OP-MLB: An Online VM Prediction-Based Multi-Objective Load Balancing Framework for Resource Management at Cloud Data Center

Deepika Saxena, Ashutosh Kumar Singh, *Senior Member, IEEE*, and Rajkumar Buyya, *Fellow, IEEE*

Abstract—The elasticity of cloud resources allows cloud clients to expand and shrink their demand for resources dynamically over time. However, fluctuations in the resource demands and pre-defined size of virtual machines (VMs) lead to lack of resource utilization, load imbalance, and excessive power consumption. To address these issues and to improve the performance of data center, an efficient resource management framework is proposed, which anticipates resource utilization of the servers and balances the load accordingly. It facilitates power saving, by minimizing the number of active servers, VM migrations, and maximizing the resource utilization. An online resource prediction system, is developed and deployed at each VM to minimize the risk of Service Level Agreement (SLA) violations and performance degradation due to under/overloaded servers. In addition, multi-objective VM placement and migration algorithms are proposed to reduce the network traffic and power consumption within data center. The proposed framework is evaluated by executing experiments on three real world workload datasets namely, Google Cluster dataset, Planet Lab, and Bitbrains VM traces. The comparison of proposed framework with the state-of-the-art approaches reveals its superiority in terms of different performance metrics. The improvement in power saving achieved by OP-MLB framework is upto 85.3 percent over the Best-Fit approach.

Index Terms—Cloud computing, communication cost, load balancing, online-prediction, oversubscription, server, virtual machine

1 Introduction

Commercial cloud service providers (CSPs) offer elastic computing benefits to cloud clients in the form of variety of VMs with different resource capacities at minimum capital investment [1] which allow them to choose appropriate VMs for their applications execution and pay accordingly. However, the resource requirement changes over time that raises challenges for CSPs including inefficient resource utilization and extravagant power consumption. For instance, the effect of resources utilization deviation from their demanded capacity can be seen in Fig. 1 that shows a snapshot of resource utilization over a period of 24 hours, collected from Google Cluster dataset [2] where the CPU and memory utilization fluctuates between 5-42 and 5-60 percent respectively. It reveals that most of the time user over-subscribes the resources that results into inefficient VM placement and leads to ample amount of resource wastage and power consumption [3]. The reason behind over-subscription is that cloud clients are unaware of the actual resource usage of their applications and to avoid the risk of failure at peak time, they tend to over-estimate the resources [4]. Moreover, the network traffic scales up due to placement of inter-dependent VMs on farther located servers. Cisco Global Cloud Index predicts that by the year 2021, network traffic among the devices within data center will grow at Compound Annual Growth Rate (CAGR) of 23.4 percent [5]. Therefore, a network aware and power efficient resource management technique is needed to consolidate cloud workload on minimum number of active servers and minimize network traffic within data center [6].

The elastic resource management includes several operations like, balanced scheduling of applications and VMs, controlling of server over/under-load by applying VM migration etc. [7], each of which has been investigated individually in the existing work. For example, task scheduling is discussed in [8], [9], VM placement and migration algorithms are presented in [10], [11] and [12], [13] respectively. In the real cloud environment, all these operations work continuously in cooperative manner because the performance achieved by optimal task scheduling degrades, if VMs are not placed effectively on the server. Similarly, optimizing VM placement and ignoring optimization during VM migration cannot bring the possible power saving. Therefore, all these operations must be considered in cooperation to achieve the real benefits of performance optimization associated at different levels. Moreover, the existing literature [14] suggests that prior knowledge of upcoming workload can assist in load balancing and reduction of performance degradation due to overloads. On the other hand, SLA violations may increase due to errors in predicting future workloads. Now, the two key challenges are: (i) How to deploy prediction system at data center to assist in

- *Deepika Saxena and Ashutosh Kumar Singh are with the Department of Computer Applications, National Institute of Technology, Kurukshetra, Haryana 136119, India.*
  E-mail: 13deepikasaxena@gmail.com, ashutosh@nitkkr.ac.in.
- *Rajkumar Buyya is with the Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, University of Melbourne, Parkville, VIC 3010, Australia.*
  E-mail: rbuyya@unimelb.edu.au.

Manuscript received 3 Jan. 2020; revised 15 Dec. 2020; accepted 4 Feb. 2021. Date of publication 12 Feb. 2021; date of current version 6 Dec. 2022.
(Corresponding author: Deepika Saxena.)
Recommended for acceptance by K. Chen.
Digital Object Identifier no. 10.1109/TCC.2021.3059096

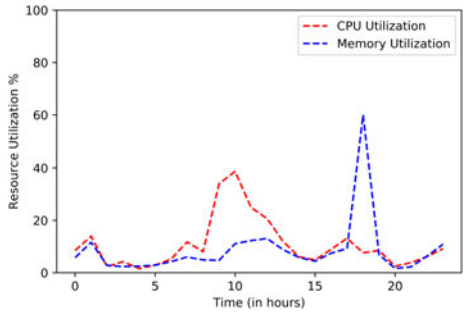


Fig. 1. From Google cluster trace: Snapshot of 24 hours resource utilization percentage on randomly chosen VM.

the resource management? (ii) How to utilize the predicted information for improving performance of data center while avoiding risk of SLA violations?

To address aforementioned challenges, an **O**nline VM **P**rediction based **M**ulti-objective **L**oad **B**alancing (OP-MLB) framework is proposed that incorporates several algorithms to work in collaboration to provide efficient resource management for cloud environment. It includes SLA aware user's application execution on VMs followed by the multi-objective VM assignment and migration for handling of over/under-loaded servers. Furthermore, to effectively exploit over-subscription by cloud clients, an online-prediction system is developed and deployed at each VM that learns and predicts the future resource demand in parallel with application execution at respective VM to allow closer estimation of the expected resource demand beforehand. The physical resources are distributed on the basis of predicted resource demand combined with the error-driven padding (EDP) to avoid risk of SLA violations.

1.1 Our Contributions

The key contributions of the proposed work are:

- A novel multi-objective load-balancing approach based on online resource prediction i.e., OP-MLB framework is proposed for elastic resource management at cloud data center.
- A neural network based online predictor tuned with adaptive evolutionary algorithm is developed to forecast resource usage with enhanced accuracy by appending error-driven padding to predicted output.
- Multi-objective load balancing is presented with proactive *VM placement* and *migration*, where VMs are allocated subject to maximum resource utilization and minimum power consumption, communication cost. It triggers VM migration before overload occurrence to improve performance of data center.
- Substantial power saving is achieved by exploiting oversubscribed cloud, applying successive optimization at different phases and reducing number of active servers and VM migrations.
- Implementation and evaluation of proposed framework by using three real benchmark datasets reveal that proposed work outperforms the state-of-art approaches in terms of performance metrics like server overload prediction, resource utilization, reduction of power consumption and communication cost.

TABLE 1
Notations

Symbols	Explaination terms
A_o, A_g	availability offered, availability guaranteed
a, b, Ω	server uptime, downtime, penalty cost
p, q, r	number of input, hidden, output nodes
N	number of neural networks
Λ, Φ	mutant vector, current vector
μ, Cr	mutation rate, crossover-rate
$\wp$	mutation scheme selection parameter
χ	new offspring for next generation or epoch
Γ	mutation selection probability
δ	number of offspring that successfully reached next generation
k	size of data samples for specific prediction interval
z_a and z_p	actual and predicted output
Ψ, n	VM allocation, number of Ψ
γ	status of server
ω	mapping of VM on server
β	mapping between user and server
L	size of neural network
W	size of VM= $CPU \times Memory$
O_s, O_v	list of overloaded server, list of VMs on O_s
c_{ij}	cost of moving i^{th} VM to j^{th} server

Organization. Section 2 discusses related work. The overview of proposed framework is given in Section 3 followed by its detailed description including task assignment, online resource predictor and load management in Sections 4, 5 and 6 respectively. Main algorithm is presented in Section 7. The performance evaluation is presented in Section 8 followed by conclusion and future scope of the proposed work in Section 9. Table 1 shows the list of symbols with their explanatory terms that have been used throughout the paper.

2 Related Work

The related work is organized into two subsections: (2.1) the work which predicts the resource usage before load balancing and (2.2) overload handling and VM migration techniques.

2.1 Resource Prediction and Allocation Techniques

Dynamic resource allocation and VM placement were presented in [15], [16] in which future demands of the resource were predicted for energy efficient resource utilization. The future state of VMs was predicted by computing exponential weighted moving average (EWMA) based on past behaviors of VMs. Nguyen *et al.* [17] presented VM consolidation method based on prediction of multiple resource utilization for power efficient data centers. There was always a risk of SLA violation and resource wastage due to prediction errors. To avoid that problem, Yu *et al.* [18] presented stochastic load balancing, in which probabilistic distribution of prediction errors were padded to the predicted output. In addition, to handle improper VM migrations occur due to overloaded servers and inefficient resource utilization,

hotspot (overloaded servers) were identified and a heuristic algorithm was proposed to decide VM migrations from hotspot to underutilized servers and fairly allocate VMs on available servers. Later, an energy-efficient VM prediction and migration framework was proposed in [4] where Wiener filter was used for prediction of resource usage and EWMA based safety margins were appended to the predicted output to avoid the SLA violation due to errors in prediction. An Online VM placement for raising cloud provider's revenue was proposed in [19] that employed First-Fit and Harmonic Algorithms(HA) for online VM placement and Decreased Density Greedy algorithm to handle SLA violation where HA outperformed FF for energy-efficient VM placement. Recently, Saxena *et al.* [20] have proposed a workload and security threats prediction based resource allocation model that considers network traffic and previous resource utilization before workload distribution on user VMs.

2.2 Overload Handling Techniques

VMs migration have been used in [21], [22] after detection of server overloads. This approach causes delay in execution of user's application and leads to SLA violation. To overcome this limitation, threshold-based VM migration is presented in [23] where migration process is triggered before actual occurrence of overload on the basis of threshold value of server utilization. The threshold value can be set statically or depending upon workload fluctuations [23]. However, this technique might trigger unnecessary VM migrations because exceeding threshold value does not always lead to overloading of server and also under-utilized servers are not addressed that causes wastage of resources and power. To tackle these limitations, overload avoidance techniques were introduced on the basis of prediction of resource demands in [4], [17]. The prediction helps to estimate unseen over/under-load situation and decide VM migration. Energy aware VM migration is presented in [4], [24]. Dabaggh *et al.* [4] considered the energy consumed during VM migration and suggested to migrate VM from overloaded server to already active server and turn-ON selected inactive server only if migration is impossible on all the active servers. However, they ignored the traffic management and network bandwidth cost required during VM migration. Meng *et al.* [25] studied the effect of network resources while optimizing VM migration on host machines [26] and proposed a two-tier approximation algorithm to solve traffic-aware VM mapping problem that resulted into increased throughput and decreased communication cost.

Unlike existing works, OP-MLB framework provides a pragmatic solution of complex and challenging elastic resource management problem by developing and incorporating all the needed operations at unified platform and allowing interaction among them to optimize and tune/learn together to bring overall performance improvement of cloud data center.

3 Framework Overview

The proposed OP-MLB framework incorporates three phases viz. task assignment, online VM prediction and load balancing to provide concrete and optimal solution to the problem of elastic resource management in cloud environment as shown in Fig. 2. These three phases work in collaboration by interacting with each other. In the *first phase*, applications $\{Application_1, Application_2, \ldots, Application_M\}$ submitted by the users $\{User_1, User_2, \ldots, User_M\}$ are divided into sub-units called as tasks $\{T1, T2, \ldots, Tz\}$, which are assigned to computing instances' called as VMs $\{VM_1, VM_2, \ldots, VM_Q\}$ for execution. The application model considered in the proposed framework is made up of a collection of independent tasks and it is known by many names: Bag of Tasks (BoT), Task Farming, Parameter Sweep, and SPMD (Single Program and Multiple Data) model. The applications created using this model are very common in scientific and commercial domains such as molecular docking for drug discovery [27] and investment risk analysis [28].

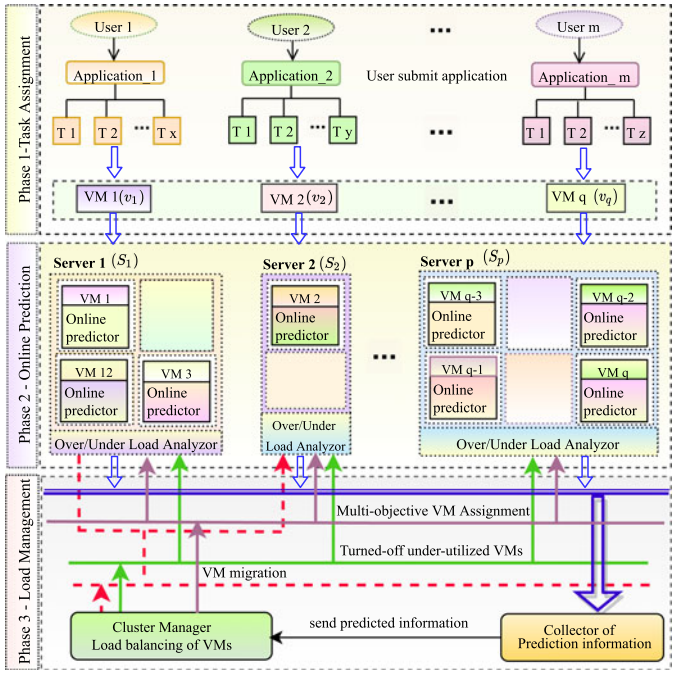


Fig. 2. OP-MLB framework.

During *second phase*, VMs loaded with user's task perform task execution as well as predict resource utilization for the next interval concurrently. The allocated VMs are shown in filled blocks whereas vacant blocks specify de/un-allocated VMs. Every active VM carries distinct online-predictor depending on its configuration to forecast resource utilization information. Additionally, each server has its own load analyzer, that aggregates the predicted resource usage information of each VM and detects the occurrence of over/under-load beforehand. If the predicted resource usage is greater than the resource capacity of the server, then overload is detected and accordingly needed operations are performed to mitigate its effect in third phase.

In the *third phase*, load balancing is achieved by performing VM assignment and migration to improve resource utilization by effective handling of over/under-loaded servers. The load analyzer informs cluster manager about the expected load at respective server in the cluster periodically, which utilizes this information during VM allocation on the selected server by applying the multi-objective VM

placement optimization. The over/under-load situation detected by the online predictor is applied to perform necessary VM migration and balance the entire workload in the cluster. All the idle servers are shut-down to scale down the power consumption. The detailed description of each phase is given in subsequent sections.

4 Task Assignment

Consider a data center having P servers such as $S \in \{S_1, S_2, \ldots, S_P\}$, where M users have purchased different types of VMs to execute their applications on Q VMs such as $v \in \{v_1, v_2, \ldots, v_Q\}$. Suppose an application A_M of M^{th} user is represented as $\{T_1, T_2, \ldots, T_z\} \in A_M$, where T_z denotes z^{th} task of the application. The tasks are scheduled on the basis of their resource requirement by applying Eq. (1) that selects most appropriate VM for execution of i^{th} task (T_i^r) where, r represents resources like CPU, memory etc. and v_S^r, v_M^r, v_L^r and v_{XL}^r are small, medium, large and extra-large type of VMs. The type of VMs can be extended to any number of type offered at particular data center. If the resource requirement of i_{th} task (T_i^r) is lesser or equals to the resource capacity of v_S, then small type of VM is assigned to it. Likewise, if $(v_S^r < T_i^r \leq v_M^r)$, then v_M is selected for the task execution and so on

$$VM_{sel.} = \begin{cases} v_S, & (T_i^r \leq v_S^r) \\ v_M, & (v_S^r < T_i^r \leq v_M^r) \\ v_L, & (v_M^r < T_i^r \leq v_L^r) \\ v_{XL}, & (otherwise.) \end{cases} \quad r \in CPU, mem. \tag{1}$$

SLA is a mutual agreement between the CSP and the cloud client that guarantees availability (A_g) of services, throughput and response time etc. The negotiated SLA terms are monitored with respect to 'availability', which is computed as shown in Eq. (2) where A_o and A_g are availability offered and availability guaranteed respectively. The variables a and b are uptime and downtime of server respectively. The uptime means server is available for task execution on allocated VM and downtime states failure of task execution due to unavailability of server. Therefore, SLA violation (SLA_v) can be defined using Eq. (3)

$$A_o = 1 - \frac{b}{a} \tag{2}$$

$$SLA_v = \begin{cases} No, & A_g \leq A_o \\ Yes, & (otherwise.) \end{cases}. \tag{3}$$

The SLA violation experienced due to unavailability of small, medium, large and extra-large types of VMs are represented as $v_S^{SLA_v}$, $v_M^{SLA_v}$, $v_L^{SLA_v}$ and $v_{XL}^{SLA_v}$ respectively. The realized penalty cost depends on amount of SLA violation i.e., difference between A_g and A_o and relative standard penalty cost associated with small (Ω_S), medium (Ω_M), large (Ω_L) and extra-large (Ω_{XL}) type of VMs as shown in Eq. (4)

$$penalty = \begin{cases} (A_g - A_o) \times \Omega_S, & v_S^{SLA_v} \\ (A_g - A_o) \times \Omega_M, & v_M^{SLA_v} \\ (A_g - A_o) \times \Omega_L, & v_L^{SLA_v} \\ (A_g - A_o) \times \Omega_{XL}, & v_{XL}^{SLA_v} \end{cases}. \tag{4}$$

5 Online Resource Predictor

Consider a neural network (NN) represented as $p - q - r$, where p, q and r stands for a number of neurons in input, hidden and output layers respectively and NN connection weights are represented as Φ which are randomly generated in the range of [0, 1] as shown in Fig. 3. It receives combination of historical and live resource usage information from its host VM which is normalized to generate and feed p input values such as $\{d_1, d_2, \ldots, d_p\}$ into input layer of NN. The training of NN begins with the randomly generated N different neural networks each of size $L = (p+1) \times q + (q \times r) = q(p + r + 1) \Rightarrow q(p+2)$ as $r = 1$. One additional input is added to input neurons as a bias value. It forecasts VM resource utilization by extracting and correlating the patterns generated from the input data during learning/training process. Since the workload arrival at VM is dynamic, an Auto Adaptive Differential Evolution (AADE) learning algorithm is developed for dynamic and adaptive optimization of network weights.

AADE algorithm consists of five key operations viz. initialization, evaluation, mutation, recombination (or crossover) and selection. It begins with initialization of N networks of size L, number of maximum generations (G_{max}), mutation rate (μ) and crossover rate (Cr). These networks are evaluated by applying a fitness function i.e., Root Mean Squared Error (RMSE) as stated in Eq. (5) where m is number of data samples, z_a and z_p are actual and predicted outputs (or resource utilization) respectively. Since accuracy is inversely proportional to RMSE, the learning objective is to minimize the fitness function

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(z_a - z_p)^2}. \tag{5}$$

In order to explore the search space in multiple directions and generate new solutions with better fitness value, mutation and crossover operators are applied during iterative optimization process. Three mutation strategies opted for proposed work are: $DE/best/1 \quad (MS_1)$, $DE/current-to-best/1 \quad (MS_2)$ and $DE/rand/1 \quad (MS_3)$. The mutation strategies MS_1, MS_2 shown in Eqs. (6) and (7) tend to be greedy as they exploit the best individual to generate mutant vectors while MS_3 stated in Eq. (8) is applicable for raising population diversity,

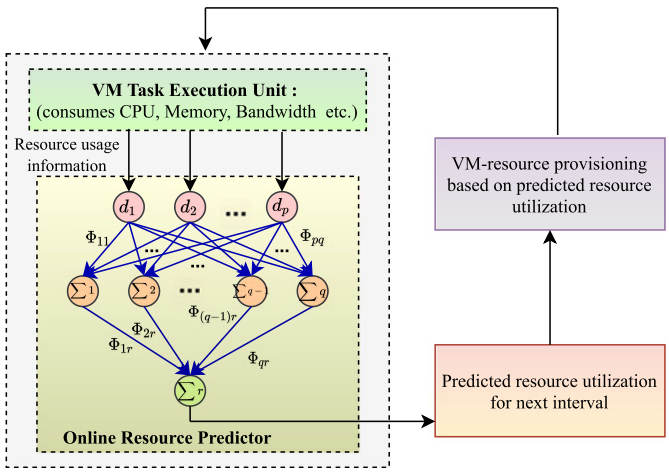


Fig. 3. Online VM resource predictor system.

$$\Lambda_i^j = \Phi_{best}^j + \mu_i \times (\Phi_{r1}^j - \Phi_{r2}^j) \tag{6}$$

$$\Lambda_i^j = \Phi_i^j + \mu_i \times (\Phi_{best}^j - \Phi_i^j) + \mu_i \times (\Phi_{r1}^j - \Phi_{r2}^j) \tag{7}$$

$$\Lambda_i^j = \Phi_{r3}^j + \mu_i \times (\Phi_{r1}^j - \Phi_{r2}^j), \tag{8}$$

where μ is mutation rate, Λ_i^j and Φ_i^j depict i^{th} mutant and current vectors respectively, of j^{th} iteration. The term Φ_{best}^j is the best solution found till j^{th} generation and $r1$, $r2$ and $r3$ are mutually distinct random numbers in the range [1, N]. To decide the mutation scheme for current iteration, a random vector i.e., mutation selection probability (msp) is generated in the range [0, 1] for each network and updated during learning process. The mutation scheme selection parameter ($\wp$) chooses most suitable mutation scheme based on the value of msp_i (where $1 \leq i \leq N$) as given in Eq. (9), where Γ_1, Γ_2 and Γ_3 are the probabilities for opting the MS_1, MS_2 and MS_3 mutation schemes respectively.

$$\wp = \begin{cases} MS_1, & If(0 < msp_i \leq \Gamma_1) \\ MS_2, & If(\Gamma_1 < msp_i \leq \Gamma_1 + \Gamma_2) \\ MS_3, & \text{otherwise} \end{cases}. \tag{9}$$

In reported experiments, initially $\Gamma_1 = \Gamma_2 = 0.33$, $\Gamma_3 = 0.34$, so that each mutation scheme get equal chance of selection. Later on, these probabilities are updated by computing the ratio of total number of successful candidates which have reached next generation through respective mutation strategy and total number of candidates generated.

Thereafter, uniform crossover is applied to mutant vector (Λ_i^j) and its corresponding current target vector (Φ_i^j) to produce new solutions called as offspring (χ_i^j) by applying Eq. (10) where $\Re$ is randomly generated number in the range [0, 1] for each gene in the chromosome (or vector). The crossover-rate is randomly generated in the range [0, 1] with mean value of 0.5 and standard deviation 0.1 during each generation. If the value of $\Re$ is smaller than i^{th} gene crossover-rate (Cr_i^j), crossover is successful and i^{th} genes swap between mutant and current vectors, else previous solution proceeds in the next generation

$$\chi_i^j = \begin{cases} \Lambda_i^j & If(\Re \in (0,1) \leq Cr_i^j) \\ \Phi_i^j & \text{otherwise.} \end{cases} \tag{10}$$

Finally, the successful candidates are selected on the basis of fitness value (i.e., least RMSE score) using Eq. (5). The population for next generation is selected by applying survival of fittest concept stated in Eq. (11), where Φ_i^{j+1} is selected candidate for next generation, χ_i^j is solution generated after crossover and Φ_i^j is current solution. Algorithm 1 gives operational summary of AADE training process for online resource prediction

$$\Phi_i^{j+1} = \begin{cases} \chi_i^j & (fitness(\chi_i^j) \leq (fitness(\Phi_i^j)) \\ \Phi_i^j & (otherwise.) \end{cases}. \tag{11}$$

Error-Driven Padding (EDP). Although the proposed on-line predictor is capable of anticipating resource demands with closer precision, still 100 percent accuracy cannot be ensured for highly dynamic resource demands. These errors may cause over/under-load and SLA violations. In order to overcome the occurrence of these issues, error-driven precaution margins are padded with predicted resource demand. At t_{th} instance, EDP is computed as $EDP_t = (1-\rho) \times EDP_{t-1} + \rho \times RMSE$ where $0.5 < \rho \leq 1$. Furthermore, it is to be noted that during EDP computation, more weightage is given to most recent error to improve the accuracy of prediction. Therefore, the improved predicted output becomes $z_p + EDP_t$.

Algorithm 1. AADE Training for Online Prediction ( )

```
Initialize Cr, μ, G_max, Γ1=Γ2=0.33, Γ3=0.34
Initialize N networks of size L randomly.
Evaluate each network on training data by computing Eq. (5)
for each generation j^th ∈ G_max do
  Generate vector msp for N networks ∈ [0,1]
  for each i^th network do
    Generate r1 ≠ r2 ≠ r3 ≠ i ∈ [1, N]
    if 0 < msp_i ≤ Γ1 then
      Apply Eq. (6)
    else if Γ1 < msp_i ≤ (Γ1 + Γ2) then
      Apply Eq. (7)
    else
      Apply Eq. (8)
    end if
    Perform uniform crossover
  end for
  Evaluate updated network by using Eq. (5)
  Select participants for next generation using Eq. (11)
  Update Γ1, Γ2, Γ3 after fixed number of generations
end for
Select network with least RMSE to predict output (z_p)
return z_p
```

6 Load Management

Load balancing includes two main processes (i) VMs placement on servers and (ii) VM migration, discussed in the following subsections.

6.1 VM Placement

Assume S_i^C and S_i^M are CPU and memory capacity of i^{th} server. If server S_i is active then $\gamma_i = 1$, means one or more VMs are placed on it, otherwise, it is idle ($\gamma_i = 0$). If server S_i hosts VM v_j, then $\omega_{ji} = 1$ else it is 0. For j^{th} VM (v_j), CPU and memory utilization are represented as v_j^C and v_j^M respectively. The CSP always seeks an optimal VM placement that maximize resource utilization and minimize power consumption and network traffic to achieve maximum return-on-investment (ROI). Therefore, multi-objective VM placement problem is formulated as stated in Eq. (12)

$$\sum_{i=j=1}^{P,Q} \omega_{ji} = Minf\left(\sum_{i=j=1}^{P,Q} \omega_{ji} \times (PW, Com, -RU)\right), \tag{12}$$

where PW, Com and RU are power consumption, communication cost and resource utilization respectively of j^{th} server. Each VM allocation is feasible only if it satisfies the following constraints given in Eq. (13)

$$\sum_{j=1}^{Q} v_j^r \times \omega_{ji} \leq S_i^r \quad r \in CPU, Mem. \tag{13}$$

To accomplish this multi-objective VM placement, following objective models are designed:

6.1.1 Resource Utilization

The resource utilization of data center can be obtained using Eqs. (14), (15), where R is number of resources. Though in formulation, only CPU (C) and memory (M) are considered, it is extendable to any number of resources

$$RU_{dc} = \int_{t_1}^{t_2} \left(\frac{RU_{dc}^C + RU_{dc}^M}{|R| \times \sum_{i=1}^{P} \gamma_i} \right) dt \tag{14}$$

$$RU_{dc}^r = \sum_{i=1}^{P} \frac{\sum_{j=1}^{Q} \omega_{ji} \times v_j^r}{S_i^r} \quad r \in CPU, Mem, etc. \tag{15}$$

6.1.2 Communication Cost

Let β_{ik} denotes mapping between server S_i and user u_k, if a user (u_k) owns VMs on server (S_i), then $\beta_{ik} = 1$ otherwise 0. The total number of users having inter-dependent VMs are computed as $\sum_{k=1}^{M} \sum_{i=1}^{P} \beta_{ik}$. The communication cost model is shown in Eq. (16) where $|U|$ is total number of active users and Com_{dc} is communication cost of data center which is minimized. The inter-dependent VMs are preferably allocated at minimum number of servers to reduce communication cost among inter-dependent VMs

$$Com_{dc} = \int_{t_1}^{t_2} \left(\frac{\sum_{k=1}^{M} \sum_{i=1}^{P} \beta_{ik}}{|U|} \times 100 \right) dt; \quad \forall \sum_{i=1}^{P} \beta_{ik} > 1. \tag{16}$$

6.1.3 Power Consumption

From existing literature [29], power consumption for i^{th} server can be formulated as PW_i and total power consumption PW_{dc} during time-interval $[t_1, t_2]$ is shown in Eq. (17)

$$PW_{dc} = \int_{t_1}^{t_2} \left(\sum_{i=1}^{P} ([{PW_i}^{max} - {PW_i}^{min}] \times RU + {PW_i}^{idle}) \right) dt, \tag{17}$$

where RU is resource utilization, ${PW_i}^{max}$, ${PW_i}^{min}$ and ${PW_i}^{idle}$ are maximum, minimum and idle state power consumption for i^{th} server.

6.1.4 Optimized VM-Allocation Approach

The proposed multi-objective VM allocation approach consists of four consecutive stages namely initialization, fitness evaluation, crossover and selection as shown in Fig. 4.

The VM allocations are represented as chromosomes (Ψ) and the step-by-step procedure is given in Algorithm 2. First, n random VM allocations are initialized as Ψ_i (step 1) which represents i^{th} VM placement, subject to $(i \leq n)$. To evaluate fitness of each chromosome, cost function $\eta(\Psi^g)$ is computed, where $f_{\Psi_g}^{RU}$, $f_{\Psi_g}^{Com}$, $f_{\Psi_g}^{PW}$ are cost values associated to resource utilization, communication cost, power consumption and can be evaluated by computing Eqs. (14), (16) and (17) respectively (step 2). A pareto-front module is called in step 3 to generate

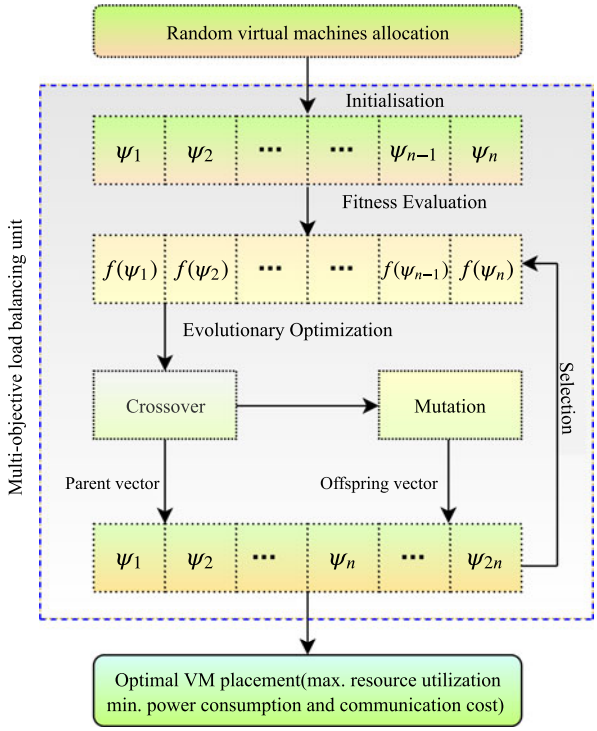


Fig. 4. Multi-objective VM allocation strategy.

Ψ_{front} i.e., chromosome with best fitness value that satisfies each objective non-dominantly using concept of NSGA-II [30]. The Ψ_i dominates Ψ_j if its cost values are better than Ψ_j on atleast one objective and same or better on other objectives. Further, steps 5-12 repeat for all n chromosomes of each generation where one-point crossover (Cr) and mutation (μ) operations are applied to generate new offspring in steps 6-9, to explore the entire search space for better solution by migrating VMs from non-optimal to selected optimal server, where cp is randomly generated crossover-point. The resultant solutions may be infeasible with respect to VM allocation constraints, which are turned into feasible solutions by re-arranging them in step 10 and multiple objective cost function is evaluated in step 11. Again, the fitness of offspring solutions (Ψ_{Off}), are evaluated (step 13) and optimal solutions for next generation (Ψ_{g+1}) are selected by calling $Pareto - front()$ in step 14.

Algorithm 2. VM Placement ()

1: Initialize n random VM allocations $\{\Psi_1, \Psi_2, \ldots, \Psi_n\} \in \Psi$
2: $[f_{\Psi}^{RU}, f_{\Psi}^{Com}, f_{\Psi}^{PW}] = \eta(\Psi)$
3: $[\Psi_{front} = Pareto - front(\Psi)]$, $\Psi_{best} \leftarrow \Psi_{front[0]}$
4: **for** $g = 1, 2, \ldots, G_{max}$ **do**
5: **for** each i=(1,2,...,n) **do**
6: $rn = random(1, n)$, $cp = random(1, P)$
7: $\Psi_{Cr_1} = [\Psi_i(1:cp), \Psi_{rn}(cp+1:p)]$
8: $\Psi_{Cr_2} = [\Psi_{rn}(1:cp), \Psi_i(cp+1:p)]$
9: $\Psi_{Off} = [\Psi_{Cr}, \mu(\Psi_{Cr_1}), \mu(\Psi_{Cr_2})]$
10: Ψ_{Off}=Feasible VM Allocation(Ψ_{Off})
11: $[f_{\Psi_{Off}}^{RU}, f_{\Psi_{Off}}^{Com}, f_{\Psi_{Off}}^{PW}] = \eta(\Psi_{Off})$
12: **end for**
13: $\Psi_g = [\Psi_g, \Psi_{Off}]$
14: $[\Psi_{g+1} = Pareto - front(\Psi_g)]$
15: **end for**

6.2 VM Migration

The VMs allocation is re-optimized during consecutive prediction intervals by migrating VMs from predicted under/over-utilized to optimal servers. The cost of VM migration mainly depends on size of VM and network bandwidth [31]. In OP-MLB, largest resource capacity VM is chosen for migration from overloaded server to allow efficient resource utilization while mitigating the effect of current overload and need of VM migration in the near future. To allow network aware VM migration, inter-dependent VMs are shifted closer to each other which reduces total network traffic and VM migration cost. The selection of destination server is energy-efficient also because it first tries to shift migrating VM (v_{mig}) to already active server in closer location and switch to nearby inactive server only if the migrating VM cannot be shifted to any active server. The necessary constraint for VM migration is that required resource capacity of v_{mig} must be lesser than available resource capacity of destination server S_j as defined in Eq. (18), where, r represents resources like CPU, memory

$$v_{mig}^{r} \leq S_j^{r} \quad r \in CPU, Mem. \tag{18}$$

Moreover, to further reduce power consumption, the VMs running on under-utilized servers migrate to near-by optimal server and idle servers are shutdown. It allows consolidation of VMs with reduced power consumption and network communication cost. The total migration cost M_{cost} for period between t_1 and t_2 can be computed using Eq. (19)

$$M_{cost} = \int_{t_1}^{t_2} \Big(\sum c_{mig.j} * (D(S_k, S_j) * W(v_{mig})) + \sum n_j * d_j \Big) dt, \tag{19}$$

where $v_{mig} \in O_v$, $j \in [1, P]$, $D(S_k, S_j)$ is the distance or number of hops covered by v_{mig} from source (S_k) to destination server S_j, $W(v_{mig}) = v_{mig}^{C} \times v_{mig}^{M}$, is the size of migrating VM, O_v is the list of VMs on overloaded server (S_k). The first term $\sum c_{mig.j} * D(S_k, S_j) * W(v_{mig})$ signifies network energy consumed during VM migration. The second term $\sum n_j * d_j$ specifies server state transition energy, where if i^{th} VM is placed at j^{th} server after migration then $c_{mig.j} = 1$, otherwise, $c_{mig.j} = 0$. If j^{th} server receives one or more VMs after migration, then $n_j = 1$ else it is 0. Similarly, if $d_j = 0$ then j^{th} server is already active before migration, otherwise, $d_j = E_{tr}$ where E_{tr} is energy consumed in switching a server from sleep to active state. In proposed work, the value of E_{tr}= 4260 Joules as stated in [32]. Algorithm 3 provides operational steps of VM migration.

Algorithm 3. VM Migration (O_S)

1: **for** each j^{th} server $S_j \in O_S$ **do**
2: Sort VMs in ascending order of resource capacity $CPU \times Mem$
3: $v_{mig} \Leftarrow$ Pick the best suited VM from sorted list to mitigate the effect of overload
4: **for** each i^{th} server S_i **do**
5: **if** status((S_i))='ACTIVE' AND Eq. (18) satisfies **then**
6: Shift v_{mig}toS_i; Update status(v_{mig})='Assigned'
7: **end if**
8: **end for**
9: **if** status(v_{mig}) $\neq$'Assigned' **then**
10: **for** each k^{th} server S_k **do**
11: Shift v_{mig} to nearby SLEEPING server that satisfy Eq. (18); status((S_k))='ACTIVE' and status(v_{mig}) ='Assigned'
12: **end for**
13: **end if**
14: Compute Migration cost using Eq. (19)
15: **end for**

Algorithm 4. Proposed Elastic Resource Management: Operational Summary()

1: **for** each i^{th} VM on S and $\{S_1, S_2, \ldots, S_P\} \in S$ **do**
2: $\{v_i^{pred.C}, v_i^{pred.M}\} \Leftarrow$ Algorithm 1()+EDP
3: **end for**
4: CALL VM placement()
5: **for** t^{th} time-interval **do**
6: **for** each j^{th} server S_j **do**
7: **for** each i^{th} VM on S_j **do**
8: $\{v_i^{pred.C}, v_i^{pred.M}\} \Leftarrow$ Algorithm 1()+EDP
9: $S_j^C = v_i^{pred.C} + S_j^C$, $S_j^M = v_i^{pred.M} + S_j^M$
10: **end for**
11: **if** $S_j^C.Max\_threshold \leq S_j^C$ OR $S_j^M.Max\_threshold \leq S_j^M$ **then**
12: Server S_j is over-loaded. $O_S \Leftarrow S_j$
13: **else if** $S_j^C.Min\_threshold \geq S_j^C$ OR $S_j^M.Min\_threshold \geq S_j^M$ **then**
14: Server S_j is under-loaded, shift all VMs to nearby ACTIVE servers subject to constraints mentioned in Eq. (18) and shutdown it.
15: **end if**
16: CALL VM-migration(O_S)
17: **end for**
18: **end for**
19: Repeat above steps for next time-interval

7 OP-MLB: Main Algorithm and Complexity

Algorithm 4 depicts the operational summary of OP-MLB framework which executes periodically to process elastic resource management. In steps 1-3, resource usage of each VM is predicted by calling Algorithm 1, which provides training steps for NN predictor whose time complexity depends on size of network (L), number of networks (N), number of input nodes (p) and G_{max} that becomes $O(p^2NLG_{max})$.

Step 4 calls multi-objective VM placement module, provided in Algorithm 2 that works on number of solutions (n), generations (G_{max}), servers (P), VMs (Q) that calls Pareto-optimal() based on NSGA-II, having time complexity of $O(n^2 \times o)$ where o is number of objectives. Hence, overall time complexity is $O(on^2PQ(G_{max}))$. Steps 5-18 handle over/under-load by calling VM migration i.e., Algorithm 3. Steps 11 and 12 prepare list of overloaded servers by using predicted load information, given by steps 7-10. The steps 13-15 detect and handle under-loaded servers and step 16 calls Algorithm 3 to handle overloaded servers. The time consumption depends on number of over-loads (O_S), number of servers (P) which becomes $O(PO_sQ) \Rightarrow O(P^2Q)$ as

$O_s << P$. Therefore, the total time complexity can be computed as $O(on^2N^2P^2Q(G_{max})L)$.

8 Performance Evaluation

8.1 Experimental Set-Up

The simulation experiments are executed on a server machine assembled with two Intel® Xeon® Silver 4114 CPU with 40 core processor and 2.20 GHz clock speed. The computation machine is deployed with 64-bit Ubuntu 16.04 LTS, having main memory of 128 GB. The data center environment was set up with three different types of server and four types of VMs configuration shown in Tables 2 and 3 in Python version-3. The resource features like power consumption (P_{max}, P_{min}), MIPS, RAM and memory are taken from real server IBM [33] and Dell [34] configuration where S_1 is 'ProLiantM110G5XEON3075', S_2 is 'IBMX3250Xeonx3480' and S_3 is 'IBM3550Xeonx5675'. Furthermore, the experimental VMs configuration are inspired from the VM instances of Amazon website [35].

The resource utilization for different VMs followed the traces from three publicly available real workloads including Google Cluster Data (GCD), PlanetLab VMs (PL) and Bitbrains (BB) dataset. GCD has resources CPU, memory, disk I/O request and usage information of 672,300 jobs executed on 12,500 servers for the period of 29 days [2]. The CPU and memory utilization percentage of VMs are obtained from the given CPU and memory usage percentage for each job in every five minute over period of twenty-four hours. PL contains CPU utilization of more than 11K VMs measured every five minutes during ten random days in March-April, 2011 [36]. BB consists of performance metrics of fast storage 1,750 VMs from a distributed data center over period of 30 days [37]. It gave information about CPU usage percentage, disk I/O etc. We extracted CPU and memory usage percentage from GCD and CPU percentage from PL and BB as per their availability for various experiments. The experiments are conducted for different size of data center over a period of 24 hours for GCD and PL and 30 days for BB. Following performance metrics are evaluated:

- Accuracy of Predicted vs Actual Workload.
- Resource utilization and power consumption.
- Communication cost percentage.
- Number of predicted and unpredicted overloads.
- VM migration cost incurred during load balancing.

TABLE 2
Server Configuration

Server	PE	MIPS	RAM(GB)	Memory(GB)	PW_{max}	PW_{min}/PW_{idle}
S_1	2	2660	4	160	135	93.7
S_2	4	3067	8	250	113	42.3
S_3	12	3067	16	500	222	58.4

TABLE 3
VM Configuration

VM type	PE	MIPS	RAM(GB)	Memory(GB)
v_{small}	1	500	0.5	40
v_{medium}	2	1000	1	60
v_{large}	3	1500	2	80
v_{Xlarge}	4	2000	3	100

- Number of active servers.
- SLA compliance in terms of availability of servers.

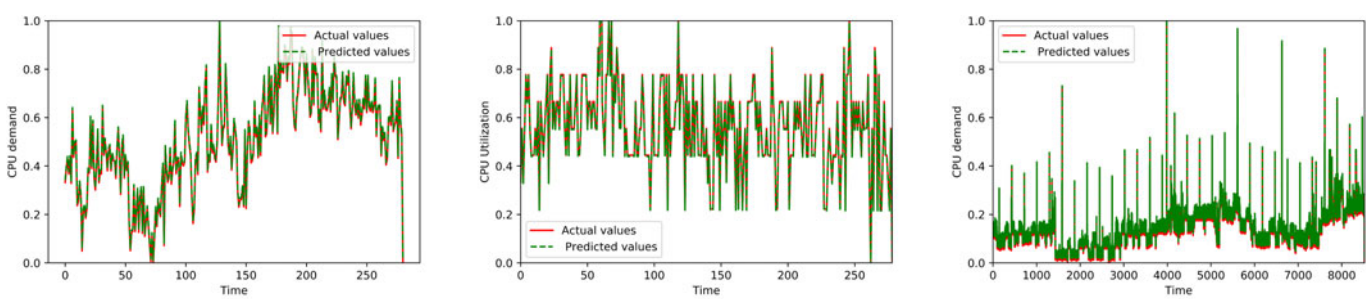
The proposed work is compared for different performance metrics with various state-of-art approaches including [4], [17], First-Fit ([38], [39]), Random-Fit ([40]) and Best-Fit ([36], [41]) heuristics. Furthermore, overall power saving achieved with OP-MLB framework with over-subscription compared to non-oversubscribed cloud is also presented.

8.2 Results

The performance evaluation of the proposed framework starts with investigation of accuracy of online-prediction system. Its effectiveness can be seen in Fig. 5 where predicted CPU usage have almost overlapped actual resource usage for GCD, PL and BB workloads. The error score of proposed prediction approach for prediction interval of 5 minutes on three workloads is shown in Table 4.

The reason behind such an accuracy is that proposed neural network based predictor periodically learns and retrains itself according to changes in live and historical workload. In addition, the application of AADE algorithm works on N number of solutions which explores the search space in multiple directions and allows efficient learning of patterns and correlations from live data which is responsible for near accurate prediction of resources.

To analyze the performance of multi-objective load balancing, numerous experiments were conducted based on different combination of VMs and servers. The experiments are executed with the ratio of number of VMs and servers



(a) Google Cluster CPU (duration=24 Hrs) (b) PlanetLab VM CPU (duration=24 Hrs) (c) Bitbrains CPU (duration=30 days)

Fig. 5. Predicted versus actual workload.

TABLE 4
RMSE for Different Workloads

Interval	GCD (CPU)	GCD (Memory)	PL (CPU)	BB (CPU)
5 min	0.0014	0.0035	0.0005	0.0031

TABLE 5
Performance Metrics for Google Cluster Dataset

VMs (Users)	RU(%)	PW	Com(%)	M_{cost} (KJ)	$SLAv$ $(10^{-3})(\%)$
200 (120)	64.98	8.80E+03	11.78	15.4	4.53
400 (240)	64.57	1.96E+04	12.8	169	2.44
600 (360)	64.05	2.17E+04	12.23	294	2.25
800 (480)	63.34	3.31E+04	12.34	278	3.22
1000 (600)	63.87	3.59E+04	12.32	306	5.52
1200 (720)	63.32	3.92E+04	12.25	384	3.93
1400 (560)	63.09	4.13E+04	12.4	404	5.63

TABLE 6
Performance Metrics for Planet Lab VM Traces

VMs (Users)	RU(%)	PW	Com(%)	M_{cost} (KJ)	$SLAv$ $(10^{-3})(\%)$
200 (120)	68.98	5.64E+03	10.14	55.7	2.69
400 (240)	69.57	1.94E+04	11.438	346	3.52
600 (360)	69.05	2.14E+04	11.43	389	3.24
800 (480)	68.34	3.14E+04	10.34	364	1.52
1000 (600)	69.81	3.75E+04	10.62	398	2.04
1200 (720)	68.32	3.96E+04	11.25	414	2.15
1400 (840)	69.19	4.13E+04	10.4	442	2.99

as 1:1. It is mentioned that resource utilization (RU) percentages per VM extracted from the three workloads are used to compute the actual resource usage of VMs under simulation. For instance, if real workload trace shows 67.3 percent of CPU usage and VM has 500 MIPS of CPU capacity then actual CPU usage of the VM is given by the product of 67.3 percent and 500. The number of users are not mentioned in the original datasets, therefore, we created random set of users, who requested different number and type of VMs to compute communication cost (Com) based on the location of inter-dependent VMs. The number of users are taken as 60 percent of size of the data center, where each user can hold VMs in the range between 0 and 5 with a constraint that at any instance, the total number of VM requests must not exceed total number of VMs of data center. Each experiment was executed for 12-15 times and a mean of the obtained results are reported. In the simulation, various performance metrics were analyzed and results are shown in Tables 5, 6 and 7 for GCD, PL and BB workloads respectively. The values of minimum and maximum threshold of CPU usage were 10 and 89 percent respectively for these experiments. The resource utilization for each workload is more than 63 percent, which varies between 63-64.8 percent for GCD, 68-69.8 percent for PL and 63-65.6 percent for BB workloads. The power consumption (PW) has increased with respect to the size of data center. The power consumption shows various trends for the three different datasets, depending upon the number of busy and idle servers. Since the communication cost depends upon the placement of

TABLE 7
Performance Metrics for Bitbrains Workload

VMs Users	RU(%)	PW	Com(%)	M_{cost} (KJ)	$SLAv$ $(\times 10^{-3})(\%)$
200 (120)	63.18	1.37E+03	15.78	62.4	3.12
400 (240)	63.57	2.84E+04	13.8	243	2.44
600 (360)	64.05	3.20E+04	13.23	267	3.85
800 (480)	64.34	3.31E+04	15.34	378	4.62
1000 (600)	65.87	4.59E+04	16.32	316	4.52
1200 (720)	64.32	6.92E+04	13.25	484	3.93
1400 (840)	64.32	8.20E+04	14.4	554	6.83

inter-dependent VMs of active users, each workload shows similar values in the range 12-16 percent. It can be observed from the Tables 5, 6 and 7 that resource utilization depends upon number of active servers shows slight change with different size of data center for each dataset. However, power consumption and VM migration costs are increasing with respect to the size of the data center. The values for SLA compliance are varying according to the availability/non-availability of servers.

The pareto-front or non-dominated solutions for 1,400 VMs placement is shown in Fig. 6 that depicts the contradictory behavior of three optimization variables viz. maximization of RU, minimization of PW and Com.

Fig. 7 shows the predicted and unpredicted overloads for all three datasets. It is noted that there is an increase in the number of correctly predicted overloads on the respective servers with increase in size of data center for all the three workloads. However, the number of unpredicted overloads are either lesser or equal to 0.07 percent independent of size of the data center for each experiment of every dataset. This is due to the efficiency of prediction system that accurately forecasts the future resource requirement. The overload prediction accuracy is around 99.94 percent for GCD during the period of 24 hours. Similarly, more than 99.91 and 99.97 percent overloads are correctly forecasted for PL and BB respectively. As a result, the number of VM migrations and SLA violations get reduced during actual VMs allocation which can be observed in Fig. 8. Since SLA violations depends upon the availability of the server, they are indirectly varying with respect to the number of unpredicted overloads for different size of the data center. The number of unpredicted overloads prompt VM migration and unavailability of server and hence account for SLA violation.

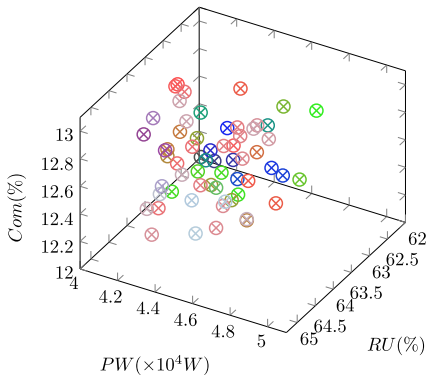
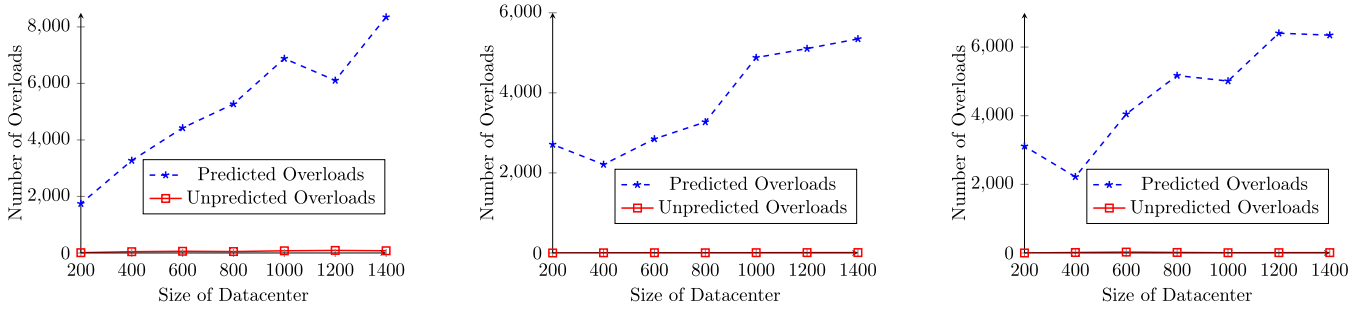
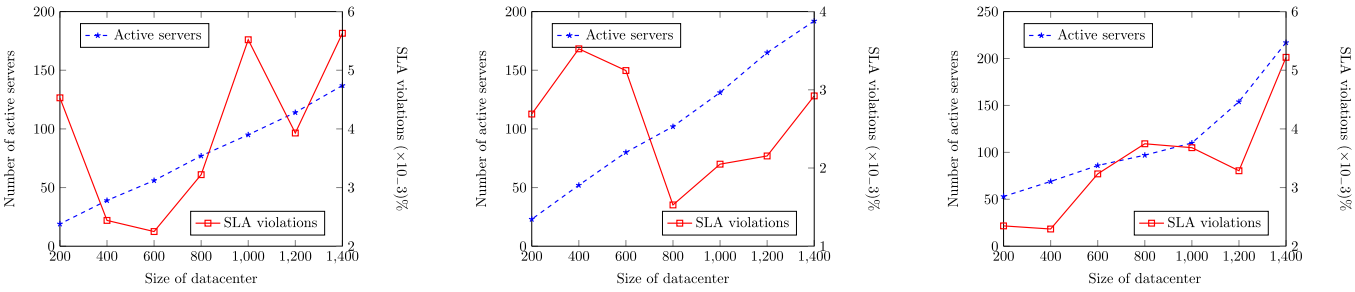


Fig. 6. Multi-objective Pareto Front for 1400 VMs with GCD.



(a) Google Cluster Data (b) PlanetLab VM traces (c) Bitbrains VM traces

Fig. 7. Predicted versus Unpredicted overloads for different workloads.



(a) Google Cluster Data (b) PlanetLab VM traces (c) Bitbrains VM traces

Fig. 8. Analysis of proposed framework on average number of active physical machines per data center for different workloads.

8.3 Comparative Analysis

The comparison of unpredicted overloads among proposed and existing approaches [4] for over-subscribed cloud is shown in Fig. 9. The reason for such comparison is that the experiments were conducted using same dataset i.e., GCD and experiments are executed for same period of 24 hours. They applied Wiener Prediction (WP) scheme for overload prediction, separately on each VM (similar to proposed approach). Therefore, it is suitable for comparison of the number of overload predicted by proposed prediction (PP) approach. It can be seen that the number of unpredicted overloads have significantly reduced in case of PP as compared to Wiener prediction (WP) with safety margin (SM) i.e., (WP+SM) and other existing approaches mentioned in [4]. The unpredicted overloads are further reduced when PP was combined with EDP.

The total number of active servers are compared with another recent work [17]. The reason for such comparison is that this work used same datasets viz. GCD and PL and their entire experimental set-up including size of different data center, types and configuration of VMs and physical servers are similar to our OP-MLB Framework. Fig. 10 entails the comparison of total number of active servers

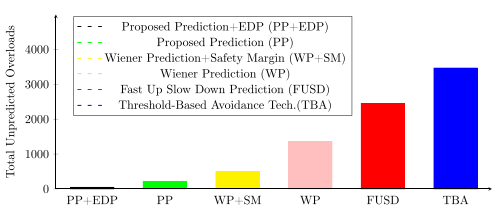
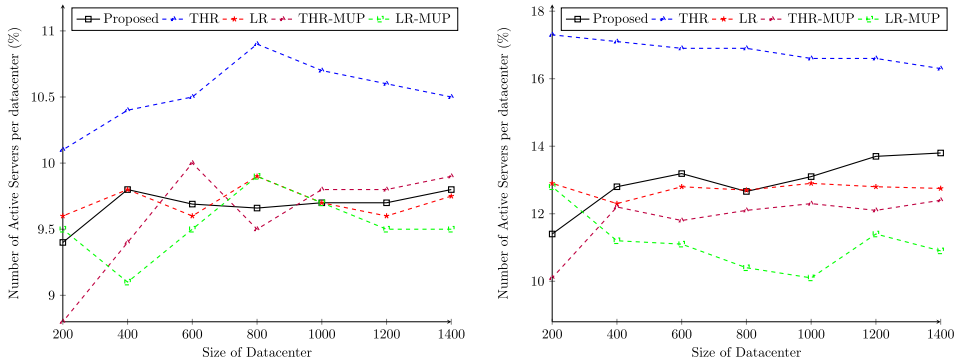


Fig. 9. Comparison of total number of unpredicted overloads during entire testing period.

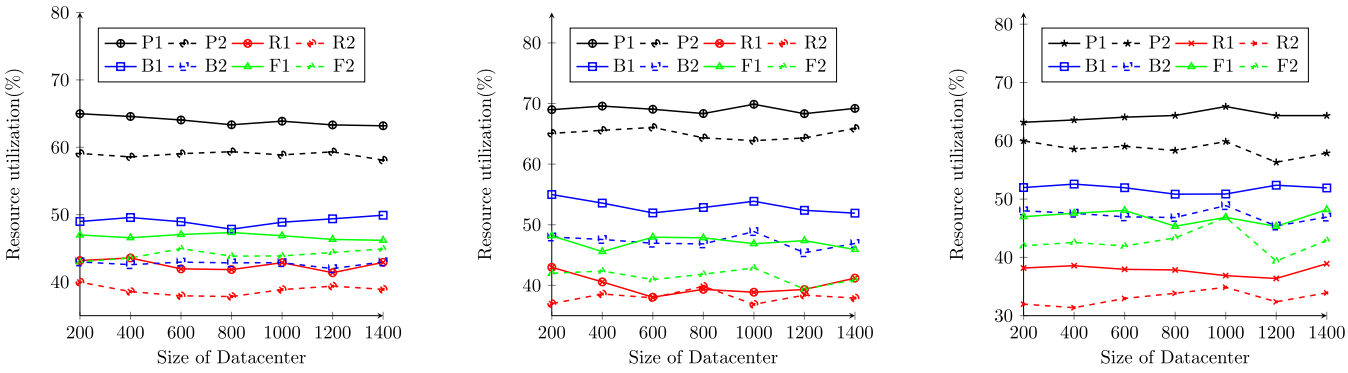
percentage for GCD and PL workloads during the entire testing period of 24 hours for 1,400 VMs among the proposed and the existing VM selection approaches, including Static THReshold (THR), Dynamic threshold based on Local Regression (LR), Static THReshold with Multiple Usage Prediction (THR-MUP) and Dynamic threshold based on Local Regression with Multiple Usage Prediction (LR-MUP) [17]. In case of both GCD and PL workloads, the percentage of active servers for the proposed approach is slightly equal or more than existing approaches due to more constrained VM placement. The proposed multi-objective approach applies an additional, but significant constraint of minimizing communication cost along with maximization of resource utilization. The VM placement is decided by the pareto-front that does not allow excessive consolidation of VMs.

Furthermore, First Fit (FF), Best Fit (BF) and Random Fit (RF) heuristics are chosen for comparison of proposed VM placement because they are baseline algorithms and many improved VM placement approaches have been developed by modifying them. Therefore, FF [38], [39], BF [36], [41] and RF [40] algorithms are implemented without oversubscription (or prediction) and with oversubscription (i.e., with our online NN based prediction) for comparison. In the figures, given ahead, proposed multi-objective load balancing (MLB) strategy with oversubscription (P1) is compared to FF, BF and RF with oversubscription denoted as F1, B1 and R1 respectively. Moreover, the proposed MLB without oversubscription (i.e., P2) is compared with FF, BF and RF without oversubscription stated as F2, B2 and R2 respectively. The comparative resource utilization is shown in Fig. 11, where proposed approach shows closer to [63-65], [68-69.5] and [63-65.8] percent of resource utilization for



(a) Google Cluster Data (b) PlanetLab VM traces

Fig. 10. Comparison of proposed framework with existing work for percentage of active servers per data center.

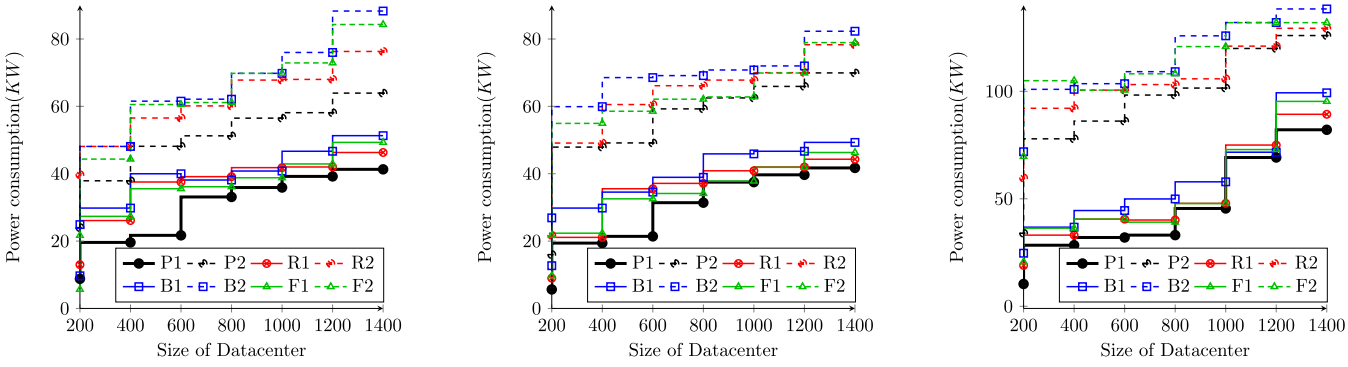


(a) Google Cluster Data (b) PlanetLab VM traces (c) Bitbrains VM traces

Fig. 11. Comparison of resource utilization trend per data center for different workloads.

GCD, PL and BB workloads respectively. The FF, BF, RF heuristics based load balancing schemes follows the resource utilization trend as BF≥FF≥RF for each workload. The OP-MLB shows substantial improvement in the resource utilization as compared to existing approaches. Specifically, it entails improvement upto 89.3 percent, 84.3 percent, 62.4 percent over R2 and 23.9 percent, 24.9 percent, 26.6 percent over B1 for BB, PL and GCD workload traces respectively. Additionally, these graphs reveals that for every type of VM allocation approach, application of online resource prediction scales up and improves their overall performance for each performance metrics (resource utilization and power consumption).

The reason for such improved resource utilization is that proposed MLB applies evolutionary optimization approach that works on N number of solutions and searches for the most optimal and feasible VM allocation, among multiple VM allocations. On the other hand, FF, BF and random heuristics are bin-packing algorithms which find out single solution that fit according to the concept of heuristic. Moreover, it is difficult to attain pareto-optimal solution that can satisfy (non-dominated) multiple constraints simultaneously with these heuristics. In addition, power consumption is computed for different VM allocations by using all the four approaches discussed above with oversubscription, and without oversubscription. Fig. 12 shows the power



(a) Google Cluster Data (b) PlanetLab VM traces (c) Bitbrains VM traces

Fig. 12. Comparison of power consumption per data center for different workloads.

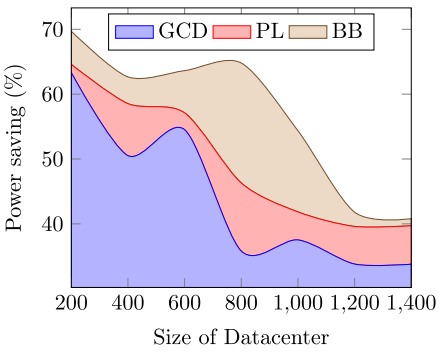


Fig. 13. Overall power saving percentage achieved by OP-MLB framework with oversubscribed compared to non-oversubscribed cloud for three different workloads.

consumption per data center for different workloads. It was observed that proposed approach scales down power consumption upto 84.01 percent, 78.9 percent over B2 and 77.8 percent over R2 for BB, PL and GCD workloads respectively. Furthermore, the power consumption by the proposed approach, is reduced upto 69.67 percent with oversubscription, as compared to without oversubscription for BB workload.

Fig. 13 shows the comparison of overall power saving percentage per data center for three different workloads, which is achieved by adapting proposed framework in oversubscribed over non-oversubscribed cloud. There is substantial improvement in power savings which follows the order: GCD $\leq$PL$\leq$BB. Furthermore, it is observed that power saving percentage is highest for data center with 200 VMs and then slowly decreases with the increase in the size of the data center and finally the power saving becomes static and reaches between 35-42 percent independent of size of the data center. This due to the reduction in the number of active servers and VM migrations with utilization of proposed framework in oversubscribed cloud. However, maximum reduction of power consumption is noticed in case of BB workloads. This is due to the fact that there were approximately 40-45 percent of VMs in BB dataset that shows CPU utilization percent less than 1 percent, which is correctly predicted by our online resource forecast system and therefore, power consumption is reduced with reduction in the number of active servers.

9 Conclusions and Future Work

In this work, elastic resource management problem is addressed by proposing an online prediction based multi-objective load balancing framework. The objectives of the proposed framework are to effectively utilize the oversubscribed cloud environment to reduce power consumption and raise resource utilization while minimizing risk of SLA violation. Moreover, the communication cost aware multi-objective load balancing was applied to minimize network traffic within the data center. The performance evaluation shows that the proposed work maximizes resource utilization and minimizes performance degradation due to overloads, number of active servers, communication cost within data center, SLA violations and power consumption. All the results are supported by the simulation and experiments executed on three different real workload traces. The comparison with state-of-art techniques states that the proposed framework can significantly improve the power savings by rightly exploiting oversubscription environment at cloud data center. In future, the proposed framework can be extended with more objectives like trust and reliability based VM allocation scheme. Additionally, the tasks can be grouped based on predicted resource utilization to allow proactive autoscaling of VMs and improve performance of cloud data center.

Acknowledgments

This work was supported in part by the National Institute of Technology, Kurukshetra, India, and in part by the University of Melbourne, Australia.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.

[2] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: Format+ schema," White Paper, 2011. [Online]. Available: https://uni2u.tistory.com/attachment/cfile30.uf@2654263359759255 2C15C1.pdf

[3] E. K. Lee, H. Viswanathan, and D. Pompili, "Proactive thermal-aware resource management in virtualized HPC cloud datacenters," *IEEE Trans. Cloud Comput.*, vol. 5, no. 2, pp. 234–248, Apr.-Jun. 2017.

[4] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "An energy-efficient VM prediction and migration framework for overcommitted clouds," *IEEE Trans. Cloud Comput.*, vol. 6, no. 4, pp. 955–966, Oct.–Dec. 2018.

[5] 'Cisco visual networking index: Forecast and methodology, 2016–2021, White Paper, 2016–2021. [Online]. Available: https://www.reinvention.be/webhdfs/v1/docs/complete-white-paper-c11–481360.pdf

[6] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic consolidation of virtual machines in self-organizing cloud data centers," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 215–228, Jul.–Dec. 2013.

[7] B. Jennings and R. Stadler, "Resource management in clouds: Survey and research challenges," *J. Netw. Syst. Manage.*, vol. 23, no. 3, pp. 567–619, 2015.

[8] D. Agarwal and S. Jain, "Efficient optimal algorithm of task scheduling in cloud computing environment," 2014, *arXiv:1404.2076*.

[9] D. Saxena, R. Chauhan, and R. Kait, "Dynamic fair priority optimization task scheduling algorithm in cloud computing: Concepts and implementations," *Int. J. Comput. Netw. Info. Secur.*, vol. 8, no. 2, pp. 41–48, 2016.

[10] K. Karthikeyan *et al.*, "Energy consumption analysis of virtual machine migration in cloud using hybrid swarm optimization (ABC–BA)," *J. Supercomputing*, vol. 76, pp. 3374–3390, 2020.

[11] S. Kim, N. Pham, W. Baek, and Y.-r. Choi, "Holistic VM placement for distributed parallel applications in heterogeneous clusters," *IEEE Trans. Serv. Comput.*, to be published, doi: 10.1109/TSC.2018.2890668.

[12] V. Sivagami and K. Easwarakumar, "An improved dynamic fault tolerant management algorithm during VM migration in cloud data center," *Future Gener. Comput. Syst.*, vol. 98, pp. 35–43, 2019.

[13] Y. Sharma, W. Si, D. Sun, and B. Javadi, "Failure-aware energy-efficient VM consolidation in cloud computing systems," *Future Gener. Comput. Syst.*, vol. 94, pp. 620–633, 2019.

[14] J. Kumar, D. Saxena, A. K. Singh, and A. Mohan, "Biphase adaptive learning-based neural network model for cloud datacenter workload forecasting," *Soft Comput.*, vol. 24, no. 1, pp. 14593–14610, 2020.

[15] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1107–1117, Jun. 2013.

[16] D. Saxena and A. K. Singh, "A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center," *Neurocomputing*, vol. 426, pp. 248–264, 2020.
[17] T. H. Nguyen, M. Di Francesco , and A. Yla-Jaaski , "Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers," *IEEE Trans. Serv. Comput.*, vol. 13, no. 1, pp. 186–199, Jan.-Feb. 2020.
[18] L. Yu, L. Chen, Z. Cai, H. Shen, Y. Liang, and Y. Pan, "Stochastic load balancing for virtual resource management in datacenters," *IEEE Trans. Cloud Comput.*, vol. 8, no. 2, pp. 459–472. Apr.-Jun. 2020.
[19] L. Zhao, L. Lu, Z. Jin, and C. Yu, "Online virtual machine placement for increasing cloud provider's revenue," *IEEE Trans. Serv. Comput.*, vol. 10, no. 2, pp. 273–285, Mar.–Apr. 2017.
[20] D. Saxena and A. K. Singh, "Security embedded dynamic resource allocation model for cloud data centre," *Electron. Lett.*, vol. 56, no. 20, pp. 1062–1065, Sep. 2020.
[21] X. Wang and Y. Wang, "Coordinating power control and performance management for virtualized server clusters," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 2, pp. 245–259, Feb. 2011.
[22] S. Kumar, V. Talwar, V. Kumar, P. Ranganathan, and K. Schwan, "vManage: Loosely coupled platform and virtualization management in data centers," in *Proc. 6th Int. Conf. Autonomic Comput.*, 2009, pp. 127–136.
[23] A. Beloglazov and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers," in *Proc. 8th Int. Workshop Middleware Grids, Clouds e-Sci.*, 2010, pp. 1–6.
[24] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.
[25] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[26] X. Zhang, Z.-Y. Shae, S. Zheng, and H. Jamjoom, "Virtual machine migration in an over-committed cloud," in *Proc. IEEE Netw. Operations Manage. Sym.*, 2012, pp. 196–203.
[27] R. Buyya, K. Branson, J. Giddy, and D. Abramson, "The virtual laboratory: A toolset to enable distributed molecular modelling for drug design on the world-wide grid," *Concurrency Comput., Pract. Experience*, vol. 15, no. 1, pp. 1–25, 2003.
[28] R. Moreno-Vozmediano , K. Nadiminti, S. Venugopal, A. B. Alonso-Conde, H. Gibbins, and R. Buyya, "Portfolio and investment risk analysis on global grids," *J. Comput. Syst. Sci.*, vol. 73, no. 8, pp. 1164–1175, 2007.
[29] N. K. Sharma and G. R. M. Reddy, "Multi-objective energy efficient virtual machines allocation at the cloud data center," *IEEE Trans. Serv. Comput.*, vol. 12, no. 1, pp. 158–171, Jan.–Feb. 2019.
[30] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[31] N. Jain, I. Menache, J. S. Naor, and F. B. Shepherd, "Topology-aware VM migration in bandwidth oversubscribed datacenter networks," in *Proc. Int. Colloq. Automata, Lang., Program.*, 2012, pp. 586–597.
[32] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Exploiting task elasticity and price heterogeneity for maximizing cloud computing profits," *IEEE Trans. Emerg. Top. Comput.*, vol. 6, no. 1, pp. 85–96, Jan.–Mar. 2018.
[33] IBM, "Power model," 1999. [Online]. Available: https://www.ibm.com/
[34] Dell, "Power model," 1999. [Online]. Available: https://www.dell.com/systems/power/hardware/
[35] Amazon, "Amazon ec2 instances," 1999. [Online]. Available: https://aws.amazon.com/ec2/instance-types/
[36] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.
[37] S. Shen, V. van Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in *Proc. 15th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, 2015, pp. 465–474.
[38] S. Fang, R. Kanagavelu, B.-S. Lee, C. H. Foh, and K. M. M. Aung, "Power-efficient virtual machine placement and migration in data centers," in *Proc. IEEE Int. Conf. Green Comput. Commun. IEEE Internet Things IEEE Cyber, Phys. Social Comput.*, 2013, pp. 1408–1413.
[39] S. Jangiti, E. S. Ram, and V. S. Sriram, "Aggregated rank in first-fit-decreasing for green cloud computing," in *Proc. Cogn. Inform. Soft Comput.*, 2019, pp. 545–555.
[40] G. Jung, M. A. Hiltunen, K. R. Joshi, R. D. Schlichting, and C. Pu, "Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures," in *Proc. IEEE 30th Int. Conf. Distrib. Comput. Syst.*, 2010, pp. 62–73.
[41] S. Shirvastava, R. Dubey, and M. Shrivastava, "Best fit based VM allocation for cloud resource allocation," *Int. J. Comput. Appl.*, vol. 158, no. 9, pp. 25–27, 2017.

Deepika Saxena received the M.Tech. degree in computer science and engineering from Kurukshetra University Kurukshetra, Haryana, India in 2014. She is currently working toward the Ph.D. degree with the Department of Computer Applications, National Institute of Technology, Kurukshetra, India. Her major research interests include neural networks, predictive analytics, evolutionary algorithms, and scheduling and security in cloud computing.

Ashutosh Kumar Singh (Senior Member, IEEE) received the Ph.D. degree in electronics engineering from Indian Institute of Technology, BHU, India. He was a Postdoc with the Department of Computer Science, University of Bristol, UK. He is currently a professor and the head with the Department of Computer Applications, National Institute of Technology Kurukshetra, India. He has more than 20 years research and teaching experience at various Universities of the India, U.K., and Malaysia. He is also a charted engineer from UK. He has authored or coauthored more than 250 research papers in different journals, conferences, and news magazines. He is the co-author of six books including the *Web Spam Detection Application using Neural Network*, the *Digital Systems Fundamentals*, and the *Computer System Organization & Architecture*. His research area includes verification, synthesis, design, and testing of digital circuits, data science, cloud computing, machine learning, security, and big data. He was an editorial board member of the *International Journal of Networks and Mobile Technologies* and the *International journal of Digital Content Technology and its Applications*. He has also shared his experience as a guest editor of *Pertanika Journal of Science and Technology*. He is involved in reviewing process of different journals and conferences including the IEEE TRANSACTIONS ON COMPUTERS, the *IET*, the IEEE conference on ITC, and the ADCOM.

Rajkumar Buyya (Fellow, IEEE) is currently a redmond barry distinguished professor and the director with Cloud Computing and Distributed Systems Laboratory, University of Melbourne, Australia. He is currently the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in cloud computing. He has authored more than 680 publications and seven textbooks including the *Mastering Cloud Computing* published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets, respectively. He is one of the highly cited authors in computer science and software engineering worldwide (h-index=142, g-index=310, 109 000+ citations).