STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **7**, June 2019, pp 468–485.
Published online in International Academic Press (www.IAPress.org)

Hypercube Based Genetic Algorithm for Efficient VM Migration for Energy Reduction in Cloud Computing

Navneet Singh [1,*], Vijay Dhir [2]

[1] *University Institute of Computer Applications, Sant Baba Bhag Singh University, Punjab, India*
[2] *Dean University Institute of Engineering and Technology, Sant Baba Bhag Singh University, Punjab, India*

Abstract If we choose to compare computing technology to coral reef then cloud computing technology is its very live and growing end. Its challenges are new and demand innovative measure to bring the size of its expending data centers under calipers and bridle its energy consumptions. Reduction in the consumption of energy is to be brought about without compromising quality-of-service and efficacy. For this, we purpose a Hypercube based Genetic Algorithm (HBGA) for efficient VM migration for energy reduction in cloud computing under QoS (Quality-of-service) constraint. The proposed HBGA technique can be implemented in two phases. First, in a data center the physical machines organize themselves in such a way as to acquire a highly scalable structure called Hypercube. The hypercube imperceptibly grates itself up or dips low in sympathy with VM instances as they mount up or get depleted. Secondly on the basis of this representation model of the compute nodes, and given the hypercube topology in which they are organized we propose three algorithms: (a) Hypercube based Node Selection Algorithm to minimize energy consumption (b) Hypercube based VM Selection Algorithm which minimizes the number of VM to be migrated. (c) To solve the problem of VM Placement we propose Hypercube based Genetic algorithm. Experimental results of comparisons between the proposed HBGA method viz-a-viz the existing solutions show a marked reduction in energy consumption of cloud computing environment.

Keywords Cloud Computing, Hypercube, VM Migration, Genetic algorithm, Energy Efficient.

DOI: 10.19139/soic.v7i2.541

1. Introduction

The concept of Cloud Computing was first propounded by John McCathy way back in 1961, wherein he envisaged the idea of computer time sharing technology. His ideas and concept grew with time to become a reality. As they say nothing is beyond human contrivance and utilitarian aspects always egg man to strive for excellence. Today computing power and even specific applications could be sold through the utility business model like water and electricity and made available on-demand in metered way [1]. The first decade of the century saw the computing industry and researchers wrestling with the problems of cloud computing in a concerned way. Cloud computing as we know is based on Virtualization technology i.e. it abstracts physical resources of the data center as virtual resources that can be isolated from each other. Here a single Physical machine (PM) can virtualizes multiple independent virtual machines (VMs) and provide different services. It is like working through ghost machines.

In clod computing it is possible to provide services to the customer in metered way i.e. pay only to the extent of service availed. Using virtualization technology, a service provider can build a flexible, transparent, resilient and scalable computing environment that meets the requirements of various applications, and increase resource

*Correspondence to: Navneet Singh (Email: navneetsainigch@gmail.com). University Institute of Computer Applications, Sant Baba Bhag Singh University, Punjab, India.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

utilization [2, 3]. With the development and popularity of social networks, e-commerce, streaming media, search engines and other technologies, the demand for computing resources is increasing and the scale of a data center is also gradually increasing [4, 5]. Like other public utility services wiz water supply, electricity, cooking gas and telephone etc. cloud computing is also considered a utility service which can be availed of on demand [1]. We expect all electronic gadgetry to be energy efficient to possibly achievable limits. So our data centers hosting cloud application must be cost effective and the same time should avoid undue burden of carbon footprint [1].

Studies have shown that data centers around the world consumed 201.8 TWh of electricity in 2010, enough to power 19 million average U.S. households [6]. This consumption accounted for 1.1 percent to 1.3 percent of the worldwide total and the rate was expected to increase to 8 percent by 2020 [7]. The breakdown of energy consumption in a data center on IT equipment type is as under: 65 percent energy is consumed by servers, 20 percent by storage and only 10 percent by networking equipment [8].The average data center consumes as much energy as 25,000 households reported by Kaplan et al [9]. With their enormous appetite for energy, todays data centers emit as much carbon dioxide as all of Argentina [9]. Data center emissions are expected to quadruple by 2020 [9]. Gartner Says Global IT Spending to reach 3.7 Trillion Dollar in 2018 [10]. Between 2000 and 2007, the total power consumption of data centers worldwide went from 70 billion to 330 billion kWh; its projected to grow to more than 1,000 billion kWh by 2020. According to the Refrigerating and Air Conditioning Engineers (ASHRAE) [11], the Infrastructure and Energy Cost (I&E) has increased by 75 percent of the cost in 2014 while IT costs are only 25 percent [12]. It is reported that the energy consumed in data centers is about 1.5 percent of the global electricity in 2010, and the percentage will be doubled by 2020 if the current trends continue [13].

	2017 Spending	2017 Growth (%)	2018 Spending	2018 Growth (%)	2019 Spending	2019 Growth (%)
Data Center Systems	178	4.4	179	0.6	179	-0.2
Enterprise Software	355	8.9	389	9.5	421	8.4
Devices	667	5.7	704	5.6	710	0.9
IT Services	933	4.3	985	5.5	1,030	4.6
Communications Services	1,393	1.3	1,427	2.4	1,443	1.1
Overall IT	**3,527**	**3.8**	**3,683**	**4.5**	**3,784**	**2.7**

Figure 1. Worldwide IT Spending Forecast (Billions of U.S. Dollars) (Source: Gartner (January 2018).

In the actual scenario, with an average Power Usage Efficiency (PUE) of 1.8, worldwide data center energy consumption will reach 507.9 TWh by 2020, explains Mattin Grao Txapartegy, Technology & Market Analyst at Yole [14].One of the ways to address the energy inefficiency is to leverage the capabilities of the virtualization technology [15]. An idle server can consume a lot of energy but storage and networking equipment while doing nothing consumes a very little energy. A sitting idle consumes 70 percent of energy & producing heat [8]. Cost of running a server in a public sector data center is 14000 euro in Europe it includes the energy, maintenance, licensing cost etc. Utilization of server in vast majority of data center ranges from 15 percent to 25 percent in which they do useful work rest is waste [8]. In majority of data center facilities 40 percent server are more than 5 years old, they provide only 7% of total compute capacity and consume 66% energy consumption of a data center. So there is need to install new sever kits. Network equipment, cooling equipment etc. are other areas where hike in energy consumption is to be brought low. Such other areas being beyond the scope of study of this work. We aim to concentrate energy saving in data center through efficient allocation of resources. Many of the researchers have tried to improve upon VM migration and consolidation in their effort to bring down energy consumption in data centers. But unfortunately migration technique has a flip side to it.

It certainly has some drawbacks: (1) The good effect of VM migration on the QoS of cloud application as by many researchers do not find favor with VM consolidation and migration algorithms. (2) As VM migration time impinges adversely on QoS of the cloud application we propose a Hypercube based VM Selection Algorithm based on minimum migration policy which is attractive in minimizing the number of VMs to be migrated. (3) VM placement still remains a problem which we proposed to tackle by Hypercube based Genetic Algorithm for VM placement. The suggested solution effectively solves the NP-hard problems of VM placements.

For allocating the virtual resources of a data center, this paper proposes a Hypercube based genetic algorithm for efficient VM migration for energy reduction in cloud computing under QoS (Quality-of-service) constraint.
The proposed HBGA technique can be implemented in two phases.

Phase I First in a data center the physical machines organize themselves in such a way as to acquire a highly scalable structure called hypercube. The hypercube imperceptibly grates itself up or dips low in sympathy with VM instances as they mount up or get depleted. Underutilized nodes attempt to shift their workload to their hypercube neighbors and switch off. On the other, overutilized nodes attempt to migrate a subset of their VM instances so as to reduce their power consumption and prevent degradation of their own resources, which in turn may lead to SLA violations.

Phase II Secondly on the basis of this representation model of the compute nodes, and given the hypercube topology in which they are organized, we present Hypercube based genetic algorithm for efficient VM migration for energy reduction in cloud computing under QoS (Quality-of-service) constraint.

The reminder of this paper is organized as follows: we provide a literature survey of the existing VM migration techniques for energy reduction in cloud computing in Section 2. In Section 3, we discuss Proposed Approach. In Section 4, we describe HBGA algorithm with example. In section 5, we describe the experimental design and present the experimental results. Finally, we present conclusions in Section 6.

2. Related Work

Data centers are known to consume lot of energy and are becoming an area of concern in computing technology where special efforts need to be made to reduce power consumption to possible achievable limits. This is what the concept of green computing is all about.

Liu et al. [16] suggested ant colony system (ACS) algorithm for VMP (Virtual machine placement. Combined with order exchange and migration (OEM) local search techniques, the resultant algorithm is called as OEMACS. The results of extensive simulation and experiments showed that the proposed OEMACS performed better as compared to conventional heuristic and existing evolutionary-based methods in terms energy saving and efficient utilization of resources. Shen et al. [17] proposed a VM allocation strategy known as CompVM. CompVM integrated complementary VMs with temporal awareness. Complementary VM are the VM has total resource request of each resource type dimension reached their node/hosts capacity during life period of VM. Based on this fact, a method is proposed that collects & analyses VMs resource utilization level, consolidates complementary VM on same computer node. This method increases resource utilization by minimizing the number of active computer nodes in cloud. It shutdown idle nodes. Simulation based on real-world traces showed that CompVM strategy should significantly reduces SLA violation, no. of VM migrations & also reduces no. of active nodes in cloud. Wang et al. [18] suggested DVMC (Dynamic virtual machine consolidation) framework that significantly reduces energy consumption while maintaining SLA agreement. Proposed VM policy is known as SABFD (Space aware best fit decreasing) & VM Selection policy called as High CPU utilization based migration VM selection (HS). Expensive simulation of DVMC along with SABFD & HS showed remarkable reduction in energy consumption of a data center. Liu et al. [19] suggested that VM consolidation & migration techniques helps to improve energy efficiency & resource utilization in cloud environment. While exercising this technique outmost care should be given to QoS of cloud applications. Proposed Energy efficient & QoS dynamic virtual machine consolidation known as EQVC framework. Extensive simulation using workload traces from real world traces showed that EQVC performs better as compared to existing techniques in respect to no of VM migration, SLA violation & energy consumption in cloud.

Rahman et al. [20] suggested now days most of application runs on cloud, more and more users are moving their application on cloud. So ICT consumes large portion of worldwide, most of this energy related to ICT is consumed by servers that runs in data center worldwide. Unit price of electricity varies from region to region & operator to operator. Proposed technique called dynamic workload aware algorithm that uses Spatio-temporal variations of electricity cost to minimize the energy cost of ICT. Spatio-temporal variations of electricity price can be exploited to reduce electricity cost i.e. workload can be shifted to regions with lower electricity cost. Proposed algorithm also taken into account electricity consumed by cooling, network equipments. Simulation results showed that algorithm significantly reduced operational cost of data center. Mishra et.al. [21] suggested efficient mapping of tasks to virtual machines & assignment of VM to computer node is a challenging task. Optimal mapping is required to ensure energy efficiency & QoS of cloud applications. In this paper authors introduced novel mapping algorithm for VM placement & VM assignment task. The proposed algorithm showed significant reduction in energy consumption by minimizing the no of active nodes while maintaining minimum makespan & task rejection rate. Cloudsim simulation tool is used for experimental setup showed effectiveness of proposed algorithm over existing technique in terms of energy efficiency.

Domanal et al. [22] proposed novel efficient & cost effective scheduling algorithm for a Bag of Tasks(BoT) on virtual machines. It used Artificial neural network to predict the future values of spot instances. The algorithm efficiently utilized cloud resources (VM instances, CPU & memory). Simulation setup used Spearmans Rho Test. Alguliyev et al. [23] suggested that while performing anomaly detection in big data selection of right tool is urgent task. Proposed algorithm for data clustering and outlier detection that considers compactness & separation of clusters is outlined. Central theme of proposed algorithm is to improve the detection of anomaly in big data. Extensive simulation based on numerical experiments on real data showed effectiveness of algorithm. Comparison was made using six data sets consisting of anomalous values. The quality of clustering result is determined using six evaluation metrics. Proposed approach enhances the accuracy of anomalous values detection using clustering. From UCI repository six data sets are used namely Diabetic, Magic 04, Banknote authentication, credit card clients, NSL-KDD-ALL, cover type & phishing are some data sets used in this simulation process. Diabetic dataset contains characteristic that are taken from messidor image set. Magic 04 data set was generated to simulate registration of high energy. Banknote authentication data set was derived from images that were taken from genuine and forged banknote like specimen. Credit card clients dataset contains information on default payments, demographic factors, credit data and bill statement of Taiwan from April 2005 to September 2005. Phishing dataset contains 11055 phishing websites. Performance of proposed algorithm is better as compared to K-mean algorithm. Algorithm works well on real data set of different sizes. This technique can be used in different research areas. The effectiveness of proposed algorithm is evaluated using MatLAab 2016a simulation toolkit using 64-bit window based Intel core (i7) 2.5 Ghz processor with 8GB of RAM. Patel et al. [24] suggested with growth of distributed computing the cloud is becoming more popular & computer system needs to be more energy efficient. Virtualization technology helps in efficient utilization of IT resources & thus reduces power consumption in cloud computing. Authors proposed novel machine learning techniques known as Deep learning. It is capable of accurately predicting the VM loads using the past workload traces that runs on the VM loads using past workload traces that runs on the VM. VM workload prediction helps cloud provider in capacity planning and apply suitable VM placement & migration techniques. Proposed algorithm is tested using real workload traces from PlanetLab. Results showed that algorithm helped in predicting VM workloads and thus improves performance of cloud computing. According to Torkestani et al. [25] management of resources in cloud computing is crucial task. It consists of allocation of resources like CPU power, storage & network bandwidth to cloud applications. Cloud provider should efficiently manage ICT resources in cloud computing while meeting the SLA constraints; otherwise cloud provider will not be able to provide services at competitive rates. Proposed algorithm used the concept of learning automata that improves resource utilization & reduces energy consumption. This framework prevents server overloads, improves server utilization, reduces number of VM migration and shutdown the idle servers to save on electricity in cloud. Simulation is performed using Cloudsim with traces from PlanetLab. Proposed algorithm performed better than existing techniques such as DVFS, NPA in terms of no of closing computer nodes and energy consumption.

Ye et al.[26] introduced an EEKnEA (energy-efficient knee point-driven evolutionary algorithm) for Energy-Efficient Many-Objective Virtual Machine Placement Optimization in a Cloud Computing Environment. The

EEKnEA algorithm performed better as compared to its counterparts in terms of energy saving, load balancing and robustness. Zhang et al. [27] proposed a heuristic task scheduling algorithm termed as Energy and Deadline Aware with Non-Migration Scheduling (EDA-NMS) algorithm, which exploits the looseness of task deadlines and tries to put off the execution of the tasks that have loose deadlines so as to avoid arousal new PMs. The results of extensive simulation and experiments showed that the proposed EDA-NMS algorithm performed better than other existing algorithms in terms of energy potency without introducing VM migration overhead and ensuring QoS(Quality-of-Service) of cloud applications.

Wen et al. [28] suggested an energy-efficient virtual resource dynamic integration (VRDI) method. This method used live migration technology of VM, that helped in reducing energy consumption of a data center by integrating the virtual resources. The proposed VRDI method was implemented in three parts: (1) the integration timing and the set of PMs that need to be integrated was decided on the basis of resource utilization and the thresholds of the PMs (2) Selected a minimal set of VMs which need to be migrated based on the load statics of the VM and the calculated Euclidean distance between the VM and a PM. (3) Finally a VM placement algorithm was proposed based on improved genetic algorithm, denoted IGAVP. Using the IGAVP, they discovered an effective VM placement solution to solve the bin-packing problem. The results of extensive simulation and experiments showed that the proposed VRDI method helped in reducing the energy consumption of data center and guaranteed the quality of service of the cloud applications developed on the VMs. The VRDI method saved about 45% of energy when the resource utilization of PM is less than 50%.

Wang et al. [29] introduced a new task model that illustrated the QoS requirements of tasks with the minimum frequency. Energy consumption ratio (ECR) was suggested to assess the efficiency of different frequencies under which to perform a take. ECR saved more than 15% energy as compared to FFD algorithm.

Laredo et al. [30] presented a self-organized criticality approach for dynamically load-balancing computational workloads. The model was based on Bak-Tang-Wiesenfelds and pile, it is a cellular automaton reaching critical states at the edge of chaos that are released in the form of avalanches. The proposed method reduced energy consumption and ensured QoS of tasks in cloud applications. Son et al. [31] presented dynamic overbooking techniques that assign host and network resources dynamically adapting based on the utilization. . The proposed method ensured reduced energy consumption and SLA violation as compared to baseline algorithms (No Over and ConnNone). Khosravi et al. [32] proposed a Dynamic VM Placement technique for reducing Energy and Carbon Cost in Geographically Distributed Cloud Data Centers. The results of extensive simulation and experiments showed that the proposed the approach which considers dynamic PUE, renewable energy sources, and changes in the total energy consumption performed than existing techniques while meeting service level agreements.

Zheng et al. [33] proposed a Non-dominated sorting genetic algorithm (NSGA-II) and a local selection strategy based on fuzzy. It was used to produce a Hybrid Energy-Aware Resource Allocation Approach in Cloud Manufacturing Environment. The proposed algorithm significantly reduced energy consumption as compared to base line algorithm. Wang et al.. [34] suggested a Multiagent (MA) based VM allocation approach for efficient allocation of VM resources to physical machines (PMs) in a data center. The proposed algorithm significantly reduced energy consumption & migration cost. Minarolli et al. [35] suggested a novel technique to make long-term predictions of resource demands of virtual machines for host overload detection. The proposed algorithm showed better performance and higher stability compared to existing techniques. Ashraf et al. [36] proposed a Multi-objective ant colony system [MOACS] algorithm for virtual machine consolidation in cloud data centers. The proposed algorithm significantly reduced energy consumption and maximized the number of released PMs as compared to two existing ant colony optimization based VM consolidation algorithms [Feller-ACO algorithm & single-objective, single-colony ACS VM consolidation algorithm]. Poola et al. [37] proposed two just-in-time adaptive workflow scheduling heuristics for clouds. These techniques used on-demand and spot instances to give fault-tolerant schedules whilst minimizing time and cost. Hieu et al.[38] proposed virtual machine consolidation algorithm with multiple usage prediction (VMCUP-M) to improve the energy efficiency of cloud data centers. VMCUP-M reduced energy consumption compared to the multiple resource black-box and graybox (BG) scheme. Pantazoglou et al. [39] suggested decentralized approach towards scalable and energy-efficient management of virtual machine (VM) instances that are provisioned within a large enterprise clouds. Wu et al. [40] proposed VM launching overhead reference model. The proposed model accurately predicted the VM launching overhead within

a mean square weighted deviation less than three from all four variables, i.e. VM CPU utilization, system I/O utilization, system CPU utilization, and VM launching time.

Xu et al. [41] proposed Brownout. It is based on the concept of brownout in electric grids. In brownout approach we perform the voltage shutdown to cope with emergency cases, in which light bulbs emit fewer lights and consume less power. Same approach can be used in cloud. Using brownout approach, optional components of cloud application can be disabled to save energy. The proposed algorithm saved 20 percent of energy. But there is always a trade-offs between energy saving and discount offered to users. Huang et al. [42] proposed a VM consolidation framework using a quasi M-convex optimization framework. The proposed framework attained a balance among multiple administrative objectives (e.g., power cost, network cost) during the VM consolidation process. Results of extensive simulations using real-world workload traces showed that the proposed framework is efficient, scalable and highly practical. Tao et al. [43] suggested a binary graph matching-based bucket-code learning algorithm (BGM-BLA) to solve problem of dynamic migration of VMs (DM-VM) in cloud. BGM-BLA algorithm performed better in terms of the Pareto sets obtained and computational time as compared to two optimization algorithms, i.e., Non-dominated Sorting Genetic Algorithm (NSGA-II) and binary graph matching-based common-coding algorithm. Dai et al. [44] proposed two greedy approximation algorithms namely minimum energy virtual machine (VM) scheduling algorithm (MinES) and minimum communication virtual machine scheduling algorithm (MinCS) to reduce the energy consumption while satisfying service level agreements of cloud users. The proposed algorithm demonstrated that MinES and MinCS gave scheduling that was within 4.3 to 6.1 percent energy consumption of the optimal solution while being computationally efficient. Vakilinia et al. [45] proposed a Platform for virtual machine (VM) placement/migration algorithm to reduce the total power consumption of cloud data centers. Results of extensive simulations using real-world workload traces showed that the algorithm explores the optimal solution with an optimality gap of at most 1% in 3 minutes Computation time.

Wu et al. [46] proposed a genetic algorithm for a new virtual machine placement problem that take into account the energy consumption in both the servers and the communication network in the data center. Experimental results showed that the proposed algorithm performed well when tackling test problems of different kinds, and scaled up and down when the problem size increases/decreases.

3. Proposed Approach

The proposed HBGA technique can be implemented in two phases:

Phase I First, in a data center the computer nodes organize themselves in such a way as to acquire a highly scalable structure called hypercube. The hypercube imperceptibly grates itself up or dips low in sympathy with VM instances as they mount up or get depleted. Underutilized nodes attempt to let their workload flow to their hypercube neighbors and themselves switch off. Alternatively, overloaded nodes attempt to shift a subset of their VM instances so as to reduce their own power consumption and degradation of their resources. Both ways help us to avoid SLA violations.

3.1. Formation of Data center

Hypercube particularly possess a series of attributes, which are also essential to our approach: (a) Network Uniformity: All nodes in a hypercube topology are at par i.e. on node takes precedence over the other nodes in any manner. (b) Economic Viability: The hypercube topology exhibits an $O(log_2 N)$ complexity. (c) Degree of resilience: It is always possible for the hypercube topology to exhibit high degree of resilience i.e. to cope up with sudden node losses of any magnitude.

In the case of hypercube structure compute nodes get themselves arranged in such a formation that each one of them is directly connected to x number of neighbors (at the most). Which can be represented as $X = 2^x$ on as such. Major Hypervisors technologies like VMware [47, 48] support it. This facilitates the migration of VM instances from one computer node to another within the data center in minimum space of time.

3.2. *Computer Nodes*

Compared to the other system resources of a compute node such as network resources, the CPU & memory consumes the main part of its power, and its utilization is typically proportional to the overall system load. Based on this fact, we focus on managing the CPU & memory power consumption of the compute nodes in the data center.

Each node in a data center is represented by a Set defined as: $W = /X_i, N_h/$, where X_i is the unique ID of a computer node in a data center. N_h represents computer nodes cache that maintains the position of a node in a hypercube.

3.3. *Hypercube Topology Construction and Maintenance Algorithm in Cloud Computing*

The construction and maintenance of hypercube of type P2P can be explained as under. Hypercube are also known as Cayley graphs[49] or start graphs.This can be better exemplified by having nine peers in a network with one peer deserting the network in the process. *Slightly shaded nodes represents temporary nodes.*
Start. To start with only peer P0 is active.
Step a. Peer P_1 wants to join the P2P network for which it contacts node. As Peer P0 has no existing neighbor it tends to integrate peer P_1 as its 0-neighbor. Normally a peer accommodates incoming peer in its first vacant dimension, the dimensions are so organized that lower dimensions always come first. As shown in Figure 2a

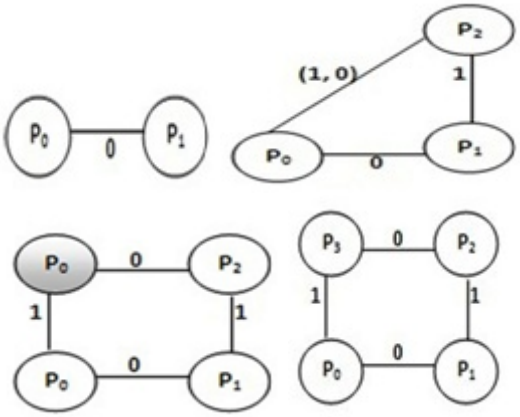

Figure 2. Hypercube Topology in data center. From upper-left to lower-right: (a), (b), (c) and (d).

Step b. Peer P_2 contacts one of the two peers (here, we assume that it contacts peer P_1) to join the network. The first vacant dimension of peer P_1 is 1 as it already has a 0-neighbor i.e. peer P0 . So peer 1 creates a new dimension for the hypercube, as shown in Figure 2b. Peer P_0 now occupies a vacant position in the hypercube, it acts as if it occupies two positions in the hypercube, as shown by the thin copy of peer P_0 (as slightly shaded node) in Figure 2c.

Step c. Peer P_3 wants to enter into the network. Here we have three cases, viz. peer P3 contacting peer P0, P1 or P2 to enter into the network. In case peer P3 communicates with peer P0 to enter into the network, peer P0 follows the general rule i.e. new node always occupies in its first vacant dimension of node which is 1 because P0 already has zero neighbor but no one-neighbor as shown in Figure2d.

Step d. Peer P4 arrives and communicates peer P0. At this stage Hypercube with two dimension accept and accommodate five peers, so a third dimension is opened. As shown in Figure 3a.

Step e. New Peer P5 communicates with Peer P1 to join the network. Peer P1 is still require a Two-neighbor, thus peer 5 will be integrated on this position as shown in Figure 3b.

Step f. Now peer P0 suddenly deserts the network.Now the question is which node takes position of P0 peer. Here in our example peer P4 comes into position P0.As shown in Figure 3c.

Step g. Peer P6 wants to join the network. It takes place of temporary node P5 in the hypercube as shown in Figure

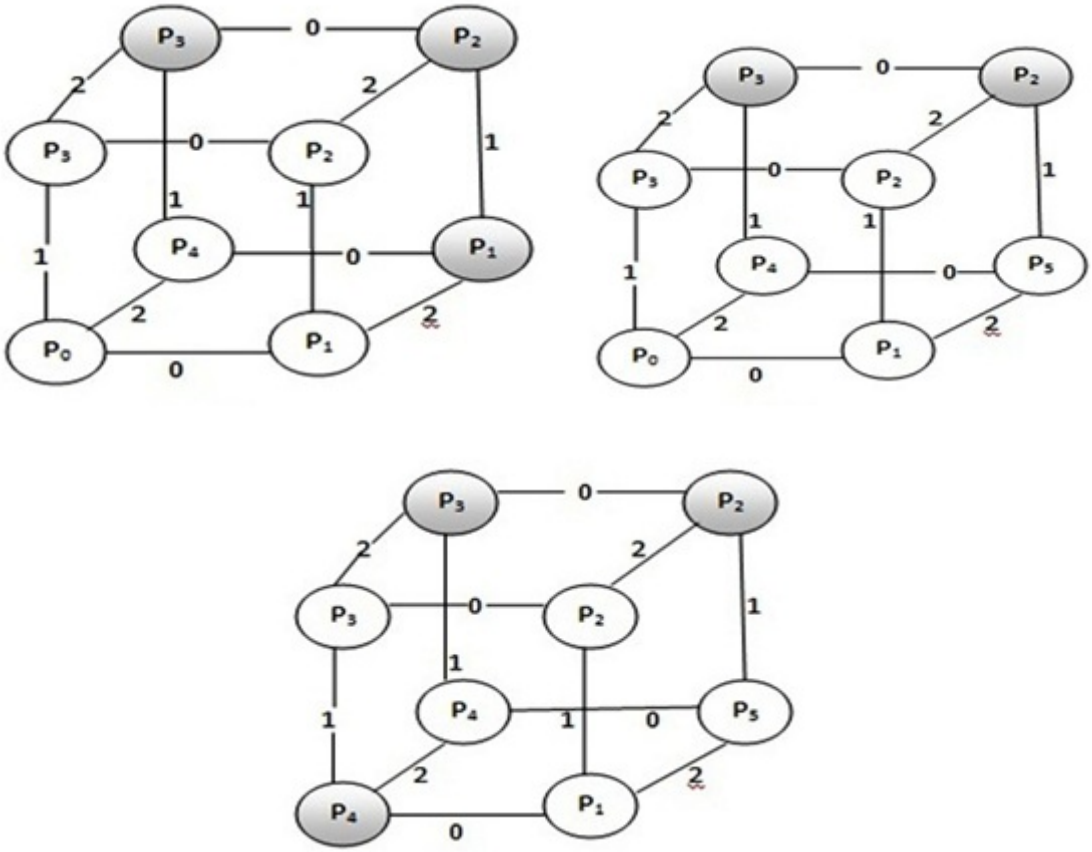

Figure 3. Hypercube Topology construction continued. From upper-left to lower-center: (a), (b) and (c)

4a.

Step h. Peer P6 is contacted by peer P7, who wants to join the network. P7 node takes place of temporary node P2 as shown in Figure 4b.

Step I. Now P8 arrives and wants to join the network. It comes into place of P4 nodes temporary i.e become first neighbor of P8 node as shown in Figure 4c.

The complexity of algorithm is $O(log_2 N)$.

Phase II On the basis of this representation model of the compute nodes, and given the hypercube topology in which they are organized, we present Hypercube based genetic algorithm for efficient VM migration for energy reduction in cloud computing under QoS (Quality-of-service) constraint. We propose a Hypercube based Node Selection Algorithm to minimize energy consumption. It is achieved by specifically defining the thresholds of resource utilization based on the Type of load to the PMs.As the VM migration time is linked with QoS of the cloud application for this we suggest a Hypercube based VM Selection Algorithm based on time space viz. minimum migration time policy which minimizes the number of VM to be migrated.To solve the problem of VM placement we propose Hypercube based Genetic algorithm.

Set of definitions used in this paper:

Definition 1: Each node in a data center is represented by a Set defined as: W = (X_i, N_h), where X_i is the unique ID of a computer node in a data center. N_h represents computer nodes cache that maintains the position of a node in a hypercube.

Definition 2: Type of load: It donates resource usage of a computer node or virtual machine. It denoted as X_i =

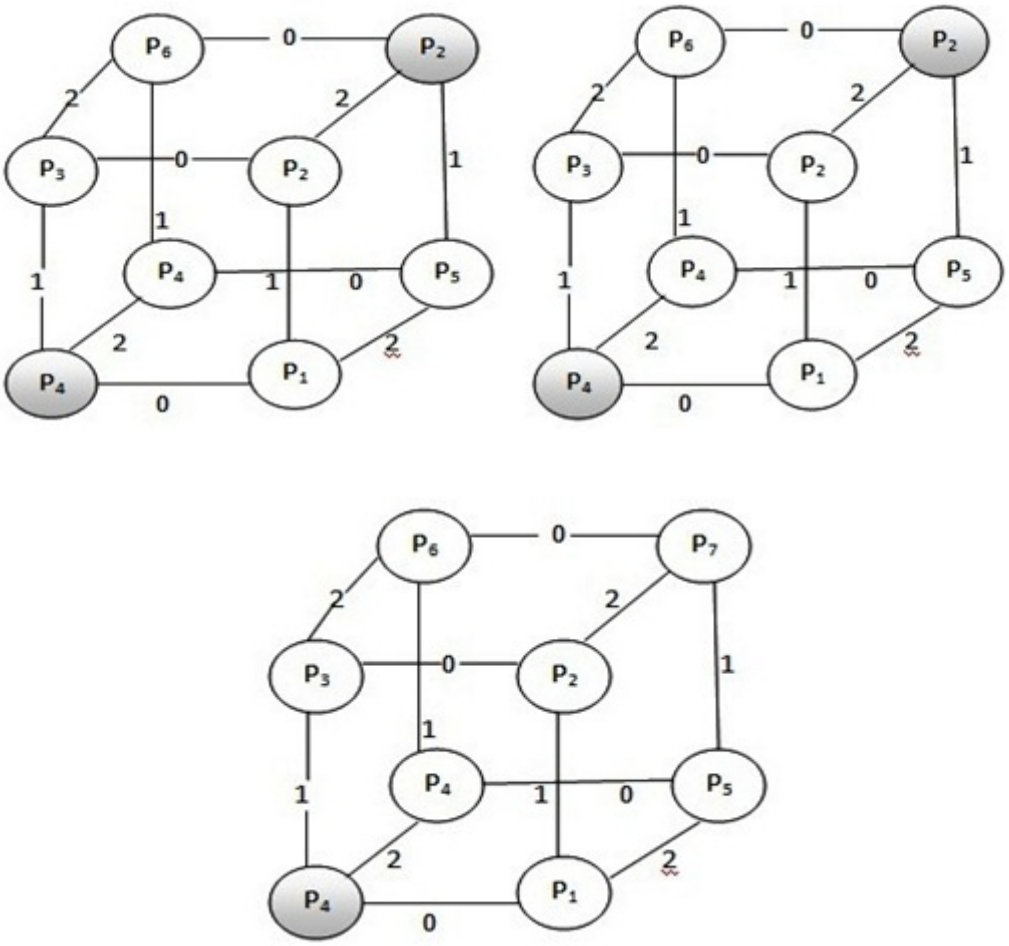

Figure 4. Hypercube Topology construction continued. From upper-left to lower-right: (a), (b) and (c)

$(X_i^{cpu+mem})^t$, where $(X_i^{cpu+mem})$ represents the cpu and memory utilization respectively of ith computer node at time t.
Definition 3: R represents resource request of a virtual machine. It is defined as $R_i = (R_i^{cpu+mem})^t$.
Definition 4: P donates the maximum resource that a node can provide. It is defined as $P_i = (P_i^{cpu+mem})^t$.
Definition 5: Lower threshold of a computer node is defined as $Mini_i$=$(Mini^{cpu+mem})^t$.
Definition 6: Upper threshold of a computer is defined as $Maxi_i$=$(Max_i^{cpu+mem})^t$.
Definition 7(a) Case 1: Full Migration (computer node is underloaded)
$X_i = (X_i^{cpu+mem})^t < (Mini_i^{cpu+mem})^t$.— (1)
Definition 7(b) Case 2: Partial Migration (computer node is overloaded)
$X_i = (X_i^{cpu+mem})^t \geq (Max_i^{cpu+mem})^t$. — (2)

A. The Hypercube Based Node Selection Algorithm

If the association between resource utilization and upper threshold of computer node satisfies as in equation (2) we transfer some of virtual machines to other computer nodes in a data center to maintain resource utilization of node below the upper threshold. We describe this process as PARTIAL.If the association between resource utilization and upper threshold of computer node satisfies as in equation (1) we transfer all of virtual machines deployed on

computer node to other computer nodes and shut down the node itself. We describe this process as FULL. For the two scenarios described above, the Hypercube Based Node Selection Algorithm is presented as Algorithm 1.If the number of computer nodes is x and the number of virtual machines is y, complexity of Algorithm 1 is O(x*y).Detail of algorithm is give below.

Algorithm 1 The Hypercube Based Node Selection Algorithm

Input: W= $\{X_i, N_h\}$, where X_i is the unique ID of a computer node in a data center. N_h represents computer nodes cache that maintains the position of a node in a hypercube.
Output: FULL, PARTIAL

1. FULL = NULL, PARTIAL = NULL
2. For each Computer Node in Hypercube do
3. Calculate Type of load Xi of node
4. If Xi satisfies equation (1) then
5. FULL = FULL Union Node // Transfer all virtual machine on Computer Node
6. ELSE if Xi satisfies equation (2) then
7. PARTIAL = PARTIAL Union Node // Transfer some of virtual machine
8. On Computer Node in Hypercube
9. Else
10. Continue;
11. End if
12. End for
13. Return FULL,PARTIAL;

B. The Hypercube Based Virtual Machine Selection Algorithm

In Hypercube based Node Selection Algorithm, we have to treat FULL and PARTIAL case separately.In FULL case, all of the virtual machines should be transferred out of computer node and node itself needs to be put in power-saver mode. In PARTIAL case we have to select set of virtual machines that needs to be transferred. While selecting the set of virtual machines to that needs to be transferred outmost care should be given. Because virtual machine migration may affect the performance of cloud applications. So we used minimum migration policy to minimize the impact of migration process in cloud computing. In minimum migration policy we migrate the virtual machine that gets transferred in minimum frame of time. We sort the all the virtual machine on overloaded host according to their resource usage. According to the above description, Hypercube based Virtual Machine Selection Algorithm is presented as Algorithm 2. If number of nodes is x and number of virtual machines is y, then complexity of Hypercube Based Virtual Machine Selection Algorithm is O(x ? y).

Algorithm 2 The Hypercube Based Virtual Machine Selection Algorithm

Input: FULL, PARTIAL
Output: VM_Mig_List // VM migration list

1. VM_Mig_List = NULL
2. For each compute_ node in FULL inside HyperCube do
3. Get VM_List from computer node
4. Add VM_List to VM_Mig_List
5. end for
6. For each computer_node in PARTIAL inside Hypercube do
7. Get VM_List from computer node

8. Sort VM_List in terms of resource usage in ascending order (CPU+Memory)
9. Select VM from VM_List
10. Add VM to VM_Mig_List
11. Calculate the Type of load Xi of computer node
12. If Xi satisfies equation (2) then
13. Select next VM from VM_List
14. Add VM to VM_Mig_List
15. Else
16. Break;
17. End if
18. End for
19. Return VM_Mig_List;

C. Hypercube based Genetic Algorithm for Virtual Machine Placement

The VM placement is a key problem of the VM migration Algorithm. The VM placement is generally described as a bin packing problem. The bin packing problem is an NP-hard problem, and most of researchers use the global optimization tools to find a solution [50]. As a classical algorithm for solving optimization problems, genetic algorithm has been researched extensively [51, 52]. John H. Holland, father of genetic algorithms and pioneer in complex systems. In genetic algorithm as the name implies we try to follow the biological evolution process of selecting the best by strictly implementing Selecting, Encoding , Crossing and mutating to reach the best individual. The fitness of each individual is accessed iteratively which aims at eliminating individuals marked with minimum fitness. The end result depends upon the number frequency of iterations. The empirical approach as far employed to fix the number of iterations to be run has its own disadvantages. If the numbers is kept low chances are we may fail to reach the optimal solution on the other hand if the number of iterations is kept high it may result in lowering the efficiency of algorithm. In this paper we suggest a VM placement algorithm based on hypercube genetic algorithm. The HBGA approach necessities immediate termination of the algorithm once the optimal fitness of the offspring is reached.

(1) Encoding: VM Migration list denoted as VM_Mig_List = (VM1, VM2, VMn), encoding of a chromosome is denoted using as array A. The elements of A represents a mapping of a VM in the set VM_Mig_List to PM from set S. The size of the array A is m. A chromosome in HBGA consists of $\|V\|$ genes.An example VM placement and its corresponding chromosome is shown in Figure 5.

(2) The Fitness Function: Qualitative assessment of chromosomes is of great significance. To select the best

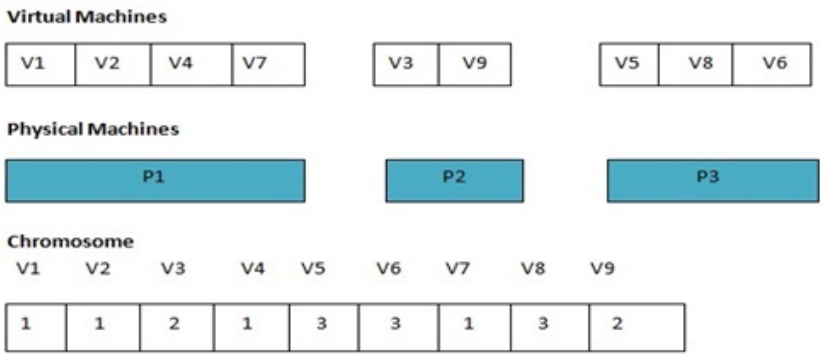

Figure 5. VM placement and its corresponding chromosome

chromosome to act as a parental collection to cross. In this paper we aim at constructing the fitness function based on resource utilization of PMs. For each chromosome G its fitness function F(G) is given as:
F(G) = 1/n$\sum_{i=1}^{n}(X_i^{cpu+mem})$
(3) Selection: To start with we select two chromosomes to generate the offspring. The fitness of the offspring is there for in direct proportion to the chromosomes selected. Going by the principle of survival of fittest we use roulette wheel to select individuals. The probability of selection of an individual is proportional to its fitness function.
(4) Crossover: In crossover set of genes from different parents combine to generate the offspring. The quality of offspring depends upon the genes inherited from parents in the matter of genes selection the crossover point is selected randomly. Using FFA, one offspring is generated by inheriting one parent from the beginning to the crossover point, and the remaining part from the crossover point to the end is inherited from the other parent. For example, if the parents are a1 = 123241 and a2 = 123341, when the crossover point is 3, based on FFA, we can get the offspring b1=123341 & b2=123241. For b1 and b2, we calculate their fitness separately, and select the offspring which has the higher fitness.
(5) Mutation: Mutation operation comes after crossover. Here two randomly selected points within the chromosome are swapped. A For example, by swapping the third and fourth positions of b1 (13213), we can get b3 = 12132. Using the described procedure, by replacing the original population of the lowest fitness with b3, we get a new population. Next, we continue the iterative evolution process. When the fitness of the optimal chromosome is no longer increasing after the generation C, the process is stopped, and the optimal chromosome corresponds to the solution of the best VM placement. Based on the described ideas, we summarize the specific process of the Hypercube based Genetic Algorithm for VM placement as follows:
Based on set of ideas described in previous section. The Hypercube based Genetic Algorithm for Virtual Machine placement is explained below:

Algorithm 3. The Hypercube based Genetic Algorithm for Virtual Machine placement

Input: VM_Mig_List, Pop_size
Output: Z //The VM placement solution

1. Generate a population of x individuals from Pop_size
2. Generate best individual from Pop_size
3. While the termination condition is not true do
4. For each individual x in Pop_size
5. Compute F (G) of each individual x in Pop_size
6. Select the parents chromosome a1 and a2 based on Roulette wheel;
7. Generate b1 and b2 by crossing a1 and a2;
8. Find the higher fitness individual b3 from b1 and b2;
9. Get the new individual Z by mutating b3;
10. Obtain the new population by replacing the lowest fitness individual of Pop_size with Z;
11. End;
12. Return Z

4. Illustration Of HBGA Algorithm With Example

Lets now demonstrate how our proposed HBGA works: consider a data center that consists of eight homogeneous compute nodes, all organized into a three dimensional binary hypercube. The compute nodes have a common power profile:

p ($Mini^{cpu+mem}$) = 210w p($Max^{cpu+mem}$) = 300w p($idle_i$) = 180w

We assume that the data center has just started its operation, and thus each PM is only aware of its immediate neighbors within the hypercube. As for the sake of simplicity, each VM consume 10 W each. The example illustrates how the VM migration scheme is applied to reduce power consumption in a data center. Yellow nodes are switched off nodes. In figure 6a only five nodes(P6, P7, P3, P5, P8) are active. Node P8 is overloaded. Nodes P3, P5, P6, P7 are underutilized. we move one VM form P8 to P3 so that status of P8 becomes ok. move two VM from P6 to P3 and switch off the P6 node. We move two VM from P7 to P5 and switch off P7 node. As shown in figure 6c only three nodes (P3, P5, P8) are in active mode. Figure 6 illustrates the concept. Figure 7 Shows summary of the data centers status before and after VM migration using HBGA approach

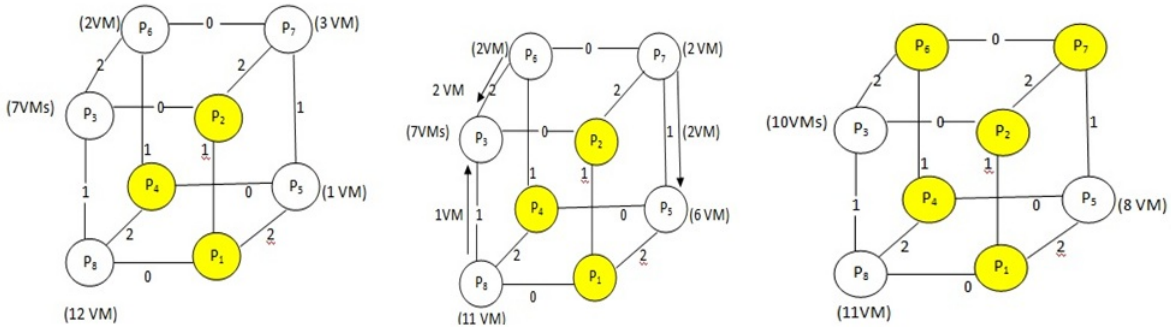

Figure 6. VM migration using HBGA in Data center. From left to right: (a) Before Migration, (b) During Migration and (c) After Migration

Physical Machine	Before VM Migration			After VM Migration		
	VM instances	Power consumption	State	VM instances	Power consumption	State
P1	0	0w	Switch off	0	0w	Switch off
P2	0	0w	Switch off	0	0w	Switch off
P3	7	250w	ok	10	280w	Ok
P4	0	0w	Switch off		0w	Switch off
P5	6	240w	Ok	8	260w	Ok
P6	2	200w	Under-loaded	0	0w	Switch off
P7	2	200w	Under-loaded	0	0w	Switch off
P8	12	310w	Overloaded	11	290w	Ok
	29	**1200W**		**29**	**830W**	

Figure 7. Summary of the data centers status before and after VM migration

5. Experiment and Evaluation

In the following, we describe the experimental setup followed by the results of the experiments.

5.1. Experimental Setup

To verify the effectiveness of the proposed HBGA, we perform a large number of repeated experiments using CloudSim [53], a cloud computing simulation toolkit, to simulate the experiment. The CloudSim is the most prominent cloud environment simulation framework, and its core module allows its users to monitor and manage the virtual resources and contrive the virtual resource allocation strategy. The extended components can produce energy consumption and statistics of the simulation. Moreover, the toolkit can also simulate the dynamic load of cloud applications. We created a data center consisting of 100 PMs and 2000 VMs using the CloudSim toolkit. It used two types of PMs models, namely, the HP ProLiant ML110 G5 and the IBM X3550. In order to ensure the accuracy of the results; our experiments were carried out for about ten months.

We adopt the realistic workload trace from more than 1,000 PlanetLab [54] VMs to create an overloaded environment. Because of the dynamic and unpredictable nature of workloads of cloud applications, the thresholds of PMs are not constant. Therefore, we used the Local Regression (LR) method in [55] to set the utilization thresholds; it is based on the historical data of the Type of loads collected by the VMM (Virtual Machine Monitor). In this work, we performed a series of experiments to estimate the threshold values. We set the lower and upper thresholds as (0.26, 0.26) and (0.84, 0.84) using extensive analysis.

We define an SLA violation occurs when VM cannot get the requested CPU utilization. For each SLA violation, the cloud providers have to pay a penalty to the users. First, we compare the performance of the three methods (HBGA, GA [46] and OEMACS [16]) in terms of the total energy consumption of the data center. From Figure 8, it is clear that HBGA method can save about 45 percent of energy as compared to GA and OEMACS. The performance of the HBGA is better than GA and OEMACS.Parameter Population size (Pop_size) is set to 150 and Iteration counter 'C' is set to 25.

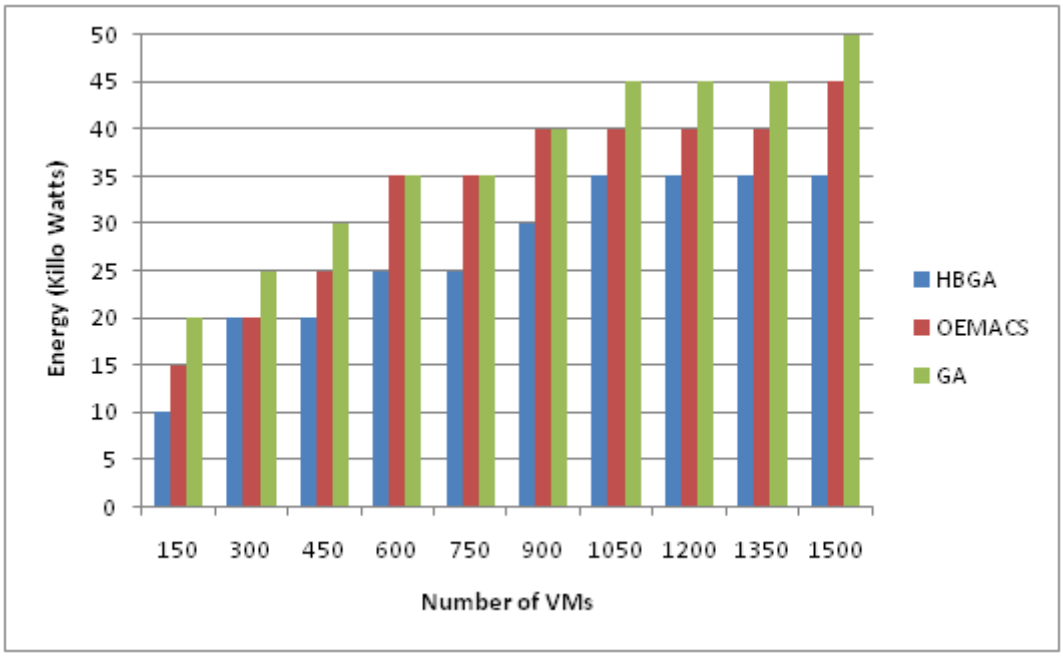

Figure 8. Comparative study of different algorithm in respect of energy consumption at data center

Secondly, we compare the three methods in term of the number of migrated VMs. In order to gain saving in energy consumption, we used VM migration technique. But the migration of VM may affect the QoS of the deployed cloud applications. So VM migration should be done only when it is needed. Figure 9 shows the results of experiment for the three methods.

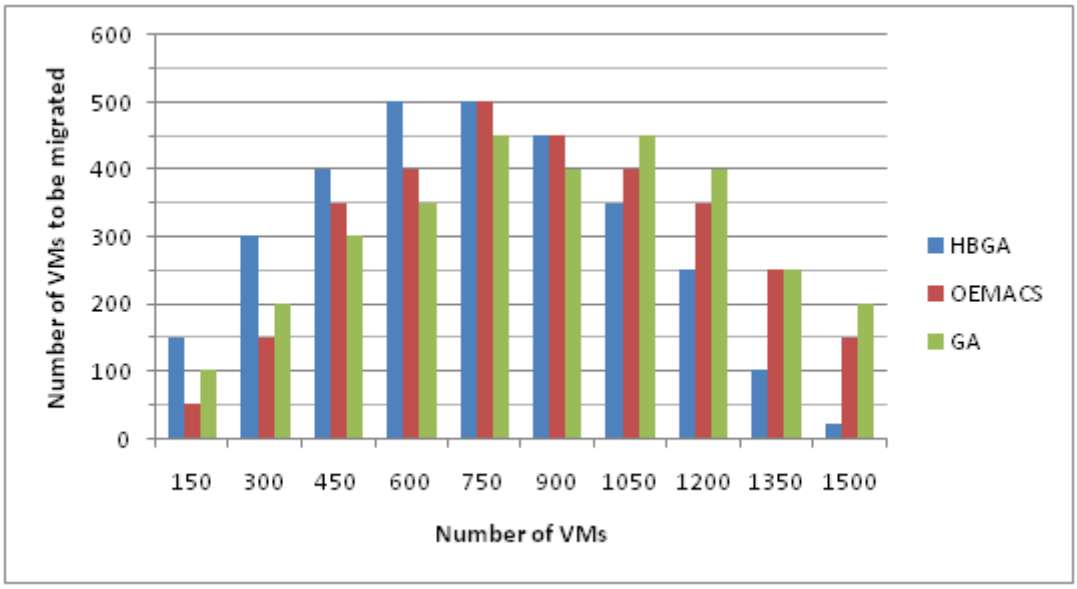

Figure 9. Comparative study of different algorithm in respect of VMs to be migrated

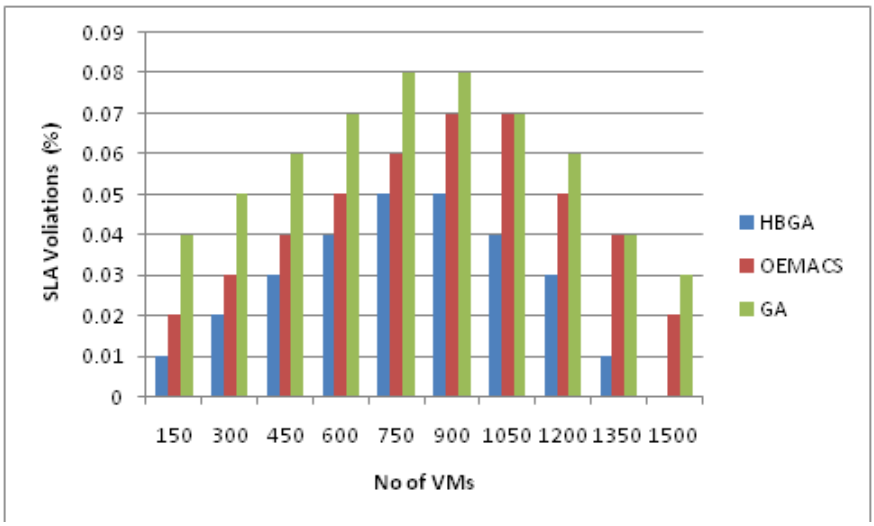

Figure 10. Comparative study of different algorithm in respect of Percentage of SLA violation of VMs.

From figure 9, it is clear that the number of VMs to be migrated in the HGBA method is larger than the GA and OEMACS methods when the number of VMs in data center is less. It makes sense because when the number of VMs is less, the HBGA approach will migrate all of the VMs developed on the underloaded PM to some other PM and shutdown the PM to save energy. Hence, in the HBGA algorithm, during the VM selection, the FULL case is active. When the number of VMs in data center is high, the PM's resource utilization is higher, and during the Hypercube based VM SELECTION ALGORITHM, the PARTIAL case becomes active. Hence, in this case, the HBGA migrates less number of VMs as compared to GA and OEMACS methods as illustrated in Figure 9. For decision on which set of VMs to migrate, the HBGA uses the minimum migration principle to select a VM whose Type of load is identical with that of the PM. Hence, the HBGA approach selects less number of VMs for migration.

Thirdly, a comparative study of the three methods in terms of the percentage of SLA violations is a must. VM migrations no doubt enhance the utilization of resources but certainly not without making a tangible dent on SLA requirements of the cloud computing applications. This negative impact of the VM migration can be countered by minimizing the required to perform VM migration process. Figure 10. shows the percentage of SLA violation of the HBGA, GA and OEMACS methods.

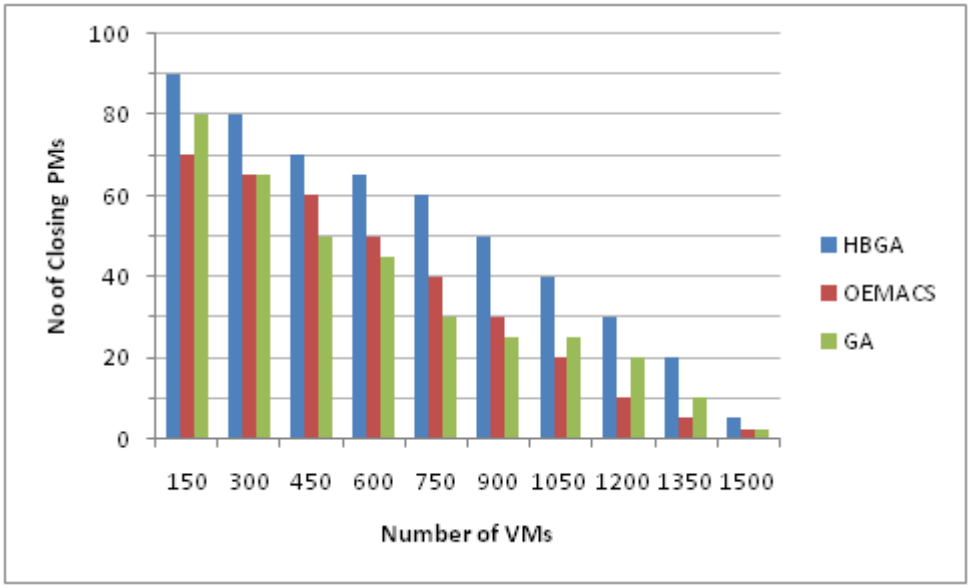

Figure 11. Comparative study of different algorithm in respect of the number of closed PMs.

HBGA can find an optimal scheme to place target VMs efficiently, hence, it can minimize the time required for VMs migration. The results from Figure 10 show that the SLA violation of the HBGA method is lower than that of the GA and OEMACS methods. Finally, in Figure 11, we show the number of PMs that were closed when using the HBGA, GA and OEMACS methods. As mentioned above, the purpose of VM migration of the data center is to transfer the VMs to close some of the PMs which have lower utilization to improve the energy efficiency of the data center. Therefore, the more the number of closed PMs, the more is the effectiveness of an algorithm. It is seen that when the resource utilization dips below 50 percent , the HBGA method tends to close about 38 percent of the PMs. In a similar situation the GA and OEMACS close only about 20 percent of the PMs. Therefore in performing the VM migration the HBGA has edge over others. Hence it makes a considerable contribution to conservation of energy in data center.

6. Conclusion

The HBGA proposed in this paper envisages live migration technology of virtual machine which helps in energy reduction of data center by VM migration. The proposed HBGA technique can be implemented in two phases. First, in a data center the computer nodes organize themselves in such a way as to acquire a highly scalable structure called hypercube. The hypercube imperceptibly grates itself up or dips low in sympathy with VM instances as they mount up or get depleted. Secondly we propose three algorithms: 1) We propose a Hypercube based Node Selection Algorithm to minimize energy consumption. It is achieved by specifically defining the thresholds of resource utilization based on the Type of load to the computer nodes. 2) As the virtual machine migration time is linked with QoS of the cloud application for this we suggest a Hypercube based Virtual Machine Selection Algorithm based on time space viz minimum migration time policy which minimizes the number of virtual machine to be transferred. 3) To solve the problem of virtual machine placement we propose Hypercube based Genetic Algorithm.

Experimental results of comparisons between the proposed HBGA method viz-a-viz the existing solutions show a marked reduction in energy consumption of cloud computing. So HBGA if implemented in the right spirit of phased structure can go a long way in realizing the concept of greener data center.

Acknowledgement

I would like to show my gratitude to Dr Jatinder Singh Bal, Vice Chancellor at Sant Baba Bhag Singh University, Punjab, India and Dr. S. M. Bhatt, Sant Baba Bhag Singh University, Punjab, India and my Mama ji Mr. Charanjit Singh for providing me help and support.

REFERENCES

1. N. Singh, and J.S. Bal, *Energy Efficient Data Center: A systematic literature review*, Advances in Computational Science and Technology, vol. 10, no. 1, pp. 121–128, 2017.
2. M. Chen et al., *Effective VM sizing in virtualized data center*, IEEE Int. Symp. Integr. Netw. Manage., pp.594–601, 2011.
3. Y. Bi, H. Zhou, W. Xu, X. Shen, and H. Zhao, *An Efficient PMIPv6-Based Handoff Scheme for Urban Vehicular Networks*, IEEE Trans. Intell. Transp. Syst., vol. 17, no. 12, pp. 3613–3628, 2016.
4. M. A. Vouk, *Cloud Computing C Issues, Research and Implementations*, Int. J. Adv. Res. Comput. Sci. Manage, vol. 16, no. 4, pp. 235–246, 2008.
5. M. N. O. Sadiku, Sarhan M. Musa, and O. D. Momoh, *Cloud computing: Opportunities and challenges*, IEEE Potentials, vol. 33, no. 1, pp. 34–36, 2014.
6. P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, *"Its not easy being green"*, SIGCOMM Comput. Commun. Rev., vol. 42, no. 4, pp. 211C-222, Oct. 2012.
7. J. Koomey et al., *Growth in Data Center Electricity Use 2005 to 2010*, Analytics Press, Oakland, CA, USA, Aug. 2011.
8. R. Bashroush, *Webinar:The Coming Data Center Energy Crisis: Fact or Fiction*, Avaliable at https://uptimeinstitute.com, May 15 2018.
9. M. K. James, W. Forrest, and N. Kindler, *Revolutionizing Data Center Energy Efficiency*, McKinsey and Company, July 2008.
10. https://www.gartner.com/newsroom/id/3845563 accessed on 4-1-2018.
11. C. L. Belady and D. Beaty, *Roadmap for datacom cooling*, ASHRAE journal, vol. 47, no. 12, p. 52, 2005.
12. R. Bolze, F. Cappello, E. Caron, M. Dayde, F. Desprez, E. Jeannot, Y. Jegou, S. Lanteri, J. Leduc, and N. Melab, *Grid5000: a large scale and highly reconfigurable experimental grid testbed*, International Journal of High Performance Computing Applications, vol. 20, no. 4, pp. 481-C494, 2006.
13. J. Koomey, *Growth in data center electricity use 2005 to 2010*, The New York Times 49 (3), 2011.
14. *http://www.yole.fr*, accessed on 17-03-2018.
15. P. Barham et al., *Xen and the art of virtualization*, Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003), Bolton Landing, NY, USA, 2003.
16. X.F. Liu, Z.H. Zhan, J. D. Deng, Y. Li, T. Gu, and J. Zhang, *An Energy Efficient Ant Colony System for Virtual Machine Placement in Cloud Computing*, IEEE Transactions On Evolutionary Computation, Vol. 22, No. 1, February 2018.
17. H. Shen, and L. Chen, *Complementary VM Allocation Mechanism for Cloud Systems*, IEEE/ACM Transactions on Networking, Volume: 26 , Issue: 3 , June 2018.
18. H. Wang, and H. Tianfield, *Energy-Aware Dynamic Virtual Machine Consolidation for Cloud Datacenters*, IEEE Access, Volume: 6, pp. 15259-C15273, 2018.
19. Y. Liu, X. Sun, W. Wei, and W. Jing, *Energy-Efficient and QoS Dynamic Virtual Machine Consolidation Method in Cloud Environment*, IEEE Access, Vol. 6, May 2018.
20. S. Rahman, A. Gupta, M. Tomatore, and B. Mukherjee, *Dynamic Workload Migration Over Optical Backbone Network To Minimize DataCenter Electricity Cost*, IEEE Transactions on Green Communications and Networking, vol. 2, issue: 2, June 2018.
21. S.K. Mishra, D. Puthal, B. Sahoo, and P.P. Jayaraman, *Energy-Efficient VM-Placement in Cloud Data Center*, Sustainable Computing: Informatics and Systems, Elsevier, February 2018.
22. S.G.Domanal, and G. Ram MohanaReddy, *An efficient cost optimized scheduling for spot instances in heterogeneous cloud environment*, Future Generation Computer Systems, Elsevier vol. 84, July 2018.
23. R. Alguliyev, R. Aliguliyev, and L. Sukhostat, *Anomaly Detection in Big Data based on Clustering*, Stat., Optim. Inf. Comput., Vol. 5, pp 325-C340, December 2017.
24. Y. S. Patel, and R. Misra, *Performance Comparison of Deep VM Workload Prediction Approaches for Cloud,Progress in Computing*, Analytics and Networking, pp. 149–160, Springer, April 2018.
25. J. A. Torkestani, and M. Ranjbari, *A learning automata based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centre*, Journal of parallel and distributed computing, vol. 113, March 2018.
26. X. Ye, Y. Yin, and L. Lan, *Energy-Efficient Many-Objective Virtual Machine Placement Optimization in a Cloud Computing Environment*, IEEE Special Section On Emerging Trends, Issues, And Challenges In Energy-Efficient Cloud Computing, August 29, 2017.
27. Y. Zhang, X. Cheng, L. Chen, and H. Shen, *Energy-efficient Tasks Scheduling Heuristics with Multi-constraints in Virtualized Clouds*, Journal of Grid Computing, Springer, 2018.

28. Y. Wen, Z. Li, S. Jin, C. Lin, and Z. Liu, *Energy-Efficient Virtual Resource Dynamic Integration Method in Cloud Computing*, IEEE Special Section On Emerging Trends, Issues, And Challenges In Energy-Efficient Cloud Computing, June 29, 2017.
29. S. Wang, Z. Qian, J. Yuag, and I. You, *A DVFS Based Energy-Efficient Tasks Scheduling in a Data Center*, IEEE Special Section On Emerging Trends, Issues, And Challenges In Energy-Efficient Cloud Computing, July 11, 2017.
30. J.L.J. Laredo, F. Guinand, D. Olivier, and P. Bouvry, *Load Balancing at the Edge of Chaos: How Self-Organized Criticality Can Lead to Energy-Efficient Computing*, IEEE Transactions On Parallel And Distributed Systems, vol. 28, no. 2, February 2017.
31. J. Son J, A.V Dastjerdi, R.N. Calheiros, and R. Buyya, *SLA-Aware and Energy-Efficient Dynamic Overbooking in SDN-Based Cloud Data Centers*, IEEE Transactions On Sustainable Computing, vol. 2, no. 2, April-June 2017.
32. A. Khosravi A, L.L.H. Andrew, and R. Buyya, *Dynamic VM Placement Method for Minimizing Energy and Carbon Cost in Geographically Distributed Cloud Data Centers*, IEEE Transactions On Sustainable Computing, vol. 2, no. 2, April-June 2017.
33. H. Zheng, Y. Feng, and J. Tan, *A Hybrid Energy-Aware Resource Allocation Approach in Cloud Manufacturing Environment*, IEEE Special Section On Emerging Cloud-Based Wireless Communications And Networks, June 2017.
34. Wang et al., *Multiagent-Based Resource Allocation for Energy Minimization in Cloud Computing Systems*, IEEE Transactions On Systems, Man, And Cybernetics: Systems, vol. 47, no. 2, February 2017.
35. M. D. Minaroll, A. Mazrekaj, and B. Freisleben, *Tracking uncertainty in long-term predictions for host overload and underload detection in cloud computing*, Journal of Cloud Computing Advances, Systems and Applications, Springer, 2017.
36. A. Ashraf, and I. Porres, *Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system*, International Journal of Parallel, Emergent and Distributed Systems, Taylor and Francis, 2017.
37. D. Poola, K. Ramamohanarao, and R. Buyya, *Enhancing Reliability of Workflow Execution Using Task Replication and Spot Instances*, ACM Transactions on Autonomous and Adaptive Systems, vol. 10, no. 4, Article 30, February 2016.
38. N. T. Hieu, M.D. Francesco, and A. Yl a-Jaaski, *Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers*, IEEE Transaction On Services Computing, March 2016.
39. M. Pantazoglou, G. Tzortzakis, and A. Delis, *Decentralized and Energy-Efficient Workload Management in Enterprise Clouds*, IEEE Transactions On Cloud Computing, vol. 4, no. 2, April-June 2016.
40. H. Wu, S. Ren, G. Garzoglio, S. Timm, and G. Bernabeu, *A Reference Model for Virtual Machine Launching Overhead*, IEEE Transactions On Cloud Computing, vol. 4, No. 3, July-September 2016.
41. M. Xu, A. V. Dastjerdi, and R. Buyya, *Energy Efficient Scheduling of Cloud Application Components with Brownout*, IEEE Transactions On Sustainable Computing, vol. 1, no. 2, July-December 2016.
42. Z. Huang et al, *M-Convex VM Consolidation: Towards a Better VM Workload consolidation*, IEEE Transactions On Cloud Computing, vol. 4, no. 4, October-December 2016.
43. F. Tao et al., *BGM-BLA: A New Algorithm for Dynamic Migration of Virtual Machines in Cloud Computing*, IEEE Transactions On Services Computing, vol. 9, no. 6, November/December 2016.
44. X. Dai et al., *Energy-Efficient Virtual Machines Scheduling in Multi-Tenant Data Centers, IEEE Transactions On Cloud Computing*, vol. 4, no. 2, APRIL-JUNE 2016.
45. S. Vakilinia et al., *Energy Efficient Resource Allocation in Cloud Computing Environments*, IEEE Special Section On Future Networks: Architectures, Protocols, And Applications, December 1, 2016.
46. G. Wu, M. Tang, Y. C. Tian, and W. Li, *Energy-efficient Virtual Machine Placement in Data Centers by Genetic Algorithm*, Lecture Notes on Computer Science, Springer Berlin Heidelberg, Renaissance Doha City Center Hotel, Doha, pp. 315–323, 2012.
47. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, *Xen and the art of virtualization*, ACM SIGOPS Oper. Syst. Rev., vol. 37, no. 5, pp. 164-C177, 2003.
48. C. A. Waldspurger, *Memory resource management in VMware ESX server*, ACM SIGOPS Oper. Syst. Rev., vol. 36, no. SI,pp. 181-C194, 2002.
49. S.B. Akers et al., *A group-theoretic model for symmetric interconnection networks*, IEEE Transactions on Computers 38 (April 1989) pp. 555-C566, 1989.
50. Z. Qi, C. Lu, and R. Boutaba, *Cloud computing: State-of-the-art and research challenges*, J. Internet Services Appl., vol. 1, no. 1, pp. 7_18, 2010.
51. E. G. Coffman, M. R. Garey, and D. S. Johnson, *Johnson, Approximation Algorithms NP-Hard Problems*, Boston, MA, USA: PWS Publishing Company, 1997.
52. E. Falkenauer, *A hybrid grouping genetic algorithm for bin packing*, J. Heuristics, vol. 2, no. 1, pp. 5–30, 1996.
53. R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, *CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms*, Softw. Pract. Exper., vol. 41, no. 1, pp. 23–50, Sep. 2011.
54. K. Park, and V. S. Pai, *A mostly-scalable monitoring system for PlanetLab*, ACM SIGOPS Operating Syst. Rev., vol. 40, no. 1, pp. 65C-74, 2006.
55. A. Beloglazov, and R. Buyya, *Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints*, IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 7, pp. 1366C-1379, Jul. 2013.