

Optimized load balancing mechanism in parallel computing for workflow in cloud computing environment

Asma Anjum, Asma Parveen

Department of Computer Science and Engineering, Khaja Banda Nawaz College of Engineering, Kalaburagi, India

Article Info

Article history:

Received Jul 29, 2022

Revised Oct 15, 2022

Accepted Dec 10, 2022

Keywords:

Cloud computing

Load balancing

Makespan

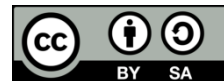
OLBP

Parallel computation

ABSTRACT

Cloud computing gives on-demand access to computing resources in metered and powerfully adapted way; it empowers the client to get access to fast and flexible resources through virtualization and widely adaptable for various applications. Further, to provide assurance of productive computation, scheduling of task is very much important in cloud infrastructure environment. Moreover, the main aim of task execution phenomena is to reduce the execution time and reserve infrastructure; further, considering huge application, workflow scheduling has drawn fine attention in business as well as scientific area. Hence, in this research work, we design and develop an optimized load balancing in parallel computation aka optimal load balancing in parallel computing (OLBP) mechanism to distribute the load; at first different parameter in workload is computed and then loads are distributed. Further OLBP mechanism considers makespan time and energy as constraint and further task offloading is done considering the server speed. This phenomenon provides the balancing of workflow; further OLBP mechanism is evaluated using cyber shake workflow dataset and outperforms the existing workflow mechanism.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Asma Anjum

Department of Computer Science and Engineering, Khaja Banda Nawaz College of Engineering

Rauza Buzurg, Kalaburagi, Karnataka, India

Email: asmacs13@gmail.com

1. INTRODUCTION

Cloud computing (CC) is nothing but specific computing paradigm that provides the absolute management and services on the internet; further, cloud computing is also referred as the fine computing environment with characteristics of being convenient, sharing and on demand services which includes servers, storage, applications, services, networks and so on. Moreover, these can be provided in rapid and efficient manner without much effort [1]. various application and resources; virtual machine (VM) distributes the various resource such as processing cores, system BUS and so on, however these resources are restricted through processing power phenomena [2]. Cloud computing have various application from industrial point as well as academic point as well as industrial point; moreover, in order to take advantage of computing resources, workflow is designed. Workflows comprises large number of dependent and independent task and possesses huge applications in scientific purpose as well as business purpose; these application includes bio-informatics, medical, weather forecasting, astronomy, and so forth [3], [4].

Figure 1 shows the simple workflow and scientific workflow is represented in the performance evaluation section. Moreover, Figure 1 shows that there are 9 tasks. Simple workflow is easy to schedule, however scientific workflow is quite complicated for example montage, epigenomics, SIPHT and cybershake workflow applied in various research like earthquakes, astronomy and so on. Moreover, scientific workflow

involves complex data of various size and requires the higher processing power; nevertheless, through providing the on-demand virtual resource, cloud-computing paradigm helps in handling these workflows.

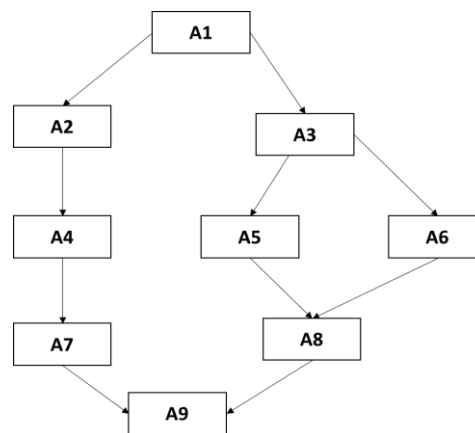


Figure 1. Simple workflow

Early work of load balancing was more focus towards traditional approach like dynamic load balancing, round robin, ant colony optimization, first come first serve (FCFS). However, most common approach used were first input first output (FIFO) and weighted round robin (WRR). Nevertheless, it also supports simulation and modelling of cloud computing environments which consists of inter-networked and single clouds, also it exposes the customized interface to implement load balancing and scheduling policies into the VMs as well as provisioning mechanism for VM allocation, Abdullahi *et al.* [5], Yaghmaee *et al.* [6] for more details. Moreover, considering the importance of workflow scheduling, several research were carried out which is extensively discussed in literature review section; few approaches are single approach or hybrid approach like meta-heuristic mechanism produce better results in comparison with other model. Moreover, methods like heterogeneous earliest finish time (HEFT) is heterogeneous mechanism where each task are mapped in priority order to virtual machine explained in [7]–[9]. Other method like gravitational search algorithm (GSA) utilizes gravitational law, as it starts with random particles set and each particle provides solution and mass are computed using the fitness function as explained by researchers [10]–[14]. Figure 2 shows the workflow of scheduling. Power consumption and simplicity and critical task is to handle the various challenges which are derived from the cloud model [13], [14].

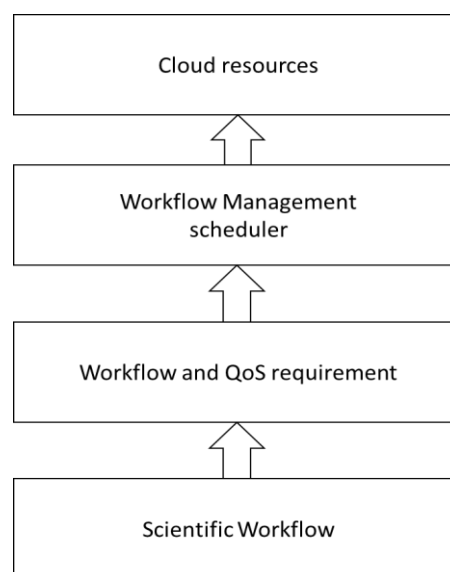


Figure 2. Workflow in cloud computing

Workflow scheduling is considered as the NP-complete problem that are widely researched for various paradigms including cluster computing and grid environment. Further, through cloud-based environment the scientific workflow can be scheduled in efficient manner. However, for huge value of m and n , achieving the optimal solution through existing method is really expensive such as brute force and its lengthy process. Although methods like metaheuristic approach have been proposed but they suffer from their own demerits. Hence motivated by the above-discussed points, in this research work we design and develop a mechanism optimized load balancing in parallel computation (OLBP) to distribute the load in efficient manner for achieving the improvisation in workflow scheduling; further contribution of research work is highlighted: at first, we carry extensive survey of existing mechanism to optimize the energy and makespan. Further, an optimized load-balancing scheme in parallel computing is designed so that loads are distributed in a manner, OLBP considers the task offloading through server speed. Moreover, execution speed and energy consumption are taken as constraint and further it is optimized integrated with load balancing approach. OLBP is evaluated considering the scientific workflow cybershake and its variant; four distinctive instances are created and compared with the existing protocol to prove the efficiency of proposed mechanism. The first component of this research begins with a backdrop of computation and the need for the cloud computing phenomenon. We next address the importance of scheduling mechanisms before concluding with motivation and the value of the research work. The second segment examines the various current protocols and their shortcomings; the third section also develops OLBP and its mathematical formulation. In the fourth segment, OLBP is assessed by taking into account numerous instances.

2. LITERATURE SURVEY

In cloud computing paradigm, the phenomena of task scheduling are to identify the optimal mapping among the virtual machine and tasks in accordance with cloud model and users' goal such as less execution time and optimal productivity. This can be achieved through load balancing and optimization mechanism as shown by Yin *et al.* [15], Gao *et al.* [16]. In general, single optimization mechanism approach were developed namely Min-Min by Patel *et al.* [17], MaxMin Mao *et al.* [18], Kong *et al.* [19]; further, improvisation has been explained by Zhang *et al.* [20] where segmented Min-Min approach were developed. In here, task was pre-ordered before the completion time, later designed sequence was segmented and MinMin approach were applied to these designed segments. Although algorithm performed better than the MinMin when task length changed dramatically through providing longer task to be executed first. Etmnani and Naghizadeh [21] adopted a novel scheduling mechanism based on MaxMin and MinMin; Further standard deviation (SD) was computed in from both algorithm on virtual machines, also Devipriya and Ramesh [22] developed improvised MaxMin algorithm depending on scheduling large, task execution time to virtual machines with less computing speed, this method successfully minimized the task execution time. Although the above improvised version of MinMin and MaxMin minimizes the task execution time and it is easy to perform but these approaches lead to unbalancing load and long-term overload affects quality of service which is eminent in workflow scheduling, for more details refer [23], [24].

Multi-objective optimization approaches were created to address these drawbacks. The researchers [25]–[27] have created an LB-ACO-based algorithm that use the ant colony optimization (ACO) mechanism to get the best results; non-domination sorting is then used to produce a Pareto solution set that reflects the tradeoff between load balancing with makespan time in a cloud computing environment. Furthermore, a task-scheduling mechanism based on the genetic ACO mechanism was created in [28]. As described by Cui and Zhang [29], genetic algorithm (GA) based task scheduling approach were developed based on GA; this approach was based on the two-dimensional coding mechanism, also genetic mutation and crossover operation were designed to produce the new offspring which for improvising the population diversity. Li *et al.* [30], Xiao [31] moreover have integrated the three mechanisms i.e., artificial bee colony (ABC), GA and ACO, a sharing module is developed for sharing the optimal solutions found through three algorithm and further solution space was created. Integration of these three model and improvised convergence accuracies mentioned by Wu and Wang [32] developed an estimation of distribution algorithm (EDA) approach for solving the parallel scheduling mechanism that has the priority constraint. Further, in literature survey we observed that most of the method ignores the load balancing mechanism and focus on single objective such as task execution time or power optimization as mentioned by Aziza and Krichen [33], Long *et al.* [34].

3. METHOD

In this section, we design and develop an optimized load balancing mechanism in Parallel computation; optimized load balancing distributes the load in efficient way such that the various constraint related to workflow is fulfilled. Task scheduling as well as resource provisioning are two obstacles that load

balancing poses in any workflow. In addition, the key goals of cloud service providers are to deliver dependable services at an affordable price with improved resource usage, availability, and energy efficiency. In case of independent task, the order of execution is not important, however for dependent task, execution order is priority, as the execution of priority task is necessary, and it gets more complicated. Direct acyclic graph (DAG) which represents the workflow where edge indicates the task and edge indicates the relation between two nodes. Figure 3 shows the load balancing mechanism and comprises various parts such as users, data center controller (DCC), load balancer where the load balancing is carried out through proposed approach, VM aka Virtual Machine manager which manages VM, VM monitor that monitor VM activity and physical server; in this work we focus on designing load balancing part.

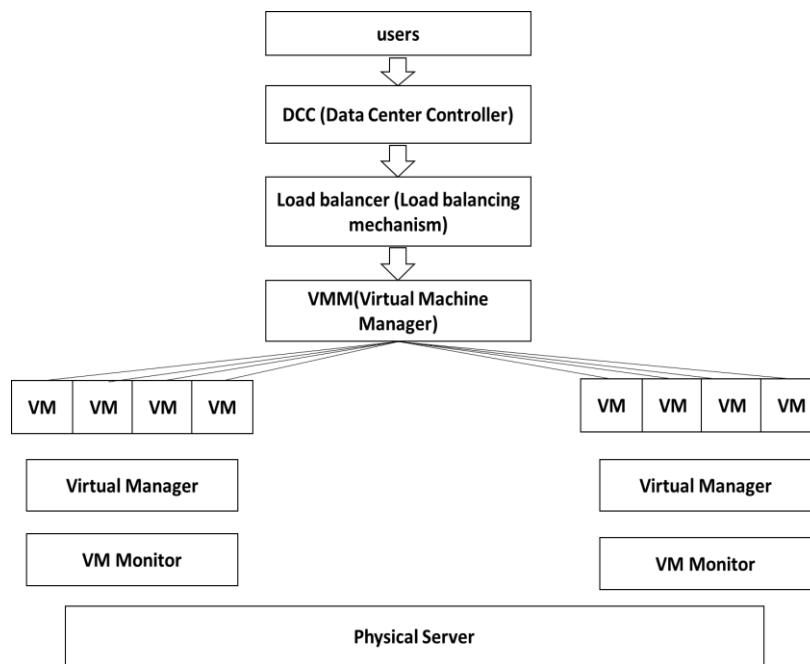


Figure 3. General load balancing

3.1. Optimized load balancing in parallel computation-OLBP steps

In this section, we go through the steps of optimized load balancing algorithm. The steps for OLBP algorithm:

Step 1: Initiate.

Step 2: Creating the analytical paradigm for parallel and heterogeneous computing.

Step 3: Power utilization is formulated.

Step 4: The designed analytical model divides the model into two constraints.

Step 4: We use the task's fixed arrival rate to calculate the service fee.

Step 5: The number of people waiting, task response time, busy server, server usage, and server speed are all calculated.

Step 6: Further, we design the load balancing through server speed and later task offloading is carried out considering makespan and power consumption.

3.2. System model

To design the load balancing mechanism in workflow, heterogeneous server model is designed; let's consider a m heterogeneous server $R_1, R_2, R_3, \dots, R_m$ of speed $r_1, r_2, r_3, \dots, r_m$ and size $l_1, l_2, l_3, l_4, \dots, l_m$. Further, this model is treated as a particular model. Further, let's assume that R_h has l_h identical server with r_h speed. Let's consider the task with arrival time A where task arrival times are independent and shared; further load distribution mechanism is parted into various sub-mechanism, hence task execution time is, let $\zeta_h = \frac{1}{w_h} = \frac{r_h}{q}$ be the service rate where average number of tasks finished through S_i server in given unit of time, then server utilization can be computed ibn (1). The above example can be considered as the average time when server is busy; let $o_{h,j}$ be the probability where there are j tasks in queue for R_h as given in (2).

$$\chi_h = \frac{A_h \bar{q}}{l_h r_h} \quad (1)$$

$$o_{h,j} = \begin{cases} o_{h,0} \frac{(l_h \chi_h)^j}{j!} & j \leq l_j \\ o_{h,0} \frac{l_h^{l_h} \chi_h^j}{l_h!} & j > l_j \end{cases} \quad (2)$$

$$o_{h,0} = \left(\sum_{j=0}^{l_h-1} \frac{(l_h \chi_h)^j}{j!} + \frac{(l_h \chi_h)^{l_h}}{l_h!} \cdot \frac{1}{1-\chi_h} \right)^{-1} \quad (3)$$

Further, probability of queueing in R_h is given as (4).

$$O_{p,h} = o_{h,0} \frac{l_h^{l_h}}{l_h!} \cdot \frac{\chi_h^{l_h}}{1-\chi_h} \quad (4)$$

Average task response is computed through (5).

$$\bar{L}_h = \sum_{j=0}^{\infty} j o_{h,j} = l_h \chi_h + \frac{\chi_h}{1-\chi_h} O_{p,h} \quad (5)$$

Average task in the model is given as:

$$S_h = \bar{w}_j \left(1 + \frac{O_{p,h}}{l_g(1-\chi_h)} \right) \quad (6)$$

$$S_h = \frac{\bar{q}}{r_j} \left(1 + o_{j,0} \frac{l_h^{l_h-1}}{l_h!} \cdot \frac{\chi_h^{l_h}}{(1-\chi_h)^2} \right) \quad (7)$$

3.3. Power consumption

Circuit delay as well as power dissipation in digital complementary metal-oxide-semiconductor (CMOS) circuits are modelled using a straightforward technique. The use of power is very important here. Can be used to solve the basic power problem.

$$O = zBU^2e \quad (8)$$

The loading capacitance is denoted by B in (8), the activity factor by a , the supply voltage by U , and the clock frequency by e . Additionally, because the system in this case is heterogeneous, it is considered that the server R_h has r_h speed. Two separate possibilities are also taken into consideration. Given that there is no task to be completed in the initial scenario, the energy consumed can be given as (9).

$$O_h = l_h O_j^* + A_h \bar{q} \vartheta_h r_h^{\delta_h-1} \quad (9)$$

Due to the server's high-power consumption, speed is $\vartheta_h r_h^{\delta_h}$ with l_h server. Even though there are no tasks remaining to complete in the second situation, the server continues to run at the specified speed r_h , allowing the suggested consumption to be calculated using the (10).

$$O_h = l_h (O_h^* + \vartheta_h r_h^{\delta_h}) \quad (10)$$

3.4. Optimal load balancing in parallel computing

Let's consider that server R_h server of l_h speed be the $r_{h,j}$, here j is the task that is queued in the system; further, the task value should not be zero. Furthermore, we consider the speed mechanism i.e., $(r_{h,0}, r_{h,1}, r_{h,3}, \dots)$ of $R_{h,j}$, these speed scheme directly represent and reflect the power and speed dependency of workload. Moreover, this are the three different scenarios. Consider that the job that is queuing in the network is called j , and that server R_h server of speed l_h is called $r_{h,j}$. The performance expectancy must not be zero. Additionally, we consider the speed mechanism of $R(h, j)$, namely $(r_{h,0}, r_{h,1}, r_{h,3}, \dots)$, and $R_{h,j}$, which directly represents and reflects the power as well as speed dependency of the workload. Additionally, the three distinct circumstances are listed after this. For instance, if $r_{h,1} = r_{h,2} = r_{h,3} \dots = R_h$ then we use a

unit speed mechanism., If $r_{h,0} = 0$, then it is said to be idle, If $r_{h,0} = r_h$ then it is said to be the constant speed mode. Additionally, arrival rate (AR) as well as service rate (SR) having various states l are processed for additional load balancing depending on speed and power, where l refers to the task within server model,

$$\delta_{h,j} = \begin{cases} j \frac{r_{h,j}}{\bar{q}} & 1 \leq j \leq l_h - 1 \\ l_h \frac{r_{h,j}}{\bar{q}} & l \geq l_h \end{cases} \tag{11}$$

$$o_{h,j} = (o_{h,0}) A_h^j (\zeta_{h,1}, \zeta_{h,2} \dots, \zeta_{h,j})^{-1} \tag{12}$$

where $o_{h,0}$ can be formulated through the (13).

$$o_{h,0} = \left(1 + \sum_{j=1}^{l_h-1} \frac{(A_h \bar{q})}{j!} \cdot (r_{h,1} r_{h,2} \dots r_{h,j})^{-1} + \sum_{j=l_h}^{\infty} \frac{(A_h \bar{q})^k}{l_h! j^{-l_h}} \cdot (r_{h,1} r_{h,2} \dots r_{h,j})^{-1} \right)^{-1} \tag{13}$$

The average number of tasks in R_h as (14).

$$\bar{M}_h = \sum_{l=1}^{\infty} j o_{h,j} \tag{14}$$

The provides the task's RT (response time),

$$M_h = \bar{M}_h (A_h)^{-1} \tag{15}$$

the number of servers in R_h is later calculated and is supplied as (16).

$$\mathcal{A}_h = \sum_{j=l_h}^{\infty} l_h j + \sum_{l=0}^{\infty} l_h o_{h,j} \tag{16}$$

Consequently, the above is used to calculate server usage,

$$\chi_j = \mathcal{A}_h (l_h)^{-1} \tag{17}$$

$$\bar{r}_h = \sum_{k=0}^{\infty} o_{h,j} j^{r_{h,j}} \tag{18}$$

the average energy consumption is calculated last, and it is also formulated for the two situations that were described before in the same section.

$$O_h = \vartheta_h \left(l_h O_h^* + \sum_{l=0}^{n_j-1} o_{h,j} j^{r_{h,j}} + \sum_{l=0}^{n_j-1} o_{h,j} j^{r_{h,j}} l_h \right) \tag{19}$$

$$O_h = l_h \vartheta_h \sum_{l=0}^{\infty} l_h O_h^* + O_{h, j} j^{r_{h,j}} \tag{20}$$

Further we compute the server speed; in here optimized server speed is formulated; consider a server speed of R_h and represented through $\vartheta_h = (a_{i,1}, \dots, m a_{h,c_h-1}; r_{h,1}, r_{h,c_h'})$. Moreover, for given server l_h . Execution speed is $s_{h,1}$. Thus, execution speed can be formulated as (21).

$$\zeta_{h,j} = \begin{cases} j \frac{r_{h,1}}{\bar{q}} & 1 \leq j \leq l_h - 1 \\ l_h \frac{r_{h,1}}{\bar{q}} & l_h \leq j \leq a_{h,1} \\ l_h \frac{r_{h,i}}{\bar{q}} & a_{h,i-1} + 1 \leq j \leq a_{h,i} \\ l_h \frac{r_{h,i}}{\bar{q}} & j \geq a_{h,i-1} + 1 \end{cases} \tag{21}$$

Further, closed form solution is derived for various quantities such as average power, average task response, server utilization, average execution speed, these can be formulated and optimized through (22).

$$\bar{O}_h = \sum_{j=1}^{\infty} l o_{h,j} \tag{22}$$

Using above task response time is formulates as (23).

$$\ddot{S}_h = \ddot{O}_h/A_h \quad (23)$$

Further, considering these parameters we design workload strategy for load balancing.

3.5. Workload designing strategy

In this section, we design and develop the workload strategy balance which optimizes two sub problem i.e., makespan time and energy. Both are considered as the sub-problem associated with load balancing. Further, in this section optimized task offloading is designed to solve the makespan time and energy problem.

3.5.1. Makespan problem

In this section, we focus on load balancing mechanism with makespan time minimization; further considering the makespan time, distribution of load must be found which can be considered as the task arrival time to model. Thus $E(A_1, A_2 \dots A_m) = A$ where

$$\begin{aligned} E(A_1, A_2 \dots A_m) &= A_1 + A_2 + \dots + A_m \\ \vartheta_h &\text{ is less than unit} \\ \forall 1 \leq h \leq m \end{aligned} \quad (24)$$

The load-balanced is achieved through server speed and dynamic. Given the m number of a server system with system size and speed $(\varphi_i = (a_{h,1}, a_{h,2}, \dots, a_{h,c_h-1}, r_{h,1}, r_{h,1}, \dots, r_{h,e}))$ of R_h , power consumption is given as $O_1^*, O_2^*, \dots, O_h^*$, task execution requirement is given through \bar{q} , arrival rate as Λ , now we need to balance the load hence the load distribution is found; can be given in (24), Further, this is optimized in the process.

$$E(A_1, A_2 \dots A_m) = A_1 + A_2 + \dots \dots + A_0 \quad (25)$$

$$(A_1, A_2 \dots A_m) = \Phi \nabla E(A_1, A_2 \dots A_m) \quad (26)$$

$$\frac{\partial S}{\partial A_1} = \Phi \frac{\partial E}{\partial A_h} = \Phi \quad (27)$$

Φ is a multiplier, given this multiplier, we find the $A_1, A_2 \dots A_0$ used for verifying the condition this generates the value of Φ .

3.5.2. Energy optimization

Let's consider number of servers as n with speed as $l_1, l_2, l_3, \dots, l_m$ with speed $\psi_h = (a_{h,1}, a_{h,2}, \dots, a_{h,c_h-1}, r_{h,1}, r_{h,2}, \dots, r_{h,c_h})$ of R_h . $\forall 1 \leq h \leq m$. Considering the energy parameter as $O_1^*, O_2^*, \dots, O_m^*$ with requirement of task execution as \bar{q} , with arrival rate A, a load balancing must be achieved such that energy $P(\lambda_1, \lambda_2, \lambda_3)$ is minimized having constraint as given (28).

$$\begin{aligned} E(A_1, A_2 \dots A_m) &= A_1 + A_2 + \dots \dots + A_0 \\ \vartheta_h &\text{ is less than unit} \\ \forall 1 \leq h \leq m \end{aligned} \quad (28)$$

Further, the above problem is minimized through Lagrange multiplier and denoted as (29).

$$\nabla S(A_1, A_2 \dots A_m) = \Phi \nabla E(A_1, A_2 \dots A_m) \quad (29)$$

$$\frac{\partial S}{\partial A_1} = \Phi \frac{\partial E}{\partial A_h} = \Phi$$

Moreover, it is observed that $\frac{\partial S}{\partial A_1}$ is complex function of λ_i , this causes highly improbable for finding the analytical solution, hence this can be solved through given steps; (a) As, we observe that $\frac{\partial S}{\partial A_1}$ is an increasing function with given Φ , using bisection approach we find the value of A_h , (b) Moreover, the calculated $A_1 +$

$A_2 + \dots + A_m$ is used for verifying the condition $E(A_1, A_2 \dots A_m) = A$. (c) Further, this verification is employed to find Φ through bisection method. (d) Further the optimized equation is given as (30)

$$\frac{\partial S}{\partial A_1} = \frac{1}{A} \left(O_h + \frac{\partial O_h}{\partial A_h} \right) \quad (30)$$

4. PERFORMANCE EVALUATION

Resources for cloud computing have become one of the most successful real-time computing models for active usage. They are accessible, affordable, and can be used from anywhere in the world over the internet. Additionally, it may be used for a variety of purposes, one of which is workflow scheduling for scientific workflows where the tasks are interdependent and independent and have a range of benefits and drawbacks. However, these computing devices' excessive energy consumption can impair their performance. Additionally, makespan is a significant restriction, thus we introduced an OLBP for heterogeneous computing systems to efficiently reduce energy consumption while also delivering superior performance to optimize these goals. Our suggested model runs on Windows 10 64-bit with 16 GB of RAM and an Intel Core i5-4460 processor. The CPU runs at 3.20 GHz. Eclipse Neon is being used to emulate this project. We utilize CloudSim as our simulator, and the 3 editor and code are written in Java [34]. We also test the effectiveness of our suggested model using the cybershake scientific workflow in four different situations, including total time comparison and power consumption. Additionally, a graphic representation of our findings that takes into account task count, execution time, and energy consumption is provided. We have taken into consideration studies with scientific workflows of 30, 50, 100, and 1000.

4.1. Energy consumption

Energy is one of the important parameters in any optimization problem; Figure 4 shows the energy consumption comparison of existing and proposed method; for first instance, existing model requires 3495.42 whereas proposed model requires 1303.740369. For second and third instance existing model requires 8518.395166 and 18966.33731 whereas OLBP requires 1330.921945 and 1436.836915 respectively. Similarly, for fourth instance existing model requires 236303.2878 and proposed model OLBP requires 3228.604586. Further, another parameter average power is considered to evaluate the model less power required, the more efficient is the model. Figure 5 shows average power comparison where it is observed that in case of all instances, existing model requires 19.14, 20.11, 0.30, and 20.11 whereas proposed model requires 15.81, 15.90, 15.90, and 15.90 respectively for all four instances.

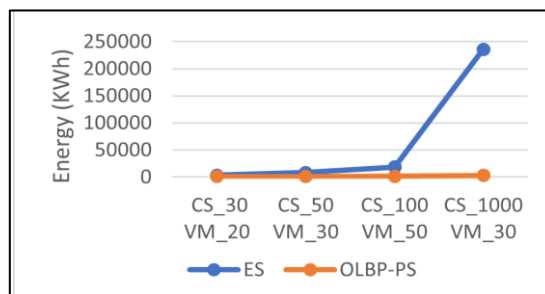


Figure 4. Energy consumption

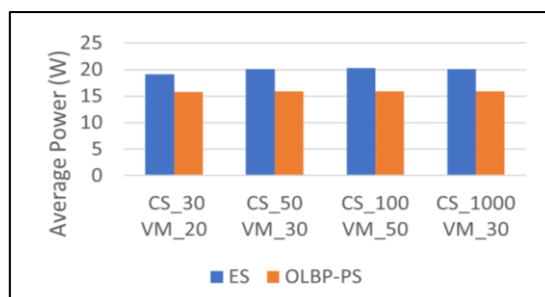


Figure 5. Average power

4.2. Makespan time

Here, for first, second, third and fourth instance takes makespan time of 6359.41, 14448.9, 30124.41, 74543 whereas OLBP requires makespan time of 2938.22, 2953.86, 3116.22, 4794.39 and 1280.85 respectively. Figure 6 displays the power sum comparison. Figure 7 shows the makespan time comparison.

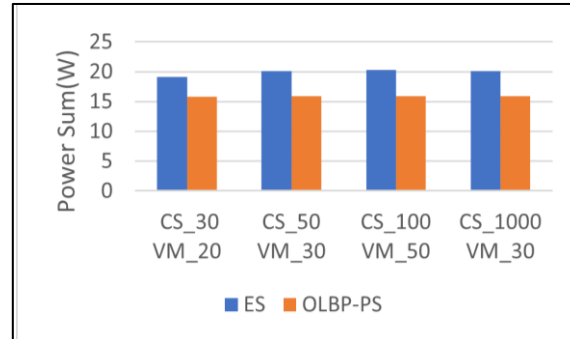


Figure 6. Power sum

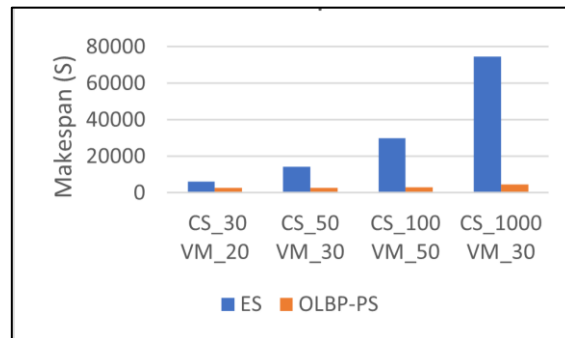


Figure 7. Makespan

4.3. Comparative analysis

In this section, we compare the effectiveness of the existing and suggested method while taking into account a number of different parameters, including energy consumption, power total, average power, and makespan time on the cybershake workflow. In the above section, we observed that, in terms of energy for instance 1, there is 62% less consumption, for instance 2, 84.37% less consumption, for instance 3, 92.42% less consumption and 98.63% less consumption for instance 4. Similarly, considering average power for instance 1, instance 2, instance 3, and instance 4, proposed model consumes 17.40%, 20.96%, 21.70%, and 20.96% efficient than the existing mechanism respectively. Furthermore, considering the power sum as parameter, proposed model consumes 61.83%, 83.41%, 91.90%, and 94.91% less power sum than the existing protocol for instance 1 to instance 4 respectively. At last, makespan time is compared and it is observed that proposed model possesses 53.79% efficient for instance 1, 79.55% efficient for instance 2, 89.65% efficient for instance 3 and 93.56% efficient for instance 4 than the existing model. Furthermore, throughout the comparison it is observed that in case of each parameter, as the workflow variant changes and as number of tasks increases performance degrades marginally, however proposed model remains either constant or decreases comparatively less.

5. CONCLUSION

Workflow management systems is considered to be eminent in construction of various application such as e-commerce, scientific application and so forth through automation process of inter-enterprise and intra-enterprise business model. Moreover, to handle the dynamic business environment and increase in global competition has led the researcher to design the scalable and flexible business environment; these phenomena can be achieved through load balancing. Further, one of the barriers to green cities and a pressing

issue that must be resolved is the high energy consumption and increased makespan time in cloud data centers. To date, many load-balancing algorithms have been created in an effort to cut down on energy use and process execution timespan. As a result, in this research effort OLBP is established, which schedules and distributes the workload in a way that important parameters like power, energy, and makespan time are maximized. This research work is aimed to address the issue. Moreover, to evaluate the OLBP, four distinctive instances are designed; each instances comprises workflow variants with random number of virtual machines. Further comparative analysis indicates that OLBP outperforms the existing model in terms of average power, energy consumption, makespan time. Although OLBP outperforms the existing protocol, there are lot of research which needs to research; in future we would be focusing on optimizing the other performance parameter such as fault tolerance, reliability, and cost.





REFERENCES

- [1] F. Tao, Y. Cheng, L. D. Xu, L. Zhang, and B. H. Li, "CCIoT-CMfg: Cloud computing and internet of things-based cloud manufacturing service system," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1435–1442, May 2014, doi: 10.1109/TII.2014.2306383.
- [2] M. Masdari and M. Zangakani, "Green cloud computing using proactive virtual machine placement: challenges and issues," *Journal of Grid Computing*, vol. 18, no. 4, pp. 727–759, Dec. 2020, doi: 10.1007/s10723-019-09489-9.
- [3] L. Mashayekhy, M. M. Nejad, D. Grosu, Q. Zhang, and W. Shi, "Energy-aware scheduling of MapReduce jobs for big data applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2720–2733, Oct. 2015, doi: 10.1109/TPDS.2014.2358556.
- [4] Y. Wu, X. Tan, L. Qian, D. H. K. Tsang, W. Z. Song, and L. Yu, "Optimal pricing and energy scheduling for hybrid energy trading market in future smart grid," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1585–1596, Dec. 2015, doi: 10.1109/TII.2015.2426052.
- [5] M. Abdullahi, M. A. Ngadi, S. I. Dishing, S. M. Abdulhamid, and B. I. E Ahmad, "An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment," *Journal of Network and Computer Applications*, vol. 133, pp. 60–74, May 2019, doi: 10.1016/j.jnca.2019.02.005.
- [6] M. H. Yaghmaee, M. Moghaddassian, and A. L. Garcia, "Autonomous two-tier cloud-based demand side management approach with microgrid," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1109–1120, Jun. 2017, doi: 10.1109/TII.2016.2619070.
- [7] M. Hosseinzadeh, M. Y. Ghafour, H. K. Hama, B. Vo, and A. Khoshnevis, "Multi-objective task and workflow scheduling approaches in cloud computing: A comprehensive review," *Journal of Grid Computing*, vol. 18, no. 3, pp. 327–356, Sep. 2020, doi: 10.1007/s10723-020-09533-z.
- [8] X. Li, P. Garraghan, X. Jiang, Z. Wu, and J. Xu, "Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 6, pp. 1317–1331, Jun. 2018, doi: 10.1109/TPDS.2017.2688445.
- [9] H. Yan *et al.*, "Cost-efficient consolidating service for aliyun's cloud-scale computing," *IEEE Transactions on Services Computing*, vol. 12, no. 1, pp. 117–130, Jan. 2019, doi: 10.1109/TSC.2016.2612186.
- [10] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, "Real-time tasks oriented energy-aware scheduling in virtualized clouds," *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 168–180, Apr. 2014, doi: 10.1109/TCC.2014.2310452.
- [11] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, and K. Vahi, "Characterizing and profiling scientific workflows," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 682–692, Mar. 2013, doi: 10.1016/j.future.2012.08.015.
- [12] H. Chen, X. Zhu, G. Liu, and W. Pedrycz, "Uncertainty-aware online scheduling for real-time workflows in cloud service environment," *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1167–1178, Jul. 2021, doi: 10.1109/TSC.2018.2866421.
- [13] G. Xie, J. Jiang, Y. Liu, R. Li, and K. Li, "Minimizing energy consumption of real-time parallel applications using downward and upward approaches on heterogeneous systems," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1068–1078, Jun. 2017, doi: 10.1109/TII.2017.2676183.
- [14] H. Rong, H. Zhang, S. Xiao, C. Li, and C. Hu, "Optimizing energy consumption for data centers," *Renewable and Sustainable Energy Reviews*, vol. 58, pp. 674–691, May 2016, doi: 10.1016/j.rser.2015.12.283.
- [15] Y. Yin, F. Yu, Y. Xu, L. Yu, and J. Mu, "Network location-aware service recommendation with random walk in cyber-physical systems," *Sensors (Switzerland)*, vol. 17, no. 9, p. 2059, Sep. 2017, doi: 10.3390/s17092059.
- [16] H. Gao, Y. Duan, H. Miao, and Y. Yin, "An approach to data consistency checking for the dynamic replacement of service process," *IEEE Access*, vol. 5, pp. 11700–11711, 2017, doi: 10.1109/ACCESS.2017.2715322.
- [17] G. Patel, R. Mehta, and U. Bhoi, "Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing," *Procedia Computer Science*, vol. 57, pp. 545–553, 2015, doi: 10.1016/j.procs.2015.07.385.
- [18] Y. Mao, X. Chen, and X. Li, "Max–min task scheduling algorithm for load balance in cloud computing," in *Proceedings of International Conference on Computer Science and Information Technology*, vol. 255, 2014, pp. 457–465, doi: /10.1007/978-81-322-1759-6_53.
- [19] X. Kong, C. Lin, Y. Jiang, W. Yan, and X. Chu, "Efficient dynamic task scheduling in virtualized data centers with fuzzy prediction," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1068–1077, Jul. 2011, doi: 10.1016/j.jnca.2010.06.001.
- [20] M. Y. Wu, W. Shu, and H. Zhang, "Segmented Min-min: A static mapping algorithm for meta-tasks on heterogeneous computing systems," in *Proceedings of the Heterogeneous Computing Workshop, HCW, 2000*, pp. 375–385, doi: 10.1109/hcw.2000.843759.
- [21] K. Etminani and M. Naghibzadeh, "A min-min max-min selective algorithm for grid task scheduling," in *2007 3rd IEEE/IFIP International Conference in Central Asia on Internet, ICI 2007*, Sep. 2007, pp. 1–7, doi: 10.1109/canet.2007.4401694.
- [22] S. Devipriya and C. Ramesh, "Improved max-min heuristic model for task scheduling in cloud," in *Proceedings of the 2013 International Conference on Green Computing, Communication and Conservation of Energy, ICGCE 2013*, Dec. 2013, pp. 883–888, doi: 10.1109/ICGCE.2013.6823559.





- [23] E. K. Tabak, B. B. Cambazoglu, and C. Aykanat, "Improving the performance of independent task assignment heuristics minmin, maxmin and sufferage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, pp. 1244–1256, May 2014, doi: 10.1109/TPDS.2013.107.
- [24] Y. Yin, S. Aihua, G. Min, X. Yueshen, and W. Shuoping, "QoS prediction for web service recommendation with network location-aware neighbor selection," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 4, pp. 611–632, May 2016, doi: 10.1142/S0218194016400040.
- [25] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," in *Proceedings - 2013 8th International Conference on Computer Engineering and Systems, ICCES 2013*, Nov. 2013, pp. 64–69, doi: 10.1109/ICCES.2013.6707172.
- [26] H. Gao, D. Chu, Y. Duan, and Y. Yin, "Probabilistic model checking-based service selection method for business process modeling," *International Journal of Software Engineering and Knowledge Engineering*, vol. 27, no. 6, pp. 897–923, Aug. 2017, doi: 10.1142/S0218194017500334.
- [27] A. Gupta and R. Garg, "Load balancing based task scheduling with ACO in cloud computing," in *2017 International Conference on Computer and Applications, ICCA 2017*, Sep. 2017, pp. 174–179, doi: 10.1109/COMAPP.2017.8079781.
- [28] C. Y. Liu, C. M. Zou, and P. Wu, "A task scheduling algorithm based on genetic algorithm and ant colony optimization in cloud computing," in *Proceedings - 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, DCABES 2014*, Nov. 2014, pp. 68–72, doi: 10.1109/DCABES.2014.18.
- [29] Y. Cui and Z. Xiaoqing, "Workflow tasks scheduling optimization based on genetic algorithm in clouds," in *2018 3rd IEEE International Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2018*, Apr. 2018, pp. 6–10, doi: 10.1109/ICCCBDA.2018.8386458.
- [30] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang, "Cloud task scheduling based on load balancing ant colony optimization," in *Proceedings - 2011 6th Annual ChinaGrid Conference, ChinaGrid 2011*, Aug. 2011, pp. 3–9, doi: 10.1109/ChinaGrid.2011.17.
- [31] F. U. Xiao, "Task scheduling scheme based on sharing mechanism and swarm intelligence optimization algorithm in cloud computing," *Computer Science*, vol. 45, no. 1, p. 303307, 2018.
- [32] C. G. Wu and L. Wang, "A multi-model estimation of distribution algorithm for energy efficient scheduling under cloud computing system," *Journal of Parallel and Distributed Computing*, vol. 117, pp. 63–72, Jul. 2018, doi: 10.1016/j.jpdc.2018.02.009.
- [33] H. Aziza and S. Krichen, "Bi-objective decision support system for task-scheduling based on genetic algorithm in cloud computing," *Computing*, vol. 100, no. 2, pp. 65–91, Feb. 2018, doi: 10.1007/s00607-017-0566-5.
- [34] W. Long, L. Yuqing, and X. Qingxin, "Using cloudsims to model and simulate cloud computing environment," in *Proceedings - 9th International Conference on Computational Intelligence and Security, CIS 2013*, Dec. 2013, pp. 323–328, doi: 10.1109/CIS.2013.75.

BIOGRAPHIES OF AUTHORS



Asma Anjum     is a full-time research scholar at Visvesvaraya Technological University, Belagavi, and Karnataka. She has completed her B. E and MTech in Computer Science And Engineering in the year 2015 and 2017 respectively. She has 5 international publications amongst which three are IEEE conferences and Scopus indexed. Her research interest includes networking and cloud computing. She can be contacted at email: asmacs13@gmail.com.



Dr. Asma Parveen     got graduated in Electrical Engineering in 1993 and completed post-graduation in Computer Science and Engineering in 2004 and in 2016 she was awarded Ph.D. in Computer Science and Engineering. She has published many research papers in leading international journals and conference proceedings. She can be contacted at email: drasma.cse@gmail.com.