Chapter 6
Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review

Md Anit Khan, Andrew Paplinski, Abdul Malik Khan, Manzur Murshed, and Rajkumar Buyya

6.1 Introduction

As envisioned by Leonard Kleinrock [1], Cloud computing has transformed the dream of "computing as a utility" into reality, so much so it has turned out as the latest computing paradigm [2]. Cloud computing is called as *Service-on-demand,* as Cloud Service Providers (CSPs) assure users about potentially unlimited amount of resources that can be chartered on demand. It is also known as *elastic computing,* since Cloud Service Users (CSUs) can dynamically scale, expand, or shrink their rented resources anytime and expect to pay for the exact tenure of resource usage under Service Level Agreements (SLA). Through such flexibilities and financial benefits, CSPs have been attracting millions of clients who are simultaneously sharing the underlying computing and storage resources that are collectively known as Cloud data centers.

However, as the number of CSUs is increasingly growing over time, the necessity of an automatic management system, referred to as Cloud Resource Management System (CRMS), has become imminent to manage the multitude of service requests of millions of CSUs in automatic fashion. Furthermore, several research unlock the fact that Cloud Data Centers (CDCs) consume a lot of electricity which on average is twenty-five thousand times more than a household's energy consumption.

Md.A. Khan (✉) • A. Paplinski • A.M. Khan
Faculty of Information Technology, Monash University, Clayton, VIC, 3168, Australia
e-mail: makha23@student.monash.edu

M. Murshed
School of Information Technology, Faculty of Science, Federation University Australia, Churchill, VIC, 3842, Australia

R. Buyya
Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Parkville, VIC, Australia

W. Rivera (ed.), *Sustainable Cloud and Energy Services*,
DOI 10.1007/978-3-319-62238-5_6

Subsequently, more carbon dioxide (CO_2) emission is driven by these data centers. A study has revealed that energy consumption of Cloud causes more carbon emission to that of two countries, Netherlands and Argentina [3]. In [4], author has highlighted that worldwide energy usage of cloud data centers climbed up to 56% between 2005 and 2010, and was projected to be between 1.1 and 1.5% of the total electricity use in 2010 [5]. Moreover, approximate carbon emission by IT industry is 2% of the global emissions which is equivalent to the emissions of the aviation industry [6]. Consequently, many developed countries are getting more concerned these days to reduce carbon emission [7]. Beyond that, Koomey [8] projected that energy consumption of CDCs would remain to rise rapidly without energy-efficient resource management solutions being applied. As such, development of energy-efficient CRMS is extremely crucial.

VM Consolidation (VMC) is one such technique incorporated in CRMS to increase the energy-efficiency of Cloud. Hardware failure of existing Physical Machines (PMs) and addition of new PMs are continuous events in data center. Furthermore, resource requirement to accomplish the remaining tasks of existing service requests evolves with the course of time. Hence, as time progresses, remapping of remaining workload to currently available resources become inevitable to uphold the optimization of Cloud resource usage. The VM consolidation technique is applied to remap the remaining workloads to currently available resources which opts to migrate VMs into lesser number of active PMs, so that the PMs which would have no VM can be kept into sleep state. Since, energy consumption by the PM which is in sleep state is significantly lower than the energy consumption by the PM which is in active state, therefore, VM consolidation minimizes energy consumption of Cloud data center.

As portrayed in Fig. 6.1, before VMC is applied, VMs are scattered in multiple PMs. VMC migrates the VMs from lower utilized PMs to higher utilized PMs and thus consolidate VMs in lesser number of PMs than before. Meanwhile, the state of those PMs having no VMs can be changed from active state (i.e., turned on state) into a lower energy consuming state, such as sleep state and consequently, energy consumption can be minimized. The more number of VMs are placed in one single PM, the lesser becomes the overall energy consumption with the lesser number of active servers. Moreover, on account of compacting more number of VMs into fewer number of PMs, *resource utilization ratio* of the PM P_i, R_{P_i} (6.1) would become higher which in turn would increase the *mean resource utilization ratio of CDC*, $\overline{R}_{\text{CDC}}$ (6.2) where N is the total number of active hosts in CDC.

$$R_{P_i} = \frac{\text{Utilized Amount of Resource of } P_i}{\text{Total Amount Resource of } P_i} \tag{6.1}$$

$$\overline{R}_{\text{CDC}} = \frac{1}{N} \sum_{i=1}^{N} R_{P_i} \tag{6.2}$$

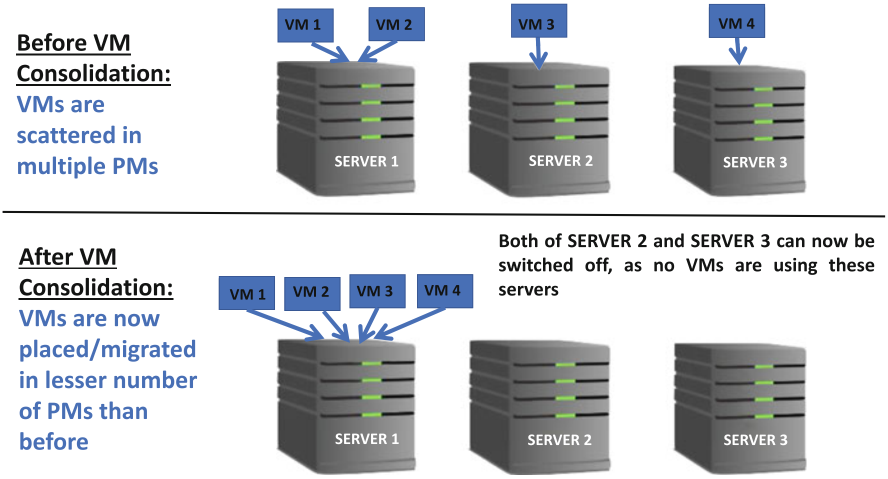

Fig. 6.1 VM consolidation

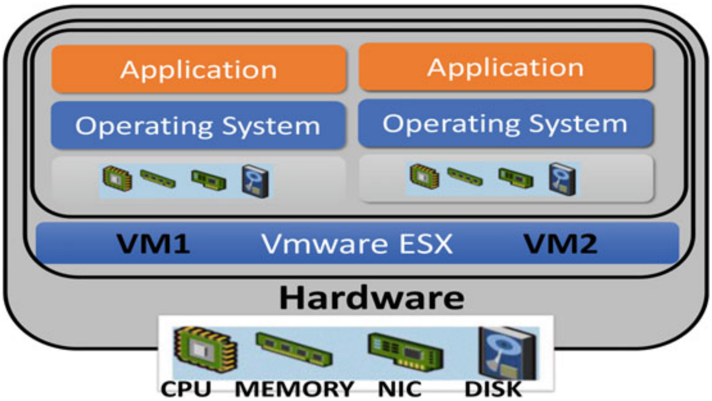

Fig. 6.2 Multiple VMs are hosted in a single physical machine [9]

However, as delineated in Fig. 6.2, VMs hosted in a PM share the underlying physical resources of that PM. Therefore, with the increased number of VMs sharing underlying resources of a single PM, waiting time for a VM prior receiving its required resources becomes higher. Thus, if more VMs are placed in a single PM, resource contention may arise which would lead towards poor QoS. Consequently, possibility of Service Level Agreement (SLA) violation arises.

To balance the tradeoff between QoS and energy-efficiency, it is extremely challenging to design such VMC algorithm which increases both resource utilization and energy-efficiency without compromising the QoS of running applications as agreed with respective CSUs during SLA. Recently, VMC has attracted interest of Cloud researchers, while designing efficient VMC algorithms is extremely challenging as it needs to be *scalable*, to the millions of VMs and PMs, as well as *robust*, such that the performance does not degrade with the fluctuation in resource demand of

VMs. In this paper, we have presented detail discussion on a wide range of VMC algorithms which complements three extremely important challenging aspects of Cloud: resource utilization maximization, energy consumption minimization, and Cloud profit maximization. In brief, our contributions are as follows:

- A group of surveys on VMC algorithms [10–20] have greatly assisted our research with VMC algorithms. However, VMC is an extremely popular research area and none of the available survey papers have presented discussion on VMC algorithms published in 2016 and onwards. Therefore, in this paper, we have primarily focused on most recent VMC algorithms published in 2016 and onwards which are not available in any other survey papers.
- We have presented critical review of different types of VMC algorithms which is missing in the available literature.
- There are several existing VMC algorithms proposed before the year of 2016 which strongly influenced subsequent researchers through introducing unique research directions. Those researches presented their own distinct techniques which are strong enough to be used as classification criterions. Therefore, we have also included elaborate discussion on such prominent VMC algorithms proposed before the year of 2016 to clarify the important concepts based on which we have reached our own classification of VMC algorithms.
- The authors of [16] have mentioned that presenting a survey and classification of VMC algorithms with an equal justice to all viewpoints is hardly possible. In the light of such admitted belief, we have proposed our own analysis and classification of existing VMC algorithms with an emphasis towards incorporation of future resource demand of Cloud resources, since considering the future resource demand is essential to prevent the SLA violation which is one of the major drawbacks of VMC algorithms.
- Based on our literature review, we have pointed out potential important research scopes which have not been explored so far.

The rest of the paper is organized as follows. In Sect. 6.2, we discuss the fundamental components of VMC. In Sect. 6.3, we present classification of VMC algorithms, followed by the extended classification and critical review of Dynamic VMC algorithms in Sect. 6.4. Next, in Sect. 6.5, we highlight the details of the notable contemporary VMC algorithms. Finally, we summarize our findings along with potential research directions in Sect. 6.6.

6.2 Fundamental Components of VMC

VMC algorithm is comprised of three core components [4, 21] which are as follows:

- *Source Host Selection*: First, among all the PMs, a set of PMs are selected from where VMs are migrated out. The *Source Host Selection* component takes all the PMs and VMs as input and selects one or more PMs as source PM(s) from where VMs would be migrated out.

- *VM Selection*: Second, one or more VM(s) are selected for migration from a source PM. The *VM Selection* component takes the PM as input which has been selected by *Source Host Selection* component and selects one or more VMs from that source PM for migration into a different PM.
- *Destination Host Selection/VM Placement*: Finally, the *Destination Host Selection/VM Placement* component selects a PM for each of the migrating VM which was selected by *VM Selection* component.

However, after a VM is created for the first time (i.e., for new VMs), the initial placement of that newly created VM can also be considered as VMC, if the corresponding destination PM is selected with an aim to minimize the total number of active PMs and increase the resource utilization, given that the hosting PM has adequate resources to fulfill the resource demand of the new VM. Hence, for new VMs, only *Destination Host Selection Algorithm/VM Placement Algorithm* does the VMC, as no source host selection and VM selection are needed for new VMs. A number of VM placement algorithms are available in the literature. In this paper, we have covered both VMC algorithms and VM placement algorithms, considering VM placement algorithms as VMC algorithms in case of new VMs. In the following section, we have presented the classification of VMC algorithms.

6.3 Classification of VMC Algorithms

In Fig. 6.3, we have delineated the classification of VMC algorithms. VMC algorithms can be broadly classified into two groups:

- *Dynamic VMC (DVMC) Algorithm*: In Cloud, received workloads are run in VMs, while these VMs accomplish the assigned workload through consuming the resources of the respective hosting PMs. With the advancement of time, progression of previously accepted workloads continues, while at the same time, new workloads keep being accepted by CSP. Furthermore, removal of some PMs due to hardware failure and addition of new PM also takes place. Thus, the overall workload with corresponding resource requirement and resource availability in the CDC keeps evolving over time. In DVMC algorithm, current VM-to-Server assignment is taken into consideration in the VMC process. Note that the workload or resource requirement of any VM and its location (i.e., its hosting PM) can be dynamic, as it changes with time. If the VMC algorithm consolidates VMs considering the dynamic (i.e., changing) workload and location of the VM (i.e., current VM-to-Server assignment), then it is called DVMC algorithm. In simple words, DVMC algorithms provide the solution of reallocation of existing VMs in lesser number of PMs such that the number of active PMs is minimized.
- *Static VMC (SVMC) Algorithm*: In contrast to DVMC algorithm, Static VMC (SVMC) algorithms, also referred to as consolidated VM placement algorithms, do not consider the current VM-to-Server assignment while choosing a new destination PM for any VM. In [22], the authors have mentioned that static

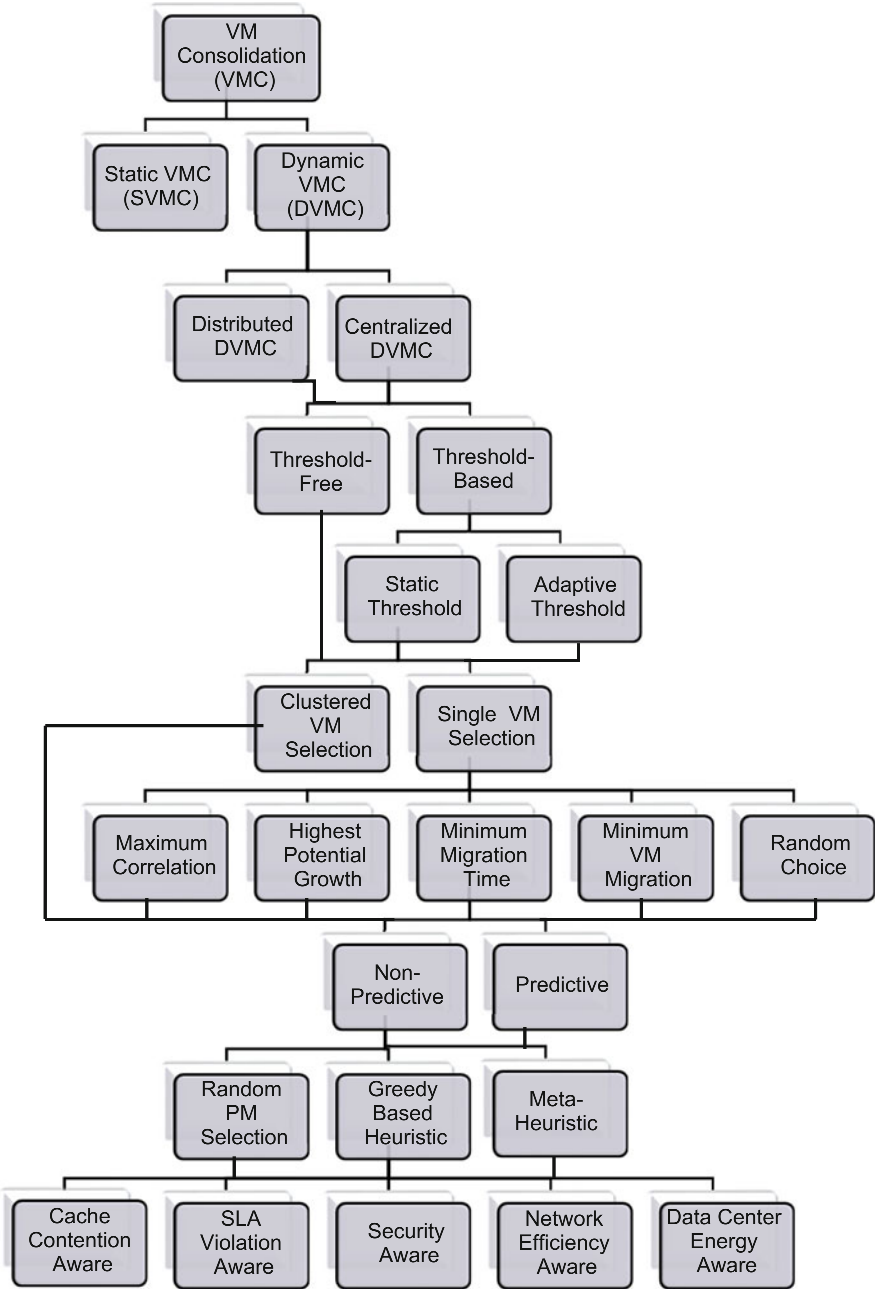

Fig. 6.3 Classification of DVMC algorithms

VMC algorithms work with a set of fully empty PMs and a set of VMs with specific resource requirement. In simple words, static VMC algorithm provides the solution of initial VM placement in minimum number of active PMs so that energy-efficiency and resource utilization of CDC increase. However, it does not provide the solution for reallocation of VMs in new PMs considering the current VM-to-Server assignment. Since, the dynamic (i.e., changing) load and placement of VMs are not considered, therefore, it is called as SVMC algorithm. [23–26] are examples of SVMC algorithm which do not consider the current VM-to-Server assignment while choosing a new destination PM for a VM.

Energy-efficiency of CDC would be hampered without the initial consolidated VM placement, as provided by SVMC algorithms. Besides, the energy-efficient initial placement keeps VMs consolidated in fewer PMs from the very beginning and consequently, the intermediate period before the awakening of the necessity to reallocate VMs can be prolonged. VMC has network overhead and it hampers QoS due to inherent service downtime. The prolonged period between initial VM placement and VMC or between two consecutive VMC reduces the overhead of VMC. However, the dynamic VM-to-Server assignment is not taken into consideration in SVMC algorithm. Therefore, the migration cost of a VM from its current hosting PM to its new destination PM is ignored. Consequently, SVMC algorithms are only applicable for initial placement of VMs or migrating VMs of one CDC into another CDC. As the time progresses, both workload and resource availability change in CDC. Therefore, apart from the initial consolidated VM placement, DVMC is one of the key techniques that uphold the energy-efficiency, resource usage optimization, and profit maximization of CSPs. As such, in this paper we have focused on DVMC algorithms. In the following section the classification of the DVMC algorithms has been presented.

6.4 Classification of DVMC Algorithms

DVMC problem focuses on run-time environments where VMs are active and already hosted by servers in the data center. Consolidation of such VMs is achieved by the VM live migration operations, where a running VM is relocated from its current host to another server while it is still running and providing service to its consumers [27]. DVMC algorithms can be classified into two groups:

- *Centralized DVMC Algorithm*: As proposed in [28–30], in centralized architecture, there is a single controller which has the information about present resource availability of all the PMs. The controller runs the centralized VMC algorithm which selects a destination PM for a migrating VM considering the resource availability of all the PMs.
- *Distributed DVMC Algorithm*: Instead of having a single controller which poses the information of present available resource availability of all the PMs and selects a destination PM for any migrating VM considering that information; in distributed architecture, PMs exchange information of their present resource

availability with their own neighbor PMs and thus, each PM has the resource availability information of its neighborhood PMs. If a PM wants to migrate out one of its VMs, it executes distributed VMC algorithm to select one of the neighbor PMs as the destination PM where the migrating VM would be hosted. Examples of distributed DVMC algorithms are [31, 32].

Majority of the DVMC algorithms found in the literature are centralized DVMC and only a few Distributed DVMC [31, 32] has been proposed. In [31], the authors have presented their distributed DVMC algorithm for a P2P network oriented CDC. According to [31, 32], the growing number of PMs becomes a bottleneck for centralized VMC at the time of selecting a destination for any migrating VM, since the asymptotic time complexity of the centralized DVMC algorithm is proportional to the number of PMs in the CDC, whereas the number of potential PMs to choose from a migrating VM is relatively small in distributed DVMC. Thus, distributed DVMC is more scalable for CDC with huge number of PMs. However, distributed DVMC has message passing overhead, as every PM must update its present resource availability to all of its neighbors. Every time a VM is migrated, both the sender PM and the destination PM must update their present resource availability status to all of their neighbors. Besides, a central monitoring system is indispensable in Cloud which monitors the accepted workload progression status and allocate/deallocate resources accordingly to accomplish the workload in time. Furthermore, at the time of accepting new workload, the overall resource availability status of CDC must be known, so that accurate decision can be made on whether the new workload would be possible to serve within deadline. Therefore, centralized DVMC can be implemented without adding any additional resources. Moreover, message passing as required by distributed DVMC algorithms increases network overhead, decreases network throughput, and increases network related energy consumption. Hence, centralized DVMC is more energy-efficient compare to distributed DVMC algorithms.

As discussed earlier in Sect. 6.2, first component of DVMC algorithm is to select the source PM. A DVMC algorithm can either randomly select a source PM from where one or more VM(s) are migrated out or VM(s) can be selected from over-utilized and under-utilized PMs. In the following section, we have presented our classification of DVMC algorithms based on different source PM selection techniques.

6.4.1 Classification of DVMC Algorithms Based on Different Source PM Selection Techniques

Based on the way source PMs are selected, DVMC algorithms can be classified into two groups:

- *Threshold-Based DVMC Algorithm*: Threshold-based DVMC algorithms use upper and lower threshold values to identify a PM as overloaded or underloaded, respectively, from the perspective of resource utilization ratio of the PM P_i, R_{P_i}

(6.1). The key point is that R_{P_i} is compared against the values of some thresholds which can be either static or adaptive (explained in detail later in this section). As presented in [33], if R_{P_i} goes past the upper utilization threshold value, then P_i is identified as overloaded or over-utilized PM and VMs are migrated out from P_i, until R_{P_i} becomes lower than the upper-threshold, since high R_{P_i} is a strong indicator of potential QoS degradation or SLA violation which is arisen because of the higher resource demand of the hosted VMs. Again, if R_{P_i} is smaller than the lower utilization threshold value, then P_i is identified as underloaded or under-utilized and potential destination PMs are looked for, where VMs of P_i can be migrated out so that P_i can be put in sleep state.

- *Threshold-Free DVMC Algorithm*: Unlike threshold-based DVMC algorithms, in threshold-free DVMC algorithms, resource utilization ratio of the PM P_i, R_{P_i} (6.1) is not compared against any threshold value to identify the PM as overloaded or underloaded. Instead, the source PMs are selected either randomly [34] or some functions are applied to favor PMs having either higher or lower R_{P_i} compare to those of other PMs. Examples of threshold-free DVMC algorithms are [25, 34, 35].

In [35], the authors have proposed two functions, one of which is used to select a source PM from where VMs would be migrated out and the other function is used to select destination PMs for those migrating VMs. The destination selection function favors PMs with neither higher nor lower R_{P_i}, while the source selection function favors PMs with lower R_{P_i}. It is noteworthy that no threshold value is used against which a PM's utilization is compared to identify it as over-utilized or under-utilized. Since, according to the proposed method, the PMs with higher R_{P_i} are not selected as source PMs to migrate out VMs; therefore, the QoS degradation or SLA violation due to increased resource demand of hosted VMs may take place.

The main difference between threshold-free and threshold-based DVMC algorithm is that threshold-based approach identifies a PM as overloaded or underloaded with respect to some thresholds, whereas threshold-free approaches select PMs either randomly or based on the lower or higher resource utilization ratio in comparison to those of rest of the PMs. As opposed to random source PM selection, the heuristic approach to always select the source PM with highest or lowest resource utilization ratio may not ensure the achieving of global best solution. Since, there is no random source PM selection policy in threshold-based DVMC algorithm; therefore, it may not provide the global best solution. Furthermore, compare to threshold-free approach, the number of VM migrations may become higher in threshold-based approach. To illustrate more, assume that overall workload in the CDC has become high. PMs would then experience high utilization as per threshold-based approach and would start migrating out their VMs, whereas threshold-free approach would consider the global picture of increase of workload in all PMs and may avoid VM migrations. However, VMC is a combinatorial optimization problem. In other words, it is a rearranging problem of VMs into different PMs, so that VMs can be placed in minimum number of PMs without violating the resource capacity of any PM and the total number of possible combinations is N^M, where N and M denote the total number of VMs and PMs, respectively. Hence, the solution search space would remain as humongous as N^M, if random selection policy is followed as per threshold-free approach. In contrast, although the solution may not be globally optimal, but selecting source PM based on upper or lower resource utilization threshold would confine the solution search space and would provide sub-optimal solution in faster time, since minimization of lower resource utilization certainly decreases the energy consumption.

From the literature review, threshold-based approaches are found as highly popular among researchers. Based on the type of used thresholds to identify a PM as overloaded or underloaded, threshold-based DVMC algorithms can be classified into two groups:

- *Static Threshold-Based DVMC Algorithm*: In static threshold-based DVMC algorithms, fixed values or predefined values are used as upper and lower thresholds to identify the PM as overloaded or underloaded. As the values of the thresholds do not change over time, therefore they are referred to as static thresholds. Examples of static threshold-based DVMC algorithms are [31, 33, 36]. In [36], the authors have used 100% CPU utilization as upper utilization threshold and 50% CPU utilization as lower utilization threshold. In other words, if the CPU utilization of a PM is found as 50%, then that PM is considered as lower utilized PM and VMs are migrated out from that PM into other PMs. Similarly, if the total resource demand of all the VMs hosted in a particular PM is found as higher than the CPU capacity of that PM, then that PM is considered as overloaded PM and VMs are migrated out from that PM into other PMs.
- *Adaptive Threshold-Based DVMC Algorithm*: In this case, the values of the thresholds based on which the PM is selected as overloaded or underloaded change dynamically as the resource utilization ratio the PM P_i, R_{P_i} (6.1) changes with time. In other words, the threshold value adapts with the change of resource utilization. Examples of adaptive threshold-based DVMC algorithms are [21, 29, 37, 38].

The authors of [21] are pioneers in proposing adaptive threshold-based DVMC algorithm, as they proposed a number of adaptive thresholds based on which a PM is detected as overloaded. One such adaptive threshold is referred to as Median Absolute Deviation (MAD). To illustrate more, let, $T = t_j \quad t_j$ denotes time j and $j \in \mathbb{N}$ where $\mathbb{N}$ is the set of positive integers and $R_{P_i}^{t_j}$ is the resource utilization ratio the PM P_i at time t_j. For each PM P_i, R_{P_i} (6.1) across different time (i.e., $R_{P_i}^{t_1}$, $R_{P_i}^{t_2}$, $R_{P_i}^{t_3}$, and so forth) would be recorded and then MAD is calculated using (6.3), while the upper utilization threshold T_u is calculated using (6.4), where $s \in \mathbb{R}^+$ a parameter which defines how strongly the system tolerates host overloads.

$$\mathrm{MAD} = \mathrm{Median}\left(\left|R_{P_i}^{t_j} – \mathrm{Median}\left(R_{P_i}^{t_j}\right)\right|\right) \tag{6.3}$$

$$T_u = 1 - s \cdot \mathrm{MAD} \tag{6.4}$$

The lower the value of s, the system is more tolerant to variation in resource utilization. If the current $R_{P_i}^{tj}$ is found as greater than T_u, then P_i is considered as overloaded [4]. Note that the value of MAD (6.3) is not fixed or static, as $R_{P_i}^{tj}$ changes with time and hence, T_u (6.4) also changes with the change in resource utilization.

As mentioned before, VMC algorithms aim to increase the resource utilization which helps to minimize the energy consumption. The high value of upper and lower utilization thresholds increases resource utilization and energy-efficiency. However, very high resource utilization or higher R_{P_i} (6.1) causes QoS degradation or potential SLA violation. Therefore, the higher is the upper utilization threshold, the higher is the SLA violation. Hence, there is a tradeoff between energy-efficiency and SLA violation, while the values of upper and lower utilization thresholds have great impact in controlling such tradeoff. Therefore, a balance is needed to be maintained between energy-efficiency and SLA violation through controlling the threshold values.

The key idea of using upper and lower static thresholds is to keep the resource utilization restricted into a certain range (i.e., in between upper and lower utilization threshold), so that a balance exists between energy-efficiency and SLA violation. However, the workload pattern as experienced by an application running inside of a VM changes over time. Besides, multiple VMs which are all hosted in a single PM may exhibit different workload pattern. Since, the threshold is static and it cannot be changed with the change of workload, therefore, SLA violation increases with the increase of workload.

Adaptive threshold-based DVMC algorithms partially mitigate this problem by changing the utilization threshold values with the variation in workload. For instance, as the workload grows in VMs, MAD (6.3) increases and the upper utilization threshold (6.4) becomes lower accordingly. Consequently, VMs are migrated out from P_i before R_{P_i} (6.1) reaches to a very high level and as a result, VMs of P_i do not suffer from degraded QoS due to high R_{P_i} (6.1). In general, compared to static threshold-based approach, adaptive threshold-based approach decreases SLA violation rate more. However, it provides less energy-efficiency than static threshold-based approach, since the lower is the upper and lower utilization threshold, the lower is the energy consumption minimization. Apart from that, adaptive-based approach causes more number of VM migration than static-based approach which increases both energy consumption and SLA violation.

Thus far, we have reviewed different types of threshold-based DVMC algorithms and previously in Sect. 4.1, we have presented our comparison between threshold-free DVMC algorithm and threshold-based DVMC algorithm. As highlighted in Sect. 6.2, one of the core components of VMC algorithm is VM selection. In the following section, we have presented our discussion on different DVMC algorithms with different VM selection policies.

6.4.2 Classification of DVMC Algorithms Based on VM Selection Policy

Once source PMs are selected, the following step of VMC is to select one or more VM(s) from source PM to migrate out. Based on the different VM selection policies used in different DVMC algorithms, the DVMC algorithms can be broadly classified into two groups:

- *Clustered VM Selection (CVS)*: Multi-layered applications comprised with load balancer(s), application server(s), and database server(s) are hosted in Cloud, while individual VM is assigned with the functionalities of each layer. To explain more, database server instance(s) are deployed in one or more VMs, while load balancer(s) are run in separate VMs and application server(s) are run in different VM(s). All these VMs which are under one application communicate with each other and the performance of the application becomes immensely hampered if these VMs are not hosted in nearby PMs. As such, instead of migrating a single VM, such group of VMs of an application, also known as Clustered VMs, are considered for migration. Prominent examples of such clustered VM selection algorithms are [24, 39].
- *Single VM Selection (SVS)*: Unlike CVS, SVS algorithms select a single VM to migrate out. Different single VM selection strategies as found in the literature are mentioned in the following:
  - *Random Choice (RC)*: Among all the VMs residing in the source PMs, a VM is randomly selected [33, 40]. Random VM Selection can select a VM in O(1) time which is faster than rest of the approaches.
  - *Minimization of VM Migration (MVM)*: Minimum number of VMs are migrated to make the current resource utilization of a PM lower than the upper utilization threshold. MVM algorithm as proposed by [33] first sorts the VMs in descending order with respect to CPU demand and then selects the VM that satisfies the two criterions: First, the VM's CPU utilization should be higher than the difference between the host's present overall CPU utilization and the upper utilization threshold; Second, that VM is selected for which the difference between the upper-threshold and the new utilization is the minimum compare to the values provided by all the VMs. If no such VM is found, then the VM with the highest utilization is selected and the process is repeated until the new utilization becomes lower than the upper utilization threshold.
  - *High Potential Growth (HPG)*: The VM with lowest ratio of actual resource usage to its initial claimed resource demand is selected [33]. Asymptotic running time of the algorithm is O(n).
  - *Minimization of Migration Time (MMT)*: The VM which requires minimum time to complete the migration is selected for migration, while the migration time is estimated as the amount of RAM utilized by a VM divided by the spare network bandwidth available for the hosting PM [21]. Asymptotic running time of the algorithm is O(n).

- *Maximum Correlation (MC)*: VM that has the highest correlation of the resource utilization with other VMs is selected [4]. Multiple Correlation Coefficient as proposed by [41] is used to determine the correlation between the resource utilization of VMs.

The RC may help to find the globally optimal solution, if source PM is selected randomly using threshold-free approach as discussed earlier in Sect. 4.1. Furthermore, RC shows the fastest run time among all the above-mentioned VM selection algorithms. However, if the solution space is confided prior, such as source PM is selected using the heuristic that the PM with highest or lowest resource utilization would be the source PM and after then that, a VM is randomly chosen from that PM, then RC cannot provide the global optimal solution. As oppose to that, after confining the solution search space by selecting source PM based on highest or lowest resource utilization ratio, rest of the heuristics such as MVM, HPG, MMT, and MC are more likely to decrease the energy consumption and SLA violations compare to RC; since, heuristics like MVM, HPG, MMT, and MC certainly provide the local best solution, whereas RC probabilistically chooses a solution which may not be locally optimal. To illustrate more, VMs experience degraded QoS during the period of migration. Therefore, selecting the VM which would take the least migration time (i.e., MMT) would certainly assist in keeping the SLA violation lower [42]. In contrast, RC may choose a VM with higher migration time. Consequently, SLA violation rate would be higher for RC.

MVM minimizes the number of VM migrations with minimal decreasing of resource utilization ratio of the PM P_i, R_{P_i}, (6.1). As a result, *mean resource utilization ratio of CDC* R_{CDC} (6.2) would remain as higher compare to rest of the above-mentioned algorithms and thus it turns out to be more energy-efficient. However, higher R_{P_i} (6.1) may cause degraded QoS and more SLA violation. Hence, MVM shows lower SLA violation than MTM.

Another critical issue with MVM is that VMs need to be sorted first with respect to resource utilization, as otherwise the asymptotic running time is exponential. However, it is not possible to sort the VMs with respect to resource demand, since a VM has three different types of resource demand, such as CPU demand, memory demand, and network bandwidth demand which are not related. For instance, a VM may have high CPU demand and low network bandwidth demand, whereas another VM may have low CPU demand and high network bandwidth demand. Therefore, it is not possible to sort VMs based on VM resource demand, since one distinctive feature of CDC is location transparency [43] which arises from the fact that a VM can be placed in any of the PMs. Hence, a PM may have VMs with varied resource demand with respect to different resource types [44].

Because of such diverse resource utilization value of a VM across various types of resources, it is not possible to select a VM with highest potential growth ratio (i.e., ratio of actual resource usage to a VM's initial claimed resource demand) among all the VMs across all resource types. Consequently, HPG is only possible to be implemented considering one resource type, such as CPU or memory or network bandwidth and thus, it does not ensure the minimization of energy-efficiency or

SLA violation. Similar issue exists with MC algorithm. In contrast, since MTM primarily selects VM based on memory size and hence, it is free from such issue.

Thus far, we have analyzed different DVMC algorithms with different VM selection policies. Another critical distinguishing aspect among DVMC algorithms is that whether estimated future resource demand has been considered in the VMC process or not, as we have presented our discussion about it in the following section.

6.4.3 Classification of DVMC Algorithms based on Consideration of Estimated Future Resource

From the literature, it can be viewed that consideration of estimated future workload in a PM is commonplace in a wide range of DVMC algorithms. However, there are still numerous DVMC algorithms which make the consolidation decision based on the current resource utilization of PMs instead of the estimated future resource utilization. Consideration of future can create a significant difference on the performance of VMC algorithm, compare to those VMC algorithms which takes the decision based on the current resource utilization. Hence, in this section, we have reviewed both types of VMC algorithms from that perspective.

– *Non-Predictive Dynamic VMC Algorithm (NPDVMC)*: Instead of considering the estimated future resource utilization of the PM, NPDVMC algorithms consider the current aggregated resource demand of VMs. Note that the aggregated resource demand of hosted VMs in the PM is equal to the resource utilization of that PM. VM migration decisions are taken when the current resource utilization of the PM P_i, R_{P_i} (6.1) becomes very high or very low so that SLA violation can be avoided or energy consumption can be minimized.

  One such example of NPDVMC algorithm is [28], where source and destination PMs for consolidation of VMs are selected based on the current resource utilization status of the PM. If the current R_{P_i} is found as equal or greater than 90%, then P_i is considered as overloaded or over-utilized and VMs are migrated out from P_i. Again, if current R_{P_i} is found as equal or lower than 10%, then P_i is considered as overloaded or over-utilized and VMs are migrated out from P_i to place in new PMs. Other prominent non-predictive VMC algorithms are [24, 25, 28–30, 34, 45].
– *Predictive Dynamic VMC Algorithm (PDVMC)*: On the contrary, PDVMC algorithms take the decision to migrate VMs from one PM to another PM considering the estimated future resource demand of VMs instead of current resource demand. Examples of PDVMC algorithms are [36, 46].
– In [36], both current and future resource utilization of the PM are considered while making the consolidation decision. Linear regression [38] is used to generate an estimated future resource utilization of a PM from analyzing its past resource utilization statistics. If the current resource utilization of the PM is found as higher than the upper-utilization threshold (explained previously in

Sect. 4.1), then that PM is identified as *overloaded*. Furthermore, although the current resource utilization of the PM is found as lower than the upper-utilization threshold, yet its estimated future resource utilization is found as higher than the upper-utilization threshold, then that PM is identified as *predicted overloaded*. Both *overloaded* and *predicted overloaded* PMs are elected as source PMs from where one or more VM(s) are selected to migrate out into new PMs. Again, both *overloaded* and *predicted overloaded* PMs are excluded from the list of potential destination PMs where migrating VMs would be placed.

– Consolidation of VMs considering the estimated workload is a more proactive approach than NPDVMC, as VMs are migrated out from those PMs which are predicted to be overloaded in future. The aim of such proactive approach is to move VMs out prior QoS degradation or SLA violation takes place. Consequently, compare to NPDVMC algorithms, PDVMC algorithms will display lower SLA violations due to less occurrences of resource contention. However, because of migrating more VMs out of the higher utilized hosts than NPDVMC, the *mean resource utilization ratio of CDC* $\overline{R}_{\text{CDC}}$ (6.2) would become lower and thus, total number of inactive PMs may become less for PDVMC. Hence, PDVMC would display lower energy consumption minimization than that of NPDVMC.

Another challenging aspect of PDVMC is that PDVMC relies on prediction techniques to estimate the future resource utilization of PMs. Predictive techniques are based on the correlation between the past history of the system behavior and its near future [4]. The efficiency of prediction-based techniques greatly depends on the actual correlation between past and future events and the quality of adjustment with a specific workload type. In Cloud environment, different VMs are hosted in a single PM, while these VMs are expected to exhibit different behavioral pattern from each other in terms of resource demand. Consequently, no single prediction technique would be a perfect fit for all PMs. A non-ideal prediction causes over- or underprediction which lead towards either inefficient resource usage or more SLA violation.

Until now we have reviewed diverse approaches to select source PMs and VMs, as we have also analyzed both prediction-based and non-prediction-based DVMC algorithms. One of the core components of DVMC algorithms is destination PM selection where migrating VMs are placed. This is also referred as *VM placement problem*. In the following section, we have presented our discussion on diverse approaches to select destination PMs for migrating VMs as incorporated in different DVMC algorithm.

6.4.4 Classification of DVMC Algorithms based on Destination PM Selection Strategies

Destination PM selection strategy plays a gigantic role in increasing the energy-efficiency of CDC. The aim of destination PM selection is to select such new PMs for migrating VMs so that the total number of active PMs becomes minimum without violating the resource constraint of any PM. However, destination PM selection component itself is a NP-hard problem and hence a number of heuristic as well as meta-heuristic algorithms have been proposed in the literature. Based on different destination PM selection strategies DVMC algorithms can be classified into three groups:

- *Random PM Selection (RPS)*: As proposed in [24], the destination PM is randomly selected from the suitable PMs for a given VM. The asymptotic running time of FF is O(n), where n is the total number of VMs.
- *Greedy Heuristic*: Greedy Heuristic algorithms are most widespread in the literature to select the destination PM for migrating VMs. Several popular heuristic-based algorithms are as follows:
  - *Random Choice (RC)*: As discussed in [47], RC algorithm randomly chooses a destination PM for a migrating VM which is already turned on. If no suitable PM can be found to host the migrating/target VM, then a new PM which was in sleep state is turned on to accommodate the target VM.
  - *First Fit (FF)*: In FF [47], PMs are ordered in a sequence and for each VM, the first available PM from the ordered list of PMs is selected. In other words, for every single VM, the searching of destination PM always starts from the first PM. If the first PM cannot accommodate a VM, then the second PM is checked and if the second PM cannot accommodate it, then the third PM is checked, as the searching continues to the next PM while always following the initial order of PMs until a suitable destination PM with adequate resource capacity is found. Since, a VM may have larger resource demand than the available remaining resource of a PM, therefore, the asymptotic running time of FF is O(nm), where n is the total number of VMs and m denotes the total number of PMs.
  - *First Fit Decreasing (FFD)*: FFD is same as FF, except the VMs are first sorted in the decreasing order of their resource demand. Then the destination PM for the first VM with highest resource demand is first searched using FF algorithm, as the searching continues for the VM with second highest resource demand, and so on. The asymptotic running time of FF is O(nlogn + nm), where n is the total number of VMs and m denotes the total number of PMs. Note that O(nlogn) is running time of sorting algorithm.
  - *Next Fit (NF)/Round Robin (RR)*: Like FF, NF also performs a sequential search, except it starts from the last server selected in the previous placement [39]. To explain more, if the last VM was placed in the second PM, then checking will start from the second PM for the following VM placement, and

so on [47], whereas in FF and FFD the checking would have always started from the first PM for any VM. NF is also referred to as *Round Robin (RR)*. The asymptotic running time of NF is same as FF.

– *Best Fit (BF)*: In BF, the PM with the minimum residual resource is selected as its destination PM [46]. The residual resource of the PM is the difference between the total resource capacity of that PM and the aggregated resource demand of the hosted VMs in it along with the resource demand of the target VM for which destination PM is under search. If PMs are first sorted based on resource utilization ratio (6.1), then running time of BF would be identical to that of FFD. However, if no sorting is applied, then the running time of BF would be O(nm^2).
– *Best Fit Decreasing (BFD)*: VMs are first sorted in the decreasing order based on their resource demand. Then the destination PM for the first VM with highest resource demand is first searched using BF algorithm, as the searching continues for the VM with second highest resource demand, and so on. The asymptotic running time of BFD is same as FFD.
– *Power Aware Best Fit Decreasing (PABFD)*: PABFD proposed by [33] is a modified version of BFD, as the VMs are first sorted in decreasing order based on their CPU demand and then the destination PM is selected with the least power increase compare to all the suitable PMs which could host the target VM. The asymptotic running time of PABFD is same as FFD.

RPS is most time efficient, but least optimal compare to rest of the above-mentioned VM selection strategies from the perspective of energy-efficiency, since it does not ensure to first choose a suitable PM from the set of currently turned on PMs, so that unnecessary waking up of PMs which are currently in sleep state can be avoided.

Unlike RPS, the similarity among all the above-mentioned heuristic-based destination PM selection algorithms is that all these algorithms first attempt to select a PM from one of the PMs which are already in turned on state, so that energy consumption can be minimized. However, the difference exists in their selection strategy to select a PM for a VM among all the existing turned on PMs. Although RC first tries to select a PM from the existing turned on PMs, yet, because of its random PM selection nature, it may cause sparse VM placement or VMs may be found as more scattered compare to those of FF, FFD, BF, and BFD. Hence, for many VMs, suitable PMs would not be found which would result in turning those PMs on which were in sleep state or in switched off state. Consequently, energy consumption would be higher for RC compare to rest of the heuristics.

The rationale of BF heuristic is that placing VMs on the PM with the least remaining available resource would provide other turned on PMs with large remaining available resources which can be used to support future larger VMs, while this strategy would concomitantly increase the *mean resource utilization ratio of CDC*, $\overline{R}_{\mathrm{CDC}}$ (6.2). Experimental results of [47] suggest that energy consumption is the highest for RC, while it is lowest for BF and BFD, as BF and BFD packs the VMs more tightly compare to RC, FF, and FFD.

Difference between BFD and PABFD is that BFD will select the smallest PM among all the suitable PMs for the first VM in terms of total resource capacity of PMs and then consolidating more VMs into it, whereas PABFD will initially select the most power-efficient PM among all the suitable PMs for the first VM and then consolidating more VMs into it. PABFD focuses on utilizing power-efficient PMs more which certainly has an impact on increasing the energy-efficiency of CDC. On the contrary, BFD leads towards utilizing the smallest PMs first and leaving larger PMs for future, while ignoring to ensure the usage of power-efficient PMs. With the increased number of VMs, larger PMs have to be turned on eventually. Hence, as opposed to BFD, since, PABFD ensures the more usage of power-efficient PMs, therefore PABFD is more energy-efficient compare to BFD.

- *Meta-heuristic*: Greedy Heuristic algorithms may become stuck with local minima or local maxima. Therefore, several meta-heuristic-based destination PM selection algorithms have been proposed in the literature which are discussed in the following:
  - *Evolutionary Algorithm*: The general steps of evolutionary algorithm are as follows:

    At each step (generation), the algorithm starts working with a population comprised of a range of solutions (members), while different evolutionary-based VMC algorithms use different heuristics to generate the initial solution.

    Next, a few members are first selected, also called parents from the generation to produce new solutions (children). Different evolutionary algorithms propose different methods to select parents from the population. One common method to select parents is to check value(s) of objective function(s) for each of the member and then select the members with higher values. The optimization functions are called as Pareto set and the values of the Pareto set achieved by the members are called as Pareto front [39]. For VMC, one prevalent object function is to maximize the number of physical servers with no VMs running in it or minimize the number of active physical servers.

    In order to produce a new solution (child), different parts collected from different parents are combined together. This technique is called mutation which takes place with a certain probability. The objective of mutation is the faster production of more optimized solutions.

    Furthermore, after mutation, swapping or interchanging (crossover) among different parts of a child takes place with a certain probability. The goal of crossover is to complement the faster creation of new children that are more optimized. In the context of VM placement or VMC, one widespread crossover technique is to interchange the hosts between two VMs [39].

    Through mutation and crossover, children are generated from parents which are added in the population.

    The entire process is repeated or more generations are run, until no improvements are found from consecutive repetitions of the algorithm.

Finally, a solution is chosen from the Pareto front based on an objective function.

– *Ant Colony Optimization (ACO)*: In ACO, a virtual ant selects a PM for a VM through considering two factors: Heuristic and Pheromone. Ants can either work sequentially or in parallel while constructing their own solutions. Each ant can either follow its own heuristic or all the ants can follow a common heuristic. The heuristic η is the key which guides to construct an optimal solution in aligned with the optimization function. Based on the diverse objective functions as presented in different ACO-based VM placement or VMC algorithms, the proposed η varies from one ACO-based VM placement or VMC to another. We have discussed about different heuristics previously. One common η found in a number of ACO-based VM placement or VMC is BFD [36]. Apart from η, the *Pheromone* plays a critical role in constructing an ant's solution which guides ants to find diverse solutions through exploring the search space. One key distinguishing aspect which makes ACO meta-heuristic different from heuristic-based algorithms is that some probability exists for an ant to choose the PM which is not optimal from the perspective of heuristic and thus stagnation into local minima or local maxima is avoided.
– *Simulated Annealing*: The authors of [35] have proposed a simulated annealing meta-heuristic-based VMC algorithm. In perturbation phase, instead of randomly choosing source or destination PMs, solutions are generated by selecting source PMs with lower utilization ratio and destination PMs with neither very high utilization ratio nor very low utilization ratio. Thus, VMs are consolidated in lesser number of active PMs. However, in order to avoid stagnation in local minima or local maxima, exploration is adopted through accepting solution that is even less optimal than the optimal solution found so far.

The aim of VMC is to minimize the energy-efficiency, as VMC minimizes energy consumption by placing more VMs in a single PM. The higher number of VMs is placed in a single PM, the higher is the minimization of energy consumption considering the energy spent after PMs. However, the higher number of VMs is placed in a single PM, the higher is the probability of QoS degradation or SLA violation. Hence, minimization of energy consumption and minimization of SLA violation are two confronting goals. While most researchers have focused to maintain a balance between minimization of PMs' energy consumption and SLA violation, some researchers have considered other aspects too, such as security, energy consumption by network, network throughput, and so forth. In the following section, classification of VMC algorithms based on their objectives has been presented.

6.4.5 Classification of DVMC Algorithms based on Different Objectives

The different DVMC algorithms with diverse objectives as observed in the literature are as follows:

- *SLA Violation Aware*: Since VMs share the underlying physical resources of their hosting PM such as CPU, RAM, and network bandwidth, therefore, the waiting time to receive the required resources for each VM increases with the increase of number of VMs in a single PM. Besides, the services which the VM is providing to its users' needs to be suspended temporarily at the time of migrating that VM from one PM to another PM. Hence, VMC causes SLA violation. Many VMC algorithms focus to minimize such SLA violation by limiting the number of VM migrations as well as minimize resource oversubscription and thus decrease SLA violation. [24] is an example of SLA violation aware VMC algorithms.
- *Security Aware*: Cloud is a multi-tenant environment, where VMs of different clients are hosted in same PM, while these VMs also share the underlying physical resources. Hence, security is one of the major challenging aspects in Cloud. In [28], the authors have proposed a security-based DVMC algorithm.
- *Network Efficiency Aware*: Such algorithms, for instance, [45] aim to uphold the network efficiency through considering different network related aspects, such as minimization of network energy consumption, and reduction of network congestion. [45, 48] are examples of network efficiency aware VMC algorithm.
- *Data Center Energy Aware*: CDC is the physical backbone of any Cloud-based services. One big challenging aspect of any data center is that an appropriate temperature must be always maintained, as the challenge escalates with the increase of the volume of data center along with the growth of number of PMs, network devices, and so forth. Cooling of CDC is extremely crucial to ensure the smooth and continuous functioning of PMs, routers, switches, and so forth. However, energy spending after cooling of CDC is very high and such energy requirement rises with the increase of quantity of VMs as well as with the increase of VMs' resource demand. Therefore, researchers have presented VMC algorithms which consolidate VMs in such a way that energy spending after cooling the data center can be minimized. [34] is an example of such VMC algorithm which minimizes the energy related to data center cooling.
- *Cache Contention Aware*: Cache contention refers to the situation that a VM may experience extra cache misses, as other VMs co-located on the same CPU fetch their own data into the Lowest Level Cache (LLC), which forces to evict the VM's data from the LLC and later fetching back that VM's data into the LLC again causes cache misses to other co-located VMs. In order to minimize cache misses due to VMC, [26] has proposed a cache contention aware VMC algorithm that considers the expected cache misses at the time of destination PM selection for a migrating VM.

Until now, we have reviewed different types of VMC algorithms, as we have highlighted the comparison with strengths and weakness of each of those techniques. We have delineated the classification of VMC algorithms in Sects. 6.3 and 6.4. In order to present further details of contemporary VMC algorithms, we have discussed about different aspects of recent VMC algorithms in the following section.

6.5 Detailed Analysis of Contemporary VMC Algorithms

Table 6.1 illustrates the most significant aspects of the notable recent research works on VMC as found in the published materials. The description of the attributes which are being considered to review the existing VMC algorithms is as follows:

- Research Project: Name of the reach project.
- Type of VMC: Whether the VMC is SVMC or DVMC.
- VMC Decision Process: If destination PM selection decision is taken centrally (i.e., centralized) or a source PM itself chooses another destination PM where the migrating VM will be placed (i.e., distributed).
- Resource Considered: Which resources (i.e., CPU, memory, network bandwidth, and so forth) are considered in the VMC algorithm.
- Source PM Selection Strategy: What the source PM selection strategy is and whether any threshold has been applied to select source PMs.
- VM Selection Criteria: What VM selection algorithm has been used.
- Application of Prediction Technique: Whether any prediction technique has been incorporated in the proposed system to predict the future resource utilization of PMs.
- Destination PM Selection Strategy: What algorithm has been used to select the destination PM for the migrating VMs.
- Performance Evaluation Technique: What technique is used to evaluate the performance of the proposed system.
- Objective: What are the aspects that the algorithm aims to optimize.

6.6 Conclusion and Future Directions

Although, CDC is the hardware backbone of Cloud-based services, all the undertaken operations in CDC are originated and managed by the Cloud Orchestration software [66], also known as CRMS. CRMS is the key enabler of all types of Cloud centric services rendered towards CSUs. One major concern with CDC is that CDC consumes humongous amount of energy. Energy consumption of CDC is impossible to reduce, if this aspect is not carefully considered while designing the core modules to fulfill the functionalities of CRMS. One of the core functionalities of CRMS is dynamic load balancing. However, the authors of [67] proposed that energy-

Table 6.1 Different aspects of the notable recent VMC algorithms

Research project name	VMC type	VMC decision process	Resource considered	Source PM selection strategy	VM selection criteria	Application of prediction technique	Destination PM selection strategy	Performance evaluation strategy	Objective
Security aware and energy-efficient virtual machine consolidation in cloud computing systems [28]	DVMC	Centralized	CPU	Static and adaptive threshold based	RC, MMT, BPG, MC	Non-predictive	Greedy heuristic	Simulation using CloudSim [49]	Security aware
Dynamic virtual machine consolidation for improving energy-efficiency in cloud data centers [29]	DVMC	Centralized	CPU	Adaptive threshold based	RC, MMT, BPG, MC	Non-predictive	Greedy heuristic	Simulation using CloudSim	Network efficiency aware
Dynamic routing and virtual machine consolidation in green clouds [30]	DVMC	Centralized	CPU	Adaptive threshold based	RC, MMT, HPG, MC	Non-predictive	Greedy heuristic	No performance evaluation	Network efficiency aware
Hierarchical, portfolio theory-based virtual machine consolidation in a compute cloud [24]	DVMC	Centralized	Single resource type which is presented by a numerical value	Adaptive threshold based	RC, MMT, HPG, MC	Non-predictive	Greedy heuristic	Simulation-based performance evaluation	Minimization of intercluster VM migration
Joint VM switch consolidation for energy-efficiency in data centers [45]	DVMC	Centralized	CPU	Threshold-free approach	CVS	Non-predictive	Greedy heuristic	Simulation with real cloud workload traces	Minimization of active PMs and active switch
Thermal aware workload consolidation in cloud data centers [34]	DVMC	Centralized	CPU	Threshold-free approach	RC	Non-predictive	Meta-heuristic	Simulation-based performance evaluation	Minimization of PM and network related energy

Optimizing virtual machine consolidation in virtualized data centers using resource sensitivity [25]	SVMC	Centralized	CPU, memory, and storage			Non-predictive	A mathematical model has been proposed	Simulation in Matlab	SLA violation aware
Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers [50]	DVMC	Centralized	CPU, memory, and network bandwidth	Static and adaptive threshold	The VM with highest resource demand	Predictive	Greedy heuristic	Simulation with real cloud workload traces	SLA violation aware
An efficient resource utilization technique for consolidation of virtual machines in cloud computing environments [40]	DVMC	Centralized	CPU	Adaptive threshold	RC, MMT, HPG, MC, MVM	Non-predictive	Greedy heuristic	Simulation in CloudSim	SLA violation aware
Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud data centers [51]	DVMC	Centralized	CPU, memory, and network bandwidth	Threshold-free approach	RC	Non-predictive	Both greedy heuristics and meta-heuristic algorithms	Simulation with real cloud workload traces	Minimization of active PMs while considering the cost of VM migrations
Cache contention aware virtual machine placement and migration in cloud data centers [26]	SVMC	Centralized	CPU			Predictive	Greedy heuristic	Simulation with real cloud workload traces	Cache contention aware
Improved virtual machine migration approaches in cloud environment [52]	DVMC	Centralized	CPU	Over-utilized and under-utilized PMs are selected as source PMs	MMT	Non-predictive	Greedy heuristics	Simulation in CloudSim with real cloud workload traces	SLA violation aware

(continued)

Table 6.1 (continued)

Research project name	VMC type	VMC decision process	Resource considered	Source PM selection strategy	VM selection criteria	Application of prediction technique	Destination PM selection strategy	Performance evaluation strategy	Objective
Self-adaptive resource management system in IaaS clouds [37]	DVMC	Centralized	CPU, memory	Adaptive threshold	MMT	Non-predictive	Greedy heuristics	Simulation with real cloud workload traces	SLA violation aware
Energy-aware consolidation in cloud data centers using utilization prediction model [46]	DVMC	Centralized	CPU, memory	Static threshold	MMT, VM with minimum resource demand	Predictive	Greedy heuristics	Simulation with real cloud workload traces	SLA violation aware
Robust server consolidation: coping with peak demand underestimation [53]	Both SVMC and DVMC	Centralized	CPU, memory	Threshold-free	All VMs from least power-efficient PMs	Predictive	Greedy heuristic	Simulation with real cloud workload traces	SLA violation aware
Achieving intelligent traffic-aware consolidation of virtual machines in a data center using learning automata [48]	DVMC	Centralized	Network bandwidth	Threshold-free	CVS	Non-predictive	Meta-heuristic	Simulation in CloudSim with real cloud workload traces	Minimize network traffic
Energy optimized VM placement in cloud environment [55]	SVMC	Centralized	CPU			Non-predictive	Greedy heuristics	Simulation in CloudSim	Restrict VM migration and minimize total number of PMs
GLAD: Distributed dynamic workload consolidation through gossip-based learning [32]	DVMC	Centralized	CPU	Threshold-free	VM with least migration cost	Predictive	One of the neighbor PMs is selected	Simulation in PeerSim [54] with real cloud workload traces	Restrict VM migration and minimize total number of PMs

A consolidation strategy supporting resources oversubscription in cloud computing [56]	SVMC	Centralized	CPU			Non-predictive	Greedy heuristics	Simulation in CloudSim	Restrict VM migration and minimize total number of PMs
Bayesian networks-based selection algorithm for virtual machine to be migrated [42]	DVMC	Centralized	CPU	Threshold-free	VM with least migration cost	Predictive	One of the neighbor PMs is selected	Simulation in PeerSim with real cloud workload traces	Restrict VM migration and minimize total number of PMs
Performance-aware server consolidation with adjustable interference levels [23]	DVMC	Centralized	CPU	Static threshold which denotes a level of VM interference level caused by VMC		Non-predictive	Greedy heuristics	Simulation-based performance evaluation	Minimize active PMs while limiting VM interference
A gossip-based dynamic virtual machine consolidation strategy for large-scale cloud data centers [31]	DVMC	Distributed	CPU	Adaptive threshold		Non-predictive	Greedy heuristic	Simulation in PeerSim [54]	SLA violation aware
Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing [57]	DVMC	Centralized	CPU, storage	Static threshold	Over-loaded PMs: VM with highest resource demand. Under-loaded PMs: All VMs	Non-predictive	Meta-heuristic	Simulation in CloudSim	SLA violation aware
SOC: Satisfaction-oriented virtual machine consolidation in enterprise data centers [58]	SVMC	Centralized	All types of resources required by a VM			Non-predictive	The PM that best matches with IT manager's prefer-ences	Simulation-based performance evaluation	SLA violation aware

(continued)

Table 6.1 (continued)

Research project name	VMC type	VMC decision process	Resource considered	Source PM selection strategy	VM selection criteria	Application of prediction technique	Destination PM selection strategy	Performance evaluation strategy	Objective
Virtual machine placement optimizing to improve network performance in cloud data centers [59]	SVMC	Centralized	Network bandwidth			Non-predictive	Meta-heuristic	Simulation in C++	Minimize total network traffic and maximize link utilization
Virtual machine consolidated placement based on multi-objective biogeography-based optimization [60]	SVMC	Centralized	CPU and memory			Non-predictive	Meta-heuristic	Simulation in C++	Minimize active servers and maximize resource utilization
An energy-aware heuristic framework for virtual machine consolidation in cloud computing [61]	DVMC	Centralized	CPU	Static threshold	RC, MMT, MC, MVM	Non-predictive	Greedy heuristics	Simulation-based performance evaluation	Minimize active PMs and SLA violations while limiting VM migration
Dynamic virtual machine consolidation for energy-efficient cloud data centers [62]	DVMC	Centralized	CPU	Static threshold	VMs from under-utilized PMs	Predictive	Meta-heuristic	Simulation-based performance evaluation	Minimize active servers and maximize resource utilization
Correlation-based virtual machine migration in dynamic cloud environments [63]	DVMC	Centralized	CPU	Static threshold	MC	Non-predictive	Greedy heuristics	Simulation-based performance evaluation	SLA violation aware
VM consolidation approach based on heuristics, fuzzy logic, and migration control [64]	DVMC	Distributed	CPU	Adaptive threshold	Fuzzy VM selection	Non-predictive	Greedy heuristic	Simulation in PeerSim [54]	SLA violation aware
Server consolidation with minimal SLA violations [65]	DVMC	Centralized	CPU	Static threshold	MMT	Non-predictive	Heuristic	Simulation	SLA violation aware

efficiency is rather possible through load unbalancing or workload concentration while switching the idle nodes off, also known as VMC. Since then, researchers have adopted VMC as a strategy to reduce energy consumption in their research work and state-of-the-art VMC algorithms have been developed through following the directions of past researches.

To the best of our knowledge, critical review on VMC algorithms is not available in the existing literature. Hence, in this paper, we have critical reviewed multitude of VMC algorithms with varied viewpoints, as we have also highlighted different aspects of most contemporary VMC algorithms published in 2016 and onwards which are not analyzed in any currently published surveys. Furthermore, we have analyzed notable VMC algorithms published before 2016 which has provided prominent research directions. The summary of our findings is as follows:

- Both SVMC and DVMC algorithms are crucial for energy-efficiency of CDC. SVMC is the first step towards limiting the energy consumption, while DVMC further minimizes the energy consumption while increasing the resource utilization of CDC.
- Centralized DVMC algorithms have been found as more popular than distributed DVMC algorithms. For P2P networks, distributed DVMC algorithms are useful. Although, distributed DVMC algorithms are more robust and reliable in case of hardware failure; however, it poses more network overhead compare to centralized DVMC algorithms.
- In comparison with threshold-free approach, threshold-based approach is more time efficient. However, since threshold-based approach limits the search space by selecting PMs based on the threshold value, therefore stagnation in local minima or local maxima may arise.
- One drawback of VMC is that SLA violation may arise because of aggressive consolidation. Static threshold-based DVMC algorithms cannot control the SLA violation. In contrast, adaptive threshold-based DVMC algorithms limits SLA violation by prior migration VMs from potential overloaded PMs. However, in terms of minimization of energy consumption, adaptive threshold-based DVMC algorithm is less efficient and it causes more number of VM migrations.
- MMT is the most widely used VM selection strategy, as with the decrease of migration time, service disruption time decreases which certainly lowers the SLA violation.
- A wide range of prediction techniques have been proposed in the literature to estimate the future resource demand of VMs as well as future resource utilization of PMs, as predictive DVMC algorithms minimize SLA violation more than non-predictive DVMC algorithms. However, predictive DVMC algorithms exert the overhead of more VM migrations.
- Unlike meta-heuristic algorithms, greedy-based heuristic algorithms may become stagnant in local minima or local maxima, yet these heuristic algorithms provide acceptably sub-optimal solution in quick time. Among all the meta-heuristics, evolutionary algorithms have appeared to be most popular. Considering both heuristics and meta-heuristics, modified version of best fit decreasing is found as most prevalent destination PM selection algorithm.

From our extensive study, we have found the following potential research openings which are yet to be explored:

- Adaptive threshold-based DVMC algorithms and predictive DVMC algorithms minimize SLA violation by prior migration of VMs from those PMs which are identified as potentially overloaded PMs. However, SLA violation is intrinsic in VM migration, because of unavoidable temporary service halt at some point of migration of a VM. Moreover, VM migration increases network traffic and increases network energy consumption. Existing adaptive threshold-based DVMC algorithms and predictive DVMC algorithms do not consider the overhead of VM migration. To address this gap, incorporation of a balanced approach is highly essential which limits VM migration and yet can manage an acceptable SLA violation rate.
- The effectiveness of predictive DVMC algorithms hinges on the actual correlation between past and future resource demand and the quality of adjustment with a specific workload type. However, different VMs co-hosted in a single PM have varied behavioral pattern in terms of resource demand. Thus, no single prediction technique would fit for all PMs, whereas existing predictive DVMC algorithms apply a common prediction techniques for all PMs. Furthermore, there is always a possibility of inaccurate prediction because of potential mismatches between past and present. Hence, there exists a research gap which is yet to be addressed.
- Apart from MMT, rest of the VM selection algorithms only consider CPU demand of VMs and ignore the memory, network bandwidth, and disk I/O requirement of VMs. However, as argued by the authors of [44], selecting a VM only based on CPU will cause saturation in terms of CPU and can lead towards no further improvement in utilization while leaving other types of resource under-utilized. It is highly challenging to determine a single converging point representing the equivalent total resource demand of multitude of resource types, while different types of resources represent different dimensions.

References

1. Kleinrock L. A vision for the internet. ST J Res. 2005;2(1):4–5.
2. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst. 2009;25(6):599–616.
3. Kaplan JM, Forrest W, Kindler N. Revolutionizing data center energy efficiency. Technical report, McKinsey & Company; 2008.
4. Beloglazov A. Energy-efficient management of virtual machines in data centers for cloud computing [dissertation]. Melbourne, AU: The University of Melbourne; 2013.
5. Koomey J. Growth in data center electricity use 2005 to 2010. A report by Analytical Press, completed at the request of The New York Times. 2011;9.
6. Gartner Estimates I. Industry accounts for 2 percent of global CO_2 emissions. Press release; 2007.

7. Buyya R, Beloglazov A, Abawajy J. Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges; 2010. arXiv preprint arXiv:10060308.
8. Koomey JG. Estimating total power consumption by servers in the US and the world. Feb 2007.
9. Hopkin J. VMware ESX Server [Image on internet]. OStatic; © 2015. Available from: http://ostatic.com/vmware-esx-server/screenshot/1.
10. Mann ZÁ. Allocation of virtual machines in cloud data centers—a survey of problem models and optimization algorithms. ACM Comput Surv (CSUR). 2015;48(1):11.
11. Pietri I, Sakellariou R. Mapping virtual machines onto physical machines in cloud computing: a survey. ACM Comput Surv (CSUR). 2016;49(3):49.
12. Kaur S, Bawa S, editors. A review on energy aware VM placement and consolidation techniques. In: International conference on inventive computation technologies (ICICT). IEEE; 2016.
13. Madhan ES, Srinivasan S, editors. Energy aware data center using dynamic consolidation techniques: a survey. In: Proceedings of IEEE international conference on computer communication and systems ICCCS14. 20–21 Feb 2014.
14. Pires FL, Barán B, editors. A Virtual machine placement taxonomy. In: 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and grid computing. 4–7 May 2015.
15. Ranjana R, Raja J, editors. A survey on power aware virtual machine placement strategies in a cloud data center. In: 2013 international conference on green computing, communication and conservation of energy (ICGCE). IEEE; 2013.
16. Varasteh A, Goudarzi M. Server consolidation techniques in virtualized data centers: a survey. IEEE Syst J. 2015;11(2):772–83.
17. Ahmad RW, Gani A, Hamid SHA, Shiraz M, Yousafzai A, Xia F. A survey on virtual machine migration and server consolidation frameworks for cloud data centers. J Netw Comput Appl. 2015;52:11–25.
18. Choudhary A, Rana S, Matahai KJ. A critical analysis of energy efficient virtual machine placement techniques and its optimization in a cloud computing environment. Procedia Comput Sci. 2016;78:132–8.
19. Masdari M, Nabavi SS, Ahmadi V. An overview of virtual machine placement schemes in cloud computing. J Netw Comput Appl. 2016;66:106–27.
20. Usmani Z, Singh S. A survey of virtual machine placement techniques in a cloud data center. Procedia Comput Sci. 2016;78:491–8.
21. Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr Comput. 2012;24(13):1397–420.
22. Ferdaus MH, Murshed M. Energy-aware virtual machine consolidation in IaaS cloud computing. In: Cloud computing. Berlin: Springer; 2014. p. 179–208.
23. Jersak LC, Ferreto T. Performance-aware server consolidation with adjustable interference levels. In: Proceedings of the 31st annual ACM symposium on applied computing. Pisa, Italy. 2851625: ACM; 2016. p. 420–5.
24. Hwang I, Pedram M. Hierarchical, portfolio theory-based virtual machine consolidation in a compute cloud. IEEE Trans Serv Comput. 2016;PP(99):1.
25. Nasim R, Taheri J, Kassler AJ, editors. Optimizing virtual machine consolidation in virtualized datacenters using resource sensitivity. In: 2016 IEEE international conference on cloud computing technology and science (CloudCom). 12–15 Dec 2016.
26. Chen L, Shen H, Platt S, editors. Cache contention aware virtual machine placement and migration in cloud datacenters. In: 2016 IEEE 24th international conference on network protocols (ICNP). IEEE; 2016.
27. Ferdaus MH. Multi-objective virtual machine management in cloud data centers. Melbourne: Monash University; 2016.
28. Ahamed F, Shahrestani S, Javadi B, editors. Security aware and energy-efficient virtual machine consolidation in cloud computing systems. In: 2016 IEEE Trustcom/BigDataSE/ISPA. 23–26 Aug 2016.

29. Deng D, He K, Chen Y, editors. Dynamic virtual machine consolidation for improving energy efficiency in cloud data centers. In: 2016 4th international conference on cloud computing and intelligence systems (CCIS). 17–19 Aug 2016.
30. Fioccola GB, Donadio P, Canonico R, Ventre G, editors. Dynamic routing and virtual machine consolidation in green clouds. In: 2016 IEEE international conference on cloud computing technology and science (CloudCom). 12–15 Dec 2016.
31. Masoumzadeh SS, Hlavacs H. A gossip-based dynamic virtual machine consolidation strategy for large-scale cloud data centers. In: Proceedings of the third international workshop on adaptive resource management and scheduling for cloud computing, Chicago, IL, USA. 2962565: ACM; 2016. p. 28–34.
32. Khelghatdoust M, Gramoli V, Sun D, editors. GLAP: distributed dynamic workload consolidation through gossip-based learning. In: 2016 IEEE international conference on cluster computing (CLUSTER). IEEE; 2016.
33. Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener Comput Syst. 2012;28(5):755–68.
34. Marcel A, Cristian P, Eugen P, Claudia P, Cioara T, Anghel I, et al., editors. Thermal aware workload consolidation in cloud data centers. In: 2016 IEEE 12th international conference on intelligent computer communication and processing (ICCP). 8–10 Sept 2016.
35. Marotta A, Avallone S, editors. A Simulated annealing based approach for power efficient virtual machines consolidation. In: 2015 IEEE 8th international conference on cloud computing. June 27–July 2 2015.
36. Farahnakian F, Ashraf A, Pahikkala T, Liljeberg P, Plosila J, Porres I, et al. Using ant colony system to consolidate vms for green cloud computing. IEEE Trans Serv Comput. 2015;8(2):187–98.
37. Farahnakian F, Bahsoon R, Liljeberg P, Pahikkala T, editors. Self-adaptive resource management system in IaaS clouds. In: 2016 IEEE 9th international conference on cloud computing (CLOUD). June 27–July 2 2016.
38. Farahnakian F, Liljeberg P, Plosila J, editors. LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers. In: 2013 39th euromicro conference on software engineering and advanced applications. IEEE; 2013.
39. Pascual JA, Lorido-Botrán T, Miguel-Alonso J, Lozano JA. Towards a greener cloud infrastructure management using optimized placement policies. J Grid Comput. 2015;13(3):375–89.
40. Selim GEI, El-Rashidy MA, El-Fishawy NA, editors. An efficient resource utilization technique for consolidation of virtual machines in cloud computing environments. In: 2016 33rd national radio science conference (NRSC). 22–25 Feb 2016.
41. Abdi H. Multiple correlation coefficient. Richardson: The University of Texas at Dallas; 2007.
42. Yan C, Li Z, Yu X, Yu N, editors. Bayesian networks-based selection algorithm for virtual machine to be migrated. In: 2016 IEEE international conferences on big data and cloud computing (BDCloud), Social computing and networking (SocialCom), Sustainable computing and communications (SustainCom)(BDCloud-SocialCom-SustainCom). IEEE; 2016.
43. Tanenbaum AS. Distributed operating systems. New Delhi: Pearson Education India; 1995.
44. Ferdaus MH, Murshed M, Calheiros RN, Buyya R, editors. Virtual machine consolidation in cloud data centers using ACO metaheuristic. In: European conference on parallel processing. Springer; 2014.
45. Ma L, Liu H, Leung YW, Chu X, editors. Joint VM-switch consolidation for energy efficiency in data centers. In: 2016 IEEE global communications conference (GLOBECOM). 4–8 Dec 2016.
46. Varasteh A, Goudarzi M. Server consolidation techniques in virtualized data centers: a survey. IEEE Syst J. 2015;11(2):772–83.
47. Dabbagh M, Hamdaoui B, Guizani M, Rayes A. Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment. IEEE Netw. 2015;29(2):56–61.

48. Jobava A, Yazidi A, Oommen BJ, Begnum K, editors. Achieving intelligent traffic-aware consolidation of virtual machines in a data center using learning automata. In: 2016 8th IFIP international conference on new technologies, mobility and security (NTMS). IEEE; 2016.
49. Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw Pract Exp. 2011;41(1):23–50.
50. Nguyen TH, Francesco MD, Yla-Jaaski A. Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers. IEEE Trans Serv Comput. 2017;PP(99):1.
51. Wu Q, Ishikawa F, Zhu Q, Xia Y. Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters. IEEE Trans Serv Comput. 2016;PP(99):1.
52. Choudhary A, Govil MC, Singh G, Awasthi LK, Pilli ES, Kumar N, editors. Improved virtual machine migration approaches in cloud environment. In: 2016 IEEE international conference on cloud computing in emerging markets (CCEM). 19–21 Oct 2016.
53. Grimes D, Mehta D, O'Sullivan B, Birke R, Chen L, Scherer T, et al., editors. Robust server consolidation: coping with peak demand underestimation. In: 2016 IEEE 24th international Symposium on modeling, analysis and simulation of computer and telecommunication systems (MASCOTS). IEEE; 2016.
54. Montresor A, Jelasity M, editors. PeerSim: a scalable P2P simulator. In: IEEE ninth international conference on Peer-to-Peer computing, 2009 P2P'09. IEEE; 2009.
55. Kaur A, Kalra M, editors. Energy optimized VM placement in cloud environment. In: 2016 6th international conference cloud system and big data engineering (confluence). IEEE; 2016.
56. Liu Y, editor A consolidation strategy supporting resources oversubscription in cloud computing. In: 2016 IEEE 3rd international conference on cyber security and cloud computing (CSCloud). IEEE; 2016.
57. Li H, Zhu G, Cui C, Tang H, Dou Y, He C. Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing. Computing. 2016;98(3):303–17.
58. Li X, Ventresque A, Murphy J, Thorburn J. SOC: Satisfaction-Oriented Virtual Machine Consolidation in Enterprise Data Centers. Int J Parallel Program. 2016;44(1):130–50.
59. Dong J-k, Wang H-b, Li Y-Y, Cheng S-d. Virtual machine placement optimizing to improve network performance in cloud data centers. J China Univ Posts Telecommun. 2014;21(3): 62–70.
60. Zheng Q, Li R, Li X, Shah N, Zhang J, Tian F, et al. Virtual machine consolidated placement based on multi-objective biogeography-based optimization. Future Gener Comput Syst. 2016;54:95–122.
61. Cao Z, Dong S. An energy-aware heuristic framework for virtual machine consolidation in Cloud computing. J Supercomput. 2014;69(1):429–51.
62. Kang D-K, Alhazemi F, Kim S-H, Youn C-H. Dynamic virtual machine consolidation for energy efficient cloud data centers. In: Zhang Y, Peng L, Youn C-H, editors. Cloud computing: 6th international conference, CloudComp 2015, Daejeon, South Korea, October 28–29, 2015, Revised selected papers. Cham: Springer International Publishing; 2016. p. 70–80.
63. Liu L, Zheng S, Yu H, Anand V, Xu D. Correlation-based virtual machine migration in dynamic cloud environments. Photonic Netw Commun. 2016;31(2):206–16.
64. Monil MAH, Rahman RM. VM consolidation approach based on heuristics, fuzzy logic, and migration control. J Cloud Computing. 2016;5(1):8.
65. Patel CA, Shah JS. Server consolidation with minimal SLA violations. In: Behera HS, Mohapatra DP, editors. Computational intelligence in data mining—volume 2. Proceedings of the international conference on CIDM, 5–6 Dec 2015. New Delhi: Springer India; 2016. p. 455–62.
66. Shackleford D. Virtualization and cloud: Orchestration, automation and security gaps [video on the Internet]. 2DeCipher; 2014. [Available from: https://www.youtube.com/watch?v=mjOwQlr1LIk].
67. Pinheiro E, Bianchini R, Carrera EV, Heath T, editors. Load balancing and unbalancing for power and performance in cluster-based systems. In: Workshop on compilers and operating systems for low power. Barcelona, Spain; 2001.