuristics were greedy heuristics and two heuristics were based on genetic algorithm. All of these strategies tried to achieve an optimal solution under deadline and memory restrictions. Another task scheduling model for cloud computing based on multi-objective genetic algorithms has been proposed in(Behzad, Fotohi, and Effatparvar 2013). The proposed task scheduling method attempts to

minimize the energy consumption, maximizes the profits while meeting the deadline constraints of the application.

A heterogeneous Budget constrained scheduling algorithm has been proposed in (Arabnejad and Barbosa 2014) for service-oriented computing that maintains the execution cost under pre specified budget and also minimizes the execution time of the application to maintain the lower makespan.

Researchers in (Dorronsoro et al. 2014) presented a two level strategy focused on multi-objective problem of scheduling large workloads of parallel applications in multi-core distributed systems. Their objectives were to optimize the makespan and energy consumption. Authors also evaluated few QoS metrics that helped them to optimize their objective functions besides maximizing the Quality of Service.

Authors (Ebadifard and Babamir 2018) presented Round-Robin (RR) and Particle Swarm Optimization (PSO) hybrid algorithm focused on minimizing the execution cost but skipping the cost and other QoS parameters altogether. The study in (Abdulhamid et al. 2018) attempted to achieve two optimization objectives of minimum execution cost and minimum execution time in cloud computing environment. The major characteristic of the presented algorithm is that it also considers the fault and recovery time in case of failure. The paper (Iturriaga et al. 2016) undertakes the scheduling in a federation of distributed data centres as a multi constrained and bi-objective problem and intends to optimize cost and energy. The work is oriented to find Pareto optimal schedules i.e. where none of the scheduling order can substantially dominate the other schedules with lower cost and good bandwidth. However, their algorithm did not attend the bandwidth requirements of tasks.

Research work proposed in (Ebadifard and Babamir 2018) and (Lu and Sun, 2019) focused on balancing the load distribution of incoming tasks to attain higher average resource utilization ratio but their algorithms suffered from lower QoS, High Cost and low fault tolerance. Research study in (Praveen, Rao, and Janakiramaiah 2018) proposed unique meta-heuristic solutions for task scheduling using social group optimization and SJF algorithm but also concluded that not a single proposed technique is suitable to all variations of workload. Research work in (Dubey, Kumar, and Sharma 2018) modified the well-established conventional task scheduling algorithm namely Modified Heterogeneous Earliest Finish Time (Modified HEFT) that was able to handle only static workload.

Various researchers have tried to solve the problem of resource selection and developed algorithms in cloud computing environment. Their studies have categorized various proposed resource selection algorithms in many different classes such as static algorithms (dealing with predetermined requests of resources) /dynamic algorithms (Ding et al. 2020; Nabi, Ibrahim, and Jimenez 2021), (Chhabra and Gupta 2024) (dealing with request for varying quantity of resources), batch-mode (dealing with a bunch of tasks together) /online algorithms (dealing with interactive task requests), pre-emptive or non-pre-emptive algorithms and centralized or distributed algorithms etc. one more categorization is based on the characteristics of the incoming tasks or on the characteristics of the data-centre of provider (Cheng, Li, and Nazarian 2018; Mishra et al. 2018; Wu and Wang 2018). Here data-centre characteristics represents infrastructure attributes of the provider viz. computational capabilities of virtual machines, size of primary memory, available bandwidth etc. The cloudlets/tasks characteristics includes the resource requirements of the tasks/cloudlets viz. memory requirements, QoS required, bandwidth required amongst others.
In research work (Pradeep and Jacob 2018), researchers proposed a task scheduling algorithm that combined the merits of cuckoo search and gravitational search algorithms and overcome their

shortcomings. The proposed solution was compared with GSA, GA and PSO algorithms. The results proved that the proposed hybrid algorithm performed better on various metrics. The study evaluated the proposed algorithm on cost/profit and energy metrics whereas few important aspects of bandwidth/network latency and makespan have not been attended. One more cloudy-gravitational search algorithm (CGSA) for scheduling the tasks was proposed in (Chaudhary and Kumar 2018) with an objective to achieve lower makespan but it also incurred high cost and low QoS to do so. Whereas a study in(Wei and Zeng 2019) proposed a novel idea of hybrid differential parallel computing to process the dynamic incoming tasks on available resources to maintain low energy consumption but with a trade-off of high overhead and low Average Resource Utilization (ARU).

Research work in (Dubey and Sharma 2021) proposed a novel PSO based algorithm that has the dual properties of chemical reaction and particle swarming optimization both. The algorithm produces the set of optimal sequence depending upon the deadline and it proved to enhance the cost, makespan and energy-efficiency. The results were compared with existing peer category algorithms using CloudSim toolkit and an average improvement of 1 to 6 percent in makespan and 1 to 9 percent in energy-efficiency was claimed. The authors did not attend to the bandwidth, load balancing, and lowering the task rejection ratio.

Strategies of Linear Descending and Adaptive Inertia Weight that is based on providing an appropriate inertia weight was introduced in (Nabi et al. 2022) that adds feature of adaptability to their PSO algorithm. Due to this, the algorithm behaves optimally and attains the improvement of 12% to 60 % in makespan and average resource utilization. The algorithm fails to respond to delay-sensitive tasks and also did not attend to the requirements of bandwidth.

Authors in (Singh et al. 2021) have presented an insight on six major meta-heuristic algorithms that are nature-inspired. They also presented the comparison of these algorithms based on few metrics such as makespan, average resource utilization. The algorithms were ACO, PSO, GA, ABC, C(crow)SA, P(penguin)SO algorithm. The experimental comparisons of these algorithms proved that Crow Search algorithm produces the most optimal schedule to attain best makespan and average resource utilization and this algorithm is closely followed by Penguin Search algorithm.

(Houssein et al. 2021) presented the review of task scheduling algorithms that are based on all major meta-heuristics scheduling techniques viz. Evolutionary algorithms, swarm based scheduling algorithms, emerging scheduling algorithms and hybrid meta-heuristic algorithms. Authors also categorized the various task scheduling problems according to single-objective/multi-objective problems and restrictions such as deadline etc. The study revealed various challenges associated with all categories of algorithms and their tentative solutions also. The review presented in this paper suggest that majority of the work has been carried out considering single objective (such as scalability, throughput, reliability etc.) as a metric for resource selection whereas multi-objective algorithms are much less in number hence can be addressed in future work.

Furthermore, a two-stage task scheduling algorithm (EPETS- Energy and Performance Efficient Task Scheduling Algorithm) has been mentioned in (Hussain et al. 2021) that deals with lowering the execution time while meeting the deadlines in the first stage and to do the resource-task mapping while meeting the objective of lower energy-consumption in the second stage. Their research suggested to use energy-efficient task priority system for scheduling the tasks to achieve true balance between energy-consumption and efficient task scheduling. Their suggestion also

proved to attain the stated objectives when compared with existing energy-efficient algorithms using simulation.

(Abualigah and Alkhrabsheh 2022) proposed a novel hybrid task scheduling algorithm that combines the features of multi-verse optimizer and Genetic algorithm. The proposed algorithm is focused to improve the rate of transfer of tasks on the cloud network and thus improving the performance in the initial phase. The proposed algorithm undertook the decision of task-transfer depending on various task-specific parameters such as size of task, number of tasks, number of VMs, capacity etc. The proposed solution was simulated using MATLAB tool and proved to optimize the performance even with large task sizes hence scalable. The proposed solution did not attend the various constraints such as meeting deadlines or bandwidth.

A separate literature discussed in (Zhu et al. 2021), a two phase task scheduling algorithm has been presented that did the task-resource mapping in first phase (in lines with the requirements of security and reliability) and optimization in the second phase of the algorithm to optimize the makespan and cost. The first phase did the task-resource mapping based on QoS and trust requirements of the tasks hence took care of security and reliability constraints. The second phase undertook the multiple rounds of resource allocation to find the most optimal schedule in multi-cloud infrastructure. The proposed algorithm was simulated to compare with modified ABC algorithm, modified cuckoo search algorithm, max-min algorithm and min-min task scheduling algorithm. The results proved to be in favour of the proposed algorithm in terms of makespan, average resource utilization and cost. The proposed work did not consider deadline constraints of the task, data-transfer requirements of the task and bandwidth/network-delay of the cloud infrastructure.

Authors proposed a novel adaptive task scheduling algorithm based on Deep Q-network technique in (Peng et al. 2020). The proposed online resource scheduling environment handled the trade-off between minimum energy cost and optimal makespan. The simulation results exhibited the optimization effects as per the favoured objective. The presented algorithm yet suffered from the problem of scalability and did not give any weightage to underlying network bandwidth capacity.

Researchers in (Natesan and Chokkalingam 2020) proposed a task scheduling algorithm based on Grey Wolf optimization methodology that was focused on performance (in terms of makespan) and cost of the incoming tasks. The proposed algorithm was simulated using Cloudsim toolkit and achieved better results in terms of lower task completion time and optimal cost. Besides meeting the objectives, the algorithm also respected the deadlines of the tasks as compared to peer algorithms but did not consider other constraints such as bandwidth of the channel, energy consumption etc.

In (Zivkovic et al. 2020) proposed an improved firefly algorithm to optimally balance the workload among available and required nodes so as to minimize the energy consumption. The proposed algorithm was compared with LEACH algorithm, firefly algorithm and PSO approaches using the same network infrastructure. The main focus of the algorithm was towards the direction of solving clustering problem and skipped few important metrics viz. makespan and cost. A comprehensive hybrid approach in (Bacanin et al. 2020) proposed a monarch butterfly optimization algorithm in conjunction with two swarm-intelligence algorithms. The results of simulation proved that proposed solution is more accurate than other approaches.

Authors in (Ramamoorthy et al. 2021) proposed a novel multi-objective task scheduling algorithm that adheres to multiple restrictions while meeting the overall objectives of quickest service at minimal cost. The proposed algorithm was simulated using Cloudsim toolkit (Calheiros and

Ranjan 2009) and results were compared with existing multi-objective task scheduling algorithms which turned to be in favour of the proposed solution.

Additionally, a hybrid Bat (swarm-intelligence) algorithm has been used in (Bezdan et al. 2021) to achieve multiple objectives of minimal makespan and minimal cost. The proposed BAAEQRL algorithm's simulated results were compared with peer meta-heuristic algorithms using same scenarios of synthetic and parallel workload. While the proposed algorithm has totally omitted bandwidth availability and energy consumption metrics, it managed to surpass other compared algorithms on makespan and cost metrics.

The research work carried out in this context by various researchers may be outlined as below:

Reference No.	Year	Algorithm/Technique used	Merits of the Algorithm - (Parameters optimized)						Limitations	Tool used
			Makespan	**Deadline**	**Cost**	**Energy**	**Bandwidth**	**Others**		
(Ebadifard and Babamir 2018)	2017	PSO using Load Balancing Technique	Y	-	-	-	-	Y	QoS, High Cost and low fault tolerance	CloudSim
(Lu and Sun, 2019)	2017	Energy efficient Resource-Aware Load-Balancing Clonal Selection Algorithm	-	-	-	Y	-	-	Not scalable	CloudSim
(Praveen et al. 2018)	2017	Social Group Optimization and SJF	Y	-	-	-	-	Y	Not a single proposed technique is suitable to all variations of workload	CloudSim
(Dubey et al. 2018)	2017	Modified Heterogeneous Earliest Finish Time (Modifies HEFT)	Y	-	-	-	-	-	Static workload	CloudSim
(Wu and Wang 2018)	2018	multi-model estimation of distributed algorithm	Y	-	-	Y	-	-	Makespan and energy consumption is compromised	C on Linux
(Cheng et al. 2018)	2018	Deep Reinforcement Learning-Based Resource Provisioning and Task Scheduling	-	-	-	Y	-	-	High overhead and priorities are ignored.	-
(Mishra et al. 2018)	2018	Energy-Efficient VM-Placement	-	-	-	Y	-	Y	Cost, QoS parameters are ignored.	Cloudsim
(Pradeep and Jacob 2018)	2018	CGSA scheduler	Y		Y				Did not consider the trade-off costs.	

(Chaudhary and Kumar 2018)	2018	Cloudy-Gravitational Search Algorithm	Y	-	-	-	-	-	High cost and low QoS	CloudSim
(Wei and Zeng 2019)	2019	Hybrid differential parallel computing	Y	-	-	-	-	Y	High overhead and low ARU	C++ and MATLAB 7
(Dubey and Sharma 2021)	2021	PSO based Task Scheduling	Y	-	Y	Y	-	-	No consideration to bandwidth, load balancing and task rejection	CloudSim
(Nabi et al. 2022)	2022	Adaptive PSO based Task Scheduling	Y	-	-	-	-	Y	Not considered delay-sensitive tasks and bandwidth needs	Eclipse and CloudSim
(Hussain et al. 2021)	2021	Two stage EPETS-Energy and Performance Efficient Task Scheduling Algorithm	Y	Y	-	Y	-	-	High cost and no consideration of bandwidth requirements of tasks	-
(Abualigah and Alkhrabsheh 2022)	2022	Hybrid Multi-verse Genetic Task Scheduling algorithm	Y	-	-	-	Y	Y	Meeting deadlines not given weightage and other QoS miss.	MATLAB
(Zhu et al. 2021)	2021	Two Phase Task Scheduling algorithm with security and reliability	Y	-	Y	-	-	Y	High overhead leads to deadlines miss and bandwidth needs.	-
(Peng et al. 2020)	2020	Deep Q-Network scheduler	Y			Y			Not scalable and Bandwidth needs were ignored	
(Natesan and Chokkalingam 2020)	2020	Grey-Wolf optimization	Y	Y					Resource utilization is lower and Deadlines were ignored	
(Zivkovic et al. 2020)	2020	Modified Firefly optimization				Y		Y	Did not focus on makepan/cost	IntelliJ IDE
(Bacanin et al. 2020)	2020	Hybrid Monarch Butterfly optimization	Y	Y					Skipped to attend constraints or faults.	IntelliJ IDE
(Ramamoorthy et al. 2021)	2021	MCAMO: multi constraint aware multi-objective resource scheduling optimization	Y		Y				Not all mentioned constraints were adhered to while optimization of simultaneous objectives.	CloudSim
(Bezdan et al. 2021)	2022	Hybrid Bat algorithm	Y		Y				Low ARUs and no specific constraints were adhered to.	CloudSim
(Chaudhary and Kumar 2019)	2018	Genetic Gravitational Search Algo (GSA)	Y	-	Y	-	-	-	Fixed Bandwidth Assumption, Single Workflow assumption	C++ Coding on Linux

3. Problem Formulation

In the work done by few authors (Lu and Sun, 2019;Pradeep and Jacob, 2018) have focused on energy-efficient scheduling. They have also been able to achieve their goals and proved the efficiency of the algorithms by doing comparison with peer algorithms. But an intense literature

review clearly indicates that each algorithm has its own set of limitations, which is also evident from the table constructed above. Some of the proposed algorithms and methods have not considered the deadlines of the tasks while some others have not considered the reliability factor. Some of the available studies could not pay due heed to the data transfer cost (Dubey and Sharma 2021; Hussain et al. 2021; Nabi et al. 2022; Peng et al. 2020).

Similarly, most of papers have focused their research work on makespan only (Chaudhary and Kumar 2018; Dubey et al. 2018). Whereas other metrics viz. average resource utilization, energy efficiency and cost of execution, deadlines, bandwidth have not been given due consideration.

It is concerning that while existing research on multi-objective task scheduling has aimed to achieve specific optimization goals, it has often failed to meet other critical quality parameters. Most of the work done so far has focused on optimizing processor availability and memory resources, ensuring efficient execution within these constraints. However, very few of these approaches have taken network latency into account when scheduling tasks. Neglecting this factor can lead to inefficient job execution, increased delays, and suboptimal performance in distributed cloud environments. Addressing network latency alongside traditional resource considerations is crucial for achieving truly efficient and reliable task scheduling.

This paper presents a novel two-stage algorithm for job scheduling in a cloud federation. In the first stage, the algorithm performs network-latency-aware job allocation, ensuring that jobs are assigned to appropriate cloud resources based on network conditions. The output of this stage serves as input for the second stage, which employs a Genetic Algorithm (GA) for optimal resource allocation. This second stage focuses on minimizing job completion time while simultaneously optimizing execution costs. By integrating network-awareness in the initial stage and leveraging GA for cost-effective resource distribution, the proposed approach enhances scheduling efficiency in multi-cloud environments.

4. Proposed Scheduling Algorithm

A task scheduling and resource selection problem is comprised of mapping ‘n’ applications on federation of clouds having ‘k’ cloud providers, connected to a meta-broker as shown in Figure 1. Each provider has ‘r’ types of computational resources along with computational cost of each resource. The aim of the proposed scheme is to find a schedule that maps each application on appropriate cloud with optimal cost and execution time considering bandwidth as an important parameter. Another distinctive feature of the proposed algorithm is it being a two stage algorithm where output of the first stage acts as an input mechanism for the second stage algorithm.

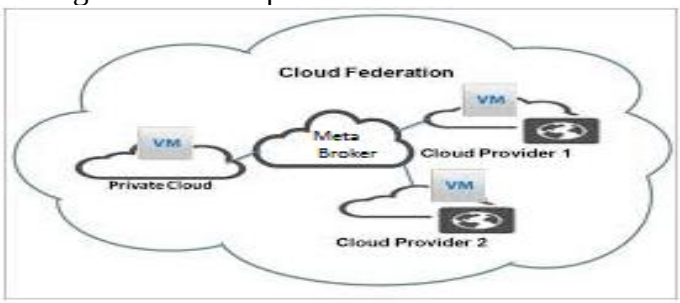


Figure 1 Federation of Clouds

Notation:

Let ‘CP’ is a set of m cloud providers, CP= {CP_1, CP_2,…..,CP_m | m>=2} and P_{ij} is a matrix that indicates the price per unit of time for using the resource where i represents the cloud provider (1<= i <= m) and j represents resource (1 <= j <= r).

Let Network_Latency (t, r) represent the network latency for task 't' on resource 'r' i.e. between the meta-broker and cloud provider. Network latency depends upon the distance and bandwidth of the channel from the broker to the cloud provider.

Let T is a set of tasks, T= { T1,T2,.... T_n } that are submitted by various users for execution to the federation and let C_{ijk} indicates the computational cost of executing i^{th} task on j^{th} resource type of the k^{th} cloud provider. It is defined as:

$$C_{ijk} = D_{ijk} \times P_{jk}$$

Where D_{ijk} is the time duration for which i^{th} task has used the j^{th} resource type of k^{th} cloud provider and P_{jk} represents the price of using j^{th} computational resource of k^{th} cloud provider.

The proposed Latency-Aware Resource Selection (LA-RS) algorithm intends to optimise the scheduling of input tasks over a multi-cloud set-up or federation of clouds. Besides being network-latency aware, it takes into account user's requirements against multiple criteria - cost and completion time. Overall, the algorithm consists of three broad steps: 1) collecting the configurations of data centres and prices of virtual machines offered by participating cloud providers, 2) use the latency-aware greedy approach to prepare task-datacentre mapping that forms the initial population for next phase 3) search an optimal schedule that meets minimal cost and time constraints using genetic algorithm.

The proposed algorithm finds a mapping-vector x for scheduling 'n' tasks over the federation with 'm' participating clouds having 'r' resource types. Mathematically $x = (x_1, x_2, \ldots\ldots, x_n)$ is derived with the following objectives:

$$\min \left(\sum_{i=1}^{n} \sum_{j=1}^{r} \sum_{k=1}^{m} C_{ijk} \right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots [1]$$

where C_{ijk} is the computational cost of executing i^{th} application on j^{th} resource type of the k^{th} cloud provider.

And

$$ET_{total} = \max_{r \in R} \left(\sum_{t \in T(r)} \text{Execution\_Time}(t, r) + \text{Network\_Latency}(t, r) \right) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots [2]$$

Where ET indicates the minimum overall execution time for T tasks on R resources.
The proposed algorithm performs federated resource allocation subject to network restrictions and mainly works in two phases, where its first phase selects a task from meta-tasks with maximum Expected Execution Time [EET] to allocate it a computing resource with Minimum Completion Time [MCT]. For each task 't' from set of tasks T and resource 'r' from the set of resources R:

EET (T) = max t ∈ T (Estimated Execution Time of Task t)
ECT (t, r) = Execution Time of t on r + Network Latency (t, r)
Makespan = max r ∈ R (∑t ∈ T(r) ECT (t, r))

The algorithm determines Estimated Completion Time [ECT] for each task i.e. estimated time that might be taken by submitted tasks to complete on different available resources. The resource with the shortest total completion time (including network latency) is then given the task with the overall maximum estimated execution time. Ultimately, after updating all recent changes in timings of tasks and resources, the process is repeated until all submitted tasks are completed. The goal of first phase of algorithm is to reduce the entire makespan, or total time after including

network-latency. Choosing the resource that takes the least amount of time to complete the largest task entails picking the fastest resource among those that are accessible.

Algorithm (Phase I)
Input:

- T = {t1, t2, ..., tn}: Set of tasks T.
- R = {r1, r2, ..., rm}: Set of available resources R.
- Network_Latency(t, r): Network latency for task 't' on resource 'r'.
- Exec_Time(t, r): Execution time of task 't' on resource 'r'.

Output:

- Resource allocation for tasks T and the makespan.

Steps:

1. **Initialize**:
    - Create an empty task-to-resource mapping table.
    - Initialize the total completion time of all resources to 0.
2. **Phase 1 - Task and Resource Selection**:
   Repeat until all tasks in T are allocated:
    - Compute EET (T): Identify the task 't*' with the maximum expected execution time.
    - Compute ECT (t∗, r) for all resources 'r'.
    - Identify resource r* with the minimum ECT (t*, r) (i.e., MCT(t*)).
    - Assign t* to r*.
    - Update the total completion time of r* and the remaining tasks in T.
3. **Update Timings**:
    - After each assignment, update the task timings and resource completion times based on the latest allocations.
4. **Repeat**:
    - Repeat the above steps until all tasks are assigned to resources.

The mapping thus produced after completion of first phase serves as initial population to the subsequent phase of the proposed algorithm. The initial population in a genetic algorithm (GA) plays a crucial role in the algorithm's performance and the efficiency with which it explores the solution space. The initial population impacts the genetic algorithm since a diverse initial population allows the algorithm to explore a wider range of potential solutions early on. This helps in avoiding premature convergence to suboptimal solutions and encourages the discovery of better solutions over time. A well-chosen initial population can help the genetic algorithm converge faster by providing a solid starting point. If the population already contains individuals that are relatively close to optimal solutions, the search can progress quickly. So, in this paper, heuristic Initialization strategy is being followed since using domain-specific knowledge to initialize the population with individuals that are likely to be closer to the optimal solution can significantly improve performance. The novel feature of the algorithm is that it considers not only the execution time while allocating the task to various providers but also the network latency of the underlying channel.

The second phase of proposed algorithm applies genetic algorithm that uses the output of first phase as initial population and aims to optimize two objectives i.e. to minimize completion cost and the overall execution time. The algorithm fetches the information about type, price and other physical configuration of various types of virtual machines available with all Cloud providers. This information helps the meta-broker to optimize the cost of execution. The meta-broker is also

capable of periodically collecting and monitoring the status of the bandwidth and latency of the channel between itself and cloud providers. The size of the population depends on the number of existing resources (both physical and virtual), virtual machine types per provider and number of input tasks. The algorithm forms a new population using crowding mechanism (to ensure diversity) from the initial population by selecting fittest chromosomes (those who dominate according to objective functions 1 and 2 as described below) using a Roulette method.

The fitness of an individual (chromosome) is determined as:

$$\text{Fitness}(x) = w1 \cdot 1/C_{total} + w2 \cdot 1/T_{total}$$

Where w1 and w2 are weights assigned to cost and runtime, respectively, based on their importance. With the aim to explore more possible hosts with better fitness, this new population is recombined using a uniform crossover. This operation requires at least two input tasks to be scheduled. This operation randomly selects two cutting points in population and all of the input tasks are swapped between these cutting points. It then performs mutation (if n>=3) that randomly selects two tasks from the population and swaps them in order to improve the fitness of individuals.

After that, the algorithm iterates until *paretoSet* (solutions) do not improve between iteration steps. At each iteration, an attempt is made to improve the fitness of the individuals in the population. It is obvious that the quality of the algorithm's interim results cannot decrease while iterating because the worst results are removed at each step. If the pareto set does not change in a number of subsequent iterations, the algorithm terminates. Furthermore, to provide an upper bound for the scheduler's runtime, there is a limit of iterations, which depends on the population size. Fitness means the quality of mapping with respect to cost and runtime. Thereby, both runtime and cost are aggregated over all iterations. The cost and runtime of each step in turn depends on the underlying resource (since VMs prices vary between different providers) and the geographical location of the provider.

Algorithm (Phase II)

Input:

- Output of Phase 1: Initial resource-task mapping.
- T = {t1, t2, ..., tn}: Set of tasks T.
- R = {r1, r2, ..., rm}: Set of available resources R.
- VM configurations, pricing, bandwidth, and latency details.

Output:

- Optimized resource-task mapping minimizing cost and execution time.

Steps:

1. **Initialization**:
   - Generate the initial population using heuristic initialization (Phase 1).
   - Compute objective functions and fitness for each individual.
2. **Main Loop**:
   Repeat until the pareto set does not improve or the iteration limit is reached:
   - **Selection**: Apply Roulette Wheel Selection to choose the fittest chromosomes.
   - **Crossover**: Perform uniform crossover to generate new individuals.
   - **Mutation**: If N≥3, mutate a subset of the population to ensure diversity.
   - **Fitness Evaluation**: Re-compute fitness for the new population.
   - **Update Pareto Set**: Add non-dominated solutions to the pareto set and remove dominated solutions.
3. **Termination**:
   - Stop the algorithm if no improvement is observed in the pareto set for k consecutive iterations or the iteration limit is reached.

5 Experimental Evaluations

In this section, the details of metrics that are used for gauging the efficiency of proposed algorithm and detailed experiment setup is presented:

5.1 Performance metric

As a measure of performance, total cost and total time for complete execution of tasks under varying workload conditions have been used as metrics for analysis in controlled simulation environment. We computed the total time and total cost of execution of a workflow using proposed Latency-Aware Resource Selection (LA-RS) algorithm and Best Resource Selection (BRS) algorithm that is based on minimum completion time by selecting a resource with maximum cost.

5.2 Data and Implementation

To simulate proposed Latency-Aware Resource Selection (LA-RS) algorithm and Best Resource Selection (BRS) algorithm, SmartFed simulator was used. A federation has been simulated with three geographically separated Cloud Providers. The cost of using their datacentres varies from provider to provider and is taken akin real life examples such as Amazon, Rackspace and Google. Table 1 depicts the details of cost assumed for each datacentre in terms of US dollar ($).

Table 1: Cost for usage of resources for each Datacentre

	First Data Center (cost in $)	Second Data Center (cost in $)	Third Data Center (cost in $)
Bandwidth	~64Mbps	~48Mbps	~56Mbps
Small VM (per Hour)	1.70	1.67	1.8
Large VM (per Hour)	3.14	3.84	3.16
Memory cost(per GB per hour)	0.05	0.05	0.05

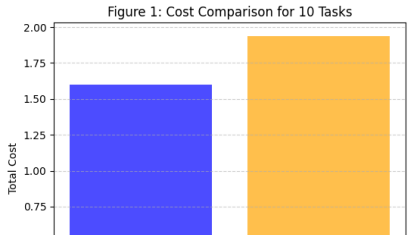
Host machines of these providers are connected to each other and meta-broker via high-speed links having a bandwidth of 64 Mbps. Storage capacity of each host is 10 TB and RAM is 8 GB and each host is equipped with 8 PEs. The processing capability of resources was confined to be same for evaluating both algorithms. Other underlying assumptions in these experiments are that in each of the datacentre, number of hosts, number of PEs and provisioning policies for VM, RAM and bandwidth are kept uniform. After input tasks are submitted, the meta-broker returns a mapping solution allocating each task to a particular provider. In this paper, allocation schemes have been implemented using SmartFed federation simulator.

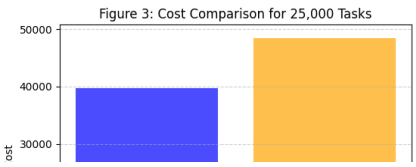
6. Results and Discussion

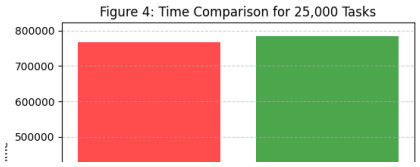
Table 2 depicts the details of output achieved from the simulation for Latency-Aware Resource Selection (LA-RS) algorithm and Best Resource Selection algorithms. The table lists the cost time and total cost of execution of tasks on the federation of clouds.

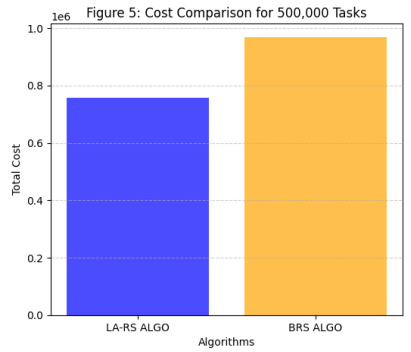
Table 2: Total Cost and Total Time of executing input tasks in a federation of three Cloud Providers in three different workload scenarios

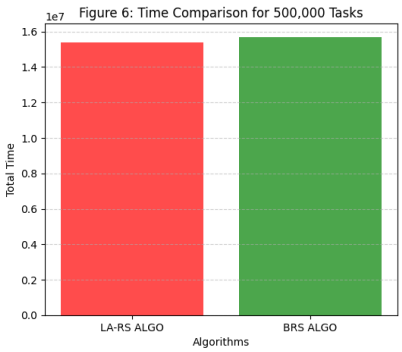
	Scaling the Input Tasks from 10 to 500000					
	10 Input Tasks to Federation		25000 Input Tasks to Federation		500000 Input Tasks to Federation	
	Total Cost	Total Time	Total Cost	Total Time	Total Cost	Total Time
LA-RS ALGO	1.59678	30.6	39690	765885	756250	15388225
BRS ALGO	1.93543	31.3	48385.75	783300	967500	15671329
% Change	21.2083067	2.2875817	21.9091711	2.27384007	27.9338843	1.83974435











Above figures depict graphically the results obtained experimentally and shows the comparison of Latency-Aware Resource Selection (LA-RS) algorithm outperforms Best Resource Selection algorithm distinctively. Figure 1, Fig. 3 and Fig. 5 display the comparison of cost efficiency of proposed LA-RS algorithm and BRS algorithm for 10, 25,000, and 500,000 tasks, showing that LA-RS algorithm consistently incurs a lower total cost, with the cost gap widening as volume of tasks increases. For 10 tasks, LA-RS algorithm's cost is about $1.75, while BRS algorithm is $2.0, making LA-RS algorithm 12.5% more cost-efficient. At 25,000 tasks, LA-RS algorithm costs 40,000, whereas BRS algorithm reaches 50,000, offering a 20% cost reduction. This trend amplifies for 500,000 tasks, where LA-RS algorithm incurs 750,000, while BRS algorithm nears 1,000,000, meaning LA-RS algorithm is 25% cheaper. The increasing cost difference suggests that BRS algorithm scales less efficiently, likely due to additional overhead or inefficiencies at higher workloads. The trade-off is that while both algorithms may perform similarly for small workloads, LA-RS algorithm becomes significantly more cost-effective as task volume increases, making it the preferred choice for large-scale scheduling in cloud computing or distributed systems.

Fig. 2, Fig. 4 and Fig. 6 depict the makespan for both resource allocation algorithms in all three workload scenarios. An insight into these figures show that out-performance of proposed Latency-Aware Resource Selection (LA-RS) algorithm over Best Resource Selection algorithm is evident on makespan metric also. But the improvement is not by huge margin. As the makespan in first workload scenario (10 tasks) is upgraded by as low as close to 2%. Moreover, as the numbers of input tasks are increased in second and third workload scenarios, the makespan of proposed LA-RS algorithm is again improved marginally only i.e. to the tune of 2%. It clearly indicates that as the workload is scaled up, the proposed algorithm sustains its trend of superseding nature of lower makespan than BRS algorithm.

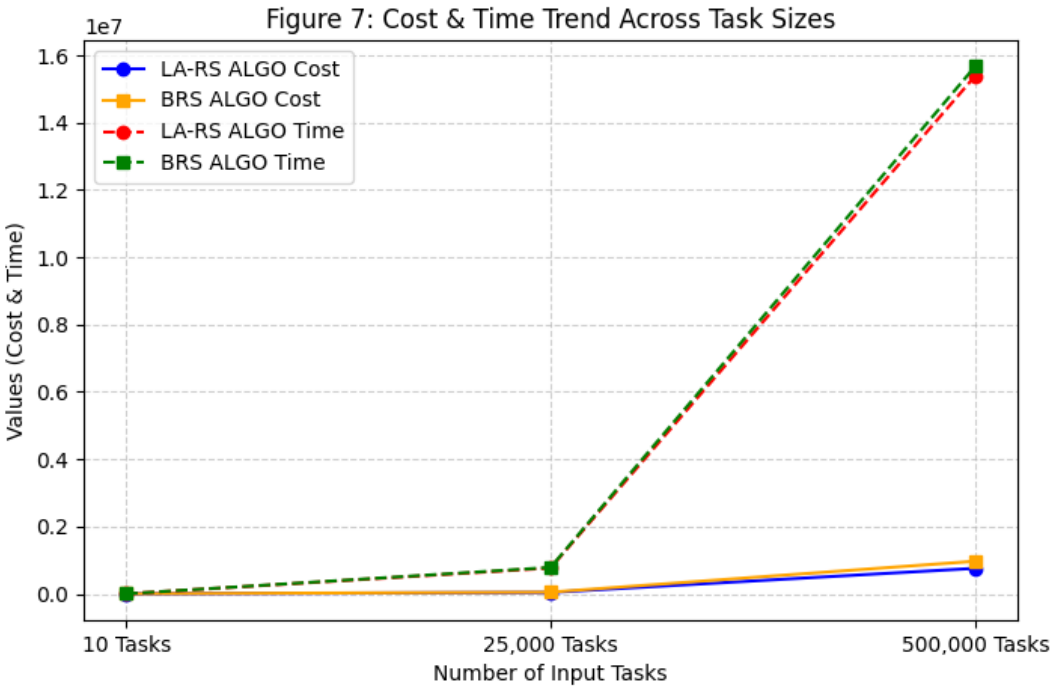


Figure 7 illustrates the trend in total cost and makespan for both algorithms under analysis, clearly highlighting the advantages of the proposed algorithm. This benefit is particularly evident due to its enhanced consideration of bandwidth availability when selecting resources in the initial stage of the algorithm.

7. Conclusion

The proposed Latency-Aware Resource Selection (LA-RS) algorithm and Best Resource Selection (BRS) algorithm evaluated experimentally in this paper have shown that proposed Latency-Aware Resource Scheduling (LA-RS) algorithm gives the lower cost for execution of tasks since this algorithm not only exploits the cheaper datacentre but also tries to optimize the makespan simultaneously. As optimal (low cost and least time) provider's datacentre runs out of resources, algorithm selects the next cheaper provider's datacentre to execute remaining tasks. As the resources are scaled up in the cheapest datacentre, the total cost of execution of tasks decreases more steeply as compared to Best Resource Selection algorithm. Amongst the three workload situations, it has been observed that proposed algorithm utilized resources in a more effective manner as compared to BRS algorithm in all workload scenarios. Rather its performance is accelerated as the input tasks are scaled up. Conclusively, adopting LA-RS algorithm for high-volume tasks could lead to substantial cost savings (20-25%) and a marginal demotion in makespan too, reinforcing its suitability for large-scale operational deployments.

References:

- Abdulhamid, Shafi'i Muhammad, Muhammad Shafie Abd Latiff, Syed Hamid Hussain Madni, and Mohammed Abdullahi. 2018. "Fault Tolerance Aware Scheduling Technique for Cloud Computing Environment Using Dynamic Clustering Algorithm." *Neural Computing and Applications* 29(1):279–93. doi: 10.1007/s00521-016-2448-8.
- Abualigah, Laith, and Muhammad Alkhrabsheh. 2022. "Amended Hybrid Multi-Verse Optimizer with Genetic Algorithm for Solving Task Scheduling Problem in Cloud Computing." *The Journal of Supercomputing* 78(1):740–65. doi: 10.1007/s11227-021-03915-0.
- Agostinho, Lucio, Guilherme Feliciano, Leonardo Olivi, Eleri Cardozo, and Eliane Guimaraes. 2011. "A Bio-Inspired Approach to Provisioning of Virtual Resources in Federated Clouds." Pp. 598–604 in *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*. Sydney, Australia: IEEE.
- Arabnejad, Hamid, and Jorge G. Barbosa. 2014. "A Budget Constrained Scheduling Algorithm for Workflow Applications." *Journal of Grid Computing* 12(4):665–79. doi: 10.1007/s10723-014-9294-7.
- Bacanin, Nebojsa, Timea Bezdan, Eva Tuba, Ivana Strumberger, and Milan Tuba. 2020. "Monarch Butterfly Optimization Based Convolutional Neural Network Design." *Mathematics* 8(6):936. doi: 10.3390/math8060936.
- Behzad, Shahram, Reza Fotohi, and Mehdi Effatparvar. 2013. "Queue Based Job Scheduling Algorithm for Cloud Computing." 7.
- Bezdan, Timea, Miodrag Zivkovic, Nebojsa Bacanin, Ivana Strumberger, Eva Tuba, and Milan Tuba. 2021. "Multi-Objective Task Scheduling in Cloud Computing Environment by Hybridized Bat Algorithm." *Journal of Intelligent & Fuzzy Systems* 42(1):411–23. doi: 10.3233/JIFS-219200.
- Calheiros, Rodrigo N., and Rajiv Ranjan. 2009. "CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services." *Grid Computing and Distributed Systems Laboratory* (1):10. doi: doi.org/10.48550/arXiv.0903.2525.
- Chaudhary, Divya, and Bijendra Kumar. 2018. "Cloudy GSA for Load Scheduling in Cloud Computing." *Applied Soft Computing* 71:861–71. doi: 10.1016/j.asoc.2018.07.046.
- Chaudhary, Divya, and Bijendra Kumar. 2019. "Cost Optimized Hybrid Genetic-Gravitational Search Algorithm for Load Scheduling in Cloud Computing." *Applied Soft Computing* 83:105627. doi: 10.1016/j.asoc.2019.105627.
- Cheng, Mingxi, Ji Li, and Shahin Nazarian. 2018. "DRL-Cloud: Deep Reinforcement Learning-Based Resource Provisioning and Task Scheduling for Cloud Service Providers." Pp. 129–34 in *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*. Jeju: IEEE.
- Chhabra, Bharat, and Shilpa Gupta. 2024. "A Critical Assessment of Task Scheduling Algorithms in Cloud Computing: A Comparative Approach." Pp. 125–35 in *Proceedings of Fifth Doctoral Symposium on Computational Intelligence*. Vol. 1086, *Lecture Notes in Networks and Systems*, edited by A. Swaroop, V. Kansal, G. Fortino, and A. E. Hassanien. Singapore: Springer Nature Singapore.
- Ding, Ding, Xiaocong Fan, Yihuan Zhao, Kaixuan Kang, Qian Yin, and Jing Zeng. 2020. "Q-Learning Based Dynamic Task Scheduling for Energy-Efficient Cloud Computing." *Future Generation Computer Systems* 108:361–71. doi: 10.1016/j.future.2020.02.018.
- Dorronsoro, Bernabé, Sergio Nesmachnow, Javid Taheri, Albert Y. Zomaya, El-Ghazali Talbi, and Pascal Bouvry. 2014. "A Hierarchical Approach for Energy-Efficient Scheduling of Large Workloads in Multicore Distributed Systems." *Sustainable Computing: Informatics and Systems* 4(4):252–61. doi: 10.1016/j.suscom.2014.08.003.

- Dubey, Kalka, Mohit Kumar, and S. C. Sharma. 2018. "Modified HEFT Algorithm for Task Scheduling in Cloud Environment." *Procedia Computer Science* 125:725–32. doi: 10.1016/j.procs.2017.12.093.
- Dubey, Kalka, and S. C. Sharma. 2021. "A Novel Multi-Objective CR-PSO Task Scheduling Algorithm with Deadline Constraint in Cloud Computing." *Sustainable Computing: Informatics and Systems* 32:100605. doi: 10.1016/j.suscom.2021.100605.
- Ebadifard, Fatemeh, and Seyed Morteza Babamir. 2018. "A PSO-Based Task Scheduling Algorithm Improved Using a Load-Balancing Technique for the Cloud Computing Environment." *Concurrency and Computation: Practice and Experience* 30(12):e4368. doi: 10.1002/cpe.4368.
- Heba, Heba. 2024. "Optimizing Task Scheduling and Resource Allocation in Computing Environments Using Metaheuristic Methods." *Fusion: Practice and Applications* 15(1):157–79. doi: 10.54216/FPA.150113.
- Houssein, Essam H., Ahmed G. Gad, Yaser M. Wazery, and Ponnuthurai Nagaratnam Suganthan. 2021. "Task Scheduling in Cloud Computing Based on Meta-Heuristics: Review, Taxonomy, Open Challenges, and Future Trends." *Swarm and Evolutionary Computation* 62:100841. doi: 10.1016/j.swevo.2021.100841.
- Huankai Chen, Frank Wang, Na Helian, and Gbola Akanmu. 2013. "User-Priority Guided Min-Min Scheduling Algorithm for Load Balancing in Cloud Computing." Pp. 1–8 in *2013 National Conference on Parallel Computing Technologies (PARCOMPTECH)*. Bangalore, India: IEEE.
- Hussain, Mehboob, Lian-Fu Wei, Abdullah Lakhan, Samad Wali, Soragga Ali, and Abid Hussain. 2021. "Energy and Performance-Efficient Task Scheduling in Heterogeneous Virtualized Cloud Computing." *Sustainable Computing: Informatics and Systems* 30:100517. doi: 10.1016/j.suscom.2021.100517.
- Iturriaga, Santiago, Sergio Nesmachnow, Andrei Tchernykh, and Bernabe Dorronsoro. 2016. "Multiobjective Workflow Scheduling in a Federation of Heterogeneous Green-Powered Data Centers." Pp. 596–99 in *2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*. Cartagena, Colombia: IEEE.
- Khan, S. U., and I. Ahmad. 2009. "A Cooperative Game Theoretical Technique for Joint Optimization of Energy Consumption and Response Time in Computational Grids." *IEEE Transactions on Parallel and Distributed Systems* 20(3):346–60. doi: 10.1109/TPDS.2008.83.
- Kofahi, Najib A., Tariq Alsmadi, Malek Barhoush, and Moy'awiah A. Al-Shannaq. 2019. "Priority-Based and Optimized Data Center Selection in Cloud Computing." *Arabian Journal for Science and Engineering* 44(11):9275–90. doi: 10.1007/s13369-019-03845-3.
- Lindberg, Peder, James Leingang, Daniel Lysaker, Samee Ullah Khan, and Juan Li. 2012. "Comparison and Analysis of Eight Scheduling Heuristics for the Optimization of Energy Consumption and Makespan in Large-Scale Distributed Systems." *The Journal of Supercomputing* 59(1):323–60. doi: 10.1007/s11227-010-0439-6.
- Lu, Yong, and Na Sun. 2019. "An Effective Task Scheduling Algorithm Based on Dynamic Energy Management and Efficient Resource Utilization in Green Cloud Computing Environment." *Cluster Computing* 22(S1):513–20. doi: 10.1007/s10586-017-1272-y.
- Malawski, Maciej, Gideon Juve, Ewa Deelman, and Jarek Nabrzyski. 2012. "Cost- and Deadline-Constrained Provisioning for Scientific Workflow Ensembles in IaaS Clouds." Pp. 1–11 in *2012 International Conference for High Performance Computing, Networking, Storage and Analysis*. Salt Lake City, UT: IEEE.
- Mezmaz, M., N. Melab, Y. Kessaci, Y. C. Lee, E. G. Talbi, A. Y. Zomaya, and D. Tuyttens. 2011. "A Parallel Bi-Objective Hybrid Metaheuristic for Energy-Aware Scheduling for

Cloud Computing Systems." *Journal of Parallel and Distributed Computing* 71(11):1497–1508. doi: 10.1016/j.jpdc.2011.04.007.

- Mishra, Sambit Kumar, Deepak Puthal, Bibhudatta Sahoo, Prem Prakash Jayaraman, Song Jun, Albert Y. Zomaya, and Rajiv Ranjan. 2018. "Energy-Efficient VM-Placement in Cloud Data Center." *Sustainable Computing: Informatics and Systems* 20:48–55. doi: 10.1016/j.suscom.2018.01.002.
- Moschakis, Ioannis A., and Helen D. Karatza. 2012. "Evaluation of Gang Scheduling Performance and Cost in a Cloud Computing System." *The Journal of Supercomputing* 59(2):975–92. doi: 10.1007/s11227-010-0481-4.
- Nabi, Said, Masroor Ahmad, Muhammad Ibrahim, and Habib Hamam. 2022. "AdPSO: Adaptive PSO-Based Task Scheduling Approach for Cloud Computing." *Sensors* 22(3):920. doi: 10.3390/s22030920.
- Nabi, Said, Muhammad Ibrahim, and Jose M. Jimenez. 2021. "DRALBA: Dynamic and Resource Aware Load Balanced Scheduling Approach for Cloud Computing." *IEEE Access* 9:61283–97. doi: 10.1109/ACCESS.2021.3074145.
- Natesan, Gobalakrishnan, and Arun Chokkalingam. 2020. "An Improved Grey Wolf Optimization Algorithm Based Task Scheduling in Cloud Computing Environment." *The International Arab Journal of Information Technology* 73–81. doi: 10.34028/iajit/17/1/9.
- Nayak, Suvendu Chandan, and Chitaranjan Tripathy. 2018. "Deadline Sensitive Lease Scheduling in Cloud Computing Environment Using AHP." *Journal of King Saud University - Computer and Information Sciences* 30(2):152–63. doi: 10.1016/j.jksuci.2016.05.003.
- Peng, Zhiping, Jianpeng Lin, Delong Cui, Qirui Li, and Jieguang He. 2020. "A Multi-Objective Trade-off Framework for Cloud Resource Scheduling Based on the Deep Q-Network Algorithm." *Cluster Computing* 23(4):2753–67. doi: 10.1007/s10586-019-03042-9.
- Pradeep, K., and T. Prem Jacob. 2018. "CGSA Scheduler: A Multi-Objective-Based Hybrid Approach for Task Scheduling in Cloud Environment." *Information Security Journal: A Global Perspective* 27(2):77–91. doi: 10.1080/19393555.2017.1407848.
- Praveen, S. Phani, K. Thirupathi Rao, and B. Janakiramaiah. 2018. "Effective Allocation of Resources and Task Scheduling in Cloud Environment Using Social Group Optimization." *Arabian Journal for Science and Engineering* 43(8):4265–72. doi: 10.1007/s13369-017-2926-z.
- Ramamoorthy, S., G. Ravikumar, B. Saravana Balaji, S. Balakrishnan, and K. Venkatachalam. 2021. "RETRACTED ARTICLE: MCAMO: Multi Constraint Aware Multi-Objective Resource Scheduling Optimization Technique for Cloud Infrastructure Services." *Journal of Ambient Intelligence and Humanized Computing* 12(6):5909–16. doi: 10.1007/s12652-020-02138-0.
- Singh, Harvinder, Sanjay Tyagi, Pardeep Kumar, Sukhpal Singh Gill, and Rajkumar Buyya. 2021. "Metaheuristics for Scheduling of Heterogeneous Tasks in Cloud Computing Environments: Analysis, Performance Evaluation, and Future Directions." *Simulation Modelling Practice and Theory* 111:102353. doi: 10.1016/j.simpat.2021.102353.
- Wei, Jing, and Xin-fa Zeng. 2019. "Optimal Computing Resource Allocation Algorithm in Cloud Computing Based on Hybrid Differential Parallel Scheduling." *Cluster Computing* 22(S3):7577–83. doi: 10.1007/s10586-018-2138-7.
- Wu, Chu-ge, and Ling Wang. 2018. "A Multi-Model Estimation of Distribution Algorithm for Energy Efficient Scheduling under Cloud Computing System." *Journal of Parallel and Distributed Computing* 117:63–72. doi: 10.1016/j.jpdc.2018.02.009.

- Zhu, Qing-Hua, Huan Tang, Jia-Jie Huang, and Yan Hou. 2021. "Task Scheduling for Multi-Cloud Computing Subject to Security and Reliability Constraints." *IEEE/CAA Journal of Automatica Sinica* 8(4):848–65. doi: 10.1109/JAS.2021.1003934.
- Zivkovic, Miodrag, Nebojsa Bacanin, Eva Tuba, Ivana Strumberger, Timea Bezdan, and Milan Tuba. 2020. "Wireless Sensor Networks Life Time Optimization Based on the Improved Firefly Algorithm." Pp. 1176–81 in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. Limassol, Cyprus: IEEE.