IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 13, No. 4, December 2024, pp. 3849~3857
ISSN: 2252-8938, DOI: 10.11591/ijai.v13.i4.pp3849-3857

A multi-core makespan model for parallel scientific workflow execution in cloud computational framework

Farha Naaz, Sameena Banu
Department of Computer Science and Engineering, Khaja Banda Nawaz University, Kalaburagi, India

Article Info

Article history:

Received Dec 28, 2023
Revised Mar 29, 2024
Accepted Apr 17, 2024

Keywords:

Cloud computing
Energy
MapReduce
Quality of services
Scientific workflow

ABSTRACT

Researchers have shown a lot of potential in optimizing cloud-based workload-scheduling over the past few years. However, executing scientific workloads inside the cloud is time-consuming and costly, making it inefficient from both a financial and productivity standpoint. As a result, there are many investigations conducted, with the general trend being to speed up the rate of processing and establish a cost-effective system, whereby customers are billed according to their actual use. In addition, energy-consumption is capable of being reduced, especially if the available resources are heterogeneous; however, few investigations have optimized multi-core with analyzing makespan parameters collectively to fulfill the quality of service (QoS) and service level agreement (SLA) of the workload task. In this research, we introduce an optimal scheduling for a heterogeneous distributed cloud computing environment called task aware makespan optimized scheduler (TAMOS) that guarantees requirements across the task levels of scientific workflows. The energy and time required to carry out specific workflows are significantly reduced by using this TAMOS strategy. The TAMOS framework was studied using the scientific workflows namely, inspiral and sipht. When compared to the conventional method of scheduling work, our methodology used less energy and makespan.

Corresponding Author:

Farha Naaz
Department of Computer Science and Engineering, KBN University
Kalaburagi, India
Email: farhanaaz_12@rediffmail.com

1. INTRODUCTION

Parallel scientific workflow tasks are extensively employed in both scientific and business investigations [1]. Several instances of workloads in various scientific fields can be observed. These involve astrophysics workloads such as Montage and Ligo, which focus on astronomical data analysis. Additionally, there is a tremor identification workload called CyberShake, which aims to identify seismic activities. Furthermore, deoxyribonucleic acid (DNA) sequencing workloads like epigenomics and sipht are employed for studying genetic information. These workloads have been identified in previous research [2]. Multiple scientific workloads are commonly utilized for a multitude of reasons. The execution of scientifically challenging workloads is observed across various platforms, including MapReduce [3] and Amazon EC2 [4]. Platforms like MapReduce [5] as shown in Figure 1, are not efficient for execution of complex workflow having multiple levels of dependency.

On the other side, the cloud-based parallel computational infrastructures as shown in Figure 2 are known for their ability to offer top-notch storing and computing capabilities, including applications, networks, and services. These resources are particularly valuable for handling the demanding computational requirements of scientific demanding workloads [6]. In contemporary times, various scientific disciplines, such as science, biology, and astrophysics, have been increasingly leveraging cloud services to simulate and analyze sophisticated scientific-workloads. This utilization of cloud-based parallel computational systems enables researchers to devise better approaches for real-time challenges [7]. Therefore, it is evident that scientific complicated tasks are effectively performed on cloud-based parallel computational systems, wherein the evaluation is conducted utilizing computational tools offered by the cloud infrastructure.

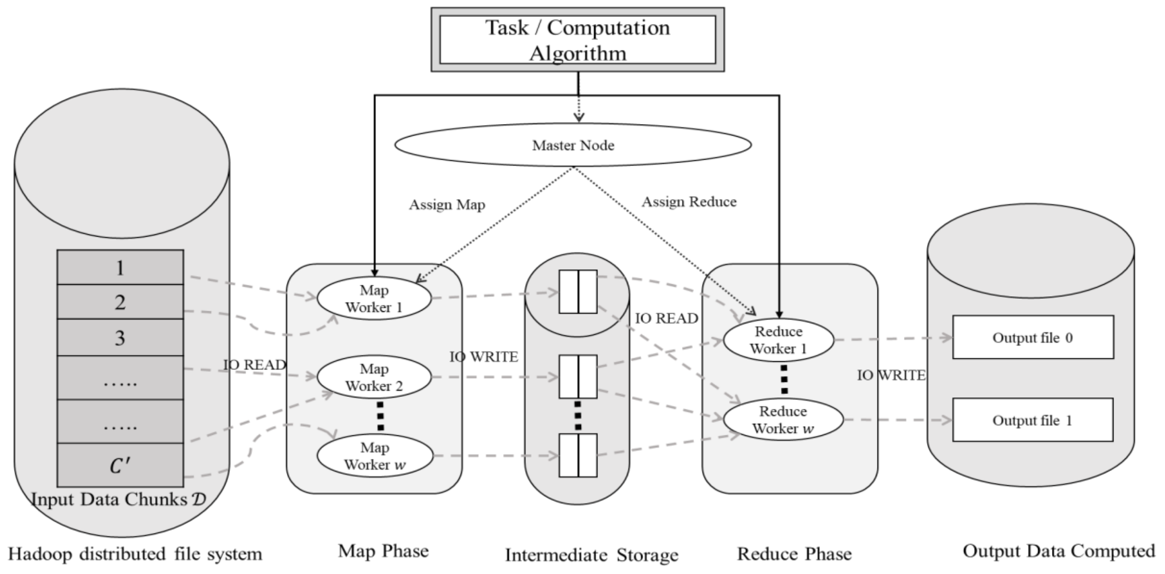


Figure 1. Standard hadoop based computational platform

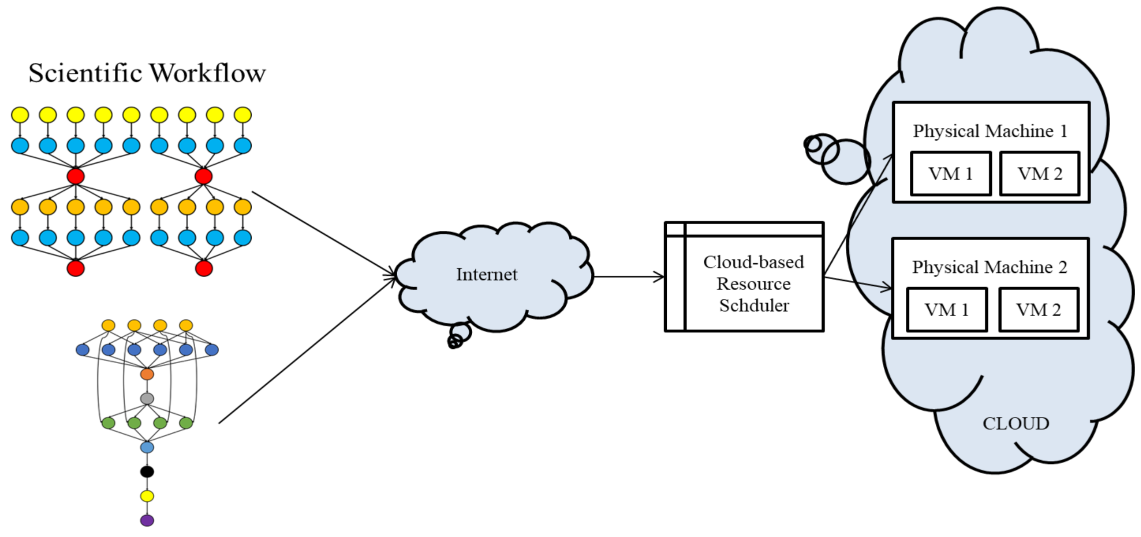


Figure 2. Cloud-based computational platform for execution of scientific workflow with multi-level dependencies

Moreover, it has been observed that the implementation of scientifically complicated tasks in cloud environments leads to a significant reduction in both execution cost and energy, as highlighted in previous research [7]. The intricate nature of tasks can be effectively characterized by employing a directed acyclic graph (DAG) framework. Within this framework, the centers of the DAG represent the interdependencies among tasks, while the edges of the DAG delineate the specific tasks themselves. Through the implementation of this process, it is anticipated that the workload can be effectively and efficiently carried out within the designated deadlines. This implies that the tasks are going to be carried out in a sequential manner, adhering to their predetermined order [8]. Furthermore, the intricate interdependencies among the different tasks pose a significant challenge when it comes to effectively managing and allocating the workload within a cloud computing environment.

In recent years, more and more people have started using cloud services for their various software needs. As a result, a significant number of researchers have opted to utilize cloud services as a means to efficiently manage and allocate their workload [9]. The provided diagram, Figure 2, illustrates a straightforward architectural representation of the workload-scheduling process within a cloud environment. Designing an effective workload-scheduling approach necessitates a thorough examination of current systems, a process that in turn poses multiple difficulties. Notably, carrying out of more complicated and bigger scientific-workloads demands increased execution-time and incurs higher costs. The execution of tasks within a specified deadline presents an increased level of difficulty. Numerous investigations were conducted by scholars from various institutions, focusing on the development of algorithms based on heuristics. These algorithms aim to offer optimal solutions for a wide range of issues. Moreover, it has been observed that the aforementioned heuristic algorithms exhibit a lack of time efficiency. Consequently, a significant number of scholars have encountered challenges in attaining the best possible outcome, resulting in adverse impacts on service level agreement (SLA) compliance and quality of service (QoS). Furthermore, it has been established in the literature that the allocation of workload is classified as an NP-hard (non-deterministic polynomial-time) problem [10]. Indeed, the optimization of both cost and time presents a formidable challenge within the domain of workload-scheduling [11]. In the context of a scheduling approach, it is observed that when the objective is to minimize costs, there is an associated rise in the time necessary for finishing a specific task. The interdependence of time and cost is a significant contributing factor to the observed phenomenon. The lack of consideration for virtual machine (VM) decision-making procedures in current approaches during schedule creation has resulted in an ongoing issue of make-span and cost [12], [13].

To effectively tackle the issue, this research introduces an innovative approach known as task aware makespan optimized scheduler (TAMOS) under multi-core platform, which aims to ensure SLA compliance at the task levels. The TMOS is designed in such a way it reduces makespan leveraging multi-core execution and thereby utilizing system resource i.e., reducing overall energy consumption meeting multi-level scientific workflow task dependencies. The significance of the research work is mentioned here.

- The work introduces a novel makespan model leveraging multi-core resource utilization.
- A makespan with task dependencies model considering multi-level workflow task interdependence of scientific workflow,
- Simulation is considered using two different scientific workflows namely inspiral and sipht.
- Experiment outcome shows the proposed model exhibits superior performance in terms of reducing makespan and energy in comparison with existing methodology.

The manuscript organization: section 2 introduces current state-of-the-art system to meets strict QoS and SLA constraint of scientific workflow application with multi-level dependencies. Section 3 introduces a novel makespan model to fully utilize resources efficiently and assures task dependencies of scientific workflow application. The performance is proposed and existing is studied using two scientific workflows in section 4. the last section the significance of research and future enhancement is provided.

2. LITERATURE SURVEY

The scientific workflow applications have grown exponentially in recent years due to availability of parallel and distributed computing platforms, and their efficiency has improved as a result. Methods created in [14] for a homogeneous framework execute unsatisfactorily in a heterogeneous framework due to the need for the input/output (I/O) and memory optimization method, despite providing lockless first in first out (FIFO) which incorporates hadoop map reduce (HMR) and other applications. In the HMR system, the scheduler technique used during the shuffling stage is the primary determinant of makespan for completing tasks [15]. As demonstrated in [16], prior approaches to performing heterogeneous tasks have not taken task dependence models into account, leading to inefficient use of available resources [17]. So, they devised a new yet another resource negotiator (YARN) scheduling scheme that cuts down the makespan of various industrial tasks. Makespan efficiency, particularly when dealing with complicated repetitive applications, is negatively impacted by the fact that the models described in [18], [19] don't account for failures at intermediary tasks considering DAG applications. The optimization of energy and cost for diverse computational structures is the main objective of the study [20]. The researchers aimed to develop a scheduling mechanism that effectively managed the workload in order to achieve optimal outcomes in terms of both energy and cost efficiency. The utilization of a minimum functionality in this context aims to optimize the use of energy and meet task-deadlines, taking into account the dispersed location. Distribution of task-related data. The deadlines have been carefully examined and organized in ascending order, starting from the smallest to the largest. In this study, they presented a novel approach to address the challenge of selecting an effective schedule for execution of processes. The proposed approach was based on adaptable searching techniques, which have shown promise in various optimization problems. By leveraging the adaptability of the search algorithm, their method aimed to identify the most suitable schedule that can maximize the efficiency of the method of execution.

This research contribution fills a gap in the existing literature by offering a practical solution to the problem at hand. As per [12], there exists an immediate connection between escalated computation expenses associated with service provision and the corresponding surge in energy usage. Timeliness and reliability are widely recognized as the primary metrics of utmost significance within the realm of customer service supply. The researchers devised a scheduling structure called energy min scheduling (EMS) to address the need for reduced energy-consumption in running workloads. This framework successfully fulfilled both the timeliness and reliability requirements. The application of non-linear mixed-integer-programming (NL-MIP) was employed in this particular scenario to obtain an optimal solution. The present study focused on the development of a meeting reliability approach, which centered on the utilization of a schedule length minimizing approach. Furthermore, the implementation of dynamic-voltage frequency-scaling (DVFS) method allowed for the reduction of energy-consumption through the method of processor merging with one another In this particular case, scaling was performed at both the processor and task stages. The findings of this study suggested that previous frameworks performed more effectively when exposed to varying degrees of disruption.

Nested particle swarm optimization (NPSO) and fast nested particle swarm optimization (FNPSO) are two evolutionary computing models developed in [21] to enhance the performance of demanding workloads. Compared to the traditional NPSO approach, the FNPSO is considerably reduced make-span and cost. As per the findings of the study conducted by [22], it is imperative for a cloud-based scheduling method to fulfill both user time constraints and SLAs. A multi-cloud system [23] was employed in order to align with the particular efficiency and cost criteria of the stream workload implementation. By leveraging a multi-cloud structure and implementing a fault-tolerant scheduling architecture, a system can be developed to address the task at hand. The utilization of a multi-cloud structure allows for the distribution of workloads across multiple cloud service providers, thereby enhancing resource availability and scalability. Additionally, the incorporation of a fault-tolerant scheduling structure ensures that the system can withstand and recover from potential failures or disruptions, thereby minimizing downtime and maximizing reliability [24]. Furthermore, the proposed model not only guarantees the fulfillment of the reliability criterion but also effectively reduces the associated cost. A comprehensive examination was conducted to assess the rates of failure and reliability by employing an ongoing probability distribution. After determining the financial implications of implementing the multi-cloud platform, the subsequent phase involves formulating a fault-tolerant workload-scheduling framework that guarantees dependability, cost reduction, minimized execution duration, and cost efficiency. The inability to fulfill the cost constraints of the application can be attributed to the absence of load balancing mechanisms [25]. The subsequent section presents a scheduling strategy that takes into consideration QoS performance factors like makespan, resource utilization and SLA constraints like workflow deadline at different level in heterogeneous cloud environments, in order to tackle the challenges previously mentioned.

3. PROPOSED METHODOLOGY

This paper introduces a novel cloud-based parallel computational framework for effective execution of parallel scientific workflow with reduced makespan as shown in Figure 3. This work introduces a multi-level workflow application specific QoS dependency aware makespan model. Namely task-aware makespan optimized scheduler (TAMOS) meeting SLA constraint with better resource utilization leveraging multi-core platform.

A comprehensive makespan approach is presented to address the issue of improving task execution efficiency in the TAMOS framework. The computation of the makespan C for carrying out a task is easily achieved by utilizing the equation shown here. The computation of the makespan is mentioned in (1).

$$C = C_T + C_M + C_R. \tag{1}$$

The variable C_T represents the makespan for the beginning worker considering both I/O and memory optimization, while C_M denotes the makespan for carrying out of map tasks, and C_R represents the makespan for carrying out reduce tasks. Assume a scenario where each worker, denoted as q, is comprised of a particular number of cores/threads, represented by n, and has a memory dimension of x.

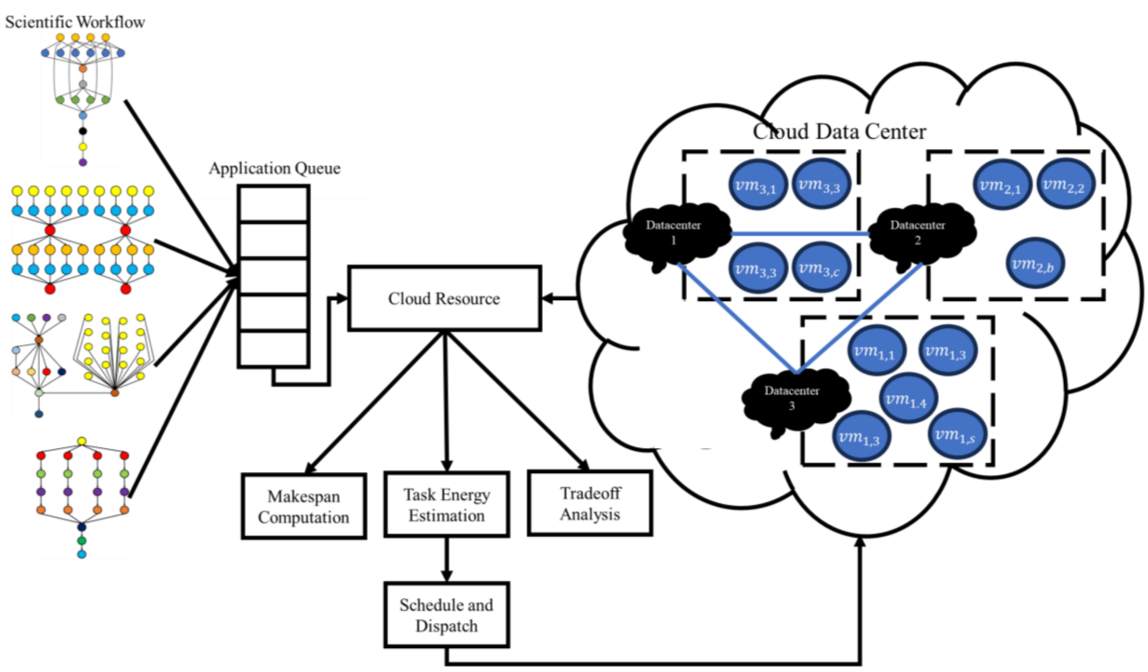


Figure 3. Architecture of proposed cloud-based computational framework for execution of parallel scientific workflow

In this context, the mean make-span for carrying out a task is able to be determined through in (2). In an identical manner, the computation of the mean make-span for the reduce task can be determined using (3). The computation of the overall make-span of TAMOS can be achieved by utilizing (2) and (3) and it is presented in (4).

$$C_M = \frac{\sum_{a=1}^{q} C_{a\_M}}{q} \tag{2}$$

$$C_R = \frac{\sum_{a=1}^{q} C_{a_R}}{q}. \tag{3}$$

$$C = C_T + \frac{\sum_{a=1}^{q} (C_{a\_M}+C_{a\_R})}{q}. \tag{4}$$

The task execution of various phases within the TAMOS framework exhibits variability. Therefore, we have established both upper limits and lower limits for the make-span required to execute task K. In (5) used to calculate the overall make-span for the K^{th} tasks within the TAMOS framework under the ideal case scenario as mentioned here.

$$U_K^{L_{lim}} = U_M^{L_{lim}} + U_R^{L_{lim}} - \left(U_M^{U_{lim}} - U_M^{L_{lim}}\right) \tag{5}$$

The variable $U_K^{L_{lim}}$ is used to denote the shortest make-span time required for the execution of task K. Similarly, $U_M^{L_{lim}}$ represents the shortest amount of time required for carrying out a map task, while $U_R^{L_{lim}}$ and $U_M^{L_{lim}}$ indicate the bare essential time required for carrying out a reduce-task. On the other hand, $U_M^{U_{lim}}$ represents the longest amount of time required for carrying out a map-task. In (5) is written in simplified representation as mentioned in (6). In the TAMOS framework, the overall makespan of the K^{th} job under the most severe circumstances is determined by applying the subsequent equation as given in (7). The above equation is written in simplified manner as mentioned in (8). In (9) provided below is utilized to derive the overall makespan of task K using the TAMOS framework. The estimation of the overall make-span $\vec{U}_K$ for carrying out task K is calculated utilizing (6) and (8) as follows in (10). In (10) is simplified as mentioned in (11). The final makespan considering multi-level workflow execution is obtained by modifying (11) is mathematically given as mentioned in (12).

$$U_K^{L_{lim}} = U_M^{L_{lim}} + U_R^{L_{lim}} - \left(U_M^{U_{lim}} - U_M^{L_{lim}}\right) \tag{6}$$

$$U_K^{U_{lim}} = U_M^{U_{lim}} + U_R^{U_{lim}} - \left(U_M^{U_{lim}} - U_M^{L_{lim}}\right) \tag{7}$$

$$U_K^{U_{lim}} = U_M^{L_{lim}} + U_R^{U_{lim}} \tag{8}$$

$$\vec{U}_K = \frac{\left(U_K^{U_{lim}} + U_K^{L_{lim}}\right)}{2} \tag{9}$$

$$\vec{U}_K = \frac{\left(\left(U_M^{U_{lim}} + U_R^{U_{lim}} - \left(U_M^{U_{lim}} - U_M^{L_{lim}}\right)\right) + \left(U_M^{L_{lim}} + U_R^{L_{lim}} - \left(U_M^{U_{lim}} - U_M^{L_{lim}}\right)\right)\right)}{2} \tag{10}$$

$$\vec{U}_K = \frac{\left(\left(U_M^{L_{lim}} + U_R^{U_{lim}}\right) + \left(2U_M^{L_{lim}} + U_R^{L_{lim}} - U_M^{U_{lim}}\right)\right)}{2} \tag{11}$$

$$\vec{U}_K = \frac{\left(3U_M^{L_{lim}} + U_R^{L_{lim}} + U_R^{U_{lim}} - U_M^{U_{lim}}\right)}{2} \tag{12}$$

The present study employs a methodology comparable to that described in previous works, specifically [3], [5]. To construct a framework for data dependence through the utilization of regression analysis. The TAMOS framework has been observed to exhibit superior performance in terms of minimizing makespan and reducing energy by utilizing resource efficiently for the execution of parallel scientific workflow which is shown through simulation study in next section.

4. RESULT AND DISCUSSION

The following section focuses on the evaluation of performance metrics, namely makespan, and energy efficiency, in the context of the suggested TAMOS framework compared to the existing reliable workflow scheduling (RWS) approach [13]. TAMOS and RWS have been developed and deployed through the utilization of the Java programming language. The CloudSim Software-defined networking-based network function virtualization (CloudSimSDN-NFV) simulator [26] has been employed as the underlying platform for the implementation of these systems. The inspiral and sipht scientific workflow is used for verifying proposed TAMOS framework. The mathematical complexity of the tasks at hand involves a significant amount of memory, processor, and input/output operations. The inspiral structure is known for its computationally demanding nature, particularly with regards to central processing unit (CPU) and memory usage. On the other side, the sipht structure is known for its computationally demanding nature, particularly with regards to CPU and I/O usage.

4.1. Makespan performance

The following section addresses the analysis regarding the makespan required to complete the execution of inspiral workflows with size ranging from 30, 50, 100, and 1000. Alongside, the section addresses the analysis regarding the makespan required to complete the execution of sipht workflows with size ranging from 30, 60, 100, and 1000. Figure 4 illustrates a visual representation of the makespan achieved over the execution of the inspiral workflow employing the TAMOS and RWS scheduling algorithms, while considering a diverse range of workflow scenarios such as 30, 50, 100. Figure 5 illustrates a visual representation of the makespan achieved over the execution of the sipht workflow employing the TAMOS and RWS scheduling algorithms, while considering a diverse range of workflow scenarios such as 30, 60, 100. Similarly, for inspiral workflow size of 1000 the visual representation of makespan is given in Figure 6 and for sipht workflow size of 1000 the visual representation of makespan is given in Figure 7. The utilization of TAMOS, as compared to RWS, results in a notable enhancement in average makespan efficiency, with an observed increase of 83.32% and 51.46% for inspiral and sipht workflow, respectively.

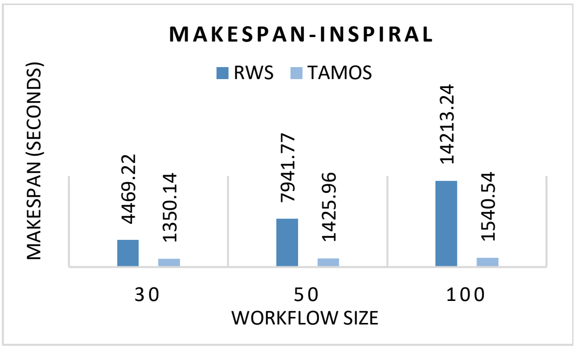


Figure 4. Makespan efficiency with different inspiral workflow size

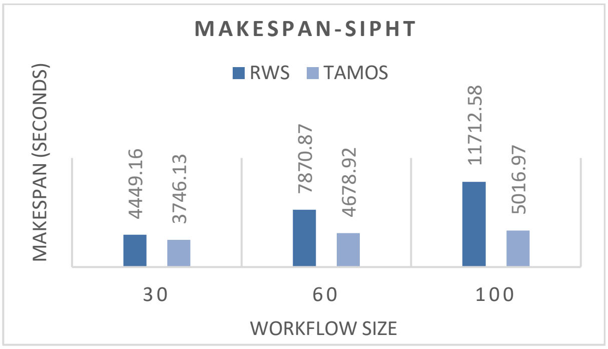


Figure 5. Makespan efficiency with different sipht workflow size

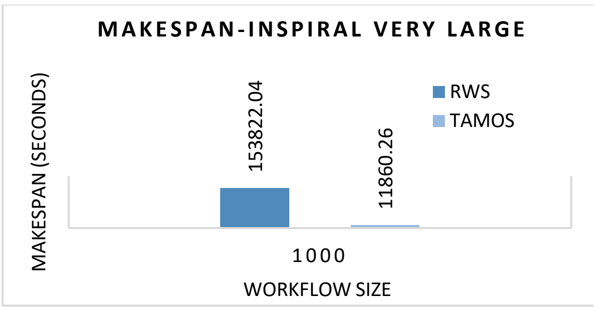


Figure 6. Makespan efficiency with inspiral workflow size of 1000

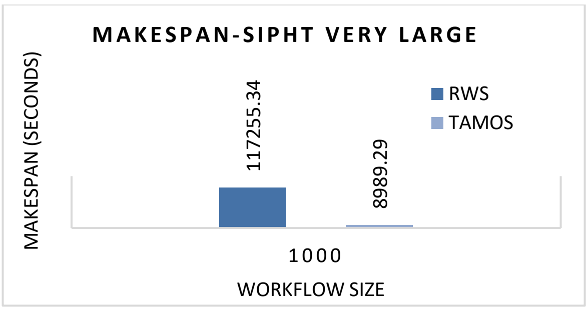


Figure 7. Makespan efficiency with sipht workflow size of 1000

4.2. Energy usage

The following section addresses the analysis regarding the energy needed to complete the execution of inspiral workflows with size ranging from 30, 50, 100, and 1000. Alongside, the work addresses the analysis regarding the energy needed to complete the execution of sipht workflows with size ranging from 30, 50, 100, and 1000. Figure 8 illustrates a visual representation of the energy efficiency achieved over the execution of the inspiral workflow employing the TAMOS and RWS scheduling algorithms, while considering a diverse range of workflow scenarios such as 30, 50, 100. Figure 9 illustrates a visual representation of the energy efficiency achieved over the execution of the sipht workflow employing the TAMOS and RWS scheduling algorithms, while considering a diverse range of workflow scenarios such as 30, 60, 100. Similarly, for inspiral workflow size of 1000 the visual representation of energy usage is given in Figure 10 and for sipht workflow size of 1000 the visual representation of energy usage is given in Figure 11. The resource utilization by reducing energy consumption of TAMOS, as compared to RWS, results in a notable enhancement in average energy efficiency, with an observed increase of 92.64% and 72.87% for inspiral and sipht workflow, respectively.

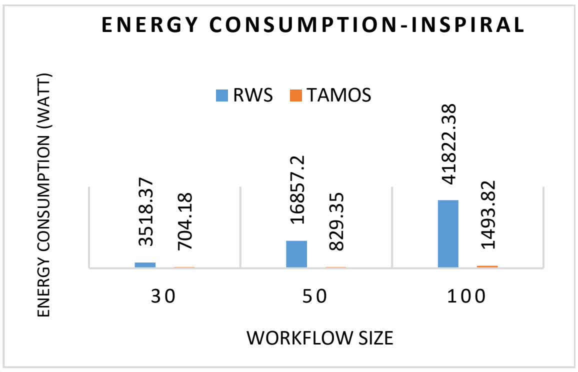


Figure 8. Energy efficiency with different inspiral workflow size

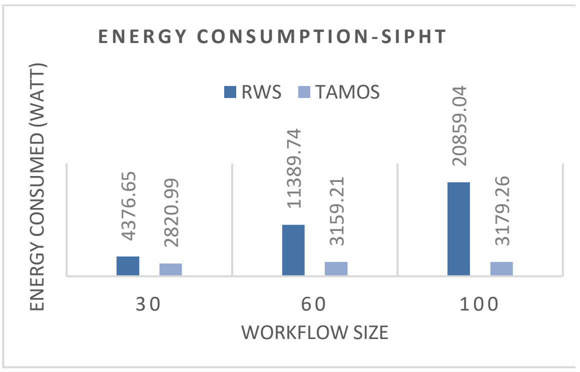
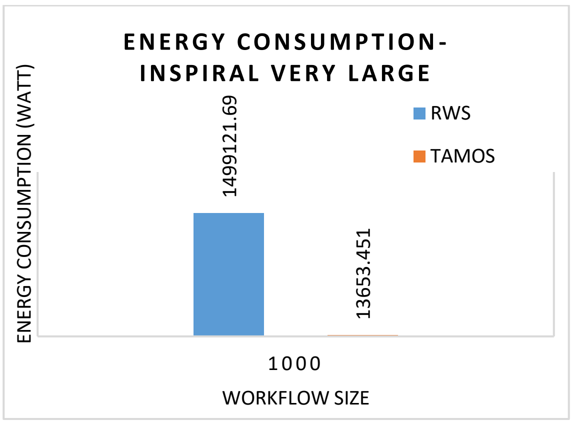


Figure 9. Energy efficiency with different sipht workflow size



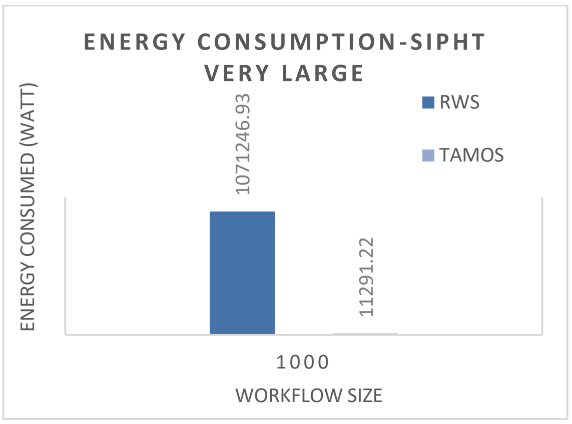


Figure 10. Energy efficiency with inspiral workflow size of 1000

Figure 11. Energy efficiency with sipht workflow size of 1000

5. CONCLUSION

The present study introduces a novel multi-core resource utilization aware makespan model for workflow scheduling which successfully guarantees the fulfillment of task-level SLAs. To our knowledge, very limited work has ever before addressed task-level SLA when planning a workflow scheduling. The present study presents an innovative approach to adaptable scheduling, which aims to minimize total makespan while simultaneously ensuring highest resource efficiency by reducing overall energy by providing execution at thread-level in multi-core processor of cloud VM. By implementing this method, a substantial reduction in overall operation makespan and energy thereby will aid in utilizing resources efficiently with minimal cost can be achieved. TAMOS demonstrates high efficiency in the allocation of CPU, memory, and I/O resources for executing parallel scientific workflows that require significant computational and data processing capabilities. This efficiency is achieved by utilizing shared computing environments, particularly in cloud environments. The experimental results demonstrate that the TAMOS system exhibits a high level of makespan and energy efficiency for execution of cybershake workflows. Specifically, when compared to the RWS system, TAMOS showcases an impressive increase of 67.39% and 82.75% for makespan and energy efficiency, respectively. In the future, it is anticipated that the suggested scheduling approach will undergo testing using a broader dataset of workloads. In addition, it is worth exploring the potential benefits of utilizing an edge-cloud structure in order to potentially achieve cost reduction and minimize execution delays.

REFERENCES

[1] J. Kim, H. Roh, and S. Park, “Selective I/O bypass and load balancing method for write-through SSD caching in big data analytics,” *IEEE Transactions on Computers*, vol. 67, no. 4, pp. 589–595, Apr. 2018, doi: 10.1109/TC.2017.2771491.

[2] N. Zhang, M. Wang, Z. Duan, and C. Tian, “Verifying properties of mapreduce-based big data processing,” *IEEE Transactions on Reliability*, vol. 71, no. 1, pp. 321–338, Mar. 2022, doi: 10.1109/TR.2020.2999441.

[3] D. Yang, D. Cheng, W. Rang, and Y. Wang, “Joint optimization of mapreduce scheduling and network policy in hierarchical data centers,” *IEEE Transactions on Cloud Computing*, vol. 10, no. 1, pp. 461–473, Jan. 2022, doi: 10.1109/TCC.2019.2961653.

[4] A. Kumar, N. Varshney, S. Bhatiya, and K. U. Singh, “Replication-based query management for resource allocation using hadoop and MapReduce over big data,” *Big Data Mining and Analytics*, vol. 6, no. 4, pp. 465–477, Dec. 2023, doi: 10.26599/BDMA.2022.9020026.

[5] X. Li, F. Chen, R. Ruiz, and J. Zhu, “MapReduce task scheduling in heterogeneous geo-distributed data centers,” *IEEE Transactions on Services Computing*, vol. 15, no. 6, pp. 3317–3329, Nov. 2022, doi: 10.1109/TSC.2021.3092563.

[6] R. Jeyaraj and A. Paul, “Optimizing MapReduce task scheduling on virtualized heterogeneous environments using ant colony optimization,” *IEEE Access*, vol. 10, pp. 55842–55855, 2022, doi: 10.1109/ACCESS.2022.3176729.

[7] X. Li, W. Yu, R. Ruiz, and J. Zhu, “Energy-aware cloud workflow applications scheduling with geo-distributed data,” *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 891–903, Mar. 2022, doi: 10.1109/TSC.2020.2965106.

[8] P. Pujar, A. Kumar, and V. Kumar, “Efficient plant leaf detection through machine learning approach based on corn leaf image classification,” *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 13, no. 1, pp. 1139–1148, Mar. 2024, doi: 10.11591/ijai.v13.i1.pp1139-1148.

[9] S. H. Sreedhara, V. Kumar, and S. Salma, “Efficient big data clustering using adhoc fuzzy C means and auto-encoder CNN,” in *Inventive Computation and Information Technologies: Proceedings of ICICIT 2022*, Singapore: Springer, 2023, pp. 353–368, doi: 10.1007/978-981-19-7402-1_25.

[10] S. Qin, D. Pi, Z. Shao, Y. Xu, and Y. Chen, “Reliability-aware multi-objective memetic algorithm for workflow scheduling problem in multi-cloud system,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 4, pp. 1343–1361, Apr. 2023, doi: 10.1109/TPDS.2023.3245089.

[11] S. Qin, D. Pi, Z. Shao, and Y. Xu, "A knowledge-based adaptive discrete water wave optimization for solving cloud workflow scheduling," *IEEE Transactions on Cloud Computing*, vol. 11, no. 1, pp. 200–216, Jan. 2023, doi: 10.1109/TCC.2021.3087642.

[12] E. Khodayarseresht and A. S. -Sendi, "A multi-objective cloud energy optimizer algorithm for federated environments," *Journal of Parallel and Distributed Computing*, vol. 174, pp. 81–99, Apr. 2023, doi: 10.1016/j.jpdc.2022.12.007.

[13] 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, Oct. 2022, doi: 10.1109/TCC.2021.3057422.

[14] G. Papadimitriou *et al.*, "End-to-end online performance data capture and analysis for scientific workflows," *Future Generation Computer Systems*, vol. 117, pp. 387–400, Apr. 2021, doi: 10.1016/j.future.2020.11.024.

[15] R. F. D. Silva, H. Casanova, A. -C. Orgerie, R. Tanaka, E. Deelman, and F. Suter, "Characterizing, modeling, and accurately simulating power and energy consumption of I/O-intensive scientific workflows," *Journal of Computational Science*, vol. 44, Jul. 2020, doi: 10.1016/j.jocs.2020.101157.

[16] S. Hedayati, N. Maleki, T. Olsson, F. Ahlgren, M. Seyednezhad, and K. Berahmand, "MapReduce scheduling algorithms in Hadoop: a systematic study," *Journal of Cloud Computing*, vol. 12, no. 1, Oct. 2023, doi: 10.1186/s13677-023-00520-9.

[17] N. Garg, Neeraj, M. Raj, I. Gupta, V. Kumar, and G. R. Sinha, "Energy-efficient scientific workflow scheduling algorithm in cloud environment," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–12, Mar. 2022, doi: 10.1155/2022/1637614.

[18] J. Wang, X. Li, R. Ruiz, J. Yang, and D. Chu, "Energy utilization task scheduling for MapReduce in heterogeneous clusters," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 931–944, Mar. 2022, doi: 10.1109/TSC.2020.2966697.

[19] Q. Liu *et al.*, "Cloud, edge, and mobile computing for smart cities," in *Urban Informatics*, Singapore: Springer, 2021, pp. 757–795, doi: 10.1007/978-981-15-8983-6_41.

[20 S. Danthuluri and S. Chitnis, "Energy and cost optimization mechanism for workflow scheduling in the cloud," *Materials Today: Proceedings*, vol. 80, pp. 3069–3074, 2023, doi: 10.1016/j.matpr.2021.07.168.

[21] N. Bacanin, M. Zivkovic, T. Bezdan, K. Venkatachalam, and M. Abouhawwash, "Modified firefly algorithm for workflow scheduling in cloud-edge environment," *Neural Computing and Applications*, vol. 34, no. 11, pp. 9043–9068, Jun. 2022, doi: 10.1007/s00521-022-06925-y.

[22] 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, Apr. 2023, doi: 10.1109/TC.2022.3191733.

[23] J. Anselmi and N. Walton, "Stability and optimization of speculative queueing networks," *IEEE/ACM Transactions on Networking*, vol. 30, no. 2, pp. 911–922, Apr. 2022, doi: 10.1109/TNET.2021.3128778.

[24] J. P. -Valero, A. Banchs, P. Serrano, J. Ortín, J. G. -Reinoso, and X. C. -Pérez, "Energy-aware adaptive scaling of server farms for NFV with reliability requirements," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4273–4284, May 2024, doi: 10.1109/TMC.2023.3288604.

[25] 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, Sep. 2022, doi: 10.1109/TSC.2021.3054754.

[26] J. Son, T. He, and R. Buyya, "CloudSimSDN-NFV: Modeling and simulation of network function virtualization and service function chaining in edge computing environments," *Software: Practice and Experience*, vol. 49, no. 12, pp. 1748–1764, Dec. 2019, doi: 10.1002/spe.2755.

BIOGRAPHIES OF AUTHOR

Farha Naaz earned her B.E. degree in CSE from K.B.N College of Engineering under VTU University, Belagavi in 2012 and master degree in M.Tech. in CSE from K.B.N College of Engineering under VTU University in 2015. Currently she is research scholar at K.B.N University doing her Ph.D. in CSE. Her areas of interest are big data analytics, cloud computing, and computer networks. She can be contacted at email: farhanaaz_12@rediffmail.com.

Sameena Banu is an associate professor in the Department of Computer Science and Engineering, Faculty of Engineering and Technology, Khaja Banda Nawaz University, Kalaburgi. She is qualified in bachelor and master degree in Computer Science and Engineering and Ph.D. in Computer Science and Engineering in the area of digital image processing. She is having 21 years of teaching experience. Her areas of interest are image processing, machine learning, artificial intelligence, and internet of things. She can be contacted at email: sameenabanu271@gmail.com.