International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Optimizing QoS and Energy using ESSA for Budget-constrained Workflows in the Cloud

C. K. Sripavithra[1, 2] and V. B. Kirubanand[1]

Submitted: 03/02/2024 **Revised**: 11/03/2024 **Accepted**: 17/03/2024

Abstract. Cloud computing has acquired colossal interest. Here resources are observed as services; hence employment of it in an organized manner is performed in two integral parts, i.e., asset provisioning as well as process organizing. Workflows are big data applications in the distributed community. Cloud infrastructures' flexibility makes them an appropriate choice for scientific workflow computations. An effective task scheduling technique must determine the order of tasks in the workflow for processing so as to satisfy the needs of the client. Most of the procedures concentrate on upgrading resource usage and fulfilling the QoS. Furthermore, these workflow computations require more computing power and the energy consumed to meet those requests are neglected. This fact persuaded us to present an Enhanced Salp Swarm Algorithm (ESSA) for scheduling scientific workflows with energy consciousness. On evaluating the potentiality, the proposed ESSA scheduling technique outperformed three other approaches in the simulation of synthetic scientific job trials. The solutions portray reduced execution time and energy usage by raising total resource consumption and employment.

Keywords: *Optimisation, QoS, cloud computing, mathematical model, makespan, budget, and ESSA.*

1 Introduction

A cloud domain structured by numerous resources and services is in demand from people and firms. It delivers the clients with exceptionally versatile software and hardware via the Internet. [1] The different cloud services and models offer a better option for executing scientific and big data workflows. [2] In the current trend, organizations are information-driven, and the cloud domain data centers are infrastructure paramount. Virtualization innovation is a crucial concept in resource utilization and keep-up the on-demand service model of the cloud. It allows the possibility of operating numerous Virtual Machines (VMs) inside the configuration of one Physical Machine (PM). The huge growth in computing requests is owing to the rapid expansion of cloud applications. It becomes unbearable for the cloud data center because of the increased utilization of energy while satisfying the requests for extensive resources. [3] The cloud services and the Service Level Agreements (SLAs) are correlated. There, preserving a deal linking power utilization and execution is critical.

A workflow is characterized by assembling several atomic tasks with data dependency [4]. Processing large-scale tasks usually have the demanding aspects. An application consists of a succession of interdependent phases, and each step includes a considerable count of autonomous jobs where there might be data exchanges among jobs of various phases. A workflow is expressed in Directed Acyclic Graph (DAG). Here jobs are mapped as nodes and the edges show the data dependency. The jobs are then doled out to processors utilizing a planning technique and this process of delineating is NP-hard [5]. A crucial problem in cloud-based process computation is the budget limitation specified by the cloud clients. The high data dependency in the tasks in the workflows and the movement of data from one resource to another provisioned resource increases the expenditure. To deal with the huge expansion in the request for workflow job assignments and ensure negligible energy cost, the data centers are in need of energy-conscious techniques for the cloud assets. However, the current research community is focusing on the development of techniques that concentrate on improving performance and energy without SLA infringement. In this environment, the provisioning of jobs to the resources by fulfilling the QoS terms is a crucial one, especially in the case of large-scale scientific complex workflows. [6] As of late, much research concentrates out on the effective consumption of cloud assets and decreasing energy utilization. Most prominently, the research works [7, 8] have utilized empirical techniques to determine the limitations in workflow planning. Still, more improvement is required in energy-conscious provisioning techniques without asset wastage. Therefore, this paper focuses on designing an energy-conscious scheduling technique coupling with VM compute capacity in budget constraints.

This paper is designed in this manner. Section II shows some related research works. Section III proposes the optimization method for budget workflow scheduling. Section IV explains implementation and the outcomes. Finally, sections V presents the conclusion and future avenues.

[1] *Central Campus, Christ (Deemed to be University), Bangalore, India kirubanand.vb@christuniversity.in*
[2] *Maharani's Science College for Women, Mysore, India sripavithra.ck@res.christuniversity.in*

2 Related work

The pace of study on the road to cloud computing is developing as surveyed over the years. [9] Ahmed et al. [10] and Zhang et al. [11] summarized the cloud environment and focused on managing the assets of the cloud. On the report of the work accomplished by Jennings [9], it is a herculean task managing the diverse assets of cloud data centers, the changing and unconventional load, in addition to the various requirements.

Niloofar et al. [12] remarked that multiple variables are responsible for influencing the performance of a scheduling mechanism. Hence, assessing the provisioning procedure with respect to one variable is not adequate. Hence, we consider in this paper four different variables for the evaluation of the suggested technique.

Numerous workflow provisioning techniques concentrated on improving the execution time and the price factors [13]. Zhou, Xiumin, et al. [14] presented Heterogeneous Earliest Finish Time (HEFT) approach to reduce the price and the execution time of the workflows. DAG splitting strategy in [15] for big data workflow job provisioning minimizes the price by raising the VM use. A list-based system in [16] offers the timeliest execution time. It is seen that these operated on bi-objective streamlining issues; still not tend to the energy objective.

The latest experimentation works considered various methods to reduce the energy usage of the work process job provisioning in clouds. In [17], an energy-aware procedure for deadline-limitation workflow, making application of the DVFS (dynamic voltage and frequency scaling) procedure, is presented. By fulfilling SLAs, the cost and utilization of resources were also taken into account in this case..

A Game-Score simulator is presented in [18] for setting up aligned jobs. It is an exchange connecting energy utilization and execution time. Li, Chunlin, et al. [19] suggested a disseminated cloud that improves asset usage by computing the job processing time. A multi-goal technique in [20] reduces price and execution time and improves asset usage using PSO.

The PESVMC procedure [21] united the VMC issue with the VM provisioning to set up the work process jobs. Li, Zhongjin, et al. [22] presented a price and energy-economical technique consisting of orderly and parallel job combing, VM reoperation and slack methods for conserving power.

Yao et al. [23] devised a job replication-type provisioning technique that creates a budget for every job and gives a proportional budget. The TDSA uses idle slots to duplicate, improving the time and asset utilization. Long et al. [24] suggested energy organized VM opening method where the picking of VM's is viable within the specified standards. Static ones are pitched using this method, which lowers energy utilization.

An ACO algorithm VM provisioning presented in [25] used the vector algebra for VM consolidation that minimized the energy consumption and increases the asset usage. R. Medara and R.S. Singh [26] presented a trustworthy and power conscious water wave technique to proficiently use VMs to conserve energy. But, real world workflows were not experimented in evaluation.

Sripavithra et al. presented a cost-based work duplicate technique Enhanced Salp Swarm Algorithm (ESSA) that reduces the overall functioning time as well as improves the asset allotment [27].

Proficient planning and provisioning are crucial for improved asset usage at a diminished price in the cloud and satisfying the QoS goals. The present provisioning schemes analyze the processing time and the expense; however, different extra attributes should likewise be regarded. It is essential to make an energy-productive planning to diminish energy utilization.

3 System Model

Fig. 1 displays the overall procedure of the proposed technique.

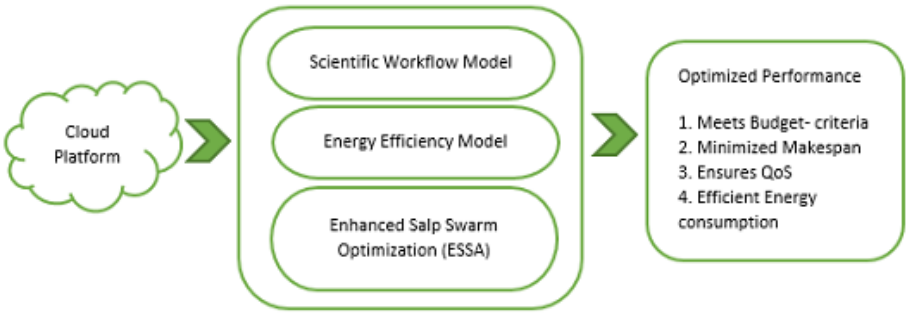

Fig. 1. Proposed Methodology

The following is this technique's primary goal:

a. Construct a budget allotment procedure to continually allot sub-allocation for every unprovisioned process to eliminate the one-side data transmission since they are costly. This guarantee the complete budget of the provisioned tasks to use pricey assets to improve the process completion of the unprovisioned jobs.

b. Enhanced Salp Swarm Algorithm optimizes the execution time and fulfills the QoS standards. A Virtual Machine's (VM) computing ability is connected to CPU and memory. Also, a job's memory utilization is inspected for tasks in the schedule, and the job's processing period is estimated on its processing and storage.

c. A mechanism to reproduce a job's descendants for suggesting duration on matching assets, thereby improving the processing period by managing the sub-expense criteria.

3.1 ESSA Algorithm

Basic Salp Swarm Algorithm (SSA).

Salp Swarm Algorithm (SSA) is a newly founded swarm optimization strategies by Mirjalili et al. [28] in 2017. Salps are associates of the Salpidae family and possess a light cylindric structure. They are identical to jellyfish in build and motion by pumping water via their body. SSA is inspired by this action and foraging mannerisms and it is known as swarming mannerism of the salp chain. The population of tiny organisms is split as leaders and supporters. The leader is the first in the chain and the supporters track it. The head salp revises its place by Eqn. (1) in the pursuit procedure for food:

$$x_i^1 = \begin{cases} y_i + r_1((ub_i - lb_i) \times r_2 + lb_i) & r_3 \le 0, \\ y_i - r_1\left((ub_i - lb_i) \times r_2 + lb_i\right) & r_3 > 0, \end{cases} \quad (1)$$

here the x_i^1 denots the place of the foremost organism in the i^{th} dimension, whereas y_i is the food place in the i^{th} dimension. lb_i and ub_i define the smallest and the highest boundary of the i^{th} dimension, and r_1, r_2, r_3 are three random numerals.

And r_1 occupies the chief position and is given as

$$r_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \quad (2)$$

here L is the greatest repetitions, l is the present repetition, and r_2, r_3 are random numerals between [0,1]. Eqn. (3) is used to update the followers place.

$$x_i^j = \frac{1}{2}\lambda t^2 + \delta_0 t \quad (3)$$

Here j≥2, x_i^j denotes the place of the j^{th} salp in the i^{th} dimension, t is the period, δ_0 is an starting speed and $\lambda = \frac{\delta_{final}}{\delta_0}$, where $\delta = \frac{x - x_0}{t}$.

Algorithm 1: Salp Swarm Algorithm (SSA)

Need: Load the species society xi(i = 1, 2, ..., n) use UB and LB.

Repeat(Termination state never met)

Compute the fitness of every forage exploring salp

F=the prime exploration result

Revise r1 using Equation (2)

for each salp (xi)

if (i == 1)

Revise the place of the foremost salp using Equation (1)

else

Revise the place of the supporters salp using Equation (3)

end

end

Confirm the place of salps using on the highest and smallest bounds

end

return F

SSA is a strong, elastic and flexible algorithm appropriate for numerous optimization situations in a fair performance duration [29]. It has a straightforward design and quick processing pace. It could obtain substantial outcomes on target procedures with more occasional best solution. Yet, it has inadequate search capacity and can easily bow out from best answers, so this acts badly on multiple target processes. Similarly, its unjust proportion of search and utilization is also a notable drawback. In order to address this issue, a mutation operator inspired by the quasi-oppositional learning

technique proposed by Rahnamayan et al. (2007) is introduced in this research. This operator aims to boost group variety and the search capability of SSA. The quasi-mutation operator modifies the positions of salps to explore the search space more effectively.

$$X_i^{q0} = rand(X_i^c, X_i^o) \quad (4)$$

$$X_i^C = \frac{X_i^{max} + X_i^{min}}{2} \quad (5)$$

$$X_i^o = X_i^{max} + X_i^{min} - X_i \quad (6)$$

The quasi-opposite individuals are generated based on the leap rate (Lr), where Lr determines the extent of changes in the search space. A smaller Lr leads to faster convergence but may not reach the global optimum, while a larger Lr takes more time to converge due to the expanded search space. To address this, an adaptive leap rate is introduced in the research to balance convergence speed and global optimization. This adaptive approach aims to prevent premature convergence by initially allowing significant changes and gradually reducing the leap rate over iterations, thus improving the algorithm's convergence rate and effectiveness in finding optimal solutions.

$$L_r = L_{r,max} - \frac{L_{r,max} - L_{r,min}}{itermax} \times iter \quad (7)$$

where, $L_{r,max}$= 0.5, $L_{r,min}$= 0.01, *itermax* denotes the maximal repetition, and *iter* indicates the ongoing loop. To avoid premature convergence Lr is high at the starting and it is progressively reduced inorder to build on the merging value.

The suggested technique uses the conventional SSA with the integration of quasi-oppositional learning-based mutation along with the adaptive leap rate, hence, it is called as Enhanced Salp Swarm Optimization Algorithm (ESSA). The algorithm of the suggested ESSA is below.

Pseudo code 2: Enhanced Binary Salp Swarm Algorithm
1. Input: UB, LB, itermax, popsize and Food fitness = ∞
2. for iter =1: itermax
3. Initialize the salp population for both binary and integer variables
4. $x_i = LB + rand\ (UB - LB) \quad i \in popsize$
5. Calculate the fitness of each salp
6. Find best fitness among the population
7. if best fitness < Food fitness
8. F = best salp corresponding to the best fitness
9. End
10. Estimate the value by using Equation (2)
11. for i = popsize
12. if i == 1
13. Estimate the place of the integer variable of leading salp using Equation (1)
14. Else
15. Estimate the place of the integer variable of trailing salp using Equation (3)
16. End
17. End
18. Find if any exploring agent surpasses the exploring area and revise it
19. Compute the leap rate Lr by Equation (7)
20. if rand(0,1) < Lr
21. Compute quasi- opposite individual for integer variable by Equation (4)
22. End
23. Compute the fitness of every exploring agent
24. Revise X* if a good result exists
25. Update the value of F and Return
26. End

4 Experimental Environment

The newly presented technique has been simulated in java using CloudSim3.0 toolkit incorporated into net beans IDE 8.0.2. CloudSim is an open source workflow simulator, which is an extension of the CloudSim framework.

The evaluation is compared with existing techniques namely GO-DNN [31], ASA-TL [32], SSA [33] and proposed technique ESSA. The performance measures namely total cost, makespan, energy consumption, and Time Consumption.

Total Cost refers to the cumulative expenditure associated with executing a specific task or process. In computational

algorithms, it encompasses factors like resource utilization, operational expenses, and overhead costs. The formula for calculating Total Cost involves summing up Resource Cost, Operational Cost, and Overhead Cost. Makespan, on the other hand, signifies the total duration needed to finish a bunch of processes in scheduling problems. It spans through initiation from primary to accomplishment of the final one. Minimizing Makespan is vital for optimizing asset utilization and meeting deadlines. Power usage quantifies the amount of energy utilized during computational task execution. It's a critical metric in energy-sensitive environments, such as wireless sensor networks or cloud computing, where reducing energy consumption extends device lifespan and cuts operational expenses.

The formula for Energy Consumption involves multiplying Power by Time. Lastly, Time Consumption denotes the overall duration or processing time required by an algorithm to complete a task, encompassing both processing time and any additional overhead. It's typically measured in milliseconds, seconds, or other time units. Time Consumption is computed by the variation between End period and Start period. These performance measures are pivotal for evaluating the fruitfulness and competence of computational techniques across diverse domains and applications.

$$Total_{Cost} = Resource_{cost} + Operational_{cost} + Overhead_{cost} \quad (8)$$

$$Makespan = Finishtime_{lasttask} + Starttime_{firsttask} \quad (9)$$

$$Energy_{Consumption} = Poer \times Time \quad (10)$$

$$Time_{Consumption} = End_{time} - Start_{time} \quad (11)$$

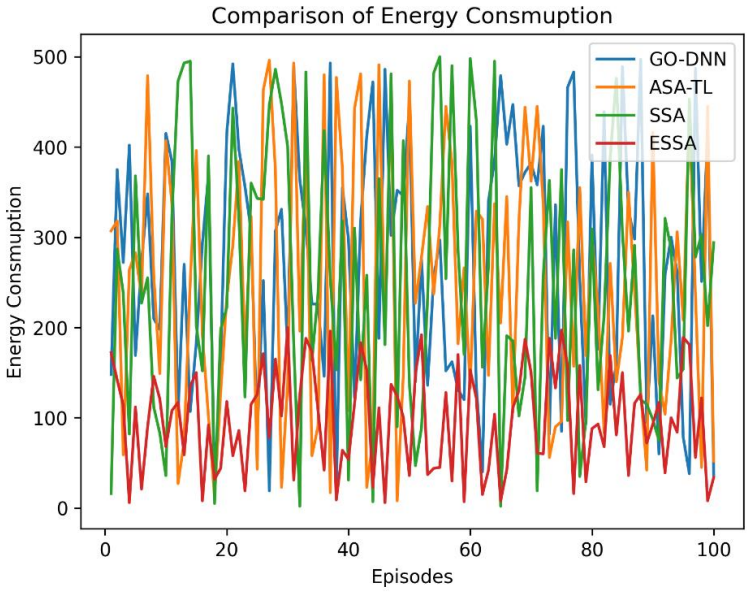

Fig 2. Comparison of Energy Consumption

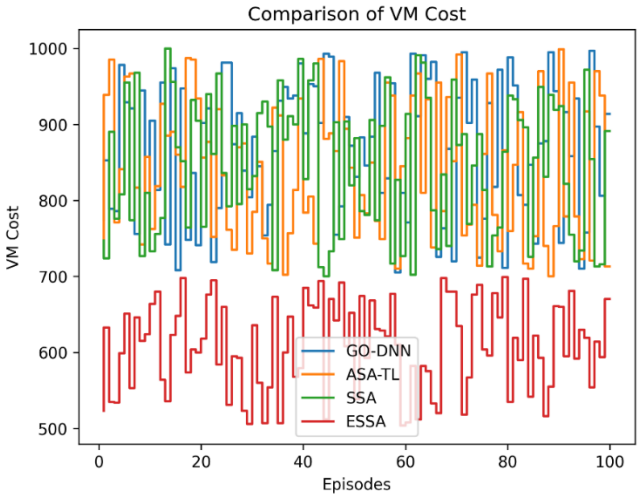

Fig 3. Comparison of VM Cost

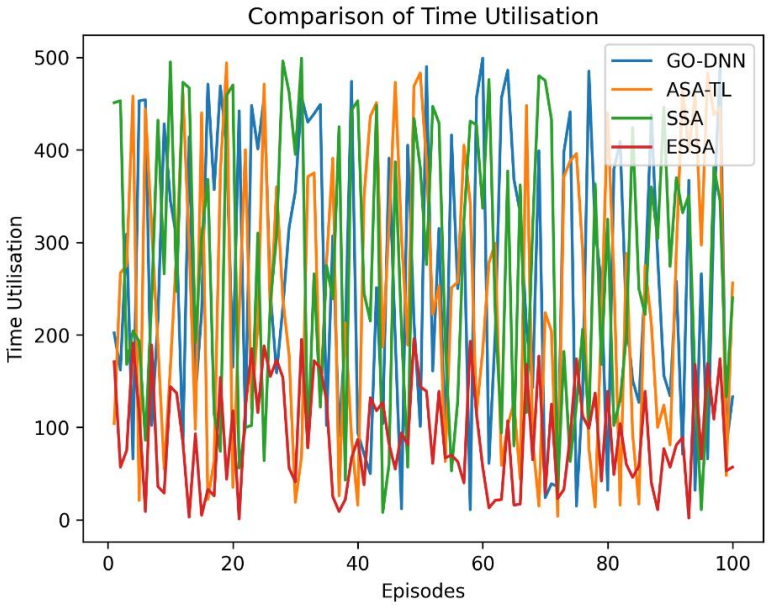

Fig 4. Comparison of Time Utilization

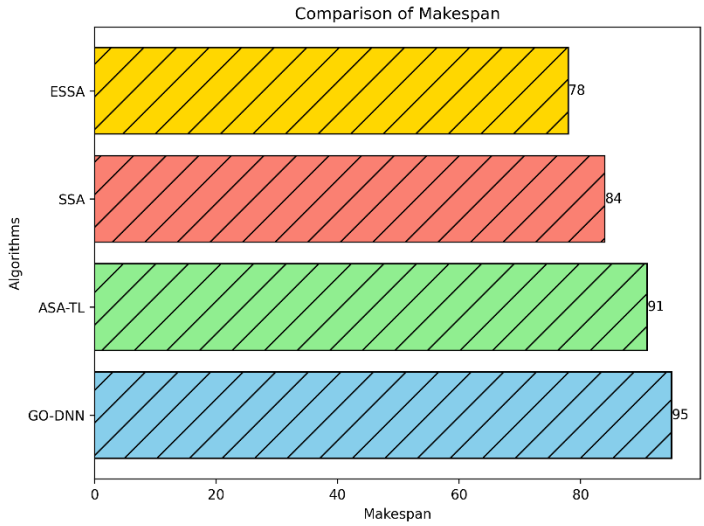

Fig 5. Comparison of Makespan

The comparative analysis of energy usage among the algorithms GO-DNN, ASA-TL, SSA, and ESSA reveals notable differences in their energy consumption patterns. Calculating the average and volatility of power utilization for each algorithm provides insights into their respective performance. GO-DNN exhibits a mean energy consumption of 266.25 with a standard deviation of 158.82, while ASA-TL demonstrates a marginally lesser average of 247.52 and a volatility of 187.95. SSA shows a mean energy consumption of 255.72 with a standard deviation of 182.36. Interestingly, ESSA stands out with the lowest mean energy consumption of 108.77 and a standard deviation of 63.57, indicating potentially higher energy efficiency compared to the other algorithms. These findings are visually represented through histograms, showcasing the distribution of energy consumption for each algorithm. While ESSA demonstrates promising energy efficiency, further considerations such as application requirements and constraints are crucial in selecting the most suitable algorithm for practical implementation.

5 Conclusion and Future Work

The proposed Enhanced Salp Swarm Optimization Algorithm (ESSA) for workflows in the cloud that are restricted by budget but with better quality of service and energy competence presents a significant advancement in cloud task scheduling algorithms. The integration of the Salp Swarm Algorithm (SSA) with dynamic workflow scheduling addresses critical challenges in cloud computing, including real-time adaptation to changing task arrivals and resource availability, optimization of makespan and Quality of Service (QoS) standards, and adherence to budget constraints. The principles of SSA, inspired by the collective movement patterns of salps, offer a robust optimization framework for

efficiently exploring solution spaces. The proposed Enhanced Salp Swarm Optimization Algorithm (ESSA) enhances SSA's effectiveness by incorporating quasi-oppositional learning-based mutation and adaptive leap rate strategies, thereby improving solution convergence and diversity.

Future enhancements could focus on further refining the adaptive mechanisms of ESSA to dynamically adjust mutation rates and leap rates based on evolving environmental conditions and task characteristics. Additionally, exploring hybridization with other nature-inspired optimization algorithms or machine learning techniques could potentially enhance the algorithm's performance and scalability. Moreover, extending the evaluation to larger-scale cloud environments and incorporating additional performance metrics such as scalability, fault tolerance, and security might offer a more thorough comprehension of ESSA's capabilities and suitability for practical deployment in diverse cloud computing scenarios. Overall, the proposed algorithm offers a promising avenue for addressing the complex challenges of dynamic workflow scheduling in budget-constrained cloud environments while improving Quality of Service and energy efficiency.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Meena, F., Kumar, M., Vardhan, M., Jain, S.: A systematic review on practical workflow scheduling algorithms in cloud under deadline constraint, International Journal of Circuit Theory and Applications, 9(41), 1160-1170 (2016).

[2] Arabnejad, V., Bubendorfer, K., and Ng, B.: Budget and Deadline Aware e-Science Workflow Scheduling in Clouds, IEEE Trans. Parallel Distrib. Syst., 30(1), 29–44 (2019).

[3] Rambabu Medara, Ravi Shankar Singh, Amit.: Energy-aware workflow task scheduling in clouds with virtual machine consolidation using discrete water wave optimization, Simulation Modelling Practice and Theory, 110 (2021). https://doi.org/10.1016/j.simpat.2021.102323

[4] Poola, D., Garg, S. k., Buyya, R., Yang, Y., and Ramamohanarao, K.: Robust Scheduling of Scientific Workflows with Deadline and Budget Constraints in Clouds, IEEE 28th International Conference on Advanced Information Networking and Applications. CONFERENCE 2014, pp. 858–865 (2014). doi: 10.1109/AINA.2014.105

[5] Ma, l., Lu, y., Zhang, F., and Sun, S.: Dynamic Task Scheduling in Cloud Computing Based on Greedy Strategy. Trustworthy Computing and Services, 156–162 (2013).

[6] Fakhfakh, F., Kacem, H.H., Kacem, A.H.: Workflow scheduling in cloud computing: a survey, IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations, CONFERENCE 2014, pp. 372–378 (2014).

[7] Liu, L., Zhang, M., Lin, Y., Qin, L.: A survey on workflow management and scheduling in cloud computing, 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 837–846, IEEE (2014).

[8] Singh, L., Singh, S.: A Survey of Workflow Scheduling Algorithms and Research Issues. International Journal of Computer Applications. 74. https://doi.org/10.5120/12961-0069.

[9] Jennings, B., Stadler, R.: Resource Management in Clouds: Survey and Research Challenges, J. Netw. Syst. Manag., 23(3), 567–619 (2015). doi: 10.1007/s10922-014-9307-7.

[10] Fakhfakh, F., Kacem, H. H., Kacem, A. H.: Workflow Scheduling in Cloud Computing: A Survey, IEEE 18th International Enterprise Distributed Object Computing Conference, Workshops and Demonstrations, CONFERENCE 2014, pp. 372–378 (2014). doi: 10.1109/EDOCW.2014.61

[11] Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: stateof-the-art and research challenges, J. Internet Serv. Appl., 1(1), 7–18 (2010). doi: 10.1007/s13174-010-0007-6

[12] Khanghahi, N., Ravanmehr, R.: Cloud Computing Performance Evaluation: Issues and Challenges, Int. J. Cloud Comput. Serv. Archit., 3(5), 29–41 (2013). doi: 10.5121/ijccsa.2013.3503.

[13] C. K. Sripavithra, V. B. Kirubanand: A review on recent scheduling algorithms in the cloud environment. AIP Conf. Proc. 28 November 2023; 2909 (1): 030007. https://doi.org/10.1063/5.0183242.

[14] Zhou,X., Zhang, G., Sun, J., Zhou,J., Wei,T., Hu, S.: Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based heft, Future Gener. Comput. Syst. 93, 278–289 (2019).

[15] Wu, H., Chen, X., Song, X., Zhang, C., Guo, H.: Scheduling large-scale scientific workflow on virtual machines with different numbers of vCPUs. J. Supercomput. 77, 1 (Jan 2021), 679–710. https://doi.org/10.1007/s11227-020-03273-3.

[16]Topcuoglu, H., Hariri, S., Wu, M. Y.: Performance-effective and low-complexity task scheduling for heterogeneous computing, IEEE Trans. Parallel Distrib. Syst. 13, 260–274 (2002).

[17]Safari, M., Khorsand, R.: Energy-aware scheduling algorithm for time-constrained workflow tasks in dvfs-enabled cloud environment, Simul. Model. Pract. Theory, 87, 311–326 (2018).

[18]Fernandez-Cerero, D., Jakobik, A., Fernandez-Montes, A., Kolodziej, J.: Game-score: Game-based energy-aware cloud scheduler and simulator for computational clouds, Simul. Model. Pract. Theory 93, 3–20 (2019).

[19]Li, C., Tang, J., Ma, T., Yang, X., Luo, Y.: Load balance based workflow job scheduling algorithm in distributed cloud, J. Netw. Comput. Appl. 152, 102518 (2020).

[20]Abazari, F., Analoui, M., Takabi, H., Fu, S.: Mows: multi-objective workflow scheduling in cloud computing based on heuristic algorithm, Simul. Model. Pract. Theory, 93, 119–132 (2019).

[21]Mohanapriya, N., Kousalya, G., Balakrishnan, P., Pethuru Raj, C.: Energy efficient workflow scheduling with virtual machine consolidation for green cloud computing, J. Intell. Fuzzy Systems, 34, 1561–1572 (2018).

[22]Li, Z., Ge, J., Hu, H., Song, W., Hu, H., Luo, B.: Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds, IEEE Trans. Serv. Comput. 11, 713–726 (2018). http://dx.doi.org/10.1109/TSC.2015.2466545

[23]F. Yao, C. Pu, and Z. Zhang, Task Duplication-Based Scheduling Algorithm for Budget-Constrained Workflows in Cloud Computing, IEEE Access, 9, 37262–37272 (2021). doi: 10.1109/ACCESS.2021.3063456

[24]Long, S., Dai, X., Pei, T., Cao, J., Sekiya, H., Choi, Y.-J.: Energy-efficient VM opening algorithms for real-time workflows in heterogeneous clouds, Neurocomputing, 483, 501–514 (2022). doi: https://doi.org/10.1016/j.neucom.2021.08.145

[25]Ferdaus, M.H., Murshed, M., Calheiros, R.N., Buyya, R.: Virtual machine consolidation in cloud data centers using aco metaheuristic, In: European Conference on Parallel Processing, Springer, 306–317 (2014).

[26]Medara, R., Singh, R.S., Kumar, U.S., Barfa, S.: Energy efficient virtual machine consolidation using water wave optimization, In: IEEE Congress on Evolutionary Computation (CEC), IEEE, 1–7 (2020).

[27]C. K. Sripavithra and V. B. Kirubanand, ESSA Scheduling Algorithm for Optimizing Budget-Constrained Workflows. In: IEEE International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME) (2022) https://doi.org/10.1109/ICECCME55909.2022.9988009 .

[28]Mirjalili, S., Amir, H. Gandomi, Mirjalili, S., Saremi, S., Faris, H., Mirjalili, S.: Salp Swarm Algorithm: A bioinspired optimizer for engineering design problems, Advances in Engineering Software 114, 163-191 (2017).

[29]Abualigah, Laith, Shehab, M., Alshinwan, M., Alabool, H.: Salp swarm algorithm: a comprehensive survey, Neural Computing and Applications, 32(15), 11195-11215 (2020).

[30]S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama. Quasi-oppositional Differential Evolution. In: IEEE Congress on Evolutionary Computation, Singapore, 2007, pp. 2229-2236, doi: 10.1109/CEC.2007.4424748.

[31]Badr, S., El Mahalawy, A., Attiya, G., & Nasr, A. A. Task consolidation based power consumption minimization in cloud computing environment. *Multimedia Tools and Applications*, *82*(14), 21385-21413 (2023). https://doi.org/10.1007/s11042-022-14009-1

[32]Murad, S. A., Muzahid, A. J. M., Azmi, Z. R. M., Hoque, M. I., & Kowsher, M. A review on job scheduling technique in cloud computing and priority rule based intelligent framework. *Journal of King Saud University-Computer and Information Sciences*, *34*(6), 2309-2331 (2022). Doi: 10.1016/j.jksuci.2022.03.027.

[33]Ahmed, O. H., Lu, J., Ahmed, A. M., Rahmani, A. M., Hosseinzadeh, M., & Masdari, M. Scheduling of scientific workflows in multi-fog environments using Markov models and a hybrid salp swarm algorithm. *IEEE Access*, *8*, 189404-189422 (2020). doi: 10.1109/ACCESS.2020.3031472