Intelligent Red Deer Algorithm based Energy Aware Load Balancing Scheme for Data Fusion in Cloud Environment

Abedallah Zaid Abualkishik *[1] , Rasha Almajed [2] , William Thompson[3]

[1,2] American University in the Emirates, Dubai, UAE

[3] Towson University, Towson University, Maryland's University, USA

Emails: : abedallah.abualkishik@aue.ae, rasha.almajed@aue.ae, wvthompson@towson.edu

Abstract

A cloud computing (CC) method was effectual if its sources were used in optimal way and an effectual consumption is attained by using and preserving proper management of cloud sources. Resource management can be attained through adoption of powerful source scheduling, allotment, and robust source scalability methods. The balancing of load in cloud is performed at VM level or physical machine level. A task use sources of VM and whenever a bunch of tasks reaches VM, the sources will be exhausted means no source is now existing for handling the extra task requests. This article develops an Intelligent Red Deer Algorithm based Energy Aware Load Balancing Scheme for data fusion in Cloud Environment, called IRDA-EALBS model. The presented IRDA-EALBS model majorly concentrates on the balancing of load among the virtual machines (VMs) in the cloud environment. The IRDA-EALBS model is mainly stimulated from the nature of red deers during a breading period. In addition, the IRDA-EALBS model derived an objective function to minimize energy consumption and maximize makespan. To demonstrate the enhanced performance of the IRDA-EALBS model, a wide range of experimental analyses is carried out. The simulation results highlighted the enhanced outcomes of the IRDA-EALBS model over other load balancers in the cloud environment.

Keywords: Data Fusion; Internet of Things; Cloud computing; Load balancing; Energy efficiency; Red deer algorithm

1. Introduction

Cloud Computing (CC) is an online organization innovation that common a fast improvement in the progresses of transmission techniques by giving assistance to users of different necessities with the guide of web computing resources. It has courses of action on software and hardware applications together with software progress stages and testing contraptions as resources [1, 2]. This resource conveyance can be achieved by providing services. Under classification of Infrastructure as a service (IaaS) cloud, the last two comprise platform as a service (PaaS) and Software as a service (SaaS) cloud independently [3]. The CC is an on-demand networking system empowered computing mechanism that shares resources as services impeached on pay-as-you-go (PAYG) strategy [4]. A portion of the monster player in innovation is Microsoft, Amazon, SAP, Google, VMware, Oracle, IBM Sales force, etc [1, 2]. Greater part of CPS is state of the art IT organizations. The CC method comprises two unmistakable headings. The initial one is the service conveyance method that describes the sort of service presented by common cloud providers. According to this, there are prominently follows three significant service mechanisms IaaS, SaaS, and PaaS [5, 6]. The other part of CC mechanism can be viewed on size of direction, access, affiliation, possession, and size. The description for CC outlines four cloud mechanisms such as public, private, local area, and mixture blend clouds [7].

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

A CC technology is successful in the event that its resources are applied in most desirable manner and its utilization is accomplished by using and keeping up with genuine administration of cloud resources. Resource management can be accomplished by embracing stronger resource booking, stronger resource adaptability, and allocation methodologies. Such resources are provided to users as Virtual Machine (VM) via communication called virtualization that uses components (software and hardware) called hypervisor [8]. The best benefit of CC is that a physical machine (PM) is changed into multiple user VMs [9, 10]. The Cloud Service Provider (CSP) performs a critical task in service conveyance to clients and is challenging with accessible virtual assets. While helping user requirements, certain VM gets lesser traffic and weighty traffic of client task. In this way, the CSP is left with uneven machine that has an enormous gradient of resource utilization and client tasks [11]. The load balancing process is depicted in Fig. 1.

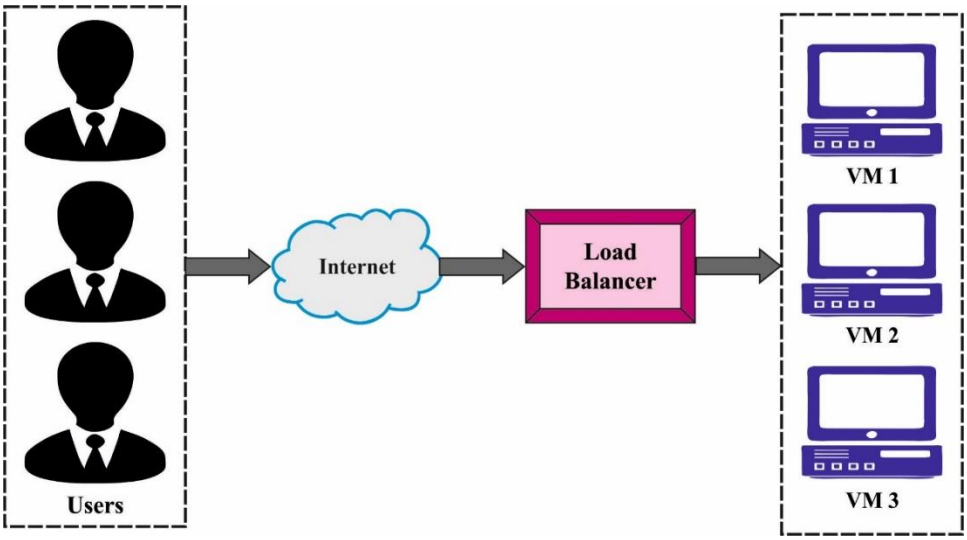

Figure 1: Load balancing in cloud

The challenge of load unbalancing is an unwanted instance in the CSP side that corrupts the efficacy and performance of the computation resource together with ensured Quality of Service (QoS) on Service Level Agreement (SLA) amongst provider and buyer. In this situation, there emerges requirement for load balancing (LB) and is hot research topic amongst research workers. The LB in CC is performed at VM or PM levels [2]. Cloud LB is the strategy engaged with computing assets and dispersing workloads across servers. This kind of circulation ensures maximal throughput in minimum reaction time. The task is isolated amongst no less than two servers, network interfaces hard drives, or other computing resources, empowering effective maximum throughput and resource utilization. Along these lines, for a higher traffic site, strong usage of cloud LB could ensure business coherence. At this time, load implies the site traffic as well as fuses memory limit, CPU load, and network burden of every server. A LB approach ensures that every structure in the organization has similar quantity of work. This implies neither any of them is pointlessly under-utilized, nor over-loaded. The load balancer transports information depending on how involved every server or hub is. In lack of a load balancer, the user ought to stand by while his cycle gets taken care of, which might be exorbitantly demotivating and tiring for them. Different data such as position waiting in line, CPU handling rate, work arrival rate, and so on are traded among the processors in the LB procedure. Failure in the correct application of load balancer prompts serious outcomes, information get lost being one of them.

This article develops an Intelligent Red Deer Algorithm based Energy Aware Load Balancing Scheme for data fusion in Cloud Environment, called IRDA-EALBS model. The presented IRDA-EALBS model majorly concentrates on the balancing of load among the virtual machines (VMs) in the cloud environment. The IRDA-EALBS model is mainly stimulated from the nature of red deers on a breading period. In addition, the IRDA-EALBS model derived an objective function to minimize energy consumption and maximize makespan. To demonstrate the enhanced performance of the IRDA-EALBS model, a wide range of experimental analyses is carried out. The simulation results highlighted the

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

enhanced outcomes of the IRDA-EALBS model over other load balancers when process in different data models in the cloud environment.

2. Related Works

Priya et al. [11] present an incorporated load balancing and resource scheduling algorithms for effective CSP. The technique creates a Fuzzy-based Multi-dimensional Resource Scheduling method for obtaining resource scheduling efficacy in cloud framework. Increased usage of VMs by using fair and effective LB is later accomplished by vigorously choosing a request from a class via Multi-dimensional Queuing Load Optimization approach. An LB method is later executed for avoiding overutilization and underutilization of the resource, enhancing latency time for every class of request. Golchi et al. [12] developed a hybrid of firefly and Improved Particle Swarm Optimization (IPSO) approaches for reaching the good average load to make and improve the significant metrics namely powerful resource usage and the response time of task correspondingly.

Jena et al. [13] developed a new technique of dynamic LB amongst the VMs using hybridization of modified Particle swarm optimization (MPSO) and better Q-learning algorithm called QMPSO. The hybridization procedure can be implemented for adjusting the velocity of the MPSO over the pbest and gbest depends on the optimal action produced by the better Q-learning. In [14], we proposed a dynamic scheduling approach which balances the workloads amongst each VM using elastic resource deprovisioning and provisioning according to the final optimum k-interval. Furthermore, the technique was tested under parameter number of tasks (10 to 30) to accomplish improved scalability.

In [15], presented a new LB technique system for Focal Load Balancer (F-LB) that reduces the traffic in the Cloud, while guaranteeing a smooth data flow in the cloud framework. The presented approach uses dynamic LB features through static balancing and prevents the impairment that a static LB cause them to fail. Thakur and Goraya [16] developed a hybrid metaheuristic based resource allocation architecture called RAFL for LB in CC infrastructure. The aim is to proactively reduce the LB over active PMs and amongst their resource capabilities (for instance, RAM and CPU). This prevents underloading or overloading of active PMs and exploits their resource ability in a balanced way. In the presented architecture, a phasor particle swarm optimization and dragonfly algorithm based hybrid optimization algorithm called PPSO-DA is utilized for generating an optimum resource allocation plan for LB system.

3. The Proposed IRDA-EALBS model

This article has developed a new IRDA-EALBS technique for Energy Aware Load Balancing Scheme on Cloud Environment. The presented IRDA-EALBS model majorly concentrates on the balancing of load among the VMs in the cloud environment.

3.1 Overview of RDA

RDA is depending on the mating behaviour of Scottish RD during the time of breading. Like numerous metaheuristic processes, the RDA initiates b an arbitrary widespread and optimum red deer (RD) is preferred as male RD while the residual RDs are called hinds. The overall objective of the optimization procedure is the recognition of near optimum or global solutions regarding the parameter of the problem. Now, RD characterizes a feasible solution X in the searching region. The dimension of the solution X is denoted by N_{var}. For a "N_{var} dimensional optimization issue, an RD is a $1xN_{var}$array is denoted by the following equation.

$$Red\ Deer\ = \left[X1, X2, X3, \dots, X_{N_{var}}\right]. \quad (1)$$

In addition, the function value is defined for each RD, as follows.

$$Value\ = f\ (Red\ Deer) = f(X1, X2, X3, \dots, X_{N_{var}}) \quad (2)$$

At first, a set of initialized population Npop is produced. Next, the optimum RD is preferred as $Nmale$ and the residuals are N_{hind} ($N_{hind} = Npop - Nmale$). It is assumed that the $Nmale$ count indicates

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

the elitist criteria of the RDA. On the other hand, the $Nmale$ count controls the intensification feature while N_{hind} consider the diversification phase of RDA.

Then, the male RD tried to improve the grace via roaring. Simultaneously, this procedure might be failure or success. Note that the male RD is the optimum solution. The neighboring RD of the male RD is defined and once the objective function of neighboring RD is superior to the male RD, then it is replaced. In fact, the RDA enables every male RD to upgrade the position as follows:

$$male_{new} = \begin{matrix} male_{o1d} + a_1 \times ((UB - LB) * a_2) + LB & if a_3 \geq 0.5 \\ male_{o1d} - a_1 \times ((UB - LB) * a_2) + LB & if a_3 < 0.5 \end{matrix} \quad (3)$$

In order to produce a feasible neighboring solution for male, the upper bound (UB) and lower bound (LB) limits the searching region. Indeed, the male RD might differ naturally where RD is fascinating, stronger, and efficiently extend the territory than other RDs and it is given by the following equation:

$$N_{Com} = round\{\gamma . N_{male}\} \quad (4)$$

In Eq. (4), N_{Com} indicates the male count and γ is determined by the primary values of RDA that ranges from zero and one. As well, it is denoted by the subsequent formula:

$$N_{stag} = N_{male} - N_{Com} \quad (5)$$

Next, all commanders fight with the stag arbitrarily. In relation to the solution space, $stag$ and commander approaches to others. Likewise, two novel solutions are attained and substituted the commander with the good one and it is expressed as follows.

$$New\ 1 = \frac{(Com + Stag)}{2} + b_1 \times ((UB - LB) * b_2) + LB) \quad (6)$$

$$New\ 2 = \frac{(Com + Stag)}{2} - b_1 \times ((UB - LB) * b_2) + LB) \quad (7)$$

In Eq. (7), New1 and New2 represent the two recently generated solutions in the fighting task.

Then, the harems are formed that is a swarm of hind in the control of male commander. The harem size is based solely on the power of male commander. In forming harem, the hinds are classified among the commanders, as follows [17, 18].

$$y_n = v_n - \max\{v_i\} \quad (8)$$

In Eq. (8), v_n indicates the strength of $n\text{-}th$ commander and y_n denotes the normalized value. In order to determine the normalization power of the commander, the subsequent formula is applied.

$$P_n = \left| \frac{V_n}{\sum_{i=1}^{N_{Com}} V_i} \right| \quad (9)$$

On the other hand, the normalized strength of male commander is the portion of the hind that should be controlled by that male and it is described below.

$$N.harem_n = round\{P_n . N_{hind}\} \quad (10)$$

In Eq. (10), $N.harem_n$ shows the hind count in $n\text{-}th$ harem and N_{hind}indicates the hind count. Like other species, RD mates with others. This process is conducted by the commander and α percentage of hind in the harem.

$$N.harem_n^{mate} = round\{\alpha . N.harem_n\} \quad (11)$$

In Eq. (11), $N.harem_n^{mate}$ indicates the hind count of $n\text{-}th$ harem. A harem is preferred randomly and the male commander mate with β percent of hinds. Indeed, the commander fought with other harem to expand the region. The hind count in the harem mate with the commander can be defined in the following equation:

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

$$N.harem_k^{mate} = round\{\beta.N.harem_k\} \quad (12)$$

In Eq. (12), $N.harem_k^{mate}$ indicates the hind count in $k\text{-}th$ harem. All stags mate with nearby hinds. In the breeding process, the male RD follows the nearby hing. It could be the favourite hind among each hind with no assumption of the harem territory. Each stage undergoes mating with the nearby hind. For finding the nearby hind, the distance among the stage and each hind in the $J\text{-}th$ parameter is defined by

$$d_i = \left(\sum_{j \in J} (stag_j - hind_j^i)^2 \right)^{1/2} \quad (13)$$

In Eq. (13), d_i indicates the distance among the $i\text{-}th$ hinds and stags. Hence, the minimal value in the matrix describes the preferred hinds. The procedure besides hind selection is mating. For choosing the succeeding generation, two strategies are followed. At first, each male RD is ket. Then, consider the remaining population in the succeeding generation. The hind is elected and offspring is generated via the mating procedure regarding the fitness value. Once the adequate iteration count is obtained, the ending condition is met.

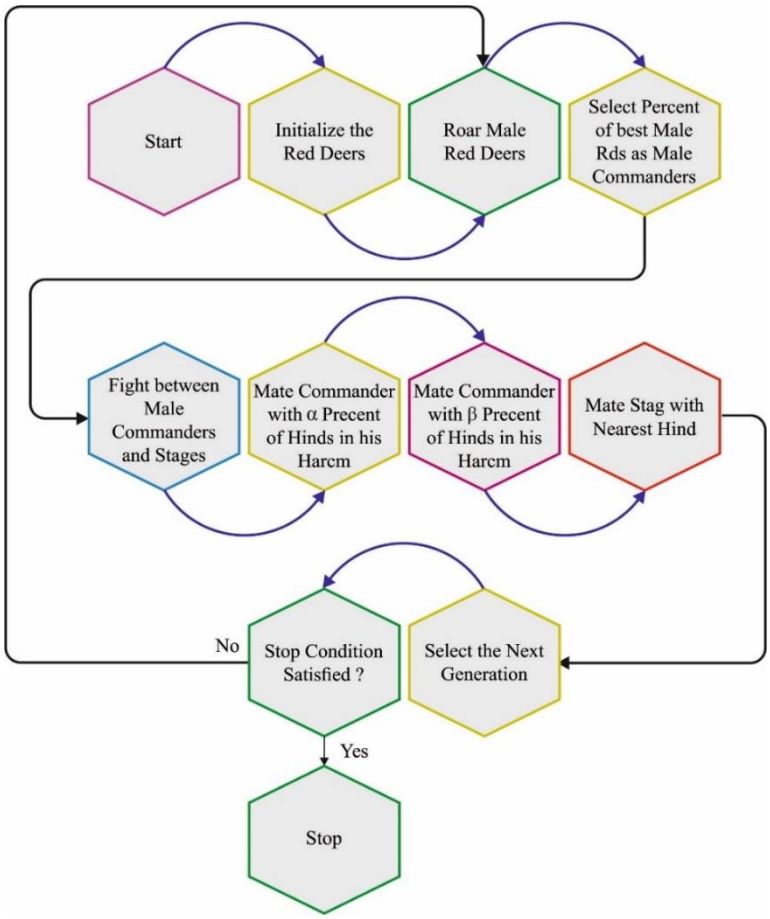

Figure 2: Flowchart of RDA

3.2 Process involved in IRDA-EALBS model

The presented IRDA-EALBS model fulfills the makespan by performing optimal TS procedures with various aspects of incoming tasks. In this approach, the cloud application was considered as collected of user jobs (UJs) that implement complex computing tasks employing cloud resources [19]. Let $UserJob = (U_1, U_2, U_3 \ldots U_N)$ represents the batch of user applications obtained at particular time. Every UJ (U_i) is represented as duplet $< a_i, d_i >$. In which a_j stands for the arrival time of UJ (U_i) and d_i stands for the purpose of UJs (U_i). During the scheduling approach, the UJ was allocated for the

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

available DCs $(D_1, D_2, D_3 \ldots D_M)$, whereas $N \leq$ M. All DCs (D_i) has linked to duplet $< C_i, m_j >. c_i$, the price per unit time charged as DC to implement UJ, m_i demonstrates the count of available Processing Elements (PEs) to apply UJs. Each DC has a set of PEs $\{PE_1, PE_2 \ldots PE_m\}$ to compute allocated UJs. Every PE is linked to duplet $< s, p >. 's'$ and $'p'$ represent the executed speed and power consumption of all PEs correspondingly. The set of nodes $V = \{T_1 \ldots T_n\}$ represents the tasks, and the set of arcs demonstrates the control or data dependency among tasks. Optimal scheduling of UJs to accessible PEs from cloud in different DC is a vital objective of this work. Considering the UJs U_i has assigned to DC (D_j). T_k implies the set of tasks of the UJs (U_i) was given to PE (P_j). Once the time need that apply T_k employ P_j is signified as G_j. The termination time of T_j is expressed as:

$$Finish(T_k) = start(T_k) + \Gamma_j \tag{14}$$

Therefore, the whole time vital to complete the UJ as D_j is demoted as $Makespan_j$ and defined as:

$$Makespan_j = \max\{Finish\ (T_k)\} \tag{15}$$

whereas $T_{(k=1\ldots n)}$ the tasks are assigned to D_j. The energy consumption for calculating the UJs (U_i) by DC D_j has measured as:

$$E_i = \sum_{k=1}^{n} (\Gamma_k \times p_k) \tag{16}$$

In which p_k denotes the power consumption per unit time through PEs (P_j) for procedure offered task (T_k). The cost of processing the UJ as DC D_j is calculated as:

$$C_j = c_j \times Makespan_j \tag{17}$$

Whereas c_j denotes the price per unit time charged by DC D_j to applied the UJs.

The consumption (U_j) of DCs (D_j) is estimated as:

$$U_j = \frac{Makespan_j}{\max\{Makespan_k\}, k = 1 \ldots M} \tag{18}$$

An objective function of this projected approach is expressed as:

$$Minimize\ Makespan_j\ j = 1..M \tag{19}$$

$$Minimize\ E_j \mathrm{j} = 1..\mathrm{M} \tag{20}$$

$$Minimize \left\{ \sum_{j=1}^{M} C_j \right\} \tag{21}$$

$$Maximize\ U_j \mathrm{j} = 1..\mathrm{M} \tag{22}$$

Subjected to:

- The UJ requirement end before deadline (d_i)
- Every UJ is allocated to only one DC.

The count of UJs is reduced to the count of present DC at a particular time.

4. Experimental Validation

This section aims to analyze the load balancing results of the IRDA-EALBS model. Table 1 and Fig. 3 indicate the comparison study of the IRDA-EALBS model with recent models interms of CPU utilization (CPUU). The outcomes implied that the IRDA-EALBS model has gained enhanced CPUU under all tasks. For instance, on small tasks, the IRDA-EALBS model has provided improved CPUU of

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

75.68% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered reduced CPUU of 51.42%, 56.91%, 67.60%, 69.62%, and 71.35% respectively. Also, on extra-large tasks, the IRDA-EALBS model has provided improved CPUU of 99.37% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered reduced CPUU of 84.06%, 89.26%, 93.59%, 96.48%, and 98.50% respectively.

Table 1: CPUU examination of IRDA-EALBS model

CPU Utilization (%)						
Type of Tasks	**DLB**	**LB-BC**	**LB-RC**	**IPSO-Firefly**	**FIMPSO**	**IRDA-EALBS**
Small	51.42	56.91	67.60	69.62	71.35	75.68
Medium	62.97	69.33	76.84	77.42	78.57	80.59
Large	73.08	78.57	85.21	88.97	92.72	95.61
Extra-Large	84.06	89.26	93.59	96.48	98.50	99.37

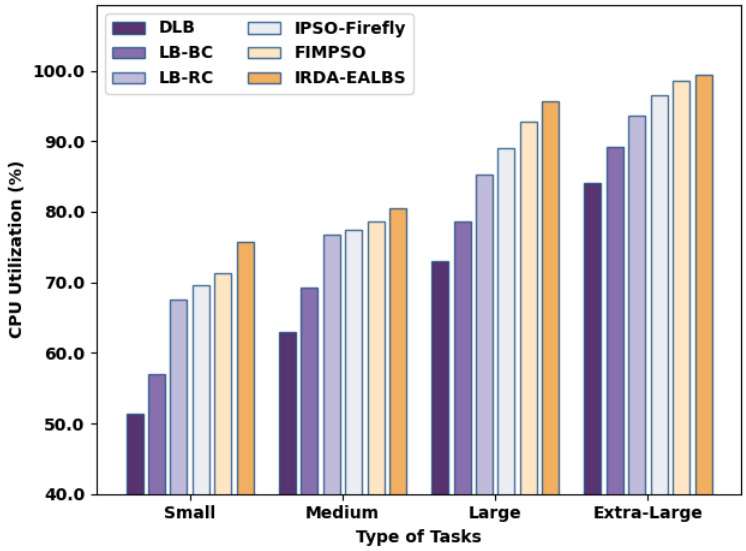

Figure 3: Comparative CPUU examination of IRDA-EALBS model

Table 2 and Fig. 4 report an analysis of the IRDA-EALBS model with recent models interms of memory utilization (MEMU). The results represented that the IRDA-EALBS model has increased enhanced MEMU under all tasks. For instance, on small tasks, the IRDA-EALBS model has delivered better MEMU of 61.85% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have accessible compact MEMU of 48.78%, 52.05%, 56.40%, 58.04%, and 59.67% respectively. Besides, on extra-large tasks, the IRDA-EALBS model has provided improved MEMU of 95.34% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered reduced MEMU of 80.36%, 83.90%, 88.80%, 90.44%, and 92.34% respectively.

Table 2: MEMU examination of IRDA-EALBS model

Memory Utilization (%)						
Type of Tasks	**DLB**	**LB-BC**	**LB-RC**	**IPSO-Firefly**	**FIMPSO**	**IRDA-EALBS**
Small	48.78	52.05	56.40	58.04	59.67	61.85
Medium	58.86	63.21	66.48	69.47	70.02	73.01
Large	68.66	71.38	78.73	82.00	84.72	86.35
Extra-Large	80.36	83.90	88.80	90.44	92.34	95.34

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

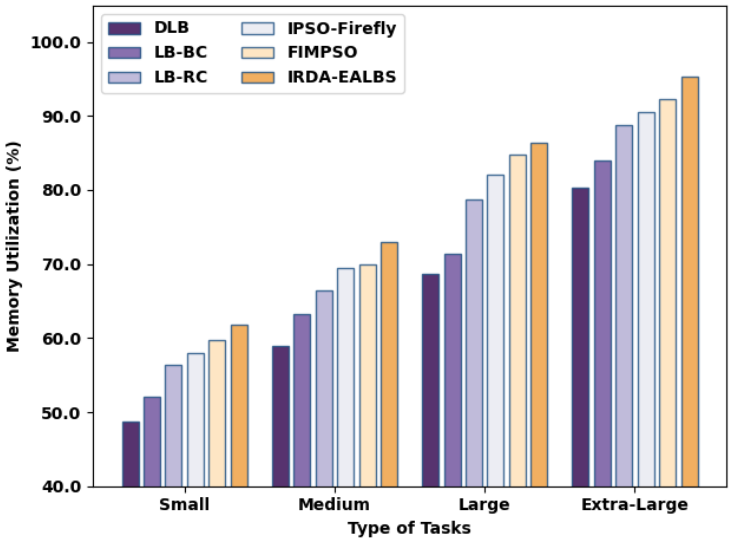

Figure 4: Comparative MEMU examination of IRDA-EALBS model

Table 3 and Fig. 5 provide an analysis of the IRDA-EALBS model with recent models interms of reliability (REL). The results characterized that the IRDA-EALBS model has amplified REL under all tasks. For instance, on small tasks, the IRDA-EALBS model has provided better REL of 99.95% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have manageable REL of 86.26%, 89.83%, 95.49%, 97.57%, and 98.47% respectively. Followed by, on extra-large tasks, the IRDA-EALBS model has provided improved REL of 70.18% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered reduced REL of 43.09%, 52.62%, 59.46%, 61.85%, and 66.01% respectively.

Table 3: REL examination of IRDA-EALBS model

Reliability (%)						
Type of Tasks	**DLB**	**LB-BC**	**LB-RC**	**IPSO-Firefly**	**FIMPSO**	**IRDA-EALBS**
Small	86.26	89.83	95.49	97.57	98.47	99.95
Medium	66.01	75.24	78.82	83.88	87.45	90.13
Large	57.08	65.42	74.35	76.73	79.41	82.09
Extra-Large	43.09	52.62	59.46	61.85	66.01	70.18

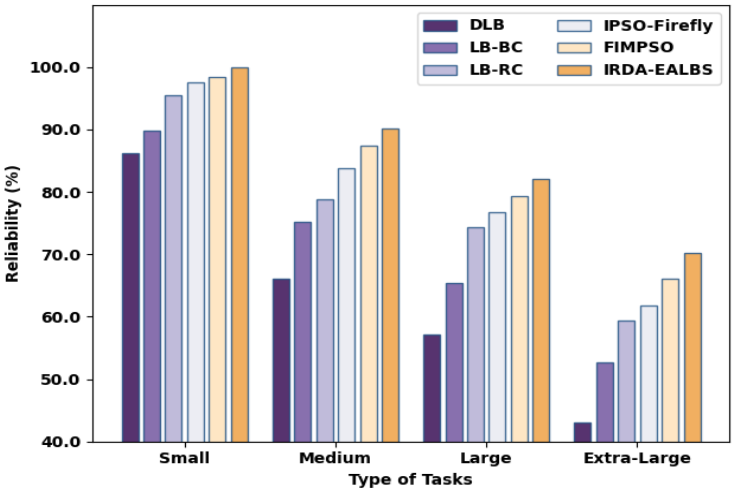

Figure 5: Comparative REL examination of IRDA-EALBS model

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

Table 4 and Fig. 6 report an analysis of the IRDA-EALBS model with recent models interms of average throughput (ATHR). The results represented that the IRDA-EALBS model has increased enhanced ATHR under all tasks. For instance, on small tasks, the IRDA-EALBS model has delivered better ATHR of 96.87% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have accessible compact ATHR of 76.41%, 84.65%, 91.19%, 91.47%, and 95.17% respectively. Besides, on extra-large tasks, the IRDA-EALBS model has provided improved ATHR of 70.73% whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered reduced ATHR of 42.88%, 50.27%, 59.36%, 63.62%, and 67.32% respectively.

Table 4: ATHR examination of IRDA-EALBS model

Average Throughput (%)						
Type of Tasks	**DLB**	**LB-BC**	**LB-RC**	**IPSO-Firefly**	**FIMPSO**	**IRDA-EALBS**
Small	76.41	84.65	91.19	91.47	95.17	96.87
Medium	65.90	71.01	79.54	79.54	82.10	83.80
Large	51.12	59.65	69.02	66.47	71.01	75.28
Extra-Large	42.88	50.27	59.36	63.62	67.32	70.73

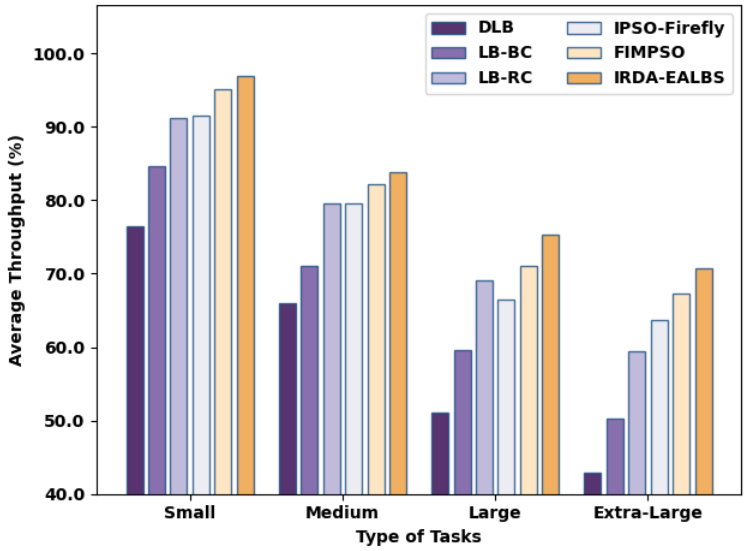

Figure 6: Comparative ATHR examination of IRDA-EALBS model

Table 5 and Fig. 7 report an analysis of the IRDA-EALBS model with recent models interms of makespan (MKSN). The results represented that the IRDA-EALBS model has reduced MKSN under all tasks. For instance, on small tasks, the IRDA-EALBS model has delivered least MKSN of 37.04 whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have accessible increased MKSN of 56.42, 49.68, 47.99, 45.47, and 44.62 respectively. On the other hand, on extra-large tasks, the IRDA-EALBS model has provided reduced MKSN of 128.05 whereas the DLB, LB-BC, LB-RC, IPSO-FF, and FIMPSO models have offered higher MKSN of 272.99, 260.35, 154.17, 150.80, and 147.43 respectively.

Table 5 : MKSN examination of IRDA-EALBS model

Makespan						
Type of Tasks	**DLB**	**LB-BC**	**LB-RC**	**IPSO-Firefly**	**FIMPSO**	**IRDA-EALBS**
Small	56.42	49.68	47.99	45.47	44.62	37.04

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

Medium	104.45	97.71	91.81	89.29	85.92	74.12
Large	187.88	179.45	171.87	170.18	165.97	148.27
Extra-Large	272.99	260.35	154.17	150.80	147.43	128.05

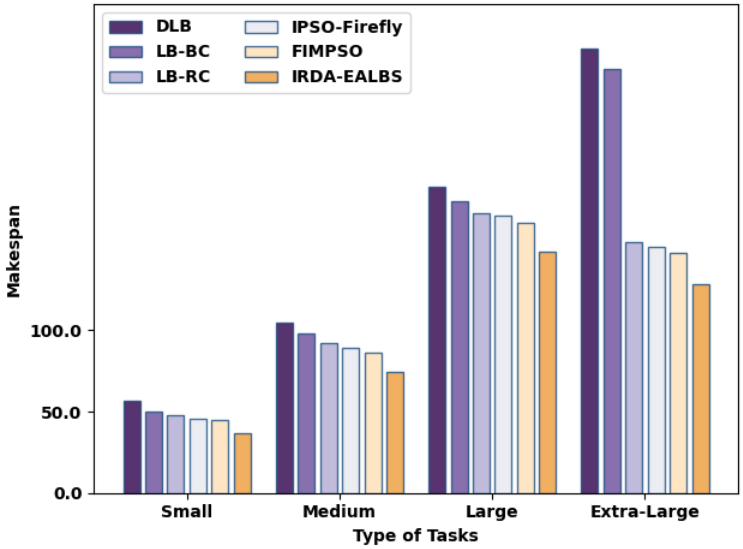

Figure 7: Comparative MKSN examination of IRDA-EALBS model

Table 6 and Fig. 8 exhibit analysis of the IRDA-EALBS model with recent models interms of average time. Based on loading time, the IRDA-EALBS model has offered lower average time of 0.121 whereas the GA, IPSO, FF, FF-IPSO, and FIMPSO models have reached higher loading time. Based on response time, the IRDA-EALBS model has offered lower average time of 12.310 whereas the GA, IPSO, FF, FF-IPSO, and FIMPSO models have reached higher response time. Thus, the results confirmed that the IRDA-EALBS model has reported better outcomes.

Table 6: Average time examination of IRDA-EALBS model

Average Time (ms)			
Methods	**loading**	**turnaround**	**response**
IRDA-EALBS	0.121	20.321	15.310
GA	0.144	26.550	19.210
IPSO Algorithm	0.813	60.930	49.050
Firefly Algorithm	0.322	55.110	49.320
FF-IPSO	0.352	24.550	13.240
FIMPSO	0.375	23.360	16.230

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

Figure 8: Comparative Average time examination of IRDA-EALBS model

5. Conclusion

This article has developed a new IRDA-EALBS technique for Energy Aware Load Balancing Scheme on Cloud Environment. The presented IRDA-EALBS model majorly concentrates on the balancing of load among the VMs in the cloud environment. The IRDA-EALBS model is mainly stimulated from the nature of red deers during a breading period. In addition, the IRDA-EALBS model derived an objective function to minimize energy consumption and maximize makespan. To demonstrate the enhanced performance of the IRDA-EALBS model, a wide range of experimental analyses is carried out. The simulation results highlighted the enhanced outcomes of the IRDA-EALBS model over other load balancers in the cloud environment.

References

[1] Kaur, A. and Luthra, M.P., 2018. A review on load balancing in cloud environment. International journal, 17(1).

[2] Mishra, S.K., Sahoo, B. and Parida, P.P., 2020. Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences, 32(2), pp.149-158.

[3] Liaqat, M., Naveed, A., Ali, R.L., Shuja, J. and Ko, K.M., 2019. Characterizing dynamic load balancing in cloud environments using virtual machine deployment models. IEEE Access, 7, pp.145767-145776.

[4] Milan, S.T., Rajabion, L., Ranjbar, H. and Navimipour, N.J., 2019. Nature inspired meta-heuristic algorithms for solving the load-balancing problem in cloud environments. Computers & Operations Research, 110, pp.159-187.

[5] Kaur, A. and Kaur, B., 2019. Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment. Journal of King Saud University-Computer and Information Sciences.

[6] Nanjappan, M. and Albert, P., 2022. Hybrid-based novel approach for resource scheduling using MCFCM and PSO in cloud computing environment. Concurrency and Computation: Practice and Experience, 34(7), p.e5517.

[7] Swarnakar, S., Bhattacharya, S. and Banerjee, C., 2021. A bio-inspired and heuristic-based hybrid algorithm for effective performance with load balancing in cloud environment. International Journal of Cloud Applications and Computing (IJCAC), 11(4), pp.59-79.

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022

[8] Joshi, A. and Munisamy, S.D., 2022. Evaluating the performance of load balancing algorithm for heterogeneous cloudlets using HDDB algorithm. International Journal of System Assurance Engineering and Management, pp.1-9.

[9] Asghari, A. and Sohrabi, M.K., 2021. Combined use of coral reefs optimization and reinforcement learning for improving resource utilization and load balancing in cloud environments. Computing, 103(7), pp.1545-1567.

[10] Lin, W., Peng, G., Bian, X., Xu, S., Chang, V. and Li, Y., 2019. Scheduling algorithms for heterogeneous cloud environment: main resource load balancing algorithm and time balancing algorithm. Journal of Grid Computing, 17(4), pp.699-726.

[11] Priya, V., Kumar, C.S. and Kannan, R., 2019. Resource scheduling algorithm with load balancing for cloud service provisioning. Applied Soft Computing, 76, pp.416-424.

[12] Golchi, M.M., Saraeian, S. and Heydari, M., 2019. A hybrid of firefly and improved particle swarm optimization algorithms for load balancing in cloud environments: Performance evaluation. Computer Networks, 162, p.106860

[13] Jena, U.K., Das, P.K. and Kabat, M.R., 2020. Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment. Journal of King Saud University-Computer and Information Sciences.

[14] Kumar, M. and Sharma, S.C., 2018. Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment. Computers & Electrical Engineering, 69, pp.395-411

[15] Mohammed, M.A., Hasan, R.A., Ahmed, M.A., Tapus, N., Shanan, M.A., Khaleel, M.K. and Ali, A.H., 2018, June. A Focal load balancer based algorithm for task assignment in cloud environment. In 2018 10th International Conference on Electronics, Computers and Artificial Intelligence (ECAI) (pp. 1-4). IEEE

[16] Thakur, A. and Goraya, M.S., 2022. RAFL: A hybrid metaheuristic based resource allocation framework for load balancing in cloud computing environment. Simulation Modelling Practice and Theory, p.102485

[17] Fathollahi-Fard, A.M., Hajiaghaei-Keshteli, M. and Tavakkoli-Moghaddam, R., 2020. Red deer algorithm (RDA): a new nature-inspired meta-heuristic. Soft Computing, 24(19), pp.14637-14665.

[18] Zitar, R.A., Abualigah, L. and Al-Dmour, N.A., 2021. Review and analysis for the Red Deer Algorithm. Journal of Ambient Intelligence and Humanized Computing, pp.1-11.

DOI: https://doi.org/10.54216/FPA.080103
Received: April 17, 2022 Accepted: August 21, 2022