Received: March 8, 2023. Revised: May 4, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Min-Max Workload Scheduling Technique Using Soft-Computing Approach in Multi-Cloud Platform

Arundhati Nelli[1]* **Rashmi Jogdand[1]**

[1]*Department of Computer Science and Engineering, Karnataka Law Society's Gogte Institute of Technology, Belagavi, Karnataka, India*
* Corresponding author's Email: arundhatinelliresearch@gmail.com

Abstract: Maximizing resource utilization by minimizing service level agreement (SLA) violation is considered to be extremely difficult process for scheduling of realtime workload on multi-cloud platform. In particular the proposed work considers both energy and performance as SLA minimization constraint for scheduling of workload in multi-cloud platform. The scheduling optimization of SLA minimization is done as single objective function using dragonfly soft computing approach. Further, the scheduling optimization of min-max is done as multi-objective for maximizing resource utilization with minimal SLA violation using dragonfly soft computing approach. The results show that the min-max workload scheduling (MMWS) has improved the resource utilization by 16.02%, 7.92%, and 2.52% respectively for montage workload and 9.92%, 4.07%, and 1.11% for SIPHT workload in comparison to the existing workload-scheduling adaptive-dragonfly algorithm (WS-ADA), reliability multi cloud-scheduler (REL-MC) and service-level agreement-based workload-scheduling (SLA-WS) method respectively. The results show that the MMWS has reduced the SLA violation rate by 81.92%, 69.23%, and 30.11% respectively for montage workload and 84.57%, 61.32%, and 12.62% for SIPHT workload in comparison to the existing WS-ADA, REL-MC and SLA-WS methods respectively. Hence, from the experiment outcomes it is shown that the proposed MMWS technique reduces SLA violation with improved resource efficiency in comparison with WS-ADA, REL-MC and SLA-WS.

Keywords: Multi-cloud, Multi-objective optimization, Resource utilization, SLA violation, Workload scheduling.

1. Introduction

Professionals have been driven to use cloud computing services for running a wide range of applications because of the popularity of cloud computing and its attractive characteristics [1], like pay per use pricing [2], high scalability [3], and resource pooling [4]. Cloud Computing is currently being put to good use in the execution of task scheduling applications, in which one task (a node) is routed to another task (another node) via a path (an edge) which symbolizes the flow of data between the nodes [5]. Different workflow scheduling (WS) algorithms have been implemented in order to connect such workflows to cloud-resources taking into account cost and time restrictions [6]. Moreover, many businesses now have access to hosting-services because to advancements in cloud computing. Cloud computing services have many practical uses in industries like business and research [7]. Applications designed for business use are task-oriented and organized as per the business processes. Alternatively, scientific-applications are organized into scientific-workflows and focus on the data and computational aspects of their work. Systems designed specifically for the scheduling and management of scientific-workflows are used to oversee their operations [8]. Many studies have been conducted recently on the topic of organizing and scheduling scientific-workflow processes [9]. In [10, 11], it provides an in-depth analysis of the state of the art in cloud-based workflow scheduling and management for scientific applications. It summarizes previous research work for managing scientific-workflows. It provides a classification scheme for scientific process uses and features. Energy-efficient, fault-tolerant and

resource scheduling are only some of the scheduling and management methods for scientific-workflows that are illustrated in this paper. In addition to providing definitions and equations, it offers discussion on a variety of performance measurement metrics. It also discusses the numerous performance measurement systems used to assess scheduling and management techniques for scientific-workflows. Using a wide range of performance measurement characteristics, [10] identifies platforms that are put to use in assessing the methods of scientific-workflows. It also identifies a variety of presentation objectives for advanced scientific- workflow management methods.

Cloud computing is amongst the most significant strategies in the domain of distributed computing, according to [12]. This approach helps to preserve higher flexibility and scalability in high performance computing, that leads to enhanced flexibility and scalability. This is made possible due to the fact that it utilizes the internet to give a variety of services and shared services at low or no cost. Cloud service providers provide a wide variety of operations to their users, with the goal of satisfying the users requirements while maintaining the highest possible standards of quality-of-service (QoS) at costs that are affordable and reasonable. Because of the need to perform operations while taking QoS requirements such as resource utilization, energy efficiency etc. into account, cloud computing presents a significant scheduling difficulty. Moreover, service-level-agreement (SLA) must be reached among users and cloud service providers in order to define the nature of their connection. Furthermore, in order to specify the terms under which their QoS obligations will be met, there must be an explicit SLA. Although there are many energy-aware resource management solutions for cloud data-centres, existing approaches focus on minimizing energy consumption while ignoring the SLA violation at the time of virtual machine (VM) deployment [13]. Also, they do not consider the types of application running in the VMs and thus may not really reduce energy consumption with minimal SLA violation under a variety of workloads. Hence due to all these issues there is a requirement to minimize the SLA violation. In this paper a min-M\max workload scheduling model is designed to maximize resource utilization with minimal SLA violation by employing dragonfly algorithm. The significance of using the proposed min-max workload scheduling (MMWS) is given below

- The MMWS is efficient in minimizing reducing SLA violation and maximizing resource utilization.
- The optimization to solve multi-objective parameter is done employing dragonfly soft computing algorithm.
- Experiments outcome shows the proposed model reduces SLA violation and maximize resource utilization in comparison with existing workload scheduling algorithm.

In this work, in the section 2, survey on various workflow scheduling methodologies has been done. Further, in section 3, the min-max workload scheduling technique for the multi-cloud platform has been proposed. In the section 4, the results have been discussed and the proposed min-max workload scheduling technique has been compared with the existing workload-scheduling adaptive-dragonfly algorithm (WS-ADA), reliability multi cloud-scheduler (REL-MC) and service-level agreement-based workload-scheduling (SLA-WS). Finally, in the section 5, the conclusion of the complete work has been given.

2. Literature survey

This section studies various existing workload scheduling method and identifies its benefits and limitation. In [14], the researchers present two effective task scheduling strategies for cloud environments, each of which takes cost and makespan into account. To begin, researchers offer deadline-constrained cost-optimization for hybrid-clouds (DCOH), a method for optimizing overall planning of workflows with a specific goal in mind: reducing the cost expense of doing so while meeting strict timeframes. In addition to DCOH, researchers present multi-objective-optimization for hybrid-clouds (MOH), a method for improving cost and makespan cost of planning operations concurrently. To prove MOH and DCOH efficiency, researchers have run several simulation studies. The findings of their simulations suggest that their DCOH method can cut consumer costs by as much as 100 percent comparing to other methods working with the same time constraints, and also that their MOH method can provide superior cost-makespan-trade-off approaches. In [15], researchers present the adaptive-dragonfly-algorithm (WS-ADA) as a revolutionary load-balancing-task-scheduling-algorithm for the cloud. Both the dragonfly method as well as the firefly method are components of the ADA. Additionally, a multi-objective mechanism dependent upon three criteria (load, completion time, and processing cost) is designed with the goal of improving efficiency. Various measures, including

executing cost and executing time, are used to assess the efficiency of the presented algorithm. As shown experimentally, the presented method achieves superior load-balancing outcomes in comparison to the alternatives. Cost optimized scheduling techniques inside the cloud were explored in [16], as well as a new task dividing algorithm called cost-optimized-heuristic-algorithm (COHA) was presented for the cloud-scheduling to optimize computation cost. To speed up this algorithm's execution, complex tasks are broken down into smaller ones. The goal of the architecture is to ensure that all jobs are completed in time. They have thoroughly evaluated the COHA's functionality by feeding it a variety of workflow inputs. Results from simulations show that COHA performs well when allocating and deploying virtual machines and therefore can manage random arriving jobs with ease. It can help keep execution-costs down while ensuring that all jobs are completed in a timely manner. Furthermore, overall execution-cost of SIPHT-workflows has been lowered by 32.5% because of their algorithmic enhancements, while montage-workflows have seen a 3.9% reduction, and CyberShake-workflows have seen a 1.2% reduction.

In [17], they built their algorithm on earlier work by utilising scientific procedures and a cost-optimized-heuristic-algorithm (COHA). MOWOS uses a tasks-splitting technique to break down long jobs into more manageable chunks. In addition, MOWOS introduces two novel algorithms regarding work allocations: the maximum virtual machine (MaxVM) decision and the minimum virtual machine (MinVM) decision. MOWOS was created so that all projects may be completed on time and under budget. MOWOS has been put through its paces using a variety of workflows to ensure its efficacy. The simulated findings show that MOWOS can efficiently allocate and deploy virtual machines as well as deal with streaming-tasks arriving at an unpredictable frequency. When comparing to the state-of-the-art algorithms, the suggested method performs much better on extra-large and large-workflow-tasks rather than on medium and small-workflow-tasks. As a result, this algorithm has the ability to to enhance resource utilization by 53%, decrease costs by 8%, decrease makespan by 10%, and ensure that all activities are completed on time. To maximize resource utilization, [18] tackles the workflow-scheduling issue. The presented QoS-based resource-allocation and resource-scheduling employs particle-swarm-based ant-colony-optimization (PS-ACO) to deliver improved reliable findings in order to avoid the scheduling-optimization challenge. Presented algorithms have been tested in a virtual cloud setting. When the presented algorithm's outcomes were measured against those of alternative strategies, it came out on top in terms of QoS metrics. It is studied in [19] how to schedule a concurrent workflow in such a way as to decrease the system's power utilization while yet meeting feedback-time and reliability-requirements. They offer a processor merging method as well as a slack-time-reclamation approach that makes use of a dynamic-voltage-frequency-scaling (DVFS) approach for cutting down on power utilization after they've developed a technique which minimizes overall completion time with satisfying reliability-requirement. Overall energy efficiency of the system is prioritized by the processor merging method, which disables less vital components. To save power, the DVFS method is used to decrease the speed of the CPU both in the task and the overall system levels. Their performance has been demonstrated experimentally on two real-world workflows and in-depth synthetic parallel-workflows.

In [20], a method for scheduling workflow-tasks with in cloud that is both energy efficient and reliable was described. This algorithm, called energy-efficient and reliability-aware workflow-task-scheduling (EERS), minimizes power consumption while increasing reliability. The EERS was tested just on WorkflowSim simulation with the CyberShake as well as montage scientific workloads to determine how well they performed. According to the findings, it is superior than competing methods in this area. To decrease carbon emissions and meet workflow-reliability limitations, the authors of [21] devised an algorithm called reliability-aware and energy-efficient workflow scheduling (REWS) for scheduling processes. To decompose the workflow-reliability-constraint into task sub-reliability-constraints, REWS takes a novel approach and demonstrates its performance using a sub-reliability-constraint prediction mechanism. In addition, an updating mechanism is used to modify overall task-sub-reliability-constraint in order to lower the overall power consumption. In contrast towards the optimization technique of REWS, a small system architecture is constructed. This architecture is made up of five parts: a workflow-analyzer, reliability-decomposer, resource-manager, workflow-scheduler, and feedback-processor. Researchers have performed studies with both simulated and real data to test the effectiveness of the REWS method. The outcomes show that REWS is far superior than the current best algorithms. In [22], a stochastic-multi-workflows-dynamic-scheduling-algorithm (SMWDSA) is developed to optimize the cost of

scheduling many workflows simultaneously while satisfying deadline restrictions. Multi-workflow pre-processing, scheduling of multiple workflows, as well as scheduling-feedback are the three components of the presented SMWDSA. For the sake of completing several workflows by their respective due dates, SMWDSA employs a new task-sub-deadlines allocation technique. Furthermore, for reducing workflow-scheduling costs while still achieving workflow-deadlines, they offer a task-scheduling approach predicated on the lowest time-slot-availability to execution-task. Furthermore, in order to further decrease workflow-scheduling costs, a scheduled-feedback technique is employed to alter the priority as well as sub-deadlines for unexpected tasks. In order to test SMWDSA, they used both simulated and real-world data in their tests. This shows that SMWDSA is better than the current best algorithms.

To address time-sensitive cloud-based workflow-scheduling challenges like hibernation and pay-per-second pricing, the authors of [23] suggested a hybrid heuristic method termed enhanced-task-type-first-algorithm (ET2FA). Overall cost and overall idle time are two of the goals to be minimized. Each stage of ET2FA consists of a "task-type-first" technique, that uses a compressed dependent VM selection technique to allocate jobs according to their topology tier and task-type. Further, delay the operation-based on the block-structure, that further optimizes the overall costs and overall idle-rate. Thirdly, a strategy for scheduling instances to hibernate while they are not being actively used is shown. ET2FA outperforms the current existing methods in extensive simulation trials that simulate the behavior of seven popular real-world workflow-applications. Execution time and cost are two of the measures used to determine how well the suggested approach performs. As shown experimentally, the proposed method achieves superior load balancing outcomes compared to the alternatives. In [24], they suggested a strategy for continuously performing vertical and horizontal cloud-resource-scaling across different types of virtual machine instances in order to execute complex-workflows. To achieve resource-scaling, a depth-first-search-coalition-reinforcement-learning (DFSCRL) provisioning strategy is introduced. This strategy combines the establishment of physical-machine (PM) coalitions with the Q-learning technique, and afterwards adaptively develops an ideal multi-type virtual-machine instance package from the physical-machine coalition. The suggested techniques outperform state-of-the-art equivalent strategies, as shown by mathematical proofs and multiple experiments using the multidimensional metrics. Comparing these findings to those of currently in use systems, they find that they are superior. In [25], this paper proposes a fault-tolerant-cost-efficient-workflow-scheduling-algorithm (FCWS) called as reliability multi cloud-scheduler (REL-MC) which reduces application processing reliability, cost, and time by defining the directed-acyclic graph (DAG) tasks somewhere at bottom layer that are most cost-effective to execute. To begin with, they developed a fault-tolerant-workflow-scheduling architecture for use across many cloud platforms, with the goal of lowering the cost of executing scientific-applications while simultaneously increasing their dependability of execution. Furthermore, they replicated the high-risk tasks using Weibull-distribution analysis of task completion accuracy and risk rate. Finally, they have established the bottom layer of cost for DAG activities and proposed a FCWS to minimize the cost and time required to execute applications while guaranteeing their dependability. Their approach has been tested experimentally through the use of scientific procedures (LIGO, Epigenomics). As can be seen from the findings, their suggested FCWS algorithm provides significant improvements over both the existing fuzzy-resource-utilization based on particle-swarm-optimization (FR-MOS) and cost-efficient workflow-scheduling (CWS) in terms of both cost and reliability.

In [14], they have only reduced the cost and maskespan by completing the task within the deadline. In [15-17, 23] they have addressed only the load-balancing and reducing the cost issues and have not addressed the SLA violation. In [18, 22] they have addressed the resource allocation and resource scheduling the tasks of the workflow and completing the task in the given deadline. In this work their main focus was to provide better QoS for the user during the execution of the workflow. In [19-21] they have reduced the energy consumption for providing better reliability during the execution of the workflow in the cloud. Further, in [24], they have provided a resource-scaling method for execution of the workflow which will enhance the QoS. Finally, in [25], they have proposed an algorithm to provide reliability and reduce the cost and time for scheduling the workflows. From all the above studies it is seen that none of the study have addressed the issue of resource utilization and SLA violation in their work. Hence, in this study a min-max workload scheduling technique to address the issues of SLA violation and resource utilization has

Table 1. Notation table

Variable	Definition
y_k	Workload
c_k	Arrival Time
f_k	Quality Requirement
I_k	Workload Dependencies
V_k	Task Set of Workload y_k
G_k	Task Set of Workload y_k
v_r^k's	Previous Sub-Task of The Workload y_k
$hv_{r,mn}^k$	Processing Time Needed for Execution of Sub-Task v_r^k'S
vv_{rl}^k	Processing Required for Communicating Between the Sub-Task v_r And v_k
$tu_{l,mn}^k$	Total Processing Time of Upcoming Task
G_k	Task Dependencies Among Sub-Task
g_{rl}^k	Edge That Belongs To G_k
hv_k	Maximum Time Needed For Completing The Execution Of The Workload y_k
p	Physical Machine Size
xt	Initialization Time
yt	Completion Time
u_m	Minimum Energy Needed for Running Machine
$r_m^{\uparrow}$	Maximum Energy Level of Physical Machines
a_m^v	Boolean parameter that represents the PMs is active or not
o	Workload size of Y and $\|V_k\|$ defines workload y_k task size
cpu_l^k	Task V_L^K'S Processing Frequency Requirement
$\mathcal{U}_l^k$	Overall Time It Has Taken for Completing Workload Task
$\mathcal{B}_m$	Active Hours of Physical Machine

been presented, which has been described in the below section.

3. Min-max workload scheduling technique in multi-cloud platform

This section present min-max workload scheduling technique in multi-cloud platform. The MMWS is focused to reduce the SLA violation and improve resource utilization. The architecture of workload scheduling in multi-cloud platform is given in Fig. 1. The notations used in the technique have been given in the Table 1.

This paper introduces an SLA-based scheduling that minimize energy and processing time for workload execution under multi-cloud platform. The workload y_k is represented as follows

$$y_k = \{c_k, f_k, I_k\}, \quad (1)$$

where c_k, f_k, I_k defines arrival time, quality requirement, and workload dependencies, respectively. The dependencies are modeled as a directed acyclic graph as follows

$$I_k = (V_k, G_k) \quad (2)$$

where V_k and $G_k$describes task sets of workloads y_k and defines task dependencies among different task.

The data dependencies among subtask minimizing SLA violation are expressed using following equation

$$hv_{r,mn}^k + vv_{rl}^k \leq tu_{l,mn}^k, \ \forall g_{rl}^k \in G_k \quad (3)$$

where $hv_{r,mn}^k$ represent the processing time needed for execution of sub-task v_r^k's (previous task), vv_{rl}^k defines processing required for communicating between the sub-task v_r and v_k, $tu_{l,mn}^k$ defines the total processing time of upcoming task, g_{rl}^k defines an edge that belongs to G_k and G_k defines task dependencies among sub-task. The maximum time needed for completing the workload y_k with given multi-cloud platform is defined as follows

$$hv_k = \max_{v_l^k \in V_k} \{hv_{l,mn}^k\}. \quad (4)$$

The proposed scheduler should meet SLA requirement given in below equation

$$hv_k \leq f_k, \ \forall y_k \in Y. \quad (5)$$

where f_k defines quality requirement of different task of respective workload. Therefore, getting the resource for execution of workload with minimal SLA violation constraint defined in Eqs. (6) and (7) is difficult as described

$$h_m^{\uparrow} - \sum_{n=1}^{|W_m|} h_{m,n} \geq 0, \ \forall j_m \in J; \quad (6)$$

$$o_m - \sum_{n=1}^{|W_m|} o_{m,n} \geq 0, \ \forall j_m \in J. \quad (7)$$

where j_m defines the physical machine that belongs to J, $h_m^{\uparrow}$ defines processing element maximum frequency level, $h_{m,n}$ defines real processing element frequency, o_m defines the memory size, and W_m defines the virtual computing nodes. The SLA violation minimization scheduler is designed through following minimization equation

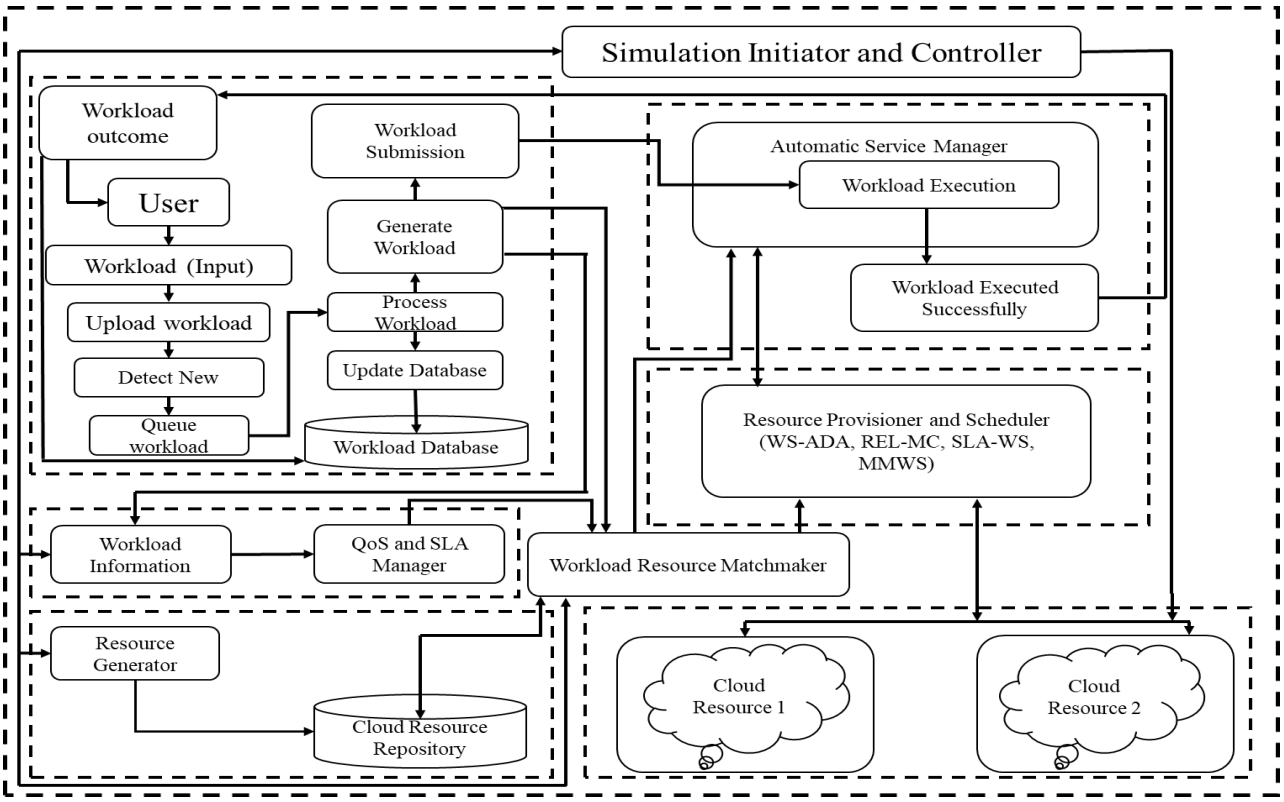

Figure. 1 Workload execution model under multi-cloud platform

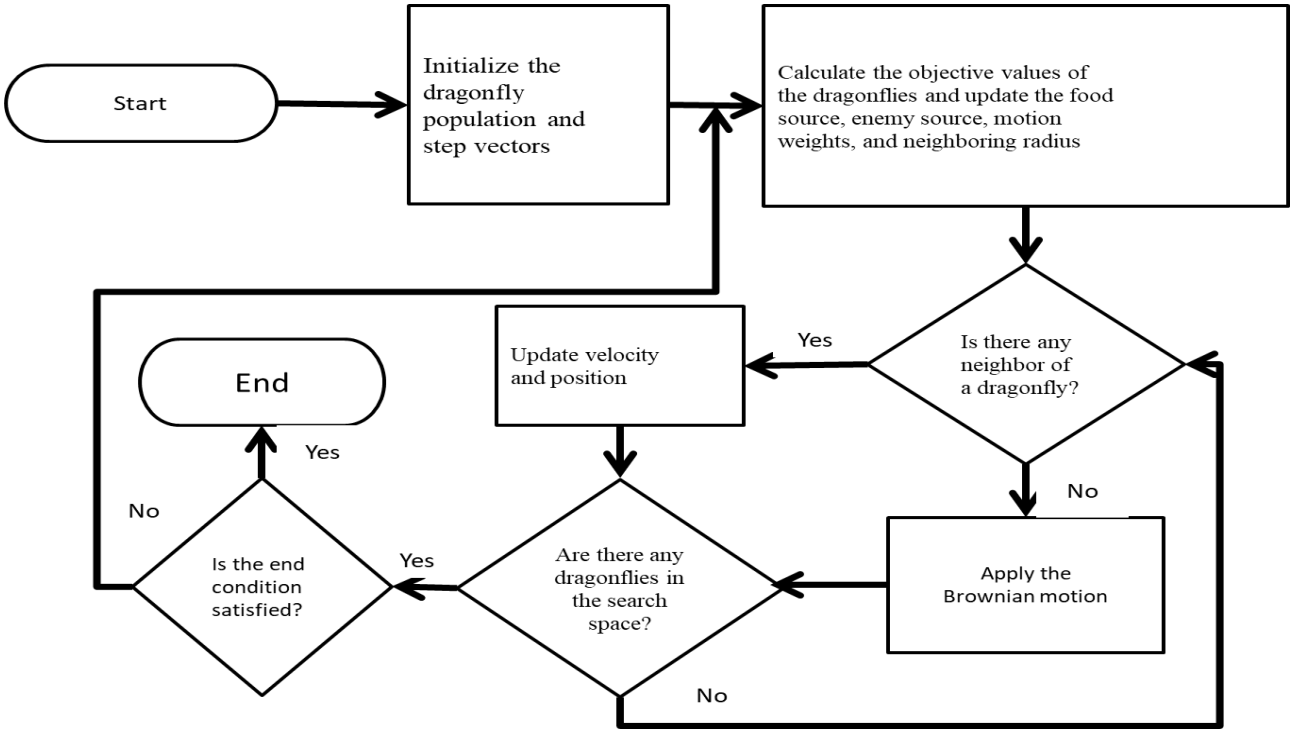

Figure 2. Flow diagram of dragonfly algorithm

$$A = Min \sum_{m=1}^{p} \int_{xt}^{yt} \left(u_m * r_m^{\uparrow} * a_m^{v} + \frac{(1-u_m)*r_m^{\uparrow}}{\left(h_m^{\uparrow}\right)^3} * \left(h_m^{f}\right)^3 \right) dt. \quad (8)$$

where p defines physical machine size, xt and yt defines initialization and completion time, respectively, u_m defines minimum energy needed for running machine, $r_m^{\uparrow}$ maximum energy level of PMs, a_m^{v} is Boolean parameter that represent the PMs is active or not. The resource can be utilized

better using following maximization equation

$$B = \frac{Max\left(\sum_{k=1}^{o}\sum_{l=1}^{|V_k|} cpu_l^k * \mathcal{U}_l^k\right)}{\left(\sum_{m=1}^{p} h_m^{\uparrow} * \mathcal{B}_m\right)} \quad (9)$$

where o represent the workload size of Y and $|V_k|$ defines workload y_k task size, cpu_l^k denote task v_l^k 's processing frequency requirement, $\mathcal{U}_l^k$ denotes the overall time it has taken for completing workload task, p describes PMs in multi-cloud platform, and $\mathcal{B}_m$ defines active hours of PMs j_m's. The dragonfly optimization algorithm is used to maximize the resource usage using Eq. (8) with minimal SLA violation using Eq. (9).

The working procedure dragonfly algorithm is shown in Fig. 2. The proposed min-max workload scheduling optimized with dragonfly algorithm achieves much improved scheduling performance in multi-cloud platform in comparison with existing workload scheduling approaches.

4. Results and discussion

In this work the performance achieved using MMWS with other workload scheduling technique namely workload-scheduling adaptive-dragonfly algorithm (WS-ADA) [15], reliability multi cloud-scheduler (REL-MC) [25] and service-level agreement-based workload-scheduling (SLA-WS) [26]. The SLA-violation rate and resource utilization are metrics used for validating workload scheduling model. The scientific workload such as SIPHT and cybershake are workload sued to validate the model. The resource utilization is measured using following equation

$$RU = \frac{S_U}{S_A} \quad (10)$$

where S_U defines the total number of slots used and S_A defines total number of slots allocated. Similarly, the SLA violation is measured using following equation

$$SLAV = \frac{SLA(t)}{l_{i+1}} \quad (11)$$

where it is measured as for an interval SI_i as the mean number of SLA violation in respective interval for leaving task l_{i+1}, $SLA(t)$ of task T is product of two metrics such as performance degradation due to migration and SLA violation time per active physical machine.

4.1 Resource utilization

In this section experiment is conducted for studying the resource utilization performance of proposed MMWS with other existing workload scheduling technique namely SLA-WS, WS-ADA, and REL-MC. The Fig. 3 shows the resource utilization performance of different workload scheduling technique for execution of montage workload in multi-cloud platform. Similarly, the Fig. 4 shows the resource utilization performance of different workload scheduling technique for execution of SIPHT workload. From Figs. 3 and 4 we can interpretate the proposed MMWS achieves much better resource utilization performance in comparison to SLA-WS, WS-ADA, and REL-MC. The results show that the MMWS has improved the resource utilization by 16.02%, 7.92%, and 2.52% respectively for Montage Workload and 9.92%, 4.07%, and 1.11% for SIPHT workload in comparison to the existing Workload-Scheduling WS-ADA, REL-MC and SLA-WS method respectively.

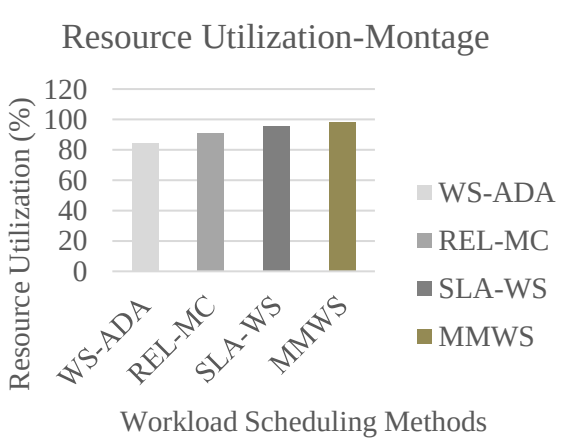

Figure. 3 Resource utilization for montage workload

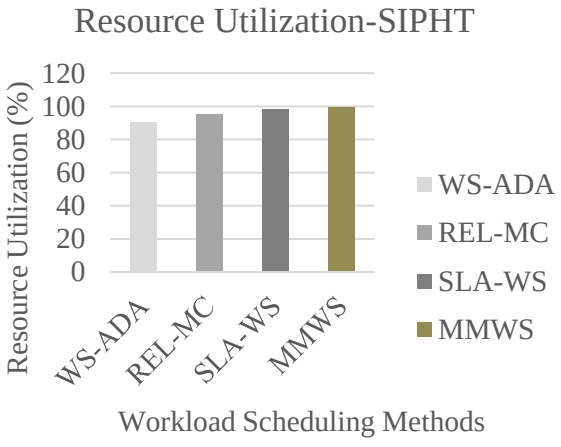

Figure. 4 Resource utilization for SIPHT workload

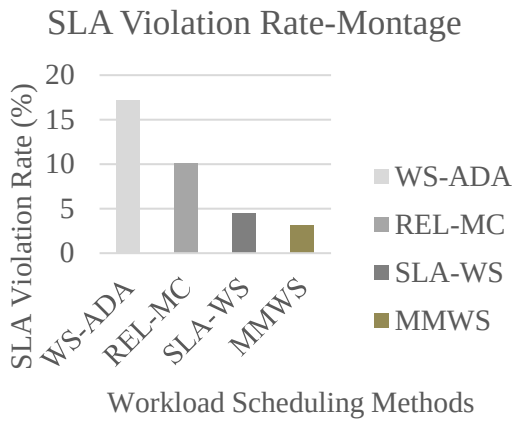

Figure. 5 SLA violation rate for montage workload

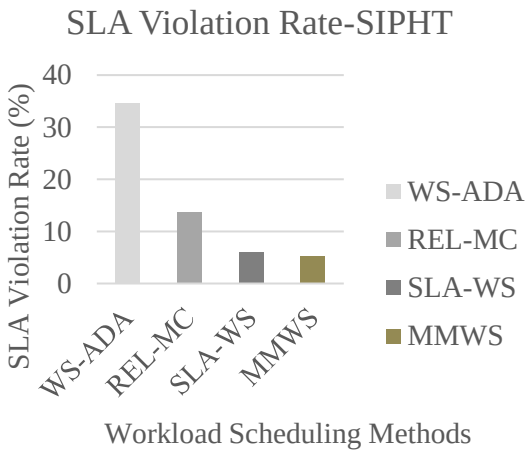

Figure. 6 SLA violation rate for SIPHT workload

4.2 SLA Violation rate

In this section experiment is conducted for studying the SLA violation rate performance of proposed MMWS with other existing workload scheduling technique namely SLA-WS, WS-ADA, and REL-MC. The Fig. 5 shows the SLA violation rate performance of different workload scheduling technique for execution of montage workload in multi-cloud platform. Similarly, the Fig. 6 shows the resource utilization performance of different workload scheduling technique for execution of SIPHT workload. From Figs. 5 and 6 we can interpretate the proposed MMWS can significantly reduce SLA violation in comparison to SLA-WS, WS-ADA, and REL-MC. The results show that the MMWS has reduced the SLA violation rate by 81.92%, 69.23%, and 30.11% respectively for Montage Workload and 84.57%, 61.32%, and 12.62% for SIPHT workload in comparison to the existing WS-ADA, REL-MC and SLA-WS methods respectively.

Table 2. Comparative study

	Cost	Energy	Resource Utilization	SLA Violation
WS-ADA [15]	✓	×	×	×
REL-MC [25]	✓	×	×	×
SLA-WS [26]	×	✓	×	×
MMWS [Proposed]	×	✓	✓	✓

4.3 Comparative study

In this section the comparative study for comparing the proposed MMWS technique with the existing WS-ADA, REL-MC and SLA-WS has been done. The comparative study has been given in Table 2. In the Table 2, the WS-ADA and REL-MC methods address only the cost metric. Further, the SLA-WS reduces the energy during the execution of the workflow. The proposed method reduces the cost by utilizing the resources efficiently and also reduces the SLA-Violation.

5. Conclusion and future work

The MMWS model achieves very good result in comparison with recent workload scheduling model such as SLA-WS, WS-ADA, and REL-MC. The MMWS is designed to reduce SLA violation and maximize the resource utilization under multi-cloud platform. The results show that the Min-Max Workload Scheduling (MMWS) has improved the resource utilization by 16.02%, 7.92%, and 2.52% respectively for Montage Workload and 9.92%, 4.07%, and 1.11% for SIPHT workload in comparison to the existing workload-scheduling adaptive-dragonfly algorithm (WS-ADA), reliability multi cloud-scheduler (REL-MC) and service-level agreement-based workload-scheduling (SLA-WS) method respectively. The results show that the MMWS has reduced the SLA violation rate by 81.92%, 69.23%, and 30.11% respectively for montage workload and 84.57%, 61.32%, and 12.62% for SIPHT workload in comparison to the existing WS-ADA, REL-MC and SLA-WS methods respectively. The adoption of dragonfly optimization algorithm aided in identifying SLA-aware resource leveraging multi-cloud platform. Using such strategies MMWS model can offer high scalability

and achieves robust performance considering both larger tasks. Future work would consider validating the MMWS considering different dataset. Alongside would further optimize the scheduling through effective task ordering or replanning mechanism.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Both authors contributed to the study, conception and design. Material preparation, data collection and analysis were performed by Arundhati Nelli and Rashmi Jogdand. Both authors read and approved the final manuscript.

References

[1] M. Bhandari, V. Gutte, and P. Mundhe, "A Survey Paper on Characteristics and Technique Used for Enhancement of Cloud Computing and Their Security Issues", *Pervasive Computing and Social Networking*, Vol. 317, pp. 217–230, 2022.

[2] N. Dimitri, "Pricing Cloud Iaas Computing Services", *Journal of Cloud Computing*, Vol. 9, No. 1, 2020.

[3] A. Ahmad and Peter Andras, "Scalability Analysis Comparisons of Cloud-Based Software Services", *Journal of Cloud Computing*, Vol. 8, No. 1, 2019.

[4] R. Tanash, A. Khalifeh, and K. A. Darabkh, "Communication over Cloud Computing: A Security Survey", In: *Proc. of 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, pp. 496-501, 2019.

[5] S. Murad, A. Muzahid, Z. Azmi, M. Hoque, and M. Kowsher, "A Review on Job Scheduling Technique in Cloud Computing and Priority Rule Based Intelligent Framework", *Journal of King Saud University Computer and Information Sciences*, Vol. 34, No. 6, pp. 2309–2331, 2022.

[6] M. Sana and Z. Li, "Efficiency Aware Scheduling Techniques in Cloud Computing: A Descriptive Literature Review", *PeerJ Computer Science*, Vol. 7, 2021.

[7] S. Bello, L. Oyedele, O. Akinade, M. Bilal, J. Delgado, L. Akanbi, A. Ajayi, and H. Owolabi, "Cloud computing in construction industry: Use cases, benefits and challenges", *Automation in Construction*, Vol. 122, pp. 103441, 2021.

[8] D. Talia, "Workflow Systems for Science: Concepts and Tools", *International Scholarly Research Notices*, Vol. 2023, No. 404525, pp. 1-15, 2023.

[9] M. Adhikari, T. Amgoth, and S. Sriram, "Multi-objective scheduling strategy for scientific workflows in cloud environment: A Firefly-based approach", *Applied Soft Computing*, Vol. 93, p. 106411, 2020.

[10] Z. Ahmad, A. Jehangiri, M. Ala'anzy, M. Othman, R. Latip, S. Zaman, and A. Umar, "Scientific Workflows Management and Scheduling in Cloud Computing: Taxonomy, Prospects, and Challenges", *IEEE Access*, Vol. 9, pp. 53491-53508, 2021.

[11] D. Hejji, A. Nassif, Q. Nasir, and M. AbuTalib, "Metaheuristics-based Approach for Workflow Scheduling in Cloud", In: *Proc. of 2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, pp. 1-5, 2020.

[12] A, Edinat, R. Sayyed, and A. Hudaib, "A Survey on Improving QoS in Service Level Agreement for Cloud Computing Environment", *International Journal of Interactive Mobile Technologies (iJIM)*, Vol. 15, No. 21, 2021.

[13] Z. Zhou, J. Abawajy, M. Chowdhury, Z. Hu, K. Li, H. Cheng, A. Alelaiwi, and F. Li, "Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms", *Future Generation Computer Systems*, Vol. 86, pp. 836-850, 2018.

[14] J. Zhou, T. Wang, P. Cong, P. Lu, T. Wei, and M. Chen, "Cost and makespan-aware workflow scheduling in hybrid clouds", *Journal of Systems Architecture*, Vol. 100, p. 101631, 2019.

[15] P. Neelima and A. Reddy, "An efficient load balancing system using adaptive dragonfly algorithm in cloud computing", *Cluster Computing*, Vol. 23, pp. 2891–2899, 2020.

[16] J. Konjaang and L. Xu, "Cost Optimised Heuristic Algorithm (COHA) for Scientific Workflow Scheduling in IaaS Cloud Environment", In: *Proc. of 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, pp. 162-168, 2020.

[17] J. Konjaang and L. Xu, "Multi-objective workflow optimization strategy (MOWOS) for cloud computing", *Journal of Cloud Computing*,

Vol. 10, No. 11, 2021.
[18] H. Singh, A. Bhasin, and P. Kaveri, "QRAS: efficient resource allocation for task scheduling in cloud computing", *SN Application and Science*, Vol. 3, No. 474, 2021.
[19] B. Hu, Z. Cao, and M. Zhou, "Energy-minimized Scheduling of Real-time Parallel Workflows on Heterogeneous Distributed Computing Systems", *IEEE Transactions on Services Computing*, Vol. 15, No. 5, pp. 2766-2779, 2022.
[20] R. Medara and R. Singh, "Energy Efficient and Reliability Aware Workflow Task Scheduling in Cloud Environment", *Wireless Personal Communication*, Vol. 119, pp. 1301–1320, 2021.
[21] L. Ye, Y. Xia, S. Tao, C. Yan, R. Gao, and Y. Zhan, "Reliability-Aware and Energy-Efficient Workflow Scheduling in IaaS Clouds", *IEEE Transactions on Automation Science and Engineering*, pp. 1-14, 2022.
[22] L. Ye, Y. Xia, L. Yang, and Y. Zhan, "Dynamic Scheduling Stochastic Multiworkflows With Deadline Constraints in Clouds", *IEEE Transactions on Automation Science and Engineering*, pp. 1-13, 2022.
[23] Z. Sun, B. Zhang, C. Gu, R. Xie, B. Qian, and H. Huang, "ET2FA: A Hybrid Heuristic Algorithm for Deadline-constrained Workflow Scheduling in Cloud", *IEEE Transactions on Services Computing*, pp. 1-14, 2022.
[24] X. Wang, J. Cao, and R. Buyya, "Adaptive Cloud Bundle Provisioning and Multi-Workflow Scheduling via Coalition Reinforcement Learning", *IEEE Transactions on Computers*, Vol. 72, No. 4, pp. 1041-1054, 2022.
[25] X. Tang, "Reliability-Aware Cost-Efficient Scientific Workflows Scheduling Strategy on Multi-Cloud Systems", *IEEE Transactions on Cloud Computing*, Vol. 10, No. 4, pp. 2909-2919, 2021.
[26] A. Nelli and R. Jogdand, "SLA-WS: SLA-based workload scheduling technique in multi-cloud platform", *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-12, 2022.