A Cost-Variant Renewable Energy-Based Scheduling Algorithm for Cloud Computing

Manas Kumar Mishra and Sanjaya Kumar Panda, ***IEEE Senior Member***
Department of Computer Science and Engineering, National Institute of Technology Warangal
Warangal - 506004, Telangana, India
mkcs20102@student.nitw.ac.in,sanjaya@nitw.ac.in

ABSTRACT

The global growth of cloud computing services is rising abruptly due to a large variety of services like computing, storage, network, etc. It expresses cloud service providers (CSPs) for better usage of existing datacenter resources, increasing agility, and reducing the need for unanticipated datacenter growth. These datacenters use a lot of energy generated from fossil fuels (i.e., non-renewable energy (NRE) sources) and omit a lot of nitrous oxide and carbon dioxide, which cause the greenhouse effect and are harmful to the environment. Moreover, NRE sources are limited in supply and cannot be sustained over a long period. As a circumstance, CSPs are moving towards renewable energy (RE) sources, such as solar, wind, hydro, and biomass, to decarbonize datacenters even though these resources are not available round the clock. Therefore, recent studies focus on using both RE and NRE sources to avoid any interruption of the datacenter services. However, these studies consider the equal cost for all the RE sources and do not consider the categorization among user requests (URs). This paper considers the different costs for RE sources and two categories of URs, namely critical and non-critical, and introduces a cost-variant RE-based scheduling (CRES) algorithm for cloud computing. Here, the critical UR does not depend on the RE resources due to the unpredictability of RE sources. On the other hand, the non-critical UR can be accommodated by both RE and NRE resources. We simulate the proposed algorithm by considering 20 to 100 URs and 5 to 25 datacenters and compare the performance with the future-aware best fit (FABEF) and highest available renewable first (HAREF) algorithms in terms of cost and usage count of RE resources to show its usefulness.

CCS CONCEPTS

• **Computing methodologies → Self-organization**.

KEYWORDS

Cloud computing, datacenter, cloud service provider, renewable energy, non-renewable energy, scheduling algorithm

ACM Reference Format:
Manas Kumar Mishra and Sanjaya Kumar Panda, ***IEEE Senior Member***. 2022. A Cost-Variant Renewable Energy-Based Scheduling Algorithm for Cloud Computing. In *2022 Fourteenth International Conference on Contemporary Computing (IC3) (IC3-2022), August 4–6, 2022, Noida, India.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3549206.3549225

IC3-2022, August 4–6, 2022, Noida, India

ACM ISBN 978-1-4503-9675-2/22/08…$15.00
https://doi.org/10.1145/3549206.3549225

1 INTRODUCTION

Cloud computing provides a large variety of services like computing, storage, network, etc., to the users without making any investment in their infrastructure [1, 13, 18]. It also provides dynamic provisioning resources on a pay-per-usage basis. Therefore, many small to medium enterprises move to the cloud to expand their businesses in order to compete in the global market. As a circumstance, the global growth of cloud computing is increasing day by day. This creates many challenges for the CSPs to maintain their infrastructure efficiently [17]. One of the challenges is to handle the increasing number of URs without increasing the data center resources. For this, an efficient scheduling algorithm needs to be developed to dispatch the URs to the resources in such a manner that the resources are appropriately utilized. At the same time, the algorithm needs to be energy-efficient to save the cost incurred by the CSPs to demand energy. In practice, the resources of the datacenter are powered by NRE energy sources, e.g., fossil fuels [10, 20]. These sources generate a lot of nitrous oxide and carbon dioxide, which pollute the environment [14, 23]. Moreover, these sources are finite and vanish over a period of time. The solution is alarming to use RE sources (i.e., solar, wind, hydro, and biomass) in addition to the NRE sources [7, 8, 11, 21, 22]. Note that RE sources are extracted from nature, i.e., solar energy from the sun, wind energy from the breeze, hydropower energy from the water, biomass energy from the wood waste, solid waste, and landfill gas. The cost of using these natural sources is free and unlimited. But, RE sources are region-specific [20]. For example, the production of solar energy in Canada is very low [4]. On the contrary, the cost of NRE sources is increasing day by day due to their scarcity.

Recent studies [2, 5, 6, 9, 10, 12, 15, 16, 19–21] focus on using both RE and NRE sources to handle all the possible situations to provide round the clock service. The limitations of these studies are as follows.

(1) The cost of RE sources is considered as constant without looking into their types.
(2) The URs are treated the same without any categorization.

In this paper, we propose a CRES algorithm, which incorporates the different costs for RE sources and categorizes the URs to provide them resources based on their importance. More specifically, we consider two categories of URs, namely critical and non-critical. In the case of critical, the UR does not depend on the RE resources. The rationality behind this is that the RE resources are unpredictable by nature. For instance, solar energy cannot be available at night and on cloudy days. In the case of non-critical, the UR can be provided with RE and/or NRE resources based on their availability. It is noteworthy to mention that the resources are supplied with RE sources followed by NRE sources in case of scarcity. The proposed algorithm is simulated in a virtual environment and tested with five datasets ranging from 20 to 100 URs and 5 to 25 datacenters. Subsequently, the existing algorithms, namely FABEF [20], and HAREF [20], are simulated in the same environment and datasets. The outcomes of these simulation runs are compared using the cost and usage count of RE resources. The comparison shows the usefulness of the proposed algorithm.

The rest of the paper is organized into the following sections. Section 2 discusses related works with their pros and cons. Section 3 presents the models and the problem studied in this paper. Section 4 introduces the proposed algorithm and shows a detailed illustration. Section 5 lists the datasets, performance metrics, and simulation results. Section 6 concludes this paper with its possible extensions.

2 RELATED WORK

Many researchers [2, 5, 6, 9, 10, 12, 15, 16, 19–21] have developed RE-based scheduling algorithms to utilize the resources of the datacenters efficiently. They have primarily focused on the RE sources and extended to NRE sources as and when required. Some of these works are listed here with their pros and cons.

Toosi and Buyya [20] have developed a fuzzy logic-based load balancing (FLB) algorithm to maximize the consumption of RE. Alternatively, this algorithm minimizes the consumption of NRE. As a result, the total cost incurred for serving the user requests is minimized. They have considered three parameters, namely utilization of RE sources, consumption of NRE, and price of electricity, to determine the suitability, which is taken as very low, low, mid, high, and very high, respectively. They have assumed that the cost of RE resources is negligible. They have taken FABEF, round-robin (RR), and HAREF algorithms to compare the performance of FLB. FABEF algorithm maps the URs to the datacentres and selects the least-cost datacenter without considering the availability of RE resources. RR algorithm assigns the URs to the datacenters evenly by following circular order. This algorithm does not consider the availability of RE and NRE resources and their costs. HAREF algorithm selects a datacenter based on the availability of RE resources. The algorithm does not consider the cost of resources incurred in the datacenters. Nayak et al. [12] have considered both RE resources availability and cost to develop a multi-objective RE-based algorithm for overcoming the drawbacks of FABEF and HAREF. However, the above works have considered the equal cost for all the RE sources.

Courchelle et al. [2] have designed a scheduler to take an offline decision. The scheduler dispatches the virtual machines (VMs) to the datacenters, resulting in the least amount of brown energy. However, their study is focused on photovoltaic (PV) energy only. Alternatively, other RE sources, namely wind, biomass, and hydro, are not taken into consideration. Nayak et al. [9] have presented an algorithm for green cloud computing called unconstrained power management. To assign the URs to the datacenters, they have examined three power sources, namely grid, PV, and battery. However, the minimum power generation of each source is not defined, resulting in datacenter power outages. Khan and Zakarya [5] have conducted a study on three main components of datacenters, namely performance, energy, and cost. They have stated that datacenters in the United States waste almost 30% of their capacity. Therefore, they have suggested controlling the same using techniques, such as virtualization and containerization, to reduce energy waste. Here, the energy efficiency methods are divided into hardware, resource management, and applications. Nayak et al. [10] have developed a RE-based scheduling algorithm by taking a linear combination of the earliest completion time and energy cost to schedule URs to the datacenters. However, they have not categorized the RE sources.

Le et al. [6] have presented a dynamic load balancing scheme to reduce expenditures related to electricity and deliver the cooling effects. This scheme handles both load placement and migration. Panda and Jana [15] have presented an approach for reducing the energy use of the datacenters. They have mapped user specifications to VMs and further mapped VMs to physical machines in the datacenters. However, the resources are not powered by RE resources. Panda and Jana [16] have addressed the issues of task consolidation and scheduling, and proposed an algorithm that balances energy usage and overall completion time of tasks. However, they have not considered the energy cost. Rajeev and Ashok [19] have developed a dynamic load shifting program for cloud computing and introduced a dynamic renewable factor. They have considered a case study with 7.5 million+ consumers on PV to show the economic and technical benefits. But, the atmospheric conditions are not taken into consideration. Xu and Buyya [21] have considered the problem of carbon emissions in datacenters and proposed an approach to transfer workloads between multi-clouds by considering various geographic locations. The goal of shifting between sites is to use RE resources in that region properly. They have suggested investigating wind turbines as a potential future work.

The proposed algorithm has the following improvements over the existing algorithms.

(1) The existing algorithms [9, 10, 12, 20] consider RE sources without varying their types in terms of cost. On the contrary, the proposed algorithm considers a variety of costs for different types of RE sources.
(2) The existing algorithms [9, 10, 12, 20] consider the URs without specifying any category. On the other hand, the proposed algorithm categorizes the URs into critical and non-critical and assigns the resources accordingly.

3 MODELS AND PROBLEM STATEMENT

3.1 System Model

Consider a set of datcenters located in different regions around the globe, and a CSP manages them. Each datacenter consists of numerous servers (also referred to as resources) and is powered by both RE and NRE sources. Initially, the resources are powered by RE sources. In case of unavailability of such sources, NRE sources are used to power the resources of datacenters. On the other hand, the user requirements are represented in the form of URs, and the CSP serves them by providing the resources. In order to handle the users and their requests, CSP manages a global queue in which URs are placed as per their arrival. These URs are categorized into critical and non-critical URs. Here, critical URs are assigned to only NRE resources, and non-critical URs can be assigned to any resources as per availability. The resources are provided to the users on a first-come, first-serve basis as per the requirements. The allotted resources may be switched (i.e., RE to NRE or NRE to RE) over time, based on the availability of sources, and the cost of this switching is assumed to be negligible.

3.2 Cost Model

The cost model is based on the type of energy required to power the resources of the datacenters. The cost of RE resources is cheaper in comparison to the cost of NRE resources. However, the cost of RE resources is further decomposed based on the availability of RE sources in a particular region. For instance, the sunniest region offers less cost to the RE resources powered by solar energy. Alternatively, the darkest region cannot offer solar energy. As a case study, Australia produces more wind energy than solar, hydro, and biomass energy. Therefore, the cost of wind energy is cheaper than solar, the cost of solar energy is comparatively cheaper than hydro energy, and so on.

3.3 Problem Statement

Consider a set of n URs, $U = \{U_1, U_2, U_3, \ldots, U_n\}$ present in a global queue Q. Each UR U_i, $1 \le i \le N$, is represented in the form of 4-tuple. The tuple can be mathematically represented as U_i = <S_i, N_j, DU_j, T_j>, where S_i is the start time of UR U_i, N_i is the number of nodes/resources required to execute UR U_i, DU_i is the duration to execute UR U_i, T_i is the application type of the UR, i.e., critical or non-critical. Note that global queue Q keeps the URs in the non-decreasing order of the S of UR. Consider a set of m datacenters, $D = \{D_1, D_2, D_3, \ldots, D_m\}$, where each data center D_j, $1 \le j \le m$, comprises of a set of K resources, $R_j = \{R_{j1}, R_{j2}, R_{j3}, \ldots, R_{jK}\}$. Each resource consists of a set of slots. Each resource slot is powered by the RE or NRE sources with respect to time. The cost of using RE and NRE resources is predetermined. The problem is to map the URs to the resources of the datacentres, such that the overall cost is minimized.

4 PROPOSED ALGORITHM

This section presents the proposed algorithm, CRES, for cloud computing. CRES aims to minimize the overall cost of assigning the URs to the datacenters. It considers the different costs for RE sources and two categories of URs, namely critical and non-critical. It works in two phases. In the first phase, it maps a UR to all the datacenter and determines the cost of executing that UR. The later phase assigns that UR to a datacenter that results in the least cost. These phases are explained using the pseudocode mentioned in Algorithm 1 in the following subsections.

4.1 First Phase

In this phase, the CRES algorithm picks a UR from the Q (i.e., UN) and determines its application type (Line 1 and Line 2 of Algorithm 1). If the application type is non-critical (Line 2), then it finds the resources that are powered by either RE sources (Line 8 to Line 14) or NRE sources (Line 15). Then it finds the cost to schedule the UR on each datacenter (Line 3 to Line 15) and selects the datacenter that holds the minimum cost (Line 33). On the other hand, if the application type is critical (Line 20), then it finds the resources that are powered by the NRE sources (Line 26 and Line 27). The rationality behind this is that NRE sources are reliable in executing the critical UR without interruption. Note that each UR starts at S and continues up to $S + DU - 1$ with N resources (Line 5 and Line 23).

Algorithm 1 Pseudocode for CRES

Inputs: 2D matrix: *slot*, 1D matrix: *cost*, *nre*, DU, S, N and T, UN, D

Output: Minimal cost datacentre

1: **for** $i \leftarrow 1$ to UN **do** ▷ UN = Number of URs
2: **if** $T[i]$ == non-critical **then**
3: **for** $j \leftarrow 1$ to D **do**
4: Set $cost[j] = 0$ and $nre[j] = 0$ ▷ nre = Number of RE slots
5: **for** $l \leftarrow S[i]$ to $S[i] + DU[i] - 1$ **do**
6: Set $k = 0$
7: **while** $k \neq N[i]$ **do**
8: **if** $slot[l, k]$ is powered by cheapest RE sources **then**
9: *cost* is updated by adding the cheapest RE slot cost, $nre[j]$++ and k++
10: **else if** $slot[l, k]$ is powered by next to cheapest RE sources **then**
11: *cost* is updated by adding the next to cheapest RE slot cost, $nre[j]$++ and k++
12: ...
13: **else if** $slot[l, k]$ is powered by expensive RE sources **then**
14: *cost* is updated by adding the expensive RE slot cost, $nre[j]$++ and k++
15: **else** *cost* is updated by adding the NRE slot cost and k++
16: **end if**
17: **end while**
18: **end for**
19: **end for**
20: **else**
21: **for** $j \leftarrow 1$ to D **do**
22: Set $cost[j] = 0$
23: **for** $l \leftarrow S[i]$ to $S[i] + DU[i] - 1$ **do**
24: Set $k = 0$
25: **while** $k \neq N[i]$ **do**
26: **if** $slot[l, k]$ is powered by NRE sources **then**
27: *cost* is updated by adding the NRE slot cost and k++
28: **end if**
29: **end while**
30: **end for**
31: **end for**
32: **end if**
33: Find the datacenter j with minimum cost
34: Assign the UR i to the datacenter j
35: Update the overall cost and the total *nre*
36: **end for**

4.2 Second Phase

In this phase, the CRES algorithm dispatches the UR to the datacenter, which results in the minimum cost (Line 33 and Line 34). Subsequently, it updates the overall cost incurred by that task for using RE and/or NRE resources and updates the total number of RE slots used by that task (Line 35). Note that the number of RE slots is irrespective of the type of RE slots.

4.3 Illustration

Let us illustrate the proposed algorithm, CRES, using eight URs (i.e., $n = 8$), U_1 to U_8 and two datacentres (i.e., $m = 2$) that are residing in Australia (D_1) and Canada (D_2). The cost of using the resources of datacentres depends on the regions in which these datacenters are deployed. For example, the region in which the rich generation of solar energy takes less cost for using the solar energy. However, if there is less production, then the cost is expensive. The URs properties are shown in Table 1 and the initial configuration of datacenters is shown in Table 2 and Table 3.

Table 1: A set of URs with their S, N, DU and T

UR	S	N	DU	T
U_1	1	1	4	Non-critical
U_2	1	1	1	Critical
U_3	1	1	4	Non-critical
U_4	3	2	5	Non-critical
U_5	4	1	3	Critical
U_6	6	1	3	Critical
U_7	7	1	2	Non-critical
U_8	7	2	3	Non-critical

Table 2: Initial configuration of datacenter D_1

7	8	7	9	8	7	9	10	8

Table 3: Initial configuration of datacenter D_2

8	10	12	9	8	9	10	7	7

In Table 2 and Table 3, the datacenters are shown with four resources for simplicity of this illustration. Each resource of the datacenter is either powered by one of the RE sources or NRE sources. The NRE sources are represented in white color in both the tables. If the RE source is solar energy, then it is represented in dark yellow color. Similarly, the wind energy is represented in dark green color, the hydropower energy is represented in blue color, and the biomass is represented in red color. In these tables, the numerical values on the top of each datacenter represent the cost of using NRE energy in that timeslot. In practice, Australia generates the RE energy in the order of wind (maximum), solar, hydropower, and biomass (minimum) [3]. Therefore, we assign the cost value as 1, 2, 3, and 4, respectively. However, Canada generates the RE energy in the order of hydropower, wind, and biomass [4]. Therefore, we assign the cost value as 1, 2, and 3, respectively. It is noteworthy to mention that solar energy production in Canada is very minimal. As a result, we have not considered solar energy.

The CRES algorithm selects UR U_1 from the Q, and the tuple of this UR is <1, 1, 4, Non-critical>. Then UR U_1 is mapped to datacenter D_1 and D_2, respectively, as shown in Table 4 and Table 5, respectively. Here, the application type of UR U_1 is non-critical. Therefore, it is mapped to both RE and NRE resources. The cost of executing the UR U_1 in the datacenter D_1 is 7 (i.e., 1 + 1 + 2 + 3) and in datacenter D_2 is 14 (i.e., 3 + 1 + 1 + 9). As datacenter D_1 takes less cost, the UR U_1 is assigned to datacenter D_1 as shown in Table 4. The overall cost of datacenter D_1 is updated to 7 and the total number of RE slots is updated to 4.

Table 4: Mapping of UR U_1 to datacenter D_1

7	8	7	9	8	7	9	10	8
U_1	U_1	U_1	U_1					

Table 5: Mapping of UR U_1 to datacenter D_2

8	10	12	9	8	9	10	7	7
U_1	U_1	U_1	U_1					

Now, the CRES algorithm selects UR U_2 from the Q, and the tuple of this UR is <1, 1, 1, Critical>. Then UR U_2 is mapped to datacenter D_1 and D_2, respectively. Here, the application type of UR U_2 is critical. Therefore, it is mapped to NRE resources only. The cost of executing the UR U_2 in the datacenter D_1 is 7, and in datacenter D_2 is 8. As datacenter D_1 takes less cost, the UR U_2 is assigned to datacenter D_1 as shown in Table 6. The overall cost of datacenter D_1 is updated to 14, and the total number of RE slots is updated to 4.

Similarly, other URs are assigned to either datacenter D_1 or D_2. The final Gannt chart of datacenter D_1 and datacenter D_2 is shown in Table 7 and Table 8, respectively. The overall cost and the total number of RE slots for assigning all the URs are 138 (i.e., 66 + 72) and 21 (i.e., 6 + 15), respectively.

For the sake of comparison, we present the Gantt chart of FABEF and HAREF as shown in Table 9 to Table 12, respectively. Note that the cost of RE resources is considered as 2.5, and the gray (white) color represents the RE (NRE) resources. The overall cost and the total number of RE slots for assigning all the URs in FABEF are 156.5 and 21, respectively. Similarly, the overall cost and the total number of RE slots for assigning all the URs in HAREF are 156.5 and 21, respectively.

Table 6: Assignment of UR U_2 to datacenter D_1

7	8	7	9	8	7	9	10	8
U_2								
U_1	U_1	U_1	U_1					

Table 7: Final Gantt chart of datacenter D_1

7	8	7	9	8	7	9	10	8
U_2				U_5	U_6			
					U_5	U_6	U_6	
			U_5					
U_1	U_1	U_1	U_1			U_7	U_7	

Table 8: Final Gantt chart of datacenter D_2

8	10	12	9	8	9	10	7	7
						U_8		
		U_4	U_4			U_8		
		U_4	U_4	U_4	U_4	U_4	U_8	U_8
U_3	U_3	U_3	U_3	U_4	U_4	U_4	U_8	U_8

Table 9: Gantt Chart for FABEF (Datacentre D_1)

7	8	7	9	8	7	9	10	8
U_2				U_5	U_6			
			U_5		U_5	U_6	U_6	
U_3	U_3	U_3	U_3					
U_1	U_1	U_1	U_1			U_7	U_7	

5 DATASETS, PERFORMANCE METRICS AND SIMULATION RESULTS

5.1 Datasets

We consider five synthetic datasets of various sizes to compare the performance of the proposed algorithm CRES and the existing algorithms. These datasets are generated using the in-built function (i.e., *randi*) of MATLAB. This function takes the range and the properties of URs as input and generates a 2D matrix. More specifically, we consider the range of S as [1, 40], N as [5, 15], DU as [1, 10] and T as 0 for critical and 1 for non-critical. The range of NRE resource costs is [7, 20] and RE resource costs is [1, 4], respectively. The range of URs is [20, 100] and datacenters is [5, 25], respectively.

Table 10: Gantt Chart for FABEF (Datacentre D_2)

8	10	12	9	8	9	10	7	7
						U_8		
						U_8		
		U_4	U_4	U_4	U_4	U_4	U_8	U_8
		U_4	U_4	U_4	U_4	U_4	U_8	U_8

Table 11: Gantt Chart for HAREF (Datacentre D_1)

7	8	7	9	8	7	9	10	8
U_2				U_5	U_6			
			U_5		U_5	U_6	U_6	
U_3	U_3	U_3	U_3					
U_1	U_1	U_1	U_1			U_7	U_7	

Table 12: Gantt Chart for HAREF (Datacentre D_2)

8	10	12	9	8	9	10	7	7
						U_8		
						U_8		
		U_4	U_4	U_4	U_4	U_4	U_8	U_8
		U_4	U_4	U_4	U_4	U_4	U_8	U_8

5.2 Performance Metrics

We consider two performance metrics, namely the overall cost (*cost*) and the total number of RE slots. The overall cost is the sum of the cost of the datacenters for executing all the URs. Mathematically,

$$cost = \sum_{j=1}^{m} cost(D_j) \tag{1}$$

where $cost(D_j)$ is the cost of the datacenter D_j for executing the URs assigned to it and m is the number of datacenters. The total number of RE slots (nre) is the sum of the RE slots used by the datacenters for executing all the URs. It can be mathematically represented as follows.

$$nre = \sum_{j=1}^{m} nre(D_j) \tag{2}$$

where $nre(D_j)$ is the number of RE slots of the datacenter D_j used for executing the URs assigned to it.

5.3 Simulation Results

The proposed algorithm is simulated using MATLAB and compared with existing algorithms, FABEF and HAREF, using five datasets and two performance metrics. The simulation results are shown in Table 13 and Table 14, respectively and shown pictorially in Figure 1 and Figure 2, respectively. The results show the superior performance of the proposed algorithm. The rationality behind this is that RE slots' cost is considered different based on their availability in a particular region. However, it is fixed in the existing algorithms. Moreover, the proposed algorithm considers two different application types for the URs compared to the existing algorithms.

Table 13: Comparison of the proposed algorithm CRES, FABEF, and HAREF in terms of the overall cost

Dataset	URs	Datacenters	CRES	HAREF	FABEF
1	20	05	5629	5784	5701
2	40	10	5989	6123	6068
3	60	15	6366	6690	6511
4	80	20	6890	7301	7156
5	100	25	7111	7654	7453

Table 14: Comparison of the proposed algorithm CRES, FABEF, and HAREF in terms of the total number of RE slots

Dataset	URs	Datacenters	CRES/HAREF	FABEF
1	20	05	777	486
2	40	10	982	654
3	60	15	1012	853
4	80	20	1352	984
5	100	25	1401	1209

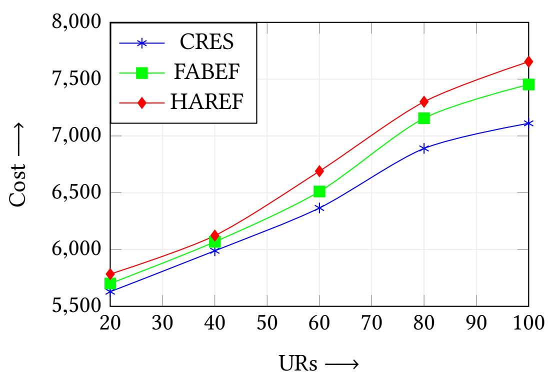

Figure 1: Pictorial comparison of the proposed algorithm CRES, FABEF, and HAREF in terms of the overall cost.

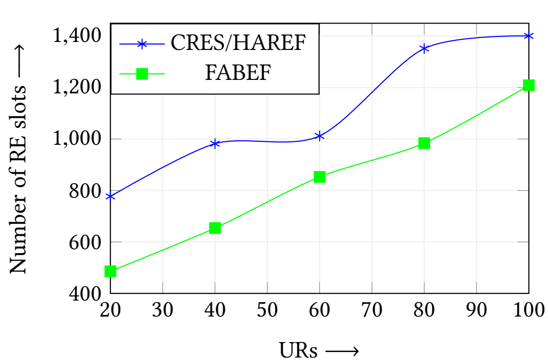

Figure 2: Pictorial comparison of the proposed algorithm CRES, FABEF, and HAREF in terms of the total number of RE slots.

6 CONCLUSION

In this paper, we have proposed a scheduling algorithm for cloud computing by considering both RE and NRE sources. The algorithm works in two phases to schedule the URs to the datacenters. It considers different costs for RE sources and categorizes the URs into critical and non-critical application types. This categorization is performed to provide resources to URs based on their importance. We have simulated the proposed algorithm using five different datasets by considering 20 to 100 URs and 5 to 25 datacenters. The simulation results of CRES are compared with the existing algorithms FABEF and HAREF using two performance metrics, namely the overall cost and the total number of RE slots, to show its superiority. In future work, the proposed algorithm will be extended to heterogeneous datacenters in which the UR will take different N and DU concerning datacenters.

REFERENCES

[1] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. 2009. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems* 25, 6 (2009), 599–616.
[2] Inès De Courchelle, Tom Guérout, Georges Da Costa, Thierry Monteil, and Yann Labit. 2019. Green energy efficient scheduling management. *Simulation Modelling Practice and Theory* 93 (2019), 208–232.
[3] IEA. 2022. Clean Energy Autralia Report 2021. https://assets.cleanenergycouncil.org.au/documents/resources/reports/clean-energy-australia/clean-energy-australia-report-2021.pdf. Accessed on 20th April 2022.
[4] IEA. 2022. Key Energy Statistics. https://www.iea.org/countries/canada. Accessed on 20th April 2022.
[5] Ayaz Ali Khan and Muhammad Zakarya. 2021. Energy, performance and cost efficient cloud datacentres: A survey. *Computer Science Review* 40 (2021), 100390.
[6] Kien Le, Ricardo Bianchini, Jingru Zhang, Yogesh Jaluria, Jiandong Meng, and Thu D Nguyen. 2011. Reducing electricity cost through virtual machine placement in high performance computing clouds. In *Proceedings of 2011 international conference for high performance computing, networking, storage and analysis*. 1–12.
[7] Sanjib Kumar Nayak, Sanjaya Kumar Panda, and Satyabrata Das. 2021. Renewable energy-based resource management in cloud computing: a review. *Advances in Distributed Computing and Machine Learning* (2021), 45–56.
[8] Sanjib Kumar Nayak, Sanjaya Kumar Panda, and Satyabrata Das. 2022. Constrained-based power management algorithm for green cloud computing. *International Journal of Computational Science and Engineering* (2022).
[9] Sanjib Kumar Nayak, Sanjaya Kumar Panda, and Satyabrata Das. 2022. Unconstrained Power Management Algorithm for Green Cloud Computing. In *Advances in Distributed Computing and Machine Learning*. Springer, 3–14.
[10] Sanjib Kumar Nayak, Sanjaya Kumar Panda, Satyabrata Das, and Sohan Kumar Pande. 2021. An efficient renewable energy-based scheduling algorithm for cloud computing. In *International Conference on Distributed Computing and Internet Technology*. Springer, 81–97.
[11] Sanjib Kumar Nayak, Sanjaya Kumar Panda, Satyabrata Das, and Sohan Kumar Pande. 2021. A renewable energy-based task consolidation algorithm for cloud computing. In *Control Applications in Modern Power System*. Springer, 453–463.
[12] Sanjib Kumar Nayak, Sanjaya Kumar Panda, Satyabrata Das, and Sohan Kumar Pande. 2022. A multi-objective renewable energy-based algorithm for geographically distributed datacentres. *International Journal of Embedded Systems* 15, 2 (2022), 119–131.
[13] Benazir Neha, Sanjaya Kumar Panda, Pradip Kumar Sahu, Kshira Sagar Sahoo, and Amir H Gandomi. 2022. A Systematic Review on Osmotic Computing. *ACM Transactions on Internet of Things* 3, 2 (2022), 1–30.
[14] Sanjaya K Panda and Prasanta K Jana. 2014. An efficient energy saving task consolidation algorithm for cloud computing systems. In *2014 International Conference on Parallel, Distributed and Grid Computing*. IEEE, 262–267.
[15] Sanjaya K Panda and Prasanta K Jana. 2017. An efficient request-based virtual machine placement algorithm for cloud computing. In *International conference on distributed computing and internet technology*. Springer, 129–143.
[16] Sanjaya K Panda and Prasanta K Jana. 2019. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. *Cluster Computing* 22, 2 (2019), 509–527.
[17] Sanjaya Kumar Panda, Shradha Surachita Nanda, and Sourav Kumar Bhoi. 2022. A pair-based task scheduling algorithm for cloud computing environment. *Journal of King Saud University-Computer and Information Sciences* 34, 1 (2022), 1434–1445.

[18] Sohan Kumar Pande, Sanjaya Kumar Panda, Satyabrata Das, Mamoun Alazab, Kshira Sagar Sahoo, Ashish Kumar Luhach, and Anand Nayyar. 2020. A smart cloud service management algorithm for vehicular clouds. *IEEE Transactions on Intelligent Transportation Systems* 22, 8 (2020), 5329–5340.

[19] T Rajeev and S Ashok. 2015. Dynamic load-shifting program based on a cloud computing framework to support the integration of renewable energy sources. *Applied Energy* 146 (2015), 141–149.

[20] Adel Nadjaran Toosi and Rajkumar Buyya. 2015. A fuzzy logic-based controller for cost and energy efficient load balancing in geo-distributed data centers. In *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 186–194.

[21] Minxian Xu and Rajkumar Buyya. 2020. Managing renewable energy and carbon footprint in multi-cloud computing environments. *J. Parallel and Distrib. Comput.* 135 (2020), 191–202.

[22] Daming Zhao and Jiantao Zhou. 2022. An Energy and Carbon-aware Algorithm for Renewable Energy Usage Maximization in Distributed Cloud Data Centers. *J. Parallel and Distrib. Comput.* (2022).

[23] Rui Zhou, Yunshan Cao, Rui Zhao, Qingwei Zhou, Jun Shen, Qingguo Zhou, and Haifeng Zhang. 2019. A novel cloud based auxiliary medical system for hypertension management. *Applied Computing and Informatics* 15, 2 (2019), 114–119.