Wireless Personal Communications (2020) 113:349–367
https://doi.org/10.1007/s11277-020-07204-6

An Energy Efficient and Adaptive Threshold VM Consolidation Framework for Cloud Environment

Nagma Khattar[1] · **Jaiteg Singh**[1] · **Jagpreet Sidhu**[2]

Published online: 21 February 2020

Abstract
Cloud-based computing, in spite of its enormous benefits has ill effects on the environment also. The release of greenhouse gases and energy consumed by cloud data centers is the most important issue that needs serious attention. Virtual machine (VM) consolidation is a prominent method for energy efficient and optimal utilization of resources. However, existing VM consolidation approaches aggressively reduce energy consumption without considering quality of service (QoS) factors. In this paper, QoS-aware VM consolidation framework is presented which reduces energy consumption and tries to minimize Service Level Agreement violations at the same time. Unlike existing solutions, the framework is generic as it works for both CPU and input/output intensive tasks. The effectiveness of proposed framework is illustrated by using real dataset of Planet lab and CloudSim platform. The proposed solution can be used in cloud data centers to enable energy efficient computing.

Keywords Energy efficiency · Cloud computing · VM consolidation · Quality of service

1 Introduction

Cloud computing enables on demand, ubiquitous, shared access to resources without much service provider interaction. Infrastructure-as-a-service (IaaS), Platform-as-a-service (PaaS) and Software-as-a-service (SaaS) are provided by cloud service providers to its users [1, 2]. All the online users are directly/indirectly getting benefits from these services. There is a remarkable impact on information technology industry with the establishment of cloud data centers. On the other hand, these data centers need massive amount of power to operate and generate high levels of carbon footprints. The demand for energy in the data centers is increasing at an alarming rate. According to the United States (US) Data Center Energy Usage Report, US data centers in 2014 consumed about 1.8% of total U.S. electricity consumption amounting to 70 billion kWh. It increased about 4% from 2010 to 2014 and 90% from 2000 to 2005. It is estimated that in 2020, the U.S. data centers will

✉ Jagpreet Sidhu
jagpreet.pu@gmail.com

[1] Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India

[2] Department of Computer Science and Information Technology, Jaypee University of Information Technology, Waknaghat, Himachal Pradesh, India

consume about 73 billion kWh [3]. The information and technology industry is predicted to use about 20% of electricity in the world by 2025 and release up to 5.5% of carbon emissions [4]. These numbers can witness a rapid hike unless noteworthy efforts are put to make energy efficient data centers. The consumption of power does not only depend on hardware or infrastructural arrangement. It is also affected by the level of resource utilization [2]. Virtual machine (VM) consolidation is an effective approach which leads to optimal utilization of resources [5–7]. VM consolidation involves shifting the VMs from less utilized servers and over utilized servers to normally loaded servers. So, less utilized servers may be switched off to save power. The process includes (1) finding underutilized servers (2) finding over utilized servers (3) selecting the VMs to be shifted (4) finding the target hosts to place the selected VMs. It ensures optimum utilization of resources by switching off the hosts which consume power at leisure. Figure 1 shows the VM consolidation process.

Existing algorithms of VM consolidation have limited focus on QoS factors. Moreover, they emphasize on one of the sub modules viz. host overload or underload detection or selection of VMs or VM placement on new hosts. Majority of existing frameworks are not generic. Furthermore, mostly the VM consolidation algorithms generally keep on aggressively reducing energy consumption without reducing violations in SLA.

The major contribution of this work is as follows:

- The novel algorithms for all the phases of VM consolidation i.e. host overload, underload detection, VM selection and VM placement are designed using four adaptive thresholds.
- The proposed framework is energy efficient. It reduces both consumption of energy and violations in SLA simultaneously.
- The framework also considers both input/output (I/O) intensive and CPU intensive tasks as the cloud environment is dynamic in nature.
- Two VM selection policies are proposed which consider CPU, memory and bandwidth at the same time.
- The proposed four threshold VM consolidation framework gives a clearer distinction of host types and shut down more number of hosts.
- The proposed framework is tested for Planet Lab workload trace and comparison with other state-of-the-art algorithms shows the efficiency of the proposed framework.

The remaining paper is framed in this manner: Sect. 2 describes the work done in the area of VM consolidation. Section 3 defines the proposed framework which includes algorithms for the four phases of VM consolidation. Section 4 presents the experimental parameters and data set for implementing the proposed algorithms. Section 5 discusses the results and Sect. 6 gives conclusive remarks on the work done.

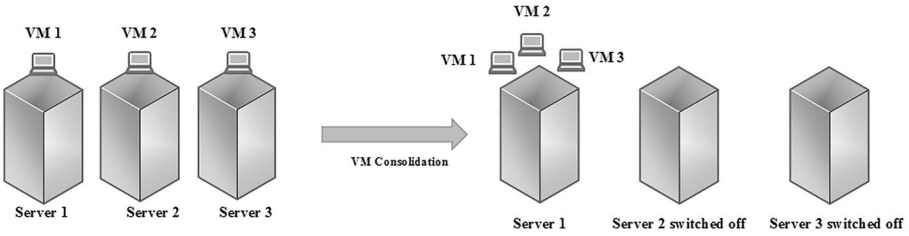


Fig. 1 VM consolidation process

2 Related Literature

There is a vast amount of literature on consolidation techniques for energy efficient cloud computing that can be classified based on consolidation type (QoS-aware or aggressive), types of threshold (static or adaptive), parameters considered in the power model (CPU, RAM, network) and data sets (synthetic or real).

Earlier energy efficient mobile devices were introduced to improve their battery lifetime [8]. Later research on energy efficient data centers and virtual computing environment started [9]. Nathuji and Schwan [10] initiated the study and designed a framework for Energy Efficient Management (EEM). The limitation was that automatic management of resources was not specifically done for global level.

Beloglazov and Buyya [11] designed a framework for EEM of resources constrained by QoS for virtual cloud data centers to reduce operational cost. Authors proposed a single and fixed threshold (no historical data) based VM consolidation algorithms and validated the algorithms on random data. However, the static threshold is not suitable for dynamic cloud environment. The drawback is that only CPU was considered in the power model and it could not handle mixed workloads. Beloglazov et al. [12] introduced an architecture based on double and fixed thresholds for VM consolidation. VM selection policies described were—Minimum Migrations (MM), Highest Potential Growth (HPG), and Random Choice Policy (RC). Only CPU was considered in the power model and system could not handle mixed workloads. They proved that the difference of 40% between upper and lower threshold is the best according to statistical tests. Beloglazov and Buyya [13] proposed heuristics and calculated the values of thresholds using adaptive double threshold based techniques using values of resource utilization. VM consolidation algorithms were based on double and adaptive thresholds. These algorithms are default algorithms in CloudSim 3.0.3. Host overload detection algorithms "[Median Absolute Deviation (MAD), Interquartile Range (IQR), Local Regression (LR), Robust Local Regression (LRR)], VM selection policies [Minimum Migration Time (MMT), Random Selection (RS), Maximum Correlation (MC) and Maximum Utilization (MU)], VM placement algorithm [bin packing heuristics like power-aware best fit decreasing (PABFD)]" were used. For underload detection, migrations of VMs were done from least utilized hosts. Planet lab dataset was used for validation. The proposed framework only considered CPU in power model.

Cao and Dong [6] improved the VM consolidation framework proposed by Beloglazov and Buyya [13]. It was based on double and adaptive thresholds. A new algorithm SLA violation decision algorithm (SLAVDA) was designed to check whether an overloaded host caused a violation in SLA. Adaptive thresholds were used based on historical values of resource utilization. Proposed system could handle mixed workloads. Limitation was that only CPU was considered in the model.

Zhou et al. [14] presented a three threshold based energy efficient algorithm for VM consolidation. VM consolidation algorithms were based on triple and fixed thresholds. Authors proposed a minimization of migration policy based on three thresholds (MIMT) for VM placement algorithm. Planet lab data was used to test the validity of proposed model. Resource utilization thresholds were not based on analysis of historical data. Proposed system could not handle mixed workloads. Various combinations of threshold values were made and statistical tests were done to know the finest combination and 40% interval between three thresholds was found to be best.

Zhou et al. [15] designed an adaptive three threshold based energy efficient algorithm for consolidating VMs. VM consolidation algorithms were based on triple and dynamic

thresholds. Authors developed two algorithms for determination of thresholds: "-Means Clustering Algorithm-Average-Median Absolute Deviation (KAM) and -Means Clustering Algorithm-Average-Interquartile Range (KAI)" and three VM selection policies: "Minimum Memory Size (MMS), Lowest CPU Utilization (LCU) and Minimum Product of both CPU Utilization and Memory Size (MPCM)". "Energy-aware best fit decreasing (EABFD)" algorithm for VM placement was used. Planet lab data was employed to test the validity of proposed model. Adaptive thresholds were calculated based on the values of resource usage. Authors focused on reducing energy consumption but not on increasing energy efficiency and proposed system was not generic.

Castro et al. [16] proposed two new approaches for dynamic consolidation of VMs that considered both CPU and RAM. VM consolidation algorithms were based on double and adaptive thresholds. Authors proposed a CPU, RAM energy-aware VM placement algorithm and an underload detection algorithm that considered both CPU and RAM. Planet lab data and Google traces were used to test the validity of proposed model. Adaptive thresholds were calculated based on the values of resource usage and system was able to handle mixed workloads.

Zhou et al. [17] developed energy efficient algorithms for VM consolidation that minimizes SLA violation by considering both memory and CPU utilization both during VM deployment. Authors used adaptive three threshold KMI algorithm to find the thresholds and proposed a novel algorithm for host overload detection. Two VM selection policies to handle both CPU bound and I/O bound tasks were developed—"Maximum ratio of CPU utilization to memory utilization (MRCU) and Minimum the product of a CPU utilization and a memory utilization (MPCU)". Authors proposed "VM placement with maximizing energy efficiency (VPME)". Planet lab data was used to evaluate the validity of the system. Other works on QoS and energy efficient resource management are [18–22].

As found from literature, the major drawbacks in the proposed VM consolidation frameworks are that they are mostly not generic or they do not consider memory and bandwidth during VM migration. This work focuses on four adaptive thresholds. This is done to shut down more number of hosts to decrease energy consumption and classification of hosts is more distinct. This work considers reducing both SLA violations and energy consumption simultaneously.

3 Proposed Framework

The framework uses four adaptive thresholds to find the category to which a host belongs. To find the thresholds and thus, overloaded hosts K-medoid interquartile range mid-range (KMIMR) algorithm is proposed. For choosing VMs from the hosts which are overloaded, two algorithms Maximum Ratio of CPU to Product of Memory and Bandwidth (MRCPMB) for CPU intensive tasks and Minimum Product of CPU, Bandwidth and Memory (MPCBM) for Input/output intensive tasks are designed. For VM deployment on new hosts, Energy efficient VM deployment (EEVD) algorithm is proposed. After that the whole process of VM placement is optimized by using Optimize VM Placement Algorithm (OVMP). Figure 2 shows the proposed framework.

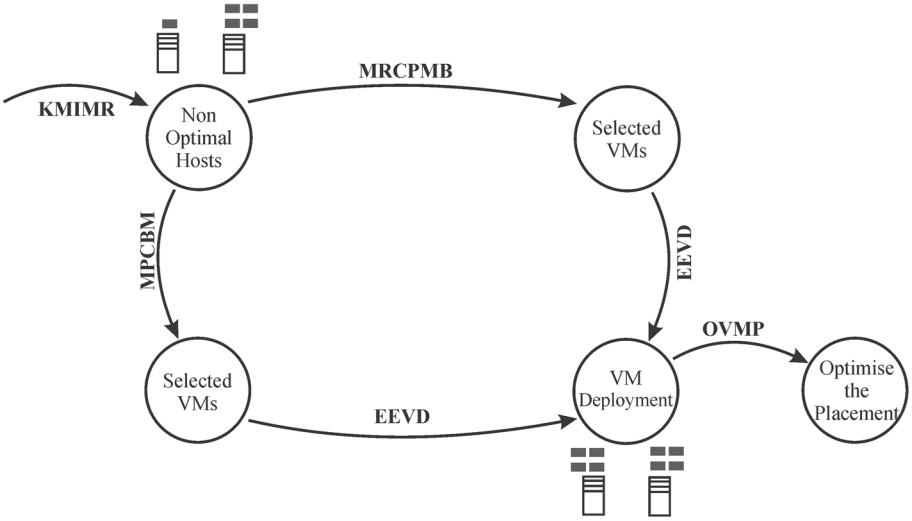


Fig. 2 Proposed framework

We have proposed a four threshold based flexible VM consolidation framework. The thresholds are decided on the basis of fuzzy values. Hosts are divided into five classes on the basis of four thresholds (T_1, T_2, T_3 and T_4). If utilization of a host is less than T_1, it is classified as very less loaded host. If utilization is larger than or equal to T_1 and lesser than T_2, it is called as less utilized host. If host utilization is larger than or equal to T_2 and lesser than T_3, it is moderately utilized host. If utilization is T_3 or lies between T_3 and T_4, the host is called normally utilized host and if utilization is T_4 or is in between T_4 and 1, then it is over utilized host. Figure 3 represents the category of hosts.

Let there are N number of hosts. The utilization of ith host is denoted as Ut_i. The working of the framework is shown in Fig. 4.

3.1 Calculating Thresholds

The unusual fact is that resources in idle state in a data center too consume high amount of power at leisure. Servers housed in cloud data centers are extremely away from energy uniformity. Even at 20% utilization, they draw 80% peak power. This is a major cause of energy inefficiency. The servers sit idle frequently as their utilization is between 10 and 50% of their peak load [14–17]. The major concern is inefficient utilization of resources which causes high power consumption. The over utilized resources degrade the perform and under-utilized resources consume high amount of power even when not in use.

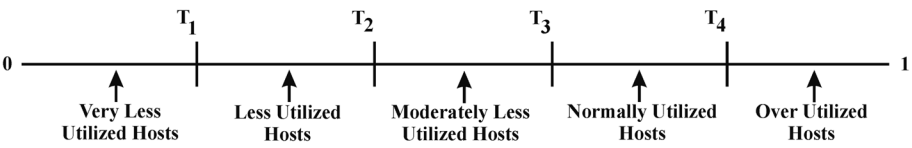


Fig. 3 Category of hosts

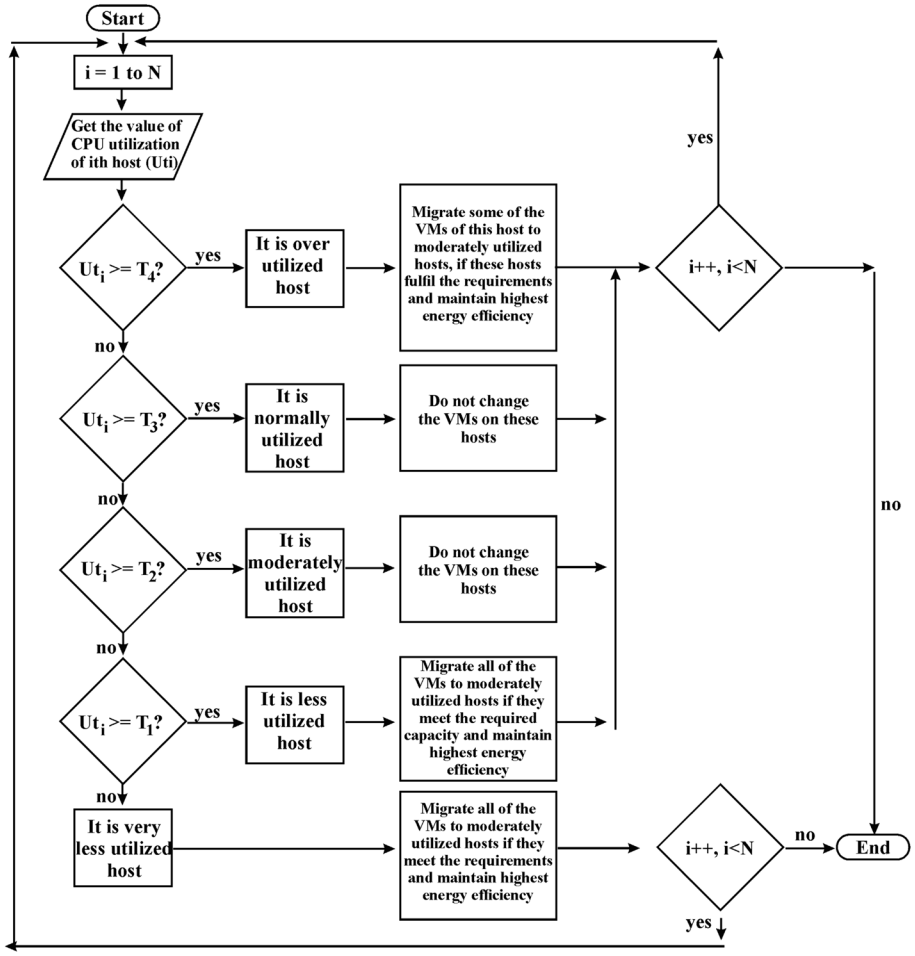


Fig. 4 Working of the framework

Literature shows detection of over loaded and under loaded hosts by finding static or adaptive single and double thresholds. This work proposes the use of four adaptive thresholds so as to classify the hosts into more distinct types for shutting down more number of hosts resulting into lesser energy consumption. The thresholds are calculated by designing a novel K-medoid interquartile range mid-range algorithm.

3.1.1 K-Medoid Interquartile Range Mid-range Algorithm (KMIMR)

To find the values of four thresholds, a KMIMR algorithm is designed using statistical analysis of resource utilization history along with some numerical computations.

Let $PH = \{H_1, H_2, H_3, \ldots H_n\}$ be a set of n physical hosts in a cloud data center. Assume $U = \{U_1, U_2, U_3, \ldots U_m\}$ be a set such that $U_i \in U$ $(1 \leq i \leq m)$ is CPU utilization of physical host

$H_j \in PH$ at time t. Algorithm 1 shows the pseudo code.

Algorithm 1: KMIMR
Input: set of utilization **(**U), number of clusters (k), safety parameter (L)
Output: Four thresholds T_1, T_2, T_3, T_4
1: C = KMedCluster (U, k)
2: **for** x = 1 to C.length **do**
3: IQR[x-1] = $TQ_3 - TQ_1$
4: **end for**
5: Iqr = (Max(IQR) + Min(IQR))/2
6: $T1 = \frac{1}{10}(1 - L.Iqr)$
7: $T2 = \frac{11}{20}(1 - L.Iqr)$
8: $T3 = \frac{9}{10}(1 - L.Iqr)$
9: $T4 = (1 - L.Iqr)$
END

The input to the algorithm is set of utilization (U), number of clusters (k) and safety parameter (L). The output is the values of four thresholds. The function KMedCluster (U, k) in Algorithm is used to partition U into k clusters C_1, C_2,…. C_k such that:

- $C_x = \{UP_{x-1+1}, UP_{x-1+2}, \ldots\ldots. UP_x\} \in D$,
- $0 = P_0 < P_1 < P_2 < P_3 \ldots\ldots. P_x = m$, and
- $C_x \cap C_y = \phi$ for $1 \leq x, y \leq k$ (the value of k can be decided using empirical approach i.e. observation through experiments)

For each cluster C_x, the algorithm calculates the interquartile range of each cluster (Eq. 1)

$$\text{IQR}\,[x-1] = \left(TQ_3\right)_x - \left(TQ_1\right)_x, \tag{1}$$

where $(TQ_3)_x$ is the third quartile of cluster x and $(TQ_1)_x$ is the first quartile of cluster x.

From the IQR array, the mid-range of interquartile range is calculated as shown in Eq. (2)

$$Iqr = \frac{Max(IQR) + Min(IQR)}{2}, \tag{2}$$

where Max(IQR) refers to maximum value of interquartile range in IQR array (which contains interquartile range of all clusters), Min(IQR) is the minimum value in the IQR array. This step obtains the value of Iqr, then the fuzzy threshold values are calculated using Eqs. (3–6).

$$T1 = \frac{1}{10}(1 - L.Iqr), \tag{3}$$

$$T2 = \frac{11}{20}(1 - L.Iqr), \tag{4}$$

$$\mathrm{T3} = \frac{9}{10}(1 - \mathrm{L.Iqr}), \tag{5}$$

$$\mathrm{T4} = (1 - \mathrm{L.Iqr}), \tag{6}$$

where L is the safety parameter for VM consolidation i.e. the extent of VM consolidation. The higher the value of safety parameter, the higher is the energy consumption and lesser SLA violations and vice versa. The thresholds are calculated motivated from [12] using the boundary values of host utilization as 10%, 55%, 90% which distinguishes the hosts into distinct types. The mathematical computations in accordance with the Iqr value obtained from KMIMR as shown in equations above give adaptive thresholds according to the resource utilization history in order to comply with dynamic cloud environment.

3.2 VM Selection Methods

VM selection methods are proposed considering both I/O and CPU intensive tasks simultaneously. A task which is CPU intensive will take more CPU cycles and its completion time depends upon the processor speed. I/O intensive tasks, on the other hand perform more of the I/O operations and their completion time depends upon the time for which they kept waiting to complete I/O operations.

3.2.1 Maximum Ratio of CPU to Product of Memory and Bandwidth (MRCPMB)

The energy consumption by CPU is more as compared to memory and bandwidth in the case of tasks which are CPU intensive. So, we choose a VM whose CPU consumption is more, to migrate on another host to reduce consumption of energy. $\mathrm{VM_i}$ is selected as compared to $\mathrm{VM_j}$ such that the following condition (Eq. 7) holds:

$$\left(\frac{CPU}{Memory * Bandwidth}\right)_i > \left(\frac{CPU}{Memory * Bandwidth}\right)_j \tag{7}$$

3.2.2 Minimum Product of CPU, Bandwidth and Memory (MPCBM)

In I/O intensive tasks, the CPU, memory and bandwidth all amount for considerable energy consumption. So, select such a VM which consumes less CPU, memory and bandwidth as its migration will be easier. Therefore, in this case, we select a $\mathrm{VM_i}$ in comparison to $\mathrm{VM_j}$ such that the following condition (Eq. 8) holds:

$$(CPU * Bandwidth * Memory)_i < (CPU * Bandwidth * Memory)_j \tag{8}$$

3.3 VM Deployment Algorithm

This subsection presents a new VM deployment algorithm i.e. Energy Efficient VM Deployment (EEVD) for placement of VMs on new target hosts. It considers energy efficiency i.e. reducing SLA violations and energy consumption at the same time. It also considers memory consumption. The pseudo code of EEVD is described in Algorithm 2.

The input to the algorithm is listofhosts, listofvms which represents the set of hosts, set of VMs in the data center respectively and four adaptive thresholds T_1, T_2, T_3, T_4 calculated using KMIMR policy. The output of the algorithm is allocation of VMs on new target hosts maintaining energy efficiency.

Algorithm 2: The EEVD algorithm

Input: listofhosts, listofvms, T_1, T_2, T_3, T_4
Output: VMs' allocation

```
1.  vmlist.arrangeByDecreasingCpuUtilization( )
2.  for each VM in the listofvms do
3.      minimEnergyEff = min
4.      hostAllocated = null
5.      for each Host in the listofhosts do
6.          if (Host matches the VM requirements) then
7.              util = getUtilAfterAlloc(Host, VM)
8.              if (util >= T2 AND util < T3) then
9.                  energyCons = Host.getPower( )
10.                 energyConsAfterAlloc = getPowAfterAlloc (Host, VM)
11.                 diffEnergyCons = energyConsAfterAlloc – energyCons
12.                 sla1before = getSlaTimePerActHost (Host)
13.                 sla1after = getSlaTimePerActHostAfterAlloc (Host, VM)
14.                 sla1 = sla1after – sla1before
15.                 sla2before = getSlaMetr (Host.getListOfVms ( ))
16.                 sla2after = getSlaMetrAfterAlloc (Host.getListOfVms ( ), VM)
17.                 sla2 = sla2after – sla2before
18.                 mem = getActualUsedMem (Host, VM) – getTotalMem (Host)
19.                 sla = sla1 * sla2
20.                 energyEff = 1/(diffEnergyCons * sla * mem)
21.                 if (energyEff > minimEnergyEff) then
22.                     minimEnergyEff = energyEff
23.                     hostAllocated = Host
24.                 end if
25.             end if
26.         end if
27.     end for
28. end for
END
```

The first step is to arrange the VMs in the decreasing order of their CPU utilization (Line 1). For every available VM in listofvms, a variable 'minimEnergyEff' is assigned a minimum value and another variable 'hostAllocated' is given a NULL value (Lines 2–4). For all the VMs, each host is examined one by one to find a suitable target host. If a host matches the requirements of a VM then the utilization of host after VM allocation is noted (Line 7). Line 8 is for hosts which are moderately utilized. Lines 9–11 is to calculate the difference between the energy consumption before and after allocation of VM to host. The difference between SVTAH (Eq. 2) before and after allocation is also calculated and stored in a variable "sla1" (Lines 12–14). The second SLA metric corresponding to performance degradation due to migration (Eq. 3) is also considered. The difference between second SLA violations before and after allocation is calculated and stored in a variable "sla2" (Lines 15–17). The difference between actual used memory

before and after allocation is stored in a variable "mem" (Line 18). The total SLAV (Eq. 4) is calculated in Line 19. Line 20 calculates the energy efficiency (Eq. 5). Lines 21–23 is to find the host with highest energy efficiency and complete the allocation of VM to that host.

3.4 Optimization of VM Placement

Optimization of VM placement is very significant to improvise the resource utilization in a data center leading to increased energy efficiency. Optimize VM placement algorithm (OVMP) is designed to optimize the VM placement from very less loaded, less loaded and over loaded hosts. Algorithm 3 shows the pseudo code for optimization of VM placement in a data center.

It takes as input the four thresholds (T_1, T_2, T_3, T_4), listofhosts, listofvms and returns the mapMigration. In the first step, CPU utilization of every host is fetched (Line 2) and it is checked whether a host is over utilized (Line 3). The selected VMs are migrated from overloaded hosts to other energy efficient hosts (Lines 5–6). Lines 8–11 finds the less loaded hosts and selected VMs from them are migrated to another hosts with highest energy efficiency. In the next step (Lines 15–19), the selected VMs from very less loaded hosts are migrated to another energy efficient hosts. The algorithm returns the migration map of all migrated VMs from over loaded, less loaded and very less loaded hosts allocated to new target machines.

Algorithm 3: The OVMP Algorithm

Input: T_1, T_2, T_3, T_4
Output: mapMigration
1. for each Host in the listofhosts **do**
2. u = host.getCpuUtil ( )
3. **if** (u >= T_4) **then**
4. overLoadedHosts = getOverLoadedHosts ( )
5. VmsForMigration = getVmsForMigrationFromHosts (OverLoadedHosts)
6. mapMigration = getNewVmPlace (VmsForMigration)
7. **end if**
8. **else if** (u < T_2 AND u >=T_1)
9. lessLoadedHosts = getLessLoadedHosts ( )
10. VmsForMigration2 = getVmsForMigrationFromLessHosts (lessLoadedHosts)
11. mapMigration2 = getNewVmPlace (VmsForMigration2)
12. **end else if**
13. mapMigration.addAll (mapMigration2)
14. **else if** (u < T_1)
15. VLessLoadedHosts = getVLessLoadedHosts ( )
17. VmsForMigration3 = getVmsForMigrationFromVLessHosts(VLessLoadedHosts ( )
18. mapMigration3 = getNewVmPlace (VmsForMigration3)
19. mapMigration.addAll (mapMigration3)
20. **end else if**
21. end for
22. return mapMigration
END

4 Experimental Parameters

The following configuration of VMs and hosts is used as shown in Table 1 and power consumption is calculated as per Spec power benchmark (Table 2). The CloudSim is used for experimental evaluation.

Planet lab workloads are utilized for experimental setup, which is a part of CoMon project. CoMon was used to monitor Planet lab nodes. The workload contains traces of CPU utilization function of VMs. Each algorithm's simulation run was designed and simulated on these workloads (Table 3).

4.1 Parameters

The output parameters in terms of which algorithms are evaluated include energy consumption, number of migrations, number of hosts shutdown, SLA violation time per active host (SVTAH), Performance degradation due to migration (PDM) and overall SLA violation.

Table 1 Configuration of VMs and hosts used in experiments

Configuration	Host	VMs
Types	2	4
MIPS	{1860, 2660} MHz	{2500, 2000, 1000, 500} EC2 Compute Units
PES	{2, 2}	{1, 1, 1, 1}
RAM	{4, 4} GB	{0.85, 1.7, 1.7, 0.6} GB
BW	1 Gbit/s	100 Mbit/s

Table 2 Power consumption in Watts

Server	Idle	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP Proliant G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP Proliant G5	93.7	97	101	105	110	116	121	125	129	133	135

Table 3 Workload traces

Workload trace name	Meaning	Number of hosts	Number of VMs
20110303	This data is collected on 3 March, 2011	800	1052
20110306	This data is collected on 6 March, 2011	800	898
20110309	This data is collected on 9 March, 2011	800	1061
20110322	This data is collected on 22 March, 2011	800	1516
20110325	This data is collected on 25 March, 2011	800	1078
20110403	This data is collected on 3 April, 2011	800	1463
20110409	This data is collected on 9 April, 2011	800	1358
20110411	This data is collected on 11 April, 2011	800	1233
20110412	This data is collected on 12 April, 2011	800	1054
20110420	This data is collected on 20 April, 2011	800	1033

4.1.1 Energy Consumption

Reducing consumption of energy by servers in data centers is crucial. It can be calculated as summation of power consumption during a time period (Eq. 9).

$$e(t) = \int_t p(t) \tag{9}$$

4.1.2 SVTAH

The percentage of SLA violation time for which a host has experienced 100% utilization. SVTAH is defined in Eq. (10).

$$\text{SVTAH} = \frac{1}{n} \sum_{x=1}^{n} \frac{t_{s_x}}{t_{a_x}} \tag{10}$$

where n is number of physical machines; t_{s_x} is time for which host x has experienced 100% utilization and t_{a_x} is active time of host x.

4.1.3 PDM

Equation (11) represents performance degradation due to migrations, where m is number of VMs in data center; C_{d_y} is predicted PDM of VM y;C_{r_y} is total CPU utilization requested by VM y during its lifetime and C_{d_y} is 10% of requested CPU utilization of y during its migration.

$$\text{PDM} = \frac{1}{m} \sum_{y=1}^{m} \frac{C_{d_y}}{C_{r_y}} \tag{11}$$

4.1.4 Overall SLA Violation

It is defined as product of SVTAH and PDM as shown in Eq. (12).

$$SLAV = SVTAH * PDM \tag{12}$$

4.1.5 Energy Efficiency

It is defined in Eq. (13).

$$\frac{1}{Energy\ Consumption * SLA * Memory} \tag{13}$$

5 Results and Discussion

This Section discusses the experimental results in comparison with other state of the art algorithms. The study presents the evaluation of VM deployment algorithm and it also evaluates the framework for both CPU and I/O intensive tasks.

5.1 Evaluation and Analysis of VM Deployment Algorithm

This Section evaluates the performance of EEVD algorithm which places the selected VMs on new hosts taking into account both consumption of energy and violations in SLA at the same time. The algorithms with which the proposed algorithm is compared are KAM-MMS-2.0 [15] and KAI-MMS-1.0 [15]. EEVD is employed with both these algorithms in order to check its performance. The combination of KAM-MMS-2.0 and EEVD is termed as KAM-MMS-2.0-EEVD and the combination of KAI-MMS-1.0 and EEVD is called KAI-MMS-1.0-EEVD. The other algorithms with which the proposed algorithm EEVD is compared with are KAI-MMS-1.0-EE [15] and KAM-MMS-2.0-EE [15].

Figure 5 shows the performance of EEVD algorithm in terms of energy efficiency in comparison to other algorithms. Figure 6 illustrates the performance of EEVD algorithm in terms of energy consumption compared with other algorithms. It is clear that algorithm KAI-MMS-1.0-EEVD is better as compared to other algorithms. It is because it saves more number of hosts by optimally utilizing the resources and considers energy, bandwidth, memory consumption and SLA violations at the same time during VM deployment.

Figures 7 and 8 compare all the algorithms in terms of SLA violations and SVTAH respectively.

Figures show that KAM-MMS-2.0-EEVD has better performance than KAM-MMS-2.0 and KAM-MMS-2.0-EE. This is because KAM-MMS-2.0-EEVD reduces excessive migrations and at the same time reduces the number of utilized hosts, SLA violations, and SVTAH. Similarly, KAI-MMS-1.0-EEVD is than KAI-MMS-1.0 and KAI-MMS-1.0-EE.

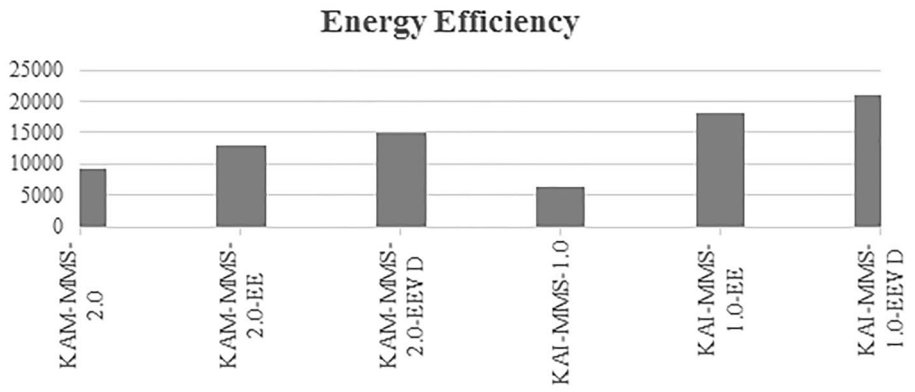


Fig. 5 Comparison of energy efficiency of algorithms

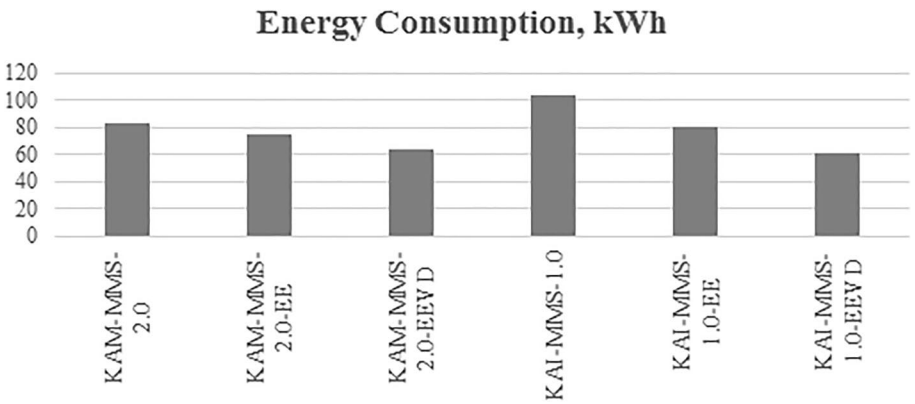


Fig. 6 Comparison of energy consumption of algorithms

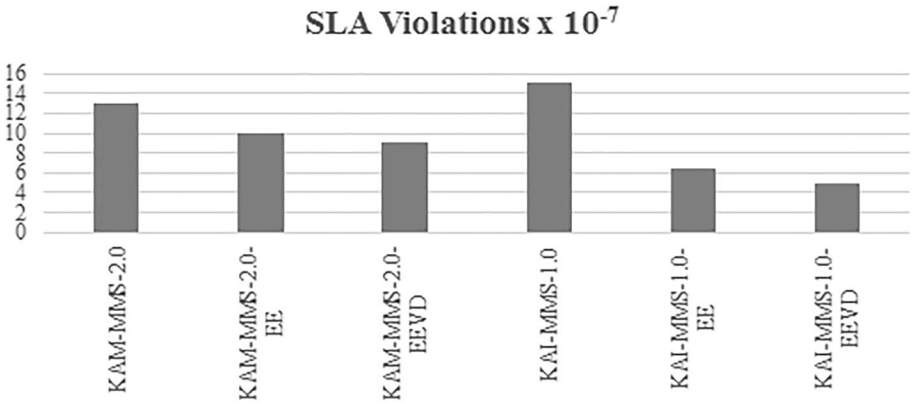


Fig. 7 Comparison of SLA violations of all algorithms

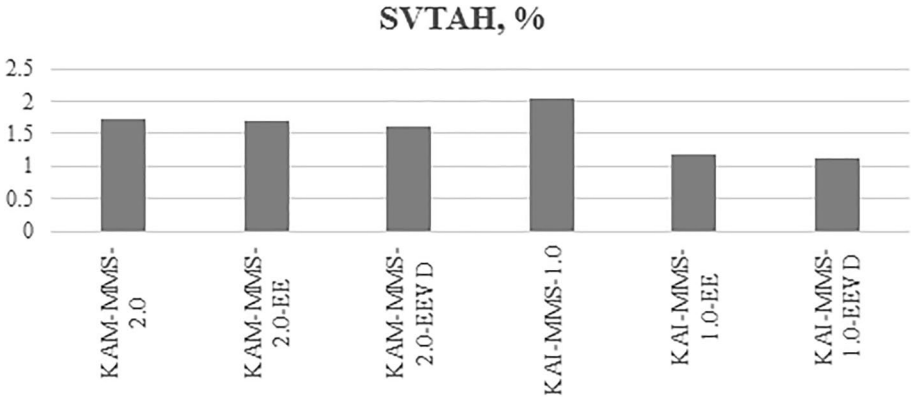


Fig. 8 Comparison of SVTAH of all algorithms

5.2 Evaluation of Framework for CPU Intensive Tasks

The workload trace named "20110303" is considered to be CPU intensive trace [23]. So, this data set is chosen for evaluation of CPU intensive tasks. The proposed framework comprising of FFT, KMIMR, MRCPMB, EEVD and safety parameter taken as 1.0 [17] is termed as KMIMR-MRCPMB-1.0. for CPU intensive tasks.

The algorithms used for comparison are "Non Power Aware (NPA) [12], Dynamic Voltage Frequency Scaling (DVFS) [12], THR-MMT-1.0 [12], THR-MMT-0.8 [12], MAD-MMT-2.5 [12], IQR-MMT-1.5 [12], KAM-MMS-2.0 [15], KAI-MMS-1.0 [15] and KMI-MRCU-1.0 [17] ".

Table 4 compares the algorithms NPA, DVFS, THR-MMT-1.0, THR- MMT-0.8, MAD-MMT-2.5, IQR-MMT-1.5, KAM-MMS-2.0, KAI-MMS-1.0, KMI-MRCU-1.0 and KMIMR-MRCPMB-1.0 on the basis of energy efficiency, energy consumption, SLA violations, SVTAH, PDM, and number of VM migrations. NPA and DVFS both do not include VM migration. Thus, energy efficiency, SLA violations, SVTAH, PDM, and number of VM migrations do not exist for them. The more the value of energy efficiency of an algorithm, and the lesser the values of energy consumption, SLA violations, SVTAH, PDM, and number of VM migrations, the better the algorithm is.

It is well clear from the table that the performance of KMIMR-MPCBM-2.0 is much better than the rest of the algorithms in terms of energy efficiency and other factors. The reason behind is that NPA and DVFS do not involve migrations, THR-MMT-0.8, THR-MMT-1.0, MAD-MMT-2.5, IQR-MMT-1.5 are based on double thresholds and only consider reducing energy consumption.

KAM-MMS-2.0, KAI-MMS-1.0, KMI-MRCU-1.0 are based on three thresholds and uses K-means clustering whereas, KMIMR-MRCPMB-1.0 is based on four flexible thresholds which results in saving more number of hosts. Moreover, it is also based on K-medoids clustering which is more robust to outliers. It also considers bandwidth in addition to CPU utilization and memory.

Table 4 Comparison of algorithms for CPU intensive tasks

Algorithms	Energy efficiency	Energy consumption (KWh)	SLA violations (10^{-7})	SVTAH (%)	PDM (%)	Number of VM migrations
NPA	–	2410.8	–	–	–	–
DVFS	–	803.91	–	–	–	–
THR-MMT-1.0	38	99.95	2613	26.97	0.10	19852
THR-MMT-0.8	170	119.40	492	4.99	0.10	26567
MAD-MMT-2.5	169	114.27	518	5.24	0.10	25923
IQR-MMT-1.5	166	117.08	514	5.08	0.10	26420
KAM-MMS-2.0	9231	83.33	13	1.73	0.01	6808
KAI-MMS-1.0	6393	104.28	15	2.03	0.01	7519
KMI-MRCU-1.0	28484	87.77	4	0.90	0.004	2821
KMIMR-MRCPMB-1.0	30089	78.98	1	0.76	0.025	2608

5.3 Analysis of Framework for I/O Intensive Tasks

To analyse I/O intensive tasks, the data set used is 20110322 [23]. For I/O intensive tasks the proposed framework uses FFR, KMIMR, MPCBM, EEVD and safety parameter as 2.0 [] and is termed as KMIMR-MPCBM-2.0. The algorithms used for comparison are "NPA [12], DVFS [12], THR-MMT-1.0 [12], THR-MMT-0.8 [12], MAD-MMT-2.5 [12], IQR-MMT-1.5 [12], KAM-MMS-2.0 [15], KAI-MMS-1.0 [15] and KMI-MPCU-1.0 [17]".

Table 5 compares the algorithms NPA, DVFS, THR-MMT-1.0, THR- MMT-0.8, MAD-MMT-2.5, IQR-MMT-1.5, KAM-MMS-2.0, KAI-MMS-1.0, KMI-MPCU-2.0 and KMIMR-MPCBM-2.0 on the basis of energy efficiency, energy consumption, SLA violations, SVTAH, PDM, and number of VM migrations. As NPA and DVFS both do not include VM migrations, there is no count of energy efficiency, SLA violations, SVTAH, PDM and number of VM migrations. It is observed that the performance of KMIMR-MPCBM-2.0 is much better than rest of the algorithms. It is because it considers SLA violations, energy, memory and bandwidth consumption at the same time compared with other methods which only consider one or two factors. It employs the use of K-medoids clustering which is more robust to outliers as compared to K-means clustering method. Moreover, the four flexible thresholds optimize the resource usage and results in saving more number of hosts. This concludes that the proposed framework is better than other state-of-the-art algorithms.

6 Conclusion

This work proposes a flexible four threshold framework for VM consolidation which not only consider energy consumption but also SLA violations, memory and bandwidth consumption. Unlike the existing frameworks, the proposed framework is generic and works for both CPU intensive and I/O intensive tasks. This paper presents KMIMR algorithm for host overload detection, MRCPMB and MPCBM policies for VM selection, and EEVD policy for VM placement and OVMP algorithm for optimizing the

Table 5 Comparison of algorithms for input output intensive tasks

Algorithms	Energy efficiency	Energy consumption (KWh)	SLA violations (10^{-7})	SVTAH (%)	PDM (%)	Number of VM migrations
NPA	–	2410.80	–	–	–	–
DVFS	–	1014.21	–	–	–	–
THR-MMT-1.0	32	101.62	3115	27.72	0.11	25560
THR-MMT-0.8	136	120.91	609	5.49	0.11	33417
MAD-MMT-2.5	148	117.88	574	5.35	0.11	32795
IQR-MMT-1.5	134	121.11	615	5.46	0.11	33061
KAM-MMS-2.0	4922	84.65	24	2.03	0.01	8736
KAI-MMS-1.0	4612	103.26	21	1.80	0.01	8190
KMI-MPCU-2.0	5694	67.55	26	2.90	0.01	12607
KMIMR-MPCBM-2.0	6009	59.89	23	2.60	0.002	11099

process of VM placement. The experimental results prove the efficiency of proposed framework. The results also show that K-medoids algorithm is more effective than K-means and four thresholds result in saving more number of hosts as compared to three threshold. Considering other factors such as bandwidth in addition to CPU utilization is more effective. Future work includes implementation in real cloud environment.

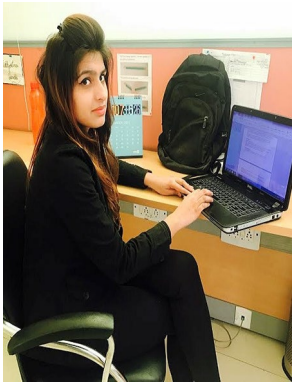
Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Buyya, R., Toosi, A. N., & Calheiros, R. N. (2014). Interconnected cloud computing environments: Challenges, taxonomy, and survey. *ACM Computing Surveys, 47,* 1–47.
2. Buyya, R. (2009). Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility. In *9th IEEE/ACM international symposium on cluster computing and the grid*.
3. Shehabi, A., & et al. (2016) United States data center energy usage report | Energy Technologies Area.
4. Adams, W. (2018). Power consumption in data centers is a global problem.
5. Cao, Z., & Dong, S. (2012). Dynamic VM consolidation for energy-aware and SLA violation reduction in cloud computing. In *Parallel and distributed computing, applications and technologies, PDCAT proceedings*.
6. Cao, Z., & Dong, S. (2014). An energy-aware heuristic framework for virtual machine consolidation in Cloud computing. *The Journal of Supercomputing, 69,* 429–451.
7. Alboaneen, D. A., Pranggono, B., & Tianfield, H. (2014). Energy-aware virtual machine consolidation for cloud data centers. In *Proceedings of IEEE/ACM 7th international conference on utility and cloud computing*.
8. Zeng, H., Ellis, C. S., Lebeck, A. R., & Vahdat, A. (2002). ECOSystem. *ACM SIGOPS Operating Systems Review, 36,* 123.
9. Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., & Doyle, R. P. (2001). Managing energy and server resources in hosting centers. In *Proceedings of 18th ACM symposium on operating systems principles*.
10. Nathuji, R., & Schwan, K. (2007). Virtual power: Coordinated power management in virtualized enterprise systems. In *Proceedings of 21st ACM SIGOPS symposium on operating systems principles*.
11. Beloglazov, A., & Buyya, R. (2010). Energy efficient resource management in virtualized cloud data centers. In *10th IEEE/ACM international conference on cluster, cloud and grid computing*.
12. Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computing Systems, 28,* 755–768.
13. Beloglazov, A., & Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience, 24,* 1397–1420.
14. Zhou, Z., Hu, Z. G., Song, T., & Yu, J. Y. (2015). A novel virtual machine deployment algorithm with energy efficiency in cloud computing. *Journal of Central South University, 22,* 974–983.
15. Zhou, Z., Hu, Z., & Li, K. (2016). Virtual machine placement algorithm for both energy-awareness and SLA violation reduction in cloud data centers. *Scientific Programming*. https://doi.org/10.1155/2016/5612039.
16. Castro, P. H. P., Barreto, V. L., Corrêa, S. L., Granville, L. Z., & Cardoso, K. V. (2016). A joint CPU-RAM energy efficient and SLA-compliant approach for cloud data centers. *Computer Networks, 94,* 1–13.
17. Zhou, Z., et al. (2018). Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms. *Future Generation Computing Systems, 86,* 836–850.
18. Goyal, S., Bawa, S., & Singh, B. (2016). Energy optimised resource scheduling algorithm for private cloud computing. *International Journal of Ad Hoc and Ubiquitous Computing, 23,* 115.
19. Kim, H. S., Shin, D. I., Yu, Y. J., Eom, H., & Yeom, H. Y. (2012). Systematic approach of using power save mode for cloud data processing services. *International Journal of Ad Hoc and Ubiquitous Computing, 10,* 63.
20. Zhao, J., Tao, J., & Frlinger, K. (2014). A framework for comparative performance study on virtualised machines. *International Journal of Ad Hoc and Ubiquitous Computing, 17,* 82–99.

21. Xu, H., Liu, Y., Wei, W., & Xue, Y. (2019). Migration cost and energy-aware virtual machine consolidation under cloud environments considering remaining runtime. *International Journal of Parallel Programming, 47,* 481–501.
22. Ramezani, F., Lu, J., & Hussain, F. K. (2014). Task-based system load balancing in cloud computing using particle swarm optimization. *International Journal of Parallel Programming, 42,* 739–754.
23. Mann, Z. A., & Szabo, M. (2017). Which is the best algorithm for virtual machine placement optimization. *Concurrency and Computation: Practice and Experience, 29*(10), e4083.



Miss Nagma Khattar is currently pursuing her Ph.D. in the domain of cloud computing. Her research work is on energy aware scheduling in cloud. Her research interests include energy efficiency, green computing, designing of scheduling algorithms and internet of things.

Dr. Jaiteg Singh holds a Ph.D. in Computer Science and Engineering with 12 years of experience in Research, Development, Training, Academics at Institutes of Higher Technical Education. His areas of expertise are software engineering, business intelligence, data and opinion mining, cartography, curriculum design, pedagogical innovation and management. Areas of interest include sustainable software engineering, education technology, offline navigation systems and cloud computing.

Dr. Jagpreet Sidhu holds a Ph.D. in Computer Science and Engineering from Panjab University, Chandigarh. He has research publications in international journals and conferences of repute to his credit. His research interests include distributed systems, distributed security architectures, distributed services like grid, cloud and web services, social network analysis, privacy and trust related issues in distributed environments. His teaching interests include distributed computing, cloud computing, data communication and computer networks. He also teaches certified courses like CCNA, CCVP and RHCE.