Computer Systems Science & Engineering
DOI:10.32604/csse.2022.018573
Article

Tech Science Press

Energy-Aware Scheduling for Tasks with Target-Time in Blockchain based Data Centres

I. Devi* **and G.R. Karpagam**

Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, 641004, India
*Corresponding Author: I. Devi. Email: deviilangovan@yahoo.com
Received: 12 March 2021; Accepted: 30 April 2021

Abstract: Cloud computing infrastructures have intended to provide computing services to end-users through the internet in a pay-per-use model. The extensive deployment of the Cloud and continuous increment in the capacity and utilization of data centers (DC) leads to massive power consumption. This intensifying scale of DCs has made energy consumption a critical concern. This paper emphasizes the task scheduling algorithm by formulating the system model to minimize the makespan and energy consumption incurred in a data center. Also, an energy-aware task scheduling in the Blockchain-based data center was proposed to offer an optimal solution that minimizes makespan and energy consumption. The established model was analyzed with a target-time responsive precedence scheduling algorithm. The observations were analyzed and compared with the traditional scheduling algorithms. The outcomes exhibited that the developed solution incurs better performance with a response to resource utilization and decreasing energy consumption. The investigation revealed that the applied strategy considerably enhanced the effectiveness of the designed schedule.

Keywords: Cloud computing; task scheduling; blockchain; energy-aware; virtual machine

1 Introduction

Cloud computing is the provisioning of computing amenities like storage, servers, network, databases, software, intelligence, and like as services through the internet. Cloud follows "Everything as a Service (XaaS)" in a pay-per-use manner. In addition to conventional web services, trillions of devices upload the data to Cloud. Thus, the volume of the data was increasing tremendously. Organizations, business firms, government agencies, and individuals realized the cloud facilities as the Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and Software as a Service (SaaS) [1]. Big business ventures like Microsoft, Amazon, Google, and IBM set up their data center for their business and payable service.

On the other hand, the placement of several DCs causes more energy consumption, leading to a financial burden to the nation. India is the second-largest marketplace for data center infrastructure and the second firmest and emerging market in Asia/Pacific, subsequently after China. By 2020, India will be the focus with $7 billion data center market. With the arrival of the Internet of Things (IoT) and Artificial

Intelligence, India is the prevalent consumer of data worldwide. As IoT and AI are power-consuming processes, they will need high-density DCs with a continuous power supply. The extensive organization and application of Cloud and virtualization have resulted in implementing a large-scale data center that offered consistent cloud services. This consequently led to a tremendous increase in energy consumption. Minimization of energy consumption was said to be a significant challenge for data centers in the Cloud. Thus, high energy costs incurred by DCs induced the researchers to find the perspectives for achieving low energy consumption. In the cloud environment, scheduling of the processing units was said to be a crucial issue. Like other processing units, tasks must be scheduled in the Cloud to maximize utilization, fast job completion, low energy consumption, and the like. The scheduling depicts the process of assigning the tasks to available resources (virtual machine). In the context of task scheduling, elasticity was termed as the point to which the system was capable of allocating and de-allocating the available resources so that the accessible resources need to match the demand. A virtuous task scheduling leads to improved performance of the system and thereby aid in attaining better Quality of Service (QoS). In general, task scheduling intended to acquire reduced makespan with increased resource utilization. Though several task scheduling algorithms were offered, more enhancement was required to make the cloud system a better one. Fig. 1: Task scheduling in the cloud environment. It comprises three layers: Application layer: offers the required services to the end-users; virtualization layer: a virtual resource group consists of physical servers through virtualization technology, and the Infrastructure layer: contains numerous kinds of servers. The process of allocating the tasks to a virtual machine (VM) was depicted as task scheduling; the process of giving VMs to a physical machine (PM) was stated as VM placement [2]. The end users in Cloud used numerous virtualized resources and it was challenging to perform task allocation by hand. With the aid of virtualization, complex task scheduling was carried out in the virtual machine layer. Therefore, scheduling was depicted as the most significant aspect in Cloud to allocate accessible resources to tasks successfully. Energy consumption turned out to be an important aspect in scheduling computational tasks in modern cloud systems. This induced the researchers to propose a useful service for executing the tasks while considering the concerns, namely energy consumption, execution time, and resource utilization.

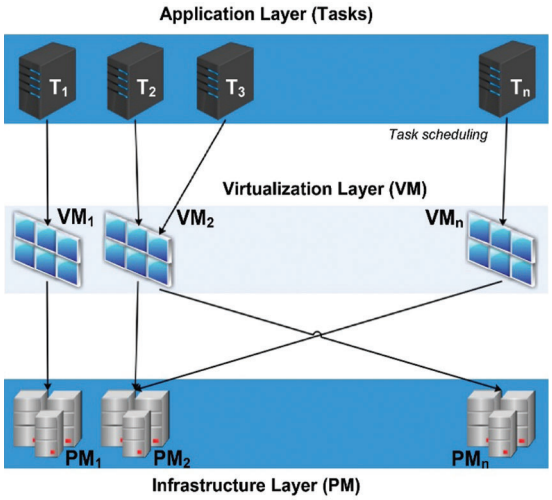

Figure 1: Task scheduling in cloud environment

The open and dynamic nature of a cloud environment is susceptible to concerns, namely accessibility and provision of physical resources, low consumption of resources, user priority management, high

energy consumption, and so forth. Thus, this work intended to propose an energy-aware task scheduling algorithm to attain considerable enhancement in the system's performance. The proposed approach to handling the issues at hand and considering the dynamic characteristics of Cloud makes our solution more robust, flexible, and more practical in real-world applications. The proposed solution intends to lessen energy consumption while attaining target-time responsiveness and realizing efficient resource utilization. The contributions of this paper were presented as follows:

• To construct the problem depicting the minimization of the energy consumption and makespan in a blockchain-based DC.

• To reduce the energy consumption incurred from task scheduling in Cloud by proposing an energy-aware task scheduling in blockchain-based DCs to exclude the energy consumption that prevails in conventional models. The failure of one DC didn't cause a single-point failure attack. Thus, it offered better robustness to the system.

• To perform simulations to analyze the performance characteristics of the proposed algorithm using google usage trace. Authors are required to adhere to this Microsoft Word template in preparing their manuscripts for submission. It will speed up the review and typesetting process.

2 Structure and Literature Review

Section 2 includes the state-of-art works related to this research work. Section 3 comprises the system model. In Section 4, concerns in Cloud were presented. Energy consumption was considered as the significant problem in the course of the scheduling of tasks in Cloud. In Section 5, the fundamentals of blockchain technology and essential components were presented. Section 6 introduced the energy-aware blockchain-based task scheduling. In Section 7, the observations from the proposed model and the comparison of the proposed algorithm with the traditional task scheduling algorithms were presented. In Section 8, the summary of the research work and future enhancements were presented.

In the cloud environment, energy-aware task scheduling was an important concern and gained more consideration. In a heterogeneous environment, an energy-efficient task scheduling algorithm was proposed to solve the shortcomings related to the consolidation of tasks and scheduling. The developed algorithm considered total resource utilization and completion time as the performance constraints, and it followed the normalization method to obtain precise scheduling decisions [3]. A scheduling algorithm that handles both load balancing and temperature was designed and developed. The proposed algorithm allocates a PM to VM in conformity with the user desires, load, and temperature of the host, thereby conserving the QoS. The VMs were assigned concerning CPU utilization and temperature of processors so that the host might not be over-loaded [4]. In energy and deadline-based task scheduling, the tasks and datasets were modeled as a binary tree following the data correlation clustering algorithm. In line with the task requirement degree, tree-to-tree task schedule was proposed. The intention was to obtain decreased global time consumption on data transmission and optimized resource utilization [5]. The tasks were scheduled to accessible VMs by considering the processing power of VM and tasks. Here, VMs were provided with the portion of the complete requested power in accordance with its power factor. The processing of tasks amongst distinct VMs and necessary power incurred by VM and tasks were considered the prime scheduling criteria [6]. The scheduling of tasks in the existence or lack of network budget by means of distantly distributed data centers. The scheduling problem was presented as linear programming (LP) problem. At that time, the convex objective function was used to convert it into a non-linear problem. Here, network bandwidth and cost were presented as the scheduling constraint [7]. In line with dynamic workflow, an algorithm was proposed which was constrained to the network. Here, a difference in the performance aspects of VM and instance attainment time was used to obtain the precise schedule of workflow [8].

Recently, blockchain gained attention by researchers in several sectors like health, finance, education, and so forth. The algorithms were implemented in the smart contract of the blockchain [9]. The decentralized data structure, namely blockchain network, was employed to provide the resources with optimization of the spent energy. With the application of smart contracts, the data were stockpiled in the distributed databases in the blocks. The lowest load tasks were allocated to the datacentres in the blocks [10]. A trusted distributed audit strategy was proposed for cloud task scheduling. With the context of scheduling, a distributed solution, namely conventional cloud server and blockchain, was employed to address cloud task scheduling's integrity and security concerns [11]. An effectual energy-oriented task scheduling with deadline in cloud computing was proposed to obtain decreased energy consumption. Based on user priorities, optimized makespan was witnessed under deadline conditions [12]. Blockchain was the emerging area of research and development that noticeably benefit the energy sector operations, markets and end users. A critical review showed the significant number of use cases and business perspectives of blockchain in the energy sector. Blockchain technology have allowed the applications of minimal and sharing economy in the energy sector, which has induced the researchers to find about the novel models [13].

From the literature review, most of the works presented task scheduling and a certain type of resources like computational resources and the power spent by these resources without considering the impact of other resource types on energy consumption. In the present work, tasks were scheduled using distinct resource types, namely compute, network, and storage resources, and task constraints like execution time and target-time. Though priority was significant, few works were considered based on target-time responsive task scheduling. In the present work, an efficient solution to address energy consumption was proposed. In a blockchain-based decentralized task management framework, the tasks were scheduled by DCs themselves without depending on the scheduler in Cloud DCs. From the literature, the significance of smart contracts in BC network was emphasized. In the BC-based task management framework, the intention was to reduce the energy consumption, makespan with maximization of resource utilization. The present work attempts to execute the algorithm to perform the tasks based on maximizing the resource application and declining the total completion time.

2.1 Novelty and Significance

In the present work, an energy-aware task scheduling in blockchain-based DC was studied. In a cloud environment, each request requires a certain amount of computational resources to accomplish the tasks. These tasks and corresponding resources (VMs) were housed by several DCs. This works aims to propose a system model and an algorithm for reducing the energy consumption in a data center by scheduling the tasks to available resources so that the user demands were satisfied. The present work proposed a task management framework according to blockchain, a distributed data structure that record all the actions on the transactions to address this issue. Compared with other frameworks, the proposed system didn't need any scheduler that incurs any energy overhead and reduces the robustness of DC. The application of blockchain with task scheduling in Cloud was an appropriate technical solution in line with the following discussions:

- The entity writer denotes the write access to blockchain, and the writer relates to a consensus member.
- If a trusted third party was not available, the writers of the system were known in the blockchain
- If the writers didn't trust each other, the application of blockchain makes sense.

3 System Model and Problem Construction

In dynamic cloud environment, task scheduling becomes hard to resolve. Thus, a virtuous task scheduling strategy needs to be designed and employed to fulfill the QoS conditions offered by Cloud

and achieve correct load balancing amongst VMs. The system model includes cloud datacentres (CDC) which consist of n hosts $H = \{h_1, h_2, h_3 \ldots h_n\}$. Each host hj was regarded as $h_j = \{RC_i, RM_j, RS_j, Re_j\}$ where RC_i was depicted as the CPU competence, RM_j was the memory size, RS_j was represented as the storage capacity and Re_j was stated as the energy consumption of the host h_j. For every host h_j, set of VM was depicted as $VM_a = \{vm_{1k}, vm_{2k}, \ldots. vm_{jk}\}$ where vm_{jk} indicates the j_{th} VM on k_{th} host. The task set $T = \{t_1, t_2, t_3 \ldots t_m\}$. Each task tm was characterized as $t_m = \{a_m, l_m, f_m, p_m\}$ where a_m, l_m, f_m, p_m was represented as arrival time, length of the task, finish time, and precedence task tm, respectively. Tab. 1 depicts the terms and terminologies in the system. Thus, the present work aimed to obtain schedule A (correct mapping of tasks to resources), which lessens the makespan and energy consumption. The mathematical model for the aforementioned multi-objective task scheduling was represented as follows:

$$\text{minimize } y = \min(\text{makespan},\ \text{energy consum})$$

$$\text{Makespan model}: \ \text{minimize makespan} = \min\{\max\{ex_{mjk}\}\}$$

where, ex_{mjk}—execution time of task tm on v_{mjk}.

Table 1: Terms and terminologies

Symbols	Description
$H = \{h_1, h_2, h_3 \ldots h_n\}$	Host set
H_j	j^{th} host
RC_i	CPU performance of j^{th} host
RM_j	Memory size of j^{th} host
RS_j	Storage capacity of j^{th} host
Re_j	Energy consumption in the j^{th} host
VM_a	Set of virtual machine for each host
VM_{jk}	j^{th} virtual machine on k^{th} host
RCV_{jk}	CPU competence of j^{th} virtual machine on k^{th} host
RMV_{jk}	Memory size of j^{th} virtual machine on k^{th} host
RSV_{jk}	Storage capacity of j^{th} virtual machine on k^{th} host
ReV_{jk}	Energy consumption of j^{th} virtual machine on k^{th} host
$T = \{t_1, t_2, t_3 \ldots t_m\}$	Task set
T_m	m^{th} task
EC_{ijk}	Energy consumption by i^{th} task on j^{th} virtual machine on k^{th} host
$H = \{h_1, h_2, h_3 \ldots h_n\}$	Host set
H_j	j^{th} host
RC_i	CPU performance of j^{th} host
RM_j	Memory size of j^{th} host

The execution time of task tm on v_{mjk} was depicted as

$$ex_{mjk} = L_m / RCV_{jk}$$

where, L_m = length of the task tm and RCV_{jk} = CPU competence of j_{th} VM on K_{th} host.

In this paper, scheduling a set of independent tasks in a cloud data center consisting of heterogeneous VMs was deliberated. For each task, suitable VM (possessed with definite measure of resources) should be efficiently assigned. The following three assumptions for practical environments were used:

- Individually a task was executed on only one VM; neither task migration nor any disruption was permitted.
- VMs to be inherent in a single data center, no break-down was concerned.
- Only the energy consumption during task scheduling was considered.
- Energy model

In the CDC, energy consumption by CPU was considered as the major portion of the energy consumed by the host. In addition, the energy consumption in CDC was categorized as static and dynamic energy consumption. The present work was aimed to focus more on dynamic energy consumption than static. Consider the task tm in v_{mjk} consumed energy EC_{mjk} and ECR_{jk} be the energy consumption rate of v_{mjk}. The energy consumption EC_{mjk} was specified as

$$EC_{mjk} = ECR_{jk} \cdot ReV_{jk}$$

Total energy consumption by the task t_m was depicted as,

$$E = EC_{exe} + EC_{idle}$$

where, EC_{exe} = energy incurred while the task is running in the VM and EC_{idle} = idle VM

Thus, the total energy consumption by VM while executing all the tasks was represented as,

$$EC^{exe} = \sum_{(k=1)}^{H} \sum_{(j=1)}^{(VMa)} \sum_{(m=1)}^{T} Z_{mjk} \cdot EC_{mjk}$$

$$EC^{exe} = \sum_{(k=1)}^{H} \sum_{(j=1)}^{(VMa)} \sum_{(m=1)}^{T} Z_{mjk} \cdot ECR_{jk} \cdot ReV_{jk}$$

The problem can be mathematically demonstrated as follows

$$\text{Minimize } EC^{exe} = \sum_{(k=1)}^{H} \sum_{(j=1)}^{(VMa)} \sum_{(m=1)}^{T} Z_{mjk} \cdot EC_{mjk}$$

where, Z_{mjk} = mapping of tasks to VM at distinct hosts in CDC.

If Z_{mjk} is 1, then task m was assigned to v_{mjk} at host k and else is '0'. If the VM was in an idle state, the host was allowed to set the low energy consumption rate for v_{mjk} as ECR_{jk}^{small}. Thus, the total energy consumption for all idle VM's were represented as,

$$EC^{idle} = \sum_{(k=1)}^{H} \sum_{(j=1)}^{(VMa)} ECR_{jk}^{small} \cdot idletime_k$$

where, $idletime_k$ = idle time while all the VM' were idle in host $h_{k.}$

The proposed work was aimed to schedule the tasks to obtain low energy consumption. Therefore, to minimize the energy consumption EC_{mjk}, the task scheduling algorithm was allowed to execute as a smart contract in the blockchain DC at the beginning of each time slot.

4 Scheduling Issues in Cloud

In cloud, the process of scheduling was stated as the mapping of tasks to accessible physical and virtual resources in the cloud. The tasks were created by the applications or conceded to the system by the users [14]. To and from the cloud, both the tasks and resources were added or removed. In general, the cloud was presented as the comprehensive, heterogeneous network. The presence of geographically distributed task contributors and existence of resource holders in distinct independent domains causes scheduling in cloud as the interesting NP-complete optimization problem [15]. Any scheduling algorithm need to:

- Affirm that all the tasks were implemented in line with the time limit and obtain the anticipated outcomes,
- Decrease task execution time so as to acquire less scheduling cost by the users.
- Be responsive and inhibit the transitory software let-downs.

The most significant scheduling aspects were as follows:

4.1 Task Handling Strategy: Batch in Competition with Instantaneous

The process of stating the task handling policy was considered as the essential factor during the detection of the certain scheduling issue. In the course of instantaneous task handling strategy, the submitted tasks were scheduled once when they were reached the system. During the batch mode, the tasks were convened into batches and the scheduler allocated the available resources to each and every batches.

4.2 Inter-relation Amongst Tasks: Dependent versus Independent Tasks

In the cloud, the tasks were allowed to execute independently or the tasks were al-lowed as parallel applications with significant measures and the association between the application modules were presented as a Directed Acyclic Graph (DAG). The notation for the determination of scheduling problem in the large-scale systems were:

$$\mu|\lambda|\delta$$

where, μ depicts the cloud architecture and resource layer signifies the resource layer, λ shows the task handling policy and δ presents the scheduling measure. With the aforementioned notations, the scheduling problem was defined as follows:

$$RM, D|b, indep|y$$

where, D—Distributed and decentralized cloud, RM—unrelated (independent) resources, b—batch handling strategy, inde- independent task handling strategy. With the schedule, the most important scheduling measure was presented as total execution time and makespan. The process of stating the task handling policy was considered as the essential factor during the detection of the certain scheduling issue. In the course of instantaneous task handling strategy, the submitted tasks were scheduled once when they were reached

the system. During the batch mode, the tasks were convened into batches and the scheduler allocated the available resources to each and every batches.

5 Energy-Aware Blockchain based Task Scheduling

Blockchain (BC) is an innovative technology, incurred an immense prospective to be a prominent approach in elucidating the multifaceted issues in high performance computing systems. The BC is the modest, time-series of immutable record of data explicitly controlled by cluster of computers not possessed by any single entity. Individually, the blocks of data (namely block) were protected and assured to each other using distinct crypto-graphic standard (chain). As it was public and immutable record of block, the information was available for any person in the BC network to view. Fig. 2: Blockchain operation. The important aspect of the blockchain was decentralization which indicates that the blockchain network was distributed among the network of nodes. Every node in the network was able to authenticate the actions of the other nodes and has the ability to create, verify and validate the new transactions in the blockchain. Except first or genesis block, each block in BC comprises of hash of the prior block, timestamp and transaction data [16]. The blocks were associated with each other block by referring to the preceding block in the chain. The data stored in the block highly relies on the service and the application being used. The important aspects of blockchain were as follows [17]:

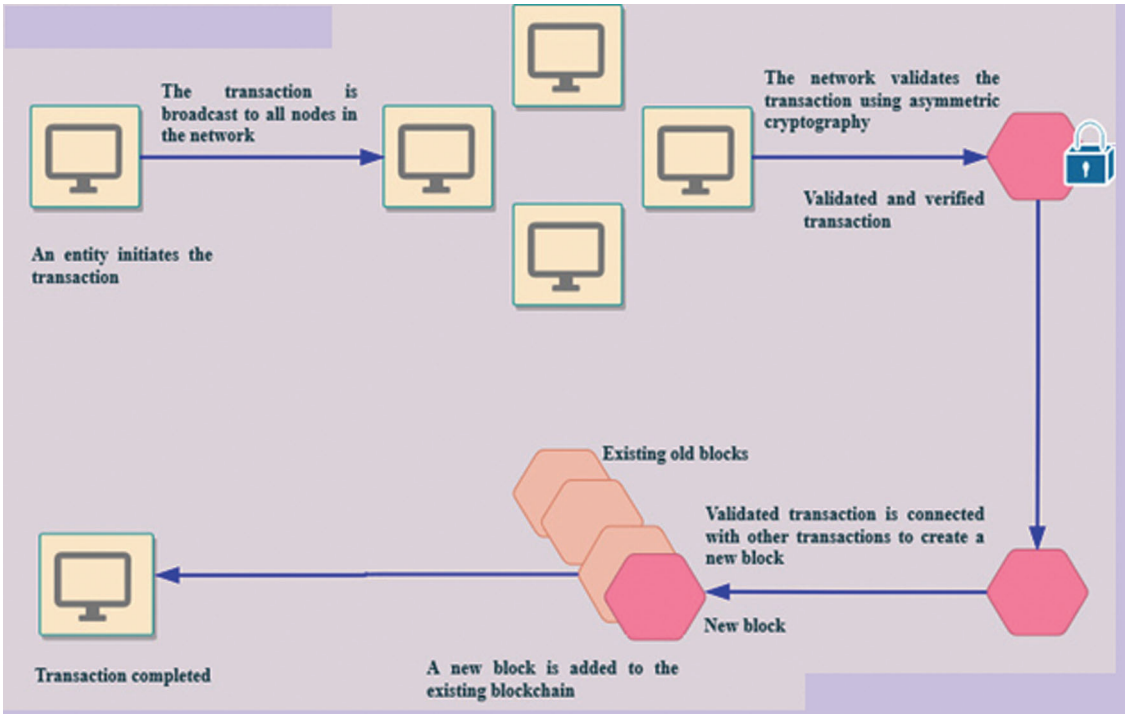

Figure 2: Notion of blockchain operation

a) Persistent: All the transactions were validated. The transactions contradictory with approved policies were held with mining nodes, blocks with inappropriate data were identified.
b) Decentralization: No central controller. Works in accordance with consensus algorithms which affirms every transaction.
c) Transparency: Transactions (with public address) were accessible for each and every user accessing blockchain network.

d) Security: Though the blocks were shared, it might not be deceived owing to tam-per-proof cryptographic hash functions.
e) Auditability: Chance to trace and authenticate each transaction.
f) Immutable: In Blockchain, the data were immutable. In the distributed ledger, every event was affirmed by the BC network. As the blocks comprises of hash value of prior blocks, a small modification in the data might reflect in the hash and thereby directs the other nodes to interrupt the change and discard it.

The important classes of blockchain were as follows [18]:

- Private BC: Here, the possessor who decides the "right to use" the blocks and capability to add and affirm the BC network.
- Public BC: All the users facilitated with the capability to read and add the events to the distributed ledger.
- Consortium BC: It was a network of selected nodes, which can add the data to BC network, data read might be private or public.

5.1 Blockchain Network

- During interaction with Node a, the user employed his private key to sign the trans-action. Therefore, the transactions were sketched by means of user's public key, digital signature reinforces the data integrity and security. The signed transaction was broadcasted to Node a's one-hop adjacent neighbour (Node b and Node c).
- Affirmed to the transaction protocol, the broadcasted transactions were verified by Node b and Node c and again the transaction were advertised to their one-hop adjacent neighbour (Node d and Node e). The aim of the transaction protocol was to inhibit any mal-function in the BC network.

By iterating the above procedure, the transaction was disseminated to the entire block chain network. During all participants approved time interval, the transactions produced by the network were altogether packaged into one block by the Node e (miner). Similarly, the miner broadcast the block to the one-hop adjacent neighbour (Node c). On receiving the block, Node c substantiates that all the transactions in the block abide to the transaction protocol and the block comprises of correct hash of the preceding block in the BC network. If the block satisfied the verification process, then the receiver adds the block to the BC and allowed to obtain the transactions from the block for updating the receiver transaction. Otherwise, the block was rejected. The choice of the miner was controlled by the consensus mechanism in the BC net-work [19]. As the blockchain was maintained by each and every participant in the BC net-work, the network might result in difference amongst the participants. Thus, the consensus mechanism was considered as more vital in BC network. Several consensus mechanisms like Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), Delegated Proof of Stake (DPoS) and the like were used in BC network [20].

5.2 Transaction

Tab. 2 shows the physical resources. In the present work, the transaction indicates the task scheduling by considering the resource allocation of requested VMs. In order to elucidate the task scheduling, the DC adapted transaction protocol was proposed. The protocol includes ID, source and target and resource allocation. Tab. 3 shows the data-center adapted transaction. In cloud environment, user request was depicted as the task. Every task comprises of its own configuration and resources namely id, length and number of processing units required. In the proposed work, M tasks (i.e., $T = \{t_1, t_2, t_3 \ldots t_m\}$) were obtained from users to be scheduled to A VMs ($VM_a = \{vm_{1k}, vm_{2k} \ldots v_{mjk}\}$) where v_{mjk} indicates the j_{th}

VM on k_{th} host. Here, tasks were non-preemptive and VMs were heterogeneous (VMs with distinct computing capabilities) in nature. The idea of the target-time responsive precedence scheduling algorithm was to schedule the tasks to VMs on account of the parameters like precedence and target-time of tasks. From the Tab. 3, aaa submitted the request which incurs a RAM, b CPU, and c Disk. The number of required resources was negative indicates that the request needs resource. Transaction ID 3 depicts that the Node0 migrates a request created by aaa to Node1. In order to update the available resource, transaction named source as nodename and target as apprise was used.

Table 2: Physical resources

Resource	Capability							
	No. of cores	Processor speed (GHz)	Memory capacity (GB)	Architecture (bits)	Capacity (TB)	Speed (GHz)	Type	Bandwidth (Gb/s)
Computational network storage	6–20	2–7	64–256	32/64	0.5–1000	2–7	SSD/ HDD	10–100

Table 3: Datacenter adapted transaction

ID	Source	Target	Capability							
			Computational				Storage			Network
			No. of cores	Processor speed (GHz)	Memory capacity (GB)	Architecture (bits)	Capacity (TB)	Speed (GHz)	Type	Bandwidth (Gb/s)
1	Node0	Apprise	–20		–128		–216			
2	aaa	Node0	–a		–b		–c			
3	Node0	Node1	–a		–b		–c			
4	End	Node1	A		B		c			

5.3 Smart Contract

A smart contract is a computer program written in block chain proposed to digitally aid, authenticate the performance of the reliable transactions. Smart contract was activated by the transactions [21]. In the present work, DCs perform task management by activating the smart contract. Let the simple DCs network includes $\{DC_1, DC_2 \ldots DC_n\}$. In the task management framework, the tasks were executed based on the algorithm depicted as smart contract in each DC. Algorithm 1 depicts the logic of the smart contract. Once the block was recognized by the DC, the smart contract on the DC perceived each and every transaction it encompasses. As soon as the smart contract identify the transaction denoting the task scheduling, it schedules the task to the resources based on the dynamic precedence excluding itself running on the DC. The precedence value varies based on the load. The dynamic precedence causes better application of available resources. The present algorithm preserves the exact sense of balance between power

management and performance. Algorithm 2 (Refer Appendix A and B) shows the target-time responsive precedence scheduling algorithm (TTR-PSA).

5.4 Mining

In the present work, permits each DC to accomplish mining process. This framework not incurred PoS and PoW mechanisms, that might lead to severe energy overhead. The DC mine the subsequent block as follows: Based on load, the list of DCs were arranged in reducing order; Discard the DCs which were utilized to mine the prior x blocks in the list; The primary DC in the list mines the subsequent block.

The remaining DCs were not allowed to mine the blocks to achieve reduced energy consumption. From the aforesaid procedure, x was the factor used to regulate the mining process. The more x denotes the maximum chance of mining heavy-loaded DC. Let every DC has the probability y of error, then the unsuccessful probability of mining was given as,

$$P_{us} = P[\text{Bino}(m, 1 - y) \leq x]$$

where, m represents the total count of DC, Bino () represents the Binomial distribution.

6 Processing and Results

In the present experimentation, the proposed scheduler was compared with traditional scheduling algorithms like Round-robin (RR), First come first serve (FCFS), and Shortest job first (SJF). Also, the proposed algorithms were evaluated based on makespan. The performance of the BC-energy aware scheduling model was analysed by using the AION blockchain in IntelliJ IDE. In the present work, requests trace from google cluster usage traces were used. The traces comprise information like arrival time of requests, CPU utilization, and the like. A data center contains of VM with an image size of 15000 MB, a number of CPU's in the VM includes 5, and the memory of the VM was fixed as 4GB. The computing capacity of VM depicts the amount of the floating-point operations carried out in a second by VM. Also, it was represented as MFLOPS (million floating-point operations per second), and MFLO (million floating-point operation) denotes the number of instructions required to complete the task. The experimental investigation was performed with a distinct number of VMs and tasks.

From Figs. 3(a–d), it was noticed that the proposed BC-energy aware task scheduling exhibited better performance assessment (low makespan) than traditional task scheduling algorithms like FCFS, RR and SJF. Fig. 4 compares the energy consumption of proposed algorithm, FCFS, RR and SJF algorithms. On comparing with other algorithms, the TTR-PSA consumed considerably minimum energy. When the number of tasks increases, the energy consumption of RR and FCFS was increased promptly, while, reduced in our proposed algorithm. The TTR-PSA performance was depicted by means of precedence strategy included the proposed model and optimal solution not only considers the energy consumption but also the resource utilization and execution time which can incur better outcomes to the incoming user tasks. Fig. 4. Resource utilization of TTR-PSA, RR and SJF algorithms with the different number of tasks. In the scheduling, resource utilization was considered the significant metric that was advantageous to the cloud service provider. The main aim was to attain maximum usage of resources. Also, the resources were maintained as active as possible to maximize the gain. Better resource utilization was observed with TTR-PSA than other algorithms. Though the number of tasks increased, the proposed algorithm showed better resource utilization than other algorithms. This observation means that the TTR-PSA uphold the occupancy rate of resource in the course of task scheduling process.

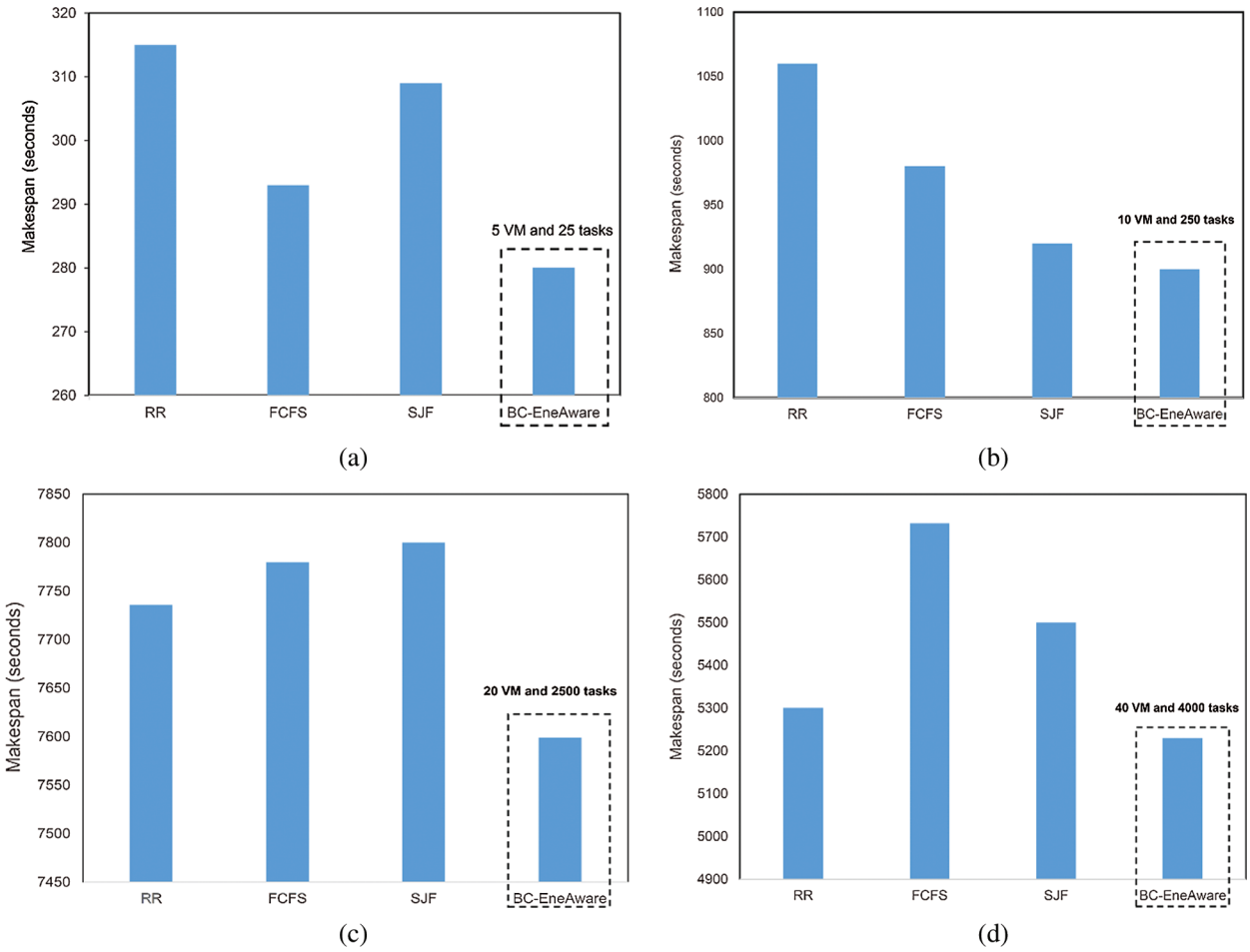

Figure 3: (a) Performance Assessment with 5 VM and 25 tasks, (b) Performance Assessment with 10 VM and 250 tasks, (c) Performance Assessment with 20 VM and 2500 tasks and (d) Performance Assessment with 40 VM and 4000 tasks

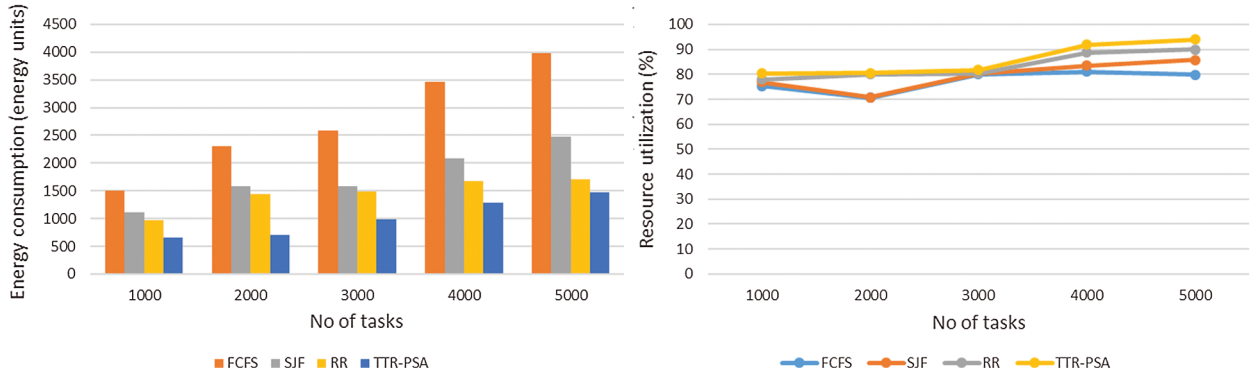

Figure 4: Energy consumption of proposed algorithm and existing algorithms and Resource utilization of TTR-PSA, and existing algorithms with different number of tasks

7 Conclusion and Future Enhancement

In a cloud environment, task scheduling problem was formulated to minimize makespan and energy consumption in the IaaS. The developed model considered storage, computational, and network resources since the units might utilize the cloud environment's power. The present work proposed an algorithm which focuses on the energy-aware task scheduling based on precedence and target-time as parameters. The proposed solution aimed to attain reduced energy consumption and makespan with maximized energy utilization. In the present work, the energy-aware task scheduling was performed in line with Blockchain technology. The intention of choosing blockchain architecture was the enhancement in the energy aspect in the task scheduling process. This novel solution offered a better schedule with minimum energy consumption and makespan than FCFS, SJF, and RR scheduling algorithms. The presented work was the primary paces on the task scheduling's energy aspect in decentralized and distributed computing environments. In the future, dependent tasks (depicted as Directed Acyclic Graph) may be modeled and analyzed. Also, an extension of this work with the thought of task pre-emption in scheduling decisions.

Acknowledgement: The authors wish to thank the Department of Computer Science and Engineering and Artificial Intelligence Research (AIR) Laboratory, PSG College of Technology, to provide the computing facilities for project execution.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Aldossary, "A review of energy-related cost issues and prediction models in cloud computing environments," *Computer Systems Science & Engineering*, vol. 36, pp. 353–368, 2021.

[2] I. Devi and G. R. Karpagam, "Task based resource scheduling in Cloud," *Proc. of IEEE Int. Conf. on Intelligent Computing & Communication for Smart World, I2C2SW*, India, pp. 249–253, 2018.

[3] S. Sotiriadis, N. Bessis and R. Buyya, "Self-managed virtual machine scheduling in Cloud systems," *Information Sciences*, vol. 433-434, pp. 381–400, 2018.

[4] Y. Mhedheb and A. Streit, "Energy-efficient task scheduling in data centers," *Proc. of 6th Int. Conf. on Cloud Computing Services Science*, Rome, vol. 1, pp. 273–282, 2016.

[5] Q. Zhao, C. Xiong, C. Yu, C. Zhang and X. Zhao, "A new energy-aware task scheduling method for data-intensive applications in the Cloud," *Journal of Network & Computer Applications*, vol. 59, pp. 14–27, 2016.

[6] E. Ibrahim, N. A. El-Bahnasawy and F. A. Omara, "Task scheduling algorithm in cloud computing environment based on cloud pricing models," in *Proc. of 2016 World Sym. Computer Applications & Research, WSCAR*, Egypt, pp. 65–71, 2016.

[7] Z. Hu, B. Li and J. Luo, "Time- and cost- efficient task scheduling across geo-distributed data centers," *IEEE Transactions on Parallel & Distributed Systems*, vol. 29, pp. 705–718, 2018.

[8] J. Sahni and P. A. Vidyarthi, "Cost-effective deadline-constrained dynamic scheduling algorithm for scientific workflows in a cloud environment," *IEEE Transactions on Cloud Computing*, vol. 6, pp. 2–18, 2018.

[9] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han *et al.,* "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Transactions on Systems, Man, Cybernetics: Systems*, vol. 49, pp. 2266–2277, 2019.

[10] F. A. Lokhandwala, "A heuristic approach to improve task scheduling in cloud computing using blockchain technology," Doctoral dissertation. National College of Ireland, Dublin, 2018.

[11] S. Ben Alla, H. Ben Alla, A. Touhafi and A. Ezzati, "An efficient energy-aware tasks scheduling with deadline-constrained in cloud computing," *Computers*, vol. 8, pp. 1–15, 2019.

[12] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach *et al.,* "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renewable & Sustainable Energy Reviews*, vol. 100, pp. 143–174, 2019.

[13] A. Wilczyński and J. Kołodziej, "Modelling and simulation of security-aware task scheduling in cloud computing based on Blockchain technology," *Simulation Modelling Practice & Theory*, vol. 99, pp. 1–24, 2020.

[14] A. E. Yahya, R. Samsudin and A. S. Ilman, "A genetic algorithm-based grey model combined with Fourier series for forecasting tourism arrivals in Langkawi Island Malaysia," *Advances in Intelligent Systems & Computing*, vol. 1073, pp. 139–151, 2020.

[15] M. Niranjanamurthy, B. N. Nithya and S. Jagannatha, "Analysis of blockchain technology: Pros, cons and SWOT," *Cluster Computing*, vol. 22, pp. 14743–14757, 2019.

[16] K. Svozil, "Quantum advantage by relational queries about equivalence classes," *Communications in Computer & Information Science*, vol. 891, pp. 504–512, 2019.

[17] H. F. Atlam and G. B. Wills, "Technical aspects of blockchain and IoT," *Advances in Computers*, vol. 115, pp. 1–39, 2019.

[18] J. Göbel, H. P. Keeler, A. E. Krzesinski and P. G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Performance Evaluation*, vol. 104, pp. 23–41, 2016.

[19] M. Du, X. Ma, Z. Zhang, X. Wang and Q. Chen, "A review on consensus algorithm of blockchain," in *IEEE Int. Conf. on Systems, Man, & Cybernetics (SMC)*, Canada, pp. 2567–2572, 2017.

[20] W. Zou, D. Lo, P. S. Kochhar, X. D. Le, X. Xia *et al.,* "Smart contract development: Challenges and opportunities," *IEEE Transactions on Software Engineering*, vol. 5589, pp. 1, 2019.

[21] C. Xu, K. Wang and M. Guo, "Intelligent resource management in blockchain-based cloud datacenters," *IEEE Cloud Computing*, vol. 4, pp. 50–59, 2017.

Appendix A

Scheduling of tasks in DC.

if transaction.target=this then

if transaction.req_resource is less than available or 0 then

If $\sum_{(j=1)}^{k}$availablej < $\sum_{(i=1)}^{m}\sum_{(j=1)}^{n}$ $Required_{I,j}$ then

Identify the DC based on target-time responsive precedence scheduling algorithm

if $\sum_{(j=1)}^{k}available_j = \sum_{(i=1)}^{m}\sum_{(j=1)}^{n}Required_{i,j}$ OR $\sum_{(j=1)}^{k}$ $available_j > \sum_{(i=1)}^{m}\sum_{(j=1)}^{n}$ $Required_{I,j}$ then

Go for standard algorithm and schedule the tasks to DCa.

end if

end if

end if

end if

Appendix B

Algorithm 2: Target time responsive Precedence Scheduling Algorithm (TTR-PSA)

Algorithm: Pseudocode for Target time responsive Precedence scheduling algorithm

Require: User submission of Jobs

Ensure: Scheduling of jobs based on resource requirement

1. Calculating the vector of criteria weights.

1.1 Creation of pairwise comparison matrix M, n × n real matrix, where n is the number of evaluation criteria considered

1.2 m_{jk} of the matrix M depicts the importance of the j_{th} criterion relative to the kth criterion

1.3 The entries m_{jk} and m_{kj} satisfy the following constraint: $m_{jk}. m_{kj} = 1$

criteria weight vector $w = \sum_{(l=1)}^{n} a_{jl}/n$

2. Calculating the matrix of option scores.

2.1 The matrix of option scores is a m × n real matrix B. Every record b_{ij} of B denotes the score of the i_{th} option in regard to the j_{th} criterion.

2.2 the score matrix B is obtained as B = [b (1) … b (n)] i.e. the j_{th} column of S corresponds to b(j)

3. Positioning the options.

3.1 When the weight vector w and the score matrix B have been computed, the vector P of global scores was obtained by multiplying S and w,

3.2 P = S · w (P precedence vector).