BAU Journal - Science and Technology

Volume 2 | Issue 1
ISSN: 2706-784X

Article 11

December 2020

CAEE: COMMUNICATION-AWARE, ENERGY-EFFICIENT VM PLACEMENT MODEL FOR MULTI-TIER APPLICATIONS IN LARGE SCALE CLOUD DATA CENTERS

Soha Rawas
PhD, Faculty of Science, Beirut Arab University, Beirut, Lebanon, rawassoha@gmail.com

Ahmed Zekri
Assisstant Professor, Faculty of Science, Beirut Arab University, Beirut, Lebanon, a.zekri@bau.edu.lb

CAEE: COMMUNICATION-AWARE, ENERGY-EFFICIENT VM PLACEMENT MODEL FOR MULTI-TIER APPLICATIONS IN LARGE SCALE CLOUD DATA CENTERS

Abstract

the increasing demand for cloud computing services has led to the adoption of large-scale cloud data centers (DCs) to meet the user's requirements. Efficiency and managing of such DCs have become a challenging problem. Consequently, energy-efficient solutions to optimize the whole DC energy consumption, optimize the application's performance and reduce the cloud provider operational cost are crucial and needed. This paper addressed the problem of Virtual Machines (VMs) placement of multi-tier applications to maximize the compute resources utilization, minimize energy consumption, and reduce network traffic inside modern large-scale cloud DCs. The VM placement problem with communication dependencies among the VMs is modeled as an optimization problem. In this context, to solve the proposed problem, that formulated as a variant of a multiple knapsack problem, an adaptive genetic algorithm is implemented to find a near-optimal solution to the NP-complete modeled optimization problem. To validate the efficacy of the proposed model, extensive simulations are conducted using CloudSimSDN simulator. The experimental results validate the usefulness of the proposed model and its effectiveness in reducing DC energy consumption and optimize network traffic inside DC.

Keywords

Cloud computing, Multi-tier applications, energy efficiency, communication-aware scheduling

1. INTRODUCTION

In recent years, energy consumption due to cloud DCs has become an emergent topic in the field of cloud computing. According to recent studies, 3.2% of the world's total carbon emission and 3-5% of the world total electricity is due to cloud DCs, and these percentages predictable to be quadruple by 2025 (Sarkar, Chatterjee & Misra, 2015) - (Vidal, 2019). Consequently, energy-efficient solutions are a must to save cloud providers' budget and for the sake of cloud future sustainability.

Cloud computing depends on virtualization that is defined as an abstracting technology to the physical cloud resources (Rawas, Itani, Zekri & Zaart, 2017). Accordingly, virtualization can improve cloud resources utilization and become a crucial demand in the operation of modern large cloud DCs (Mendiboure, Chalouf & Krief, 2019). Therefore, the problem of VM scheduling plays an important role in minimizing the cloud DCs energy consumption, through efficient resource allocation to guarantee that the available hosts' resources are not wasted.

Although the idea of consolidating and scheduling VMs to a minimum number of cloud resources leads to energy efficiency, however, this is not an easy task. A typical DC has heterogeneous computing resources, consequently, exploiting the utilizing the overall system through hosting a number of VMs is crucial work. Typically, to host a number of VMs a cloud computing resources must provide all the VMs required resources such as CPU, memory, bandwidth, storage capacity … (Mendiboure, Chalouf & Krief, 2019). However, for many applications those have dependencies among their perspective VMs that holding the application tiers, communication dependencies is an important factor to avoid the complex load interaction between the underlying computing resources (Han, Tan, Wang, Chen, Li & Lau, 2019).

Inside any DC, the traffic flows between DC servers and application tiers cause the intra-DC network traffic. It is mainly generated by the applications that require communication among services (such as 3-tier Web and MapReduce applications). Consequently, the complex load interaction among the VMs holding a multi-tier application leads to intra-DC network congestion and performance degradation (Rawas, Itani, Zekri & Zaart, 2017). This pause scalability, reliability and overall performance of cloud computing services. CISCO Global Cloud Index (Networking, 2016), a report to forecast the growth of global DC and cloud-based IP traffic trends, states that 75% of the global DC traffic is due to the intra-DC network traffic (see Fig. 1).

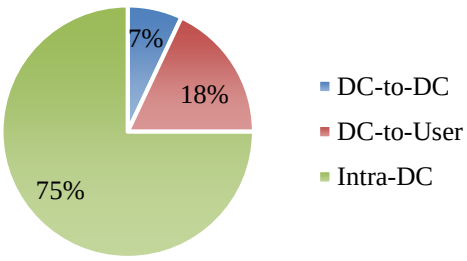

Fig.1: Global DC Traffic by Destination

Moreover, and from the cloud provider's viewpoint, minimizing the intra-Dc network traffic is an important issue to profit maximization. It leads to minimizing the packets passing through DC network devices. Consequently, reducing the overall power consumption of DC network infrastructure and guaranteed Quality of Service (QoS). The key contributions of this paper are as follows:

1- Novel VM placement model that considers communication dependencies between VMs of multi-tier application as well as the energy consumption of the hosted servers and the network devices.
2- Communication-Aware, Energy-Efficient adaptive Genetic Scheduling algorithm (CAEE-G) for VM placement to implement the model and to solve the proposed multi-objective optimization problem.

3- Extensive simulations to validate our model using both synthetic and real DC workload traces.

The paper unfolds as follows. Section 2 presents related work. Section 3 presents the DC network architecture. Section 4 describes the multi-tier applications. Section 5 shows our model formulation. Section 6 presents our proposed meta-heuristic genetic algorithm. Section 7 presents the evaluation of the proposed CAEE method. Section 8 concludes the paper

2. RELATED WORK

Energy-efficient scheduling is an emergent topic in cloud computing and can be achieved using VMs consolidations to a minimum number of operated hosts. However, interdependency among VMs is an important factor when choosing a new physical location to the requested VMs. Moreover, many researchers address the problem of minimizing the intra-DC network traffic without taking into consideration the energy cost of execution in placing VMs.

Kurdi et al. proposed LACE (Kurdi, Alismail, & Hassan, 2018), a distributed Locust-inspired scheduling algorithm, to reduce cloud DC energy consumption. The proposed model schedules and optimizes the allocation of VMs based on behavior derived from locusts. It aims to schedule load between servers rather than being centralized in one component. The proposed algorithm utilizes servers such that each one run the largest possible number of VMs based on a certain threshold. Consequently, the idle servers can be switched off inspired by the locus species in nature. However, this model doesn't address communication dependency as well as DCN energy consumption. In (Dias & Costa, 2012), Dias et al. proposed a bin-packing scheduling algorithm to reduce the traffic cost. His method based on grouping the highly communicated VMs into a number of clusters and then mapping these clusters to partitions of physical machines. Also, the proposed traffic-aware VM placement algorithm proposed by Meng et al. (Meng, Pappas, & Zhang, 2010) addressed the problem of minimizing the total communication cost among VMs. Moreover, Vivek et al. (Shrivastava, Zerfos, Lee, Jamjoom, Liu, & Banerjee, 2011), proposed the AppAware algorithm for the VM scheduling problem that formulated as a multiple knapsack optimization problem. The author considered the DC network topology, the physical machines capacity limits, as well as the real-time communication dependencies among VMs. Cao et al. (Cao, Zhang, & Liu, 2017) proposed a communication-aware and consolidation approach using modified SCAN algorithm to lower the computation cost and minimize inter-rack traffic. In (Abohamama & Hamouda, 2020), the author proposed a hybrid genetic VM placement algorithm to improve the energy consumption of cloud DC through minimizing the number of active severs. The proposed algorithm focused on reducing the resource wastage of active servers without considering the impact of network devices.

In contrast to the aforementioned, the proposed VM placement model in this paper takes into account the communication cost among the dependent VMs of multi-tier application, as well as the servers and switches energy consumption to improve the overall DC energy consumption. The basic idea is to place heavily communicated VMs on energy-efficient physical machines (PMs) that are located in the same DC rack so that inter-DC network traffic is minimized. Accordingly, CAEE differs from the other multi-tier VM placement models in the following ways:

- CAEE modeled using DC energy consumption that includes servers and switches nodes.
- The proposed model aims to optimize the intra-DC network traffic and congestion through optimizing the communicated VMs placement taking into account the available link bandwidth for data transfer .
- The proposed model exploits servers' capacities to reduce the number of active servers.
- The proposed model allocates and consolidates the requested VMs based on accurate and updated data; i.e. each physical machine allocating the new requested VMs based on a clear account of its available resources and updated load.
- The proposed algorithm initiates the consolidation approach each time a new request arrives at DC.
- The proposed model and its solution are independent on the specific network architecture type or DC specification.

3. OVERVIEW OF DATA CENTER ARCHITECTURES

3.1 Data Center Networks (DCN)

DC is a pool of resources connected using a DCN that needs to scale up to thousands of nodes to serve the cloud paradigm. Thus, scalability, high cross-section bandwidth, and fault-tolerant routing service are one of the foremost challenges to the DCNs, which provoked researchers to explore alternatives to overcome the problem of traditional 2N, tree topology. Consequently, these new topologies added a large number of network devices to achieve scalability and provide many redundant paths to ensure full bisection bandwidth and attain DC traffic needs. However, traffic at DC is far below the peak value (Najm & Tamarapalli, 2019), and the idle network devices consume significant amounts of energy since most components are always on. Therefore, using a few network devices to provide the routing service without sacrificing network performance is the key idea studied by researchers .

Recently, there are bunches of energy-aware network architecture and topologies for DCs (Irteza, Bashir, Anwar, Qazi, Dogar, 2018) - (Al-Fares, Loukissas, & Vahdat, 2008) – (Greenberg, Hamilton, Jain, Kandula, Kim, Lahiri, & Sengupta, 2009) - (Guo, Wu, Tan, Shi, Zhang, & Lu, 2008) – (Shah, Nazir, & Khan, 2017). Al-Fares et al. (Al-Fares, Loukissas, & Vahdat, 2008) proposed the fat-tree DCN architecture to overcome lack of the capability of the three-tier architecture to meet the current DC bandwidth and growth. The fat-tree architecture is scalable, energy-efficient, cost-effective, scalable, and fault-tolerant. From this discussion, we can conclude that today's DCN has redundant and over-provisioned links, resulting in high-energy consumption and redundant active links. However, the aim of this work is to minimize the packets passing through DC network devices through scheduling the heavy communicated VMs on most energy-saving PMs that located near each other without scarifying other quality constraints such as performance.

3.2 Typical DCN Architecture

Cloud computing paradigm connected through a backbone network infrastructure connected theses DCs. Cloud providers have their DC distributed among geo-locations. Therefore, there are two network connections: inter-DC network and intra-DC network as shown in Fig2. The inter DCN connect the intra-DCN as well as the inter DCN to the internet (Irteza, Bashir, Anwar, Qazi, Dogar, 2018). While the intra-DCN is made up of several layers similar to the Clos network described in (Greenberg, Hamilton, Jain, Kandula, Kim, Lahiri, & Sengupta, 2009). The bottom layer is made up of hundreds to thousands of servers connected through top-of-rack (ToR) switches use 10GbE or 40GbE Ethernet NICs (Rawas, Zekri, & El Zaart, 2018). Each bunch of servers forms a Pod (Fig2.). ToR switches connected to a second aggregate layer to form a Podset. Then the Podset is connected to a third Core layer. However, in this paper, we are focusing on intra-DCN energy optimization, while the inter DCN is out of the scope of this paper.

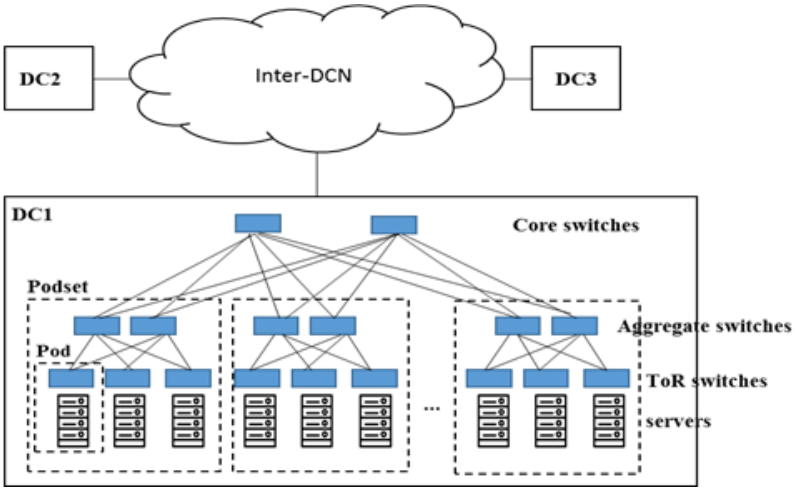

Fig.2: DCN Structure

3.3 Data Centre Energy Models

A typical DC contributes to a large amount of energy consumption due to its IT devices. However, the main devices that account for a major part of energy consumption are computing servers and DC network devices (such as switches).

3.3.1 Computing servers power model

Developing a new energy-aware VM placement model requires knowledge in computing servers' dynamic power consumption. To derive new power consumption model, real-time server power consumption monitoring is needed. However, this is out of the scope of this paper. This model uses the linear power model to predict energy consumption and saving as done by Fan et al. (Fan, Weber, & Barroso, 2007). A linear power model shows that the power consumption of a server increases linearly with its CPU utilization. Consequently, finding the server with the least CPU utilization will help reduce DC energy consumption. This following equation illustrates the computing servers' power consumption relationship:

$$P(u)=P\_idle+(P\_full-P\_idle)*u \quad (1)$$

Where Pidle is server power consumption with no load, Pfull is fully utilized server power consumption, and u is the amount of CPU utilization.

3.3.2 Network switches power model

Switches form the key enabling component of the interconnecting devices that allows passing user's request to the computing resources for execution (Rawas & Zekri, 2018). The static part includes chassis, fans, switching fabric, TCAM memory, etc. The dynamic component includes ports. Noting that the switch port consumes full power while it is active.

Based on the benchmarking suite of network devices (Irteza, Bashir, Anwar, Qazi, Dogar, 2018), the power consumption of a switch in a DC network is defined as:

$$P_{Switch} = P_{Chassis} + n_{line\ cards} * P_{line\ cards} + \sum_{i=0}^{configs} (P_{configs\ i} * Nports_{configs\ i} * Uf_i) \quad (2)$$

where $P_{Chassis}$ is the power consumed by the chassis-based hardware, $n_{line\ cards}$ is the number of line cards plugged into the switch, $P_{line\ cards}$ is the power consumed by the switch line cards with no ports turned on, $configs$ is the number of configuration for the port line rate, $P_{configs\ i}$ is the power for a port running at line rate i (i=10Mbs, 100Mbs, 1Gbs), $Nports_{configs\ i}$ is the number of ports using configuration type i, and Uf_i is the scaling factor to account for the utilization of each port.

4. MULTI-TIER APPLICATIONS

With the increasing popularity of cloud computing services, a wide variety of application types have been developed. The majority of cloud applications can be classified based on the amount of communication and computation requirement (Rawas, Zekri, & El Zaart, 2018). However, multi-tier applications dominate modern cloud applications. This type of applications divided into two or more components and the number of tiers varies by application requirements. In general, any application depends on a middleware application is known as a multi-tier application. These middleware applications can be developed, executed, reused, and tested separately.

Fig.3: Three-tier Architecture

Among different types of multi-tier applications, 3-tier applications commonly found in web services are the most commonly used architecture (Toosi, Qu, de Assunção, & Buyya, 2017). Fig3. illustrates a 3-tier web application that is decomposed into 3 main tiers:

1- Presentation tier or known as client tier, it is at the top most level of the application. It represents the user interface and application access services. The goal of this tier is to create effective operations and the interactive interface between the cloud user and the application.

Application tier or known as business, logical, or middle tier. It performs detailed processing of cloud users commands, makes logical decisions, performs calculations, and acts as a connector between the surrounding layers.

2- Data-tier is also known as a database or data store tier. It includes the database servers that manage to store and to retrieve the data needed by application tier.

As an example, a traditional 3-tier web application is made up of the front-end web server, middle tier known as an application server, and back-end database server. These tires are deployed and hosted in cloud DCs. The computing and the data components (VMs) of such applications need to exchange data between tiers. Therefore, the performance of these applications influenced by the communication latency generated by the communicated data and computing components.

5. THE VM PLACEMENT PROBLEM

The process of the VM placement problem is defined as the number of possible suitable mapping of VMs to available PMs such as the objective of the proposed problem is achieved. In this paper, the aim of the CAEE models is to minimize the total energy consumption of the DC and to minimize the volume of traffic between the communicated VMs that leads to intra-DC network optimization. This could be attained through an optimal mapping of VMs to PMs such as the following are achieved:

1. Minimize servers' energy by reducing the number of active servers and consolidating VMs to the minimum number of PMs.
2. Minimize switches' energy consumption by reducing the number of active switches
3. Minimize intra-DC network traffic by placing communicated VMs near each other.

5.1 Mathematical Formulation

Consider a DC that has a well-designed physical network topology connected its physical resources as shown in Fig2. Let P= {p1, p2,… ,pn} be the set of physical machines (PMs) inside DC, where n is the total number of PMs. Each PM pi has three different resource capacities: $p_i = (p_i^{core}, p_i^{ram}, p_i^{storage}, p_i^{bandwidth})$, where the four vector components represent the number of available cores, the amount of RAM available, the storage capacity available, and the amount of available bandwidth respectively. Let S= {s1, s2,… ,ss} be the set of network switches, where s is the total number of switches in the physical network topology of DC. Let V = {vm1, vm2, …, vmm} be the set of VMs, where m is the total

number of VMs in multi-tier application. Each VM vmj is to be placed into PM pi request three resource capacities represented by the vector $vm_{i,j} = (vm_{(i,j)}^{core}, vm_{(i,j)}^{ram}, vm_{(i,j)}^{storage}, vm_{(i,j)}^{bandwidth})$, where the four vector components represent the number of cores needed, the amount of RAM, the storage capacity requested, and the amount of bandwidth needed respectively. Table 1 summarizes the various notations used in the VM scheduling problem

Let the dependency graph G=(V, E), represents the communication dependencies among a set of VMs running multi-tier application; such that V is the set of VMs and E is the set of virtual edges representing the dependency between the VMs. Fig4. illustrates the dependency relation between VMs such as the edge $e_{i,j}$ represent a logical or virtual communication link between two communicated VMs $vm_{i,k}$ & $vm_{j,l}$ which are mapped to a set of physical network links.

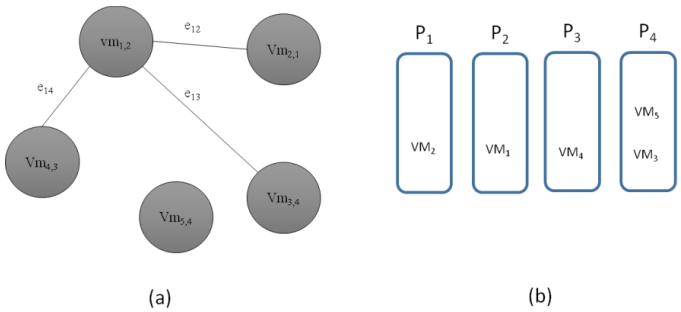

Fig.4: (a): VMs dependency graph, (b) mapped VMs to PMs

Table 1: CAEE problem notation description

Notation	Description
G	Dependency graph
E	Set of Edges between VMs
P	Set of PMs
N	Total number of PMs
V	Set of VMs
$p_i^{core,RAM,Storage}$	Amount of available CPU, RAM, and storage respectively of p_i
n	Total number of VMs
$vm_{i,j}$	vm_j hosted by PM p_i
$vm_{i,j}^{core,RAM,Storage}$	vm_j request amount of CPU, RAM, and storage respectively that allocated to p_i
$e_{i,j}$	Virtual communication link between two VMs $vm_{i,k}$ & $vm_{j,l}$
$TW(vm_{i,k}, vm_{j,l})$	Traffic weight between two dependent or communicated $vm_{i,k}$ and $vm_{j,l}$
$TL(vm_{i,k}, vm_{j,l})$	Traffic load between two dependent or communicated $vm_{i,k}$ and $vm_{j,l}$
$Power(p_i^v)$	Power consumption of a PM p_i holding number of VMs v during the slot time
$Power(s_i)$	Power consumption of a switch s_i

5.2 VMs/PMs Placement Relationship

Each PM can hold more than one VM and each VM is executed at only one PM. Let A be m x n matrix showing the mapping status of the m VMs to the n PMs as follows:

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

where a_{ij} is a binary variable (0/1) such that:

$$a_{ij} = \begin{cases} 1, & if\ vm_i\ is\ allocated\ to\ PM\ pj \\ 0, & otherwise \end{cases}$$

Since each VM is executed at only one PM, we have $\sum_{j=1}^{n} a_{ij} = 1, \forall\ i,\ i \in m.$

Therefore, the mapping process of 5 VMs to 4 PMs will be as follows (as shown in Fig.4.):

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5.3 VMs Communication Dependencies Relationship

The communication dependencies relationship among a set of VMs running the multi-tier application represented considering the following dependency relationship.

Let B be m x m matrix showing the dependencies between the m VMs as follows:

$$B = \begin{bmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mm} \end{bmatrix}$$

where b_{ij} is a binary variable (0/1) such that:

$$b_{ij} = \begin{cases} 1, & if\ vm_{i,k}\ and\ \ vm_{j,l}\ are\ dependent \\ 0, & otherwise \end{cases}$$

Therefore the matrix B that shows the dependencies between the 5 VMs as shown in Fig4. will be as follows:

$$B = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

5.4 Intra-DC Communication Traffic (IDCT)

VMs of Multi-tier application may need to communicate with each other. These communicated VMs may be hosted on different PMs. Consequently, the edge (e_{ij}), as shown in Fig4., represents the virtual communication channel that carries the communicated data between the two connected VMs (see Fig5.).

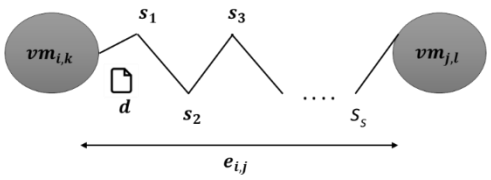

Fig.5: data (d) transfer from vmi, to vmj

The network delay, due to the intra-DC communication traffic, that measured in seconds is directly proportional to the amount of data transferred between two dependent VMs and bandwidth channel type and capacity.

Consider two dependent VMs $vm_{i,k}$ *and* $vm_{j,l}$ as shown in Fig5. The total time to transfer data d from $vm_{i,k}$ to $vm_{j,l}$ using s switches can be calculated using the following equation:

$$TT(vm_{i,k}, vm_{j,l}) = \frac{Size(d)}{AvailableLinkBw(l_1)} + delay(l_1) + \frac{Size(d)}{AvailableLinkBw(l_2)} + delay(l_2) + \cdots + \frac{Size(d)}{AvailableLinkBw(l_s)} + delay(l_s) \quad (3)$$

Therefore, using Equation 3, the total transfer time TT can be formulated using the following equation:

$$TT(vm_{i,k}, vm_{j,l}) = \sum_{a=1}^{s} \frac{Size(d)}{AvailableLinkBw(l_a)} + delay(l_a)$$

where $Size(d)$ is the size of data d (measured in bits), $AvailableLinkBw(l_a)$ is the available bandwidth capacity of the link l_a (measured in bits/seconds), and $delay(l_a)$ is the delay time due to transmission medium type (measured in seconds). Noting that, If there is no dependency between $vm_{i,k}$ and $vm_{j,l}$ then $TT(vm_{i,k}, vm_{j,l}) = 0$ as shown in matrix B related to Fig4.

Therefore, the whole intra-DC communication traffic during time T can be determined using the following equation:

$$IDCT = \sum_{i,j=0}^{m} a_{i,j} * b_{i,j} * TT(vm_{i,k}, vm_{j,l}),\ \forall(k,l),\ k,l \in n \quad (4)$$

Cloud provider can host different applications inside one DC. Therefore, different traffic types can be generated such as:

- Guest traffic: traffic that generated due to communication between multiple VMs belongs to the same tenant.
- Public traffic: traffic that generated due to and from the internet bound for VMs in DC.
- Storage traffic: traffic generated due to moving large chunks of data for running a specific type of heavy communicated applications such as Hadoop.

This traffic should be isolated from each other to prevent intra-DC network congestion and cloud provider service degradation. Therefore, cloud providers employ network virtualization techniques to improve intra-DC network performance. When customers send their application to a cloud environment, the cloud provider provides them with a virtual topology that consists of VM types and virtual links between VMs. These virtual links are mapped to a set of physical links (as Fig5. sketched), while the VMs are mapped to PMs. Nevertheless, and due to security reasons, cloud providers never provide any information about these virtual topologies, in addition to the blurring image provided by the scientific literature about the virtual network performance (Abdi, PourKarimi, Ahmadi, & Zargari, 2017).

Therefore, and since there is no general analytical model to measure the available bandwidth and the virtual transmission channel delay of the virtual route between communicated VMs, in this paper, we modeled the $TT(vm_{i,k}, vm_{j,l})$ between two dependent VMs using Traffic Weight (TW) and Traffic Load (TL) as follows:

Traffic Weight (TW) It denotes the number of hops between $vm_{i,k}$ and $vm_{j,l}$ and represented by $TW(vm_{i,k}, vm_{j,l})$. If there is no dependency between $vm_{i,k}$ and $vm_{j,l}$ then $TW(vm_{i,k}, vm_{j,l}) = 0$. Furthermore, TW between two VMs hosted on the same PM defined as $TW(vm_{i,k}, vm_{j,l}) = 0$ since the two VMs will communicate using the PM memory (Rawas, Itani, Zekri & Zaart, 2017).

Traffic load (TL) is the average size of the communication load (measured in bytes) between two adjacent switches in the virtual route connecting the two dependent VMs. Consequently, the TL between the two communicating $vm_{i,k}$ and $vm_{j,l}$ is represented by $TL(vm_{i,k}, vm_{j,l})$. If there is no dependency between $vm_{i,k}$ and $vm_{j,l}$ then $TL(vm_{i,k}, vm_{j,l}) = 0$ as shown in matrix B related to Fig4. (see Section 3).

Therefore, the whole intra-DC communication traffic during time interval T can be predicted using the following equation:

$$IDCT = \sum_{i,j=0}^{m} a_{i,j} * b_{i,j} * TW(vm_{i,k}, vm_{j,l}) * TL(vm_{i,k}, vm_{j,l}),\ \forall(k,l),\ k,l \in n \qquad (5)$$

Accordingly, for a stable DC traffic map at time T, the objective is to minimize the whole intra-DC communication traffic (IDCT).

5.5 DC Energy Consumption (DCEC)

DCEC is defined as a demand for Execution Energy Consumption (EEC) & Switches Energy Consumption (SEC).

5.5.1 Execution energy consumption (eec)

Execution Energy Consumption (EEC) is modeled using the linear power model described in Equation 1. Let's consider $Power(p_k^v)$ as the power consumption of a PM pk holding number of v VMs during the slot time [0, T]; such that each PM can hold more than one VM: $p_k^v = \sum_{i=1}^{v} vm_{i,k}$ and each VM is executed at only one PM.

Consequently, the total EEC of PMs P holding number of VMs V is modeled as follows:

$$EEC = \sum_{i=1}^{n} Power(p_i^v) * t_i \qquad (6)$$

5.5.2 Switches energy consumption (sec)

Switch chassis is the main contribution of energy consumption in the DCN network component (Shrivastava, Zerfos, Lee, Jamjoom, Liu, & Banerjee, 2011). It consumes about 90% of its maximum power as soon as it is turned on. Therefore, setting the idle devices into sleeping mode can minimize the total consumption of DCN. Consequently, this method helps to ensure reliability, performance, and availability since turning off the idle network devices takes a significant amount of time to boot computing which leads to network performance degradation.

For s numbers of switches, the total power of active switches should be minimized. Therefore, the total power consumption will be the power of all active switches such as:

$$\mathrm{SEC} = \sum_{\mathrm{i}=1}^{s} \mathrm{P}(\mathrm{s_i}) * t_i \qquad (7)$$

where $\mathrm{P}(\mathrm{s_i})$ is the power consumption of switch i calculated based on the active ports (see Equation 2).

5.6 The Optimization Problem

The main aim of the proposed CAEE VM scheduling model is to minimize the total DC energy consumption and intra-DC network traffic. This could be achieved through minimizing the operated PMs through mapping the heavy communicated VMs on energy-efficient PMs located in the same Racks that leads to VMs consolidation (as shown in the experimental results, Section 6 – Fig10.). Using Equations 5, 6, and 7, the objective function is defined as follows:

$$Minimize \begin{pmatrix} IDCT \\ DCEC \end{pmatrix} \quad (8)$$

Subject to:

1. Placement constraint: the VM placement constraint represents the relationship between the VMs and PMs. It mandates that each VM executed and assigned to only one PM such that:

$$\sum_{j=1}^{n} a_{ij} = 1 \quad , \forall\, i, i \in m \quad (9)$$

2. Capacity constraints: these constraints indicate that the resources requirements of the mapped VMs on a PMs cannot exceed the total capacity of the PMs in terms of CPU (Equation 10), RAM (Equation 11), storage (Equation 12), and bandwidth (Equation 13) as follows:

$$\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} * vm_{(i,j)}^{core} \leq p_{(i,j)}^{core} \quad (10)$$

$$\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} * vm_{(i,j)}^{RAM} \leq p_{(i,j)}^{RAM} \quad (11)$$

$$\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} * vm_{(i,j)}^{storage} \leq p_{(i,j)}^{storage} \quad (12)$$

$$\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} * vm_{(i,j)}^{bandwidth} \leq p_{(i,j)}^{bandwidth} \quad (13)$$

$$a_{ij},\ b_{ij} \in \{0, 1\}$$

6. PROPOSED GENETIC VM PLACEMENT ALGORITHM

CAEE that formulated as an optimization problem (as shown in the above section), is a variant of multiple knapsack (Rawas, Itani, Zekri & Zaart, 2017) (Meng, Pappas, & Zhang, 2010). Thus, and to solve the proposed NP-complete problem, this paper proposes an adaptive genetic-based VM scheduling algorithm that entitled as a Communication-Aware, Energy-Efficient Genetic scheduling algorithm (CAEE-G) as an approximate solution to this NP-complete problem.

Since the objective of the proposed CAEE model is a multi-objective problem (Equation 8), one can affect the optimization of the other. Therefore, a genetic algorithm (GA) is well-suited to solve this kind of problems. GA (Abdi, PourKarimi, Ahmadi, & Zargari, 2017) that works via the process of natural selection is a well-known kind of heuristic random algorithm that used to solve the multi-objective optimization problems. GA that follows the process of natural selection and evolution uses a number of genetic operators to generate such an optimum solution.

The GA complexity that denoted as O(gnm) depends on a number of factors such as: the number of generations (g), the population size (n), and the size of the individuals/chromosomes (m).

Encoding: encoding the problem into a valid chromosome is an important and critical work to solve the optimization problem correctly. In this paper, the encoding of the proposed CAEE VMs scheduling problem has done using a number of genes, such that each one stands for a VM. The gene value encoded as a binary positive integer to indicate the hosted rack. An example of 3 racks and 10 VMs are used in Table 2 to illustrate the used encoding process.

Table 2: VMs encoding representation

VM	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Gene	0001	0011	0011	0001	0011	0010	0001	0011	0010	0001
Rack	1	3	3	1	3	2	1	3	2	1

Crossover: the crossover operator used to generate a new fittest population. Consequently, and using a roulette wheel selection method, two random chromosomes are selected from the filtered fittest population and one point crossover is used in generating a new population.

Mutation: the mutation operator used to maintain genetic diversity in the successive generations to avoid generating identical generations. In this paper, the applied mutation is done according to a predefined probability.

Fitness function: the genetic fitness function that used to design a successful GA algorithm should reflect correctly the objective function of the proposed optimization problem that intended to be solved. Since the main aim of CAEE model is to optimize the DC energy consumption and the intra-DC network traffic, the fitness function is taken to be the CAEE objective function that modeled in Section 5 – Equation 8.

The CAEE-G algorithm, designed to solve the proposed VM scheduling model, illustrated in Algorithm 1 that shows a high-level pseudo-code. As shown in Algorithm 1, the GA goes through the following phases: 1- the creation of the initial random population (line 2), where each individual in the population considered a solution to the problem. 2- Fitness evaluation (line 3) (calculated using Equation 8). 3- Parents selection to generate the new fittest populations by applying crossover and mutation GA operators to the selected parents (line 4-14). After a number of iterations, the algorithm retrieves the individual with the highest fitness from the last population as a solution to the problem (line 14).

Algorithm 1: CAEE-G

Input: a set of requested VMs, Set of available PMs

Output: allocated VMs.

Processing:

1. Begin
2. Generate initial population of randomly encoded chromosomes
3. Evaluate the fitness function for each chromosome
4. Evaluate the correctness of each chromosome through checking the CAEE model constraints (Equations 9- 13)
5. Produce a valid fittest population through dropping the chromosomes that violate CAEE model constraints
6. Pass the top two fittest chromosomes to the next generation (consider them as elite)
7. Do
8. Select two parents' chromosomes using the random roulette wheel selection method
9. Apply the crossover operator
10. Move the newly created fittest children to the next generation
11. Until the new generation size = the initial population size
12. Replace the current population with the newly created generation.
13. Apply a mutation operator using a small probability Pm=0.01- 0.02
14. Return to line 3 until the end of the iteration
15. end

7. EVALUATION

To measure the effectiveness of the proposed CAEE model, CloudSim 3.0.3 has been used to design and conduct the experiments. However, CloudSim does not provide an environment for the intra-DC network level. Therefore, and to address this shortcoming, CloudSimSDN, which is an add-on library that supports and enables dynamic configuration and management of intra-DC network via a centralized controller has been added (Son, Dastjerdi, Calheiros, Ji, Yoon, Buyya, 2015). It enables dynamic configuration and management of DC network via a centralized controller. It allows developing a large variety of intra-DC network scenarios. Moreover, CloudSimSDN offered users' detailed modeling of the energy consumed by the DC resources on the compute (PMs) and different network levels (edge, aggregation, and core network level).

All the simulated experiments conducted on Intel(R) Core(TM) i7 Processor 3.4GHz, Windows 7 platform using NetBeans IDE 8.0.2 and JDK 1.8. to overcome the importance of the proposed model in different cloud environments, different scenarios were generated such as varying: the number of PMs, the DC topology, the number of VMs, the type of dependencies among VMs, the load characteristics and more… to emulate a real cloud environment, Amazon VMs specifications and characteristics are used (as shown in Table 3). The emulated servers are of two types as shown in Table 4, such that the used servers' power consumption is based on a benchmark result provided by SPEC (Amazon, 2019). Table 5 shows the parameters setting of the genetic algorithm based on a benchmark used parameters (Yin, Jin, Shen, & Huang, 2017).

Table 3: Amazon VMs Specifications

VM instance type	VM Model	Cores	MIPS	RAM (GB)	Bandwidth (Mbps)
Web Server	Standard	1	2000	0.5	100 Mbps
	Memory optimized	2	1500	2	100 Mbps
Application Server	Large	1	1500	2	100 Mbps
	XLarge	2	2400	3	100 Mbps
Database Server	Medium	2	2000	8	450 Mbps
	Large	4	2400	16	750 Mbps

Table 4: HP Servers, Host Load to Energy (Watt) mapping

Server type	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP G5	93.7	97	101	105	110	116	121	125	129	133	135

Table 5: Genetics Parameter Settings

Parameter	Value
Population size	100
Number of generations	100
Crossover rate	0.75
Mutation rate	0.02

A three-tier DCN topology was used in the simulation as shown in Fig2. The simulated DCN network has 2 core switches which connected to 8 aggregation switches which in its turn connected to 8 edge switches. The edge switches connected to 16 racks each one holds 16 servers. The servers' interconnection inside racks is done using 1 GE links. However, 10 GE links were used to form the three-tier DCN topology interconnecting core, aggregation, and edge switches.

The number of randomly generated VMs, based on the VM types specified in Table 2, varies between 300-1200 to serve 100-400 cloud customers. Each user has a 3-tier running application: Web, App, and DB server. For each user, the generated workload is based on a typical web services model (Son, Dastjerdi, Calheiros, Ji, Yoon, Buyya, 2015) to ensure switches are working during the created VMs lifetime.

To evaluate the proposed model, the CAEE algorithm compared to the following commonly used heuristics: Best Fit (BF), Worst Fit (WF), Combined Least Full First (CLFF), and Combined Most Full First (CMFF). BF policy chooses the fullest host in terms of computing power (MIPS). WF policy chooses the least full host in terms of computing power (MIPS). However, CLFF and CMFF are joint host-network energy-efficient policies. CLFF policy chooses the least full host in terms of both compute power and network bandwidth while CMFF policy chooses the fullest host in terms of both computing power and network bandwidth.

In this experiment, three metrics are used to find the effectiveness of the proposed model: energy consumption of PMs and DCN devices, the maximum number of instantaneously utilized hosts and switches, and the intra-DC network traffic.

7.1 Results and Analysis

Intra-DC Network Traffic. Fig6. shows the importance of the proposed model in reducing the intra-DC network traffic (using Equation 5). As figure plots, CAEE-G outperforms all the other competing algorithms even the network-aware ones (CLFF and CMFF). CAEE-G placed the communicated VMs in the same rack and near each other. Therefore, it decreases the intra-DC network traffic since most of the communicated VMs are now connected through servers' memory instead of DCN devices. As the result reveals CAEE-G achieves the workload consolidation that leads to minimize the network congestion. Consequently, minimizing the network congestion ensures that no additional delays in communicated VMs are caused due to intra-DC network overloading.

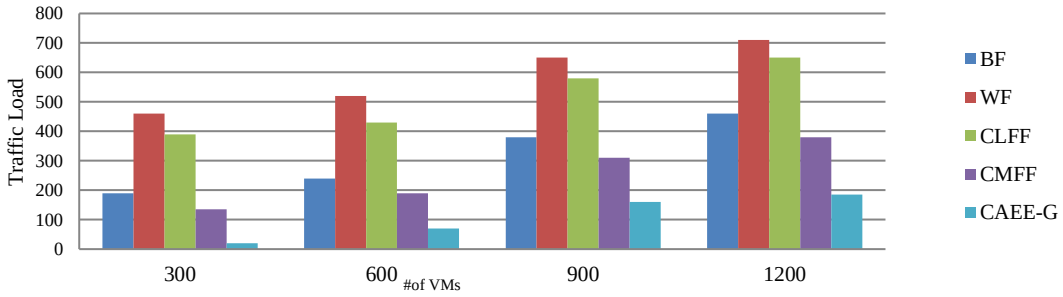

Fig.6: Intra-DC Network Traffic

DCN's Power Consumption. Fig7. depicts that using CAEE-G 30% of energy consumption of DCN devices can be saved. In details, this energy-saving is due to the consolidation of the communicated VMs which can in its turns consolidate the intra-DC network traffic that reveals the causes of 30% deactivation of DCN devices as Fig8. exposes and reduce the number of active switches.

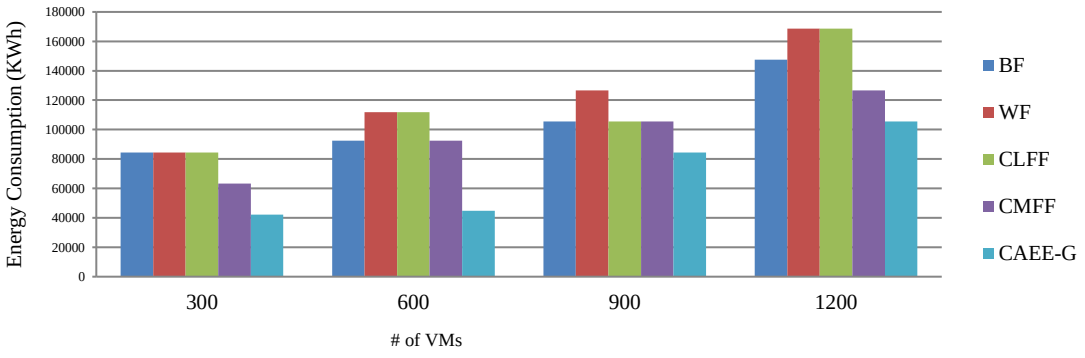

Fig.7: DCN Energy Consumption

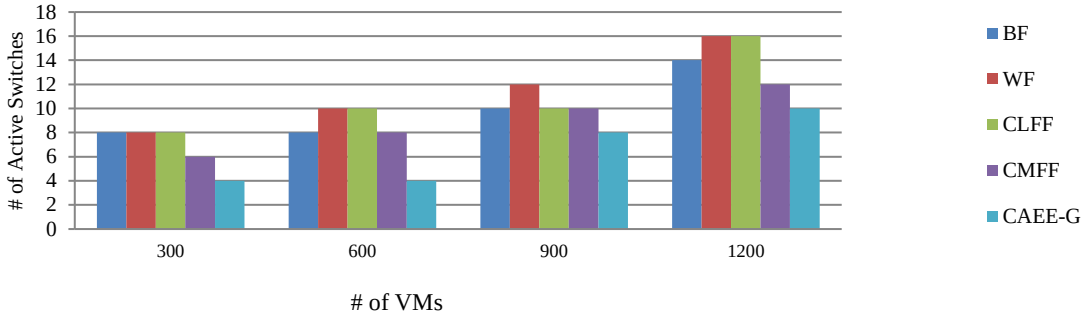

Fig.8: # of Active Switches

DC servers' Energy Consumption. Fig9. shows 40% of servers' energy consumption can be saved when CAEE-G is applied. This energy efficiency is justified by the decreases in the number of active servers due to workload consolidation. As a result, depicts, the WF and CLFF do not optimize energy consumption since it tends to use as the least full host in terms of computing power to host VMs. However, BF and CMFF provide better power efficiency compared to WF and CLFF. Nevertheless, CAEE-G outperforms all the other competing algorithms even with different workload since the CAEE-G model consolidates the VMs to the fewest servers to minimize the network traffic as well as use the most efficient server to host the VMs.

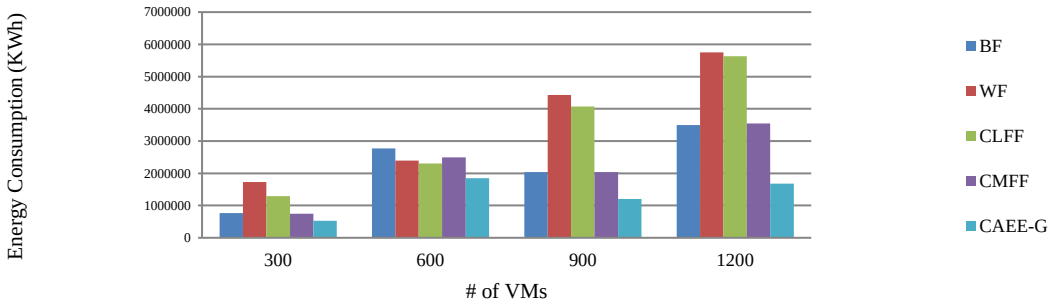

Fig.9: DC Server's Energy Consumption

Fig10. reflects the result of energy reduction. As shown the number of active servers reduced by 50% compared to WF and CLFF, and by 30% compared to BF and CMFF. This result reveals the importance of considering the cost of communication between VMs in consolidating the number of active servers.

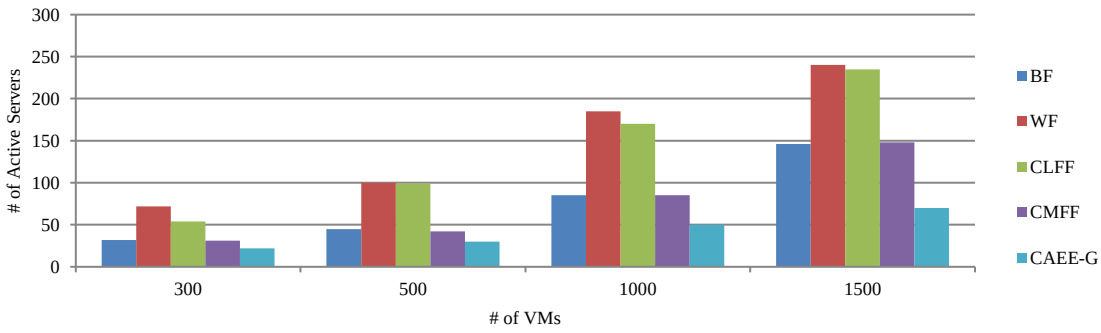

Fig.10: # of Active Servers

Total DC Energy Consumption. Fig11. shows the total energy efficiency attained by using CAEE model. In this paper, the total energy efficiency is considered as the total reduction including both DCN devices and DC servers. Fig11. shows that as the number of workload increase the percentage of total energy efficiency decreases. This is because when the number of VMs increases highly, the number of unused servers is decreasing to accommodate and host all the requested VMs. According to the above analysis, DC servers have more impact on total energy consumption when compared to DCN effect. However, in order to ensure optimum energy efficiency and minimum environmental footprint, it is necessary to consider the DCN energy efficiency.

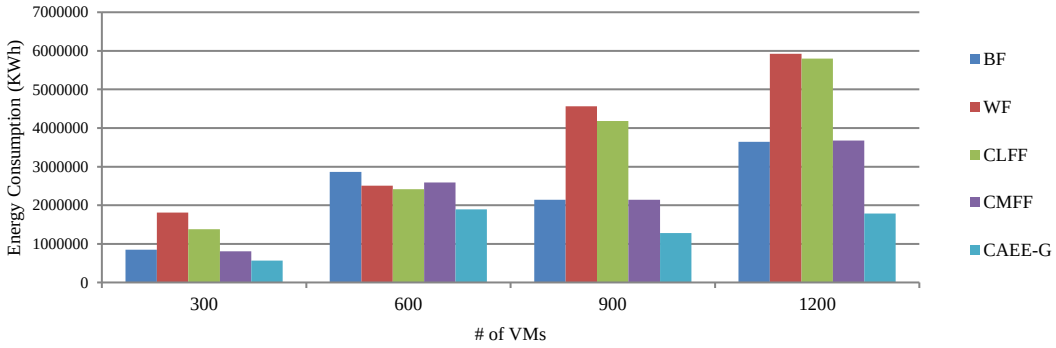

Fig.11: Total DC Energy Consumption

Genetic Convergence. Genetic convergence to find the optimal solution is a critical issue in solving optimization problems. The convergence depends on tuning different genetic operators such as crossover and mutation to avoid premature convergence that leads to suboptimal solutions (Yin, Jin, Shen, & Huang, 2017). Fig12. indicates that CAEE-G characterized by a high convergence that leads up to almost 40% of processing time-saving.

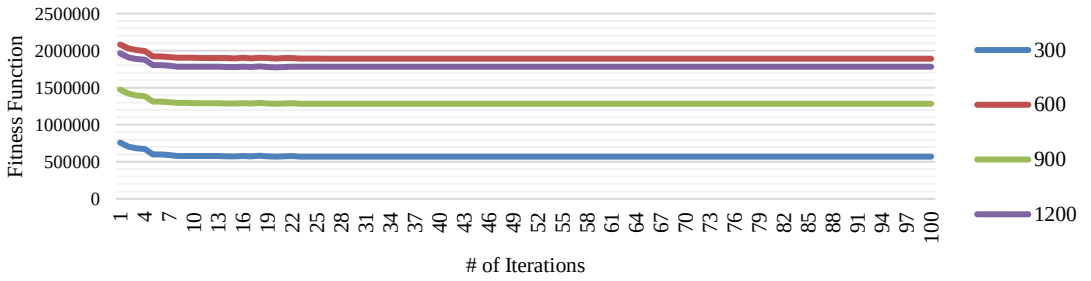

Fig.12: Genetic Convergence

Genetic Optimality. As shown in Fig13, 79% of the communicated VMs mapped to the same rack and this obviously leads to Intra-DC network traffic minimization. However, 61% of the racks' hosts contain communicated VMs. Moreover, Fig13. reveals that 76% of the complete cloud users' request processed using one rack; i.e. the communicated cloud user request (as shown in Table 3) that made up of the 3 communicated VMs (database, web, application) is handled using one rack.

Fig8. and Fig10. show the effect of CAEE-G algorithm in reducing the number of active hosts and switches which leads to minimizing the whole DC energy consumption (as shown in Fig11.). This result is due to the fact that the communicated VMs consolidated to a minimum number of hosts (as shown in Fig13.). Combining these facts disclose the validity of CAEE-G algorithm in achieving its fitness function that reflects our model objective in minimizing the IDCT and DCEC (as shown in Equation 8). On the other hand, although the experimental results proved the efficacy of the CAEE-G algorithm in achieving our model aim over a number of competing algorithms, however, the biggest limitation of GA is that it cannot guarantee optimality (Amazon, 2019).

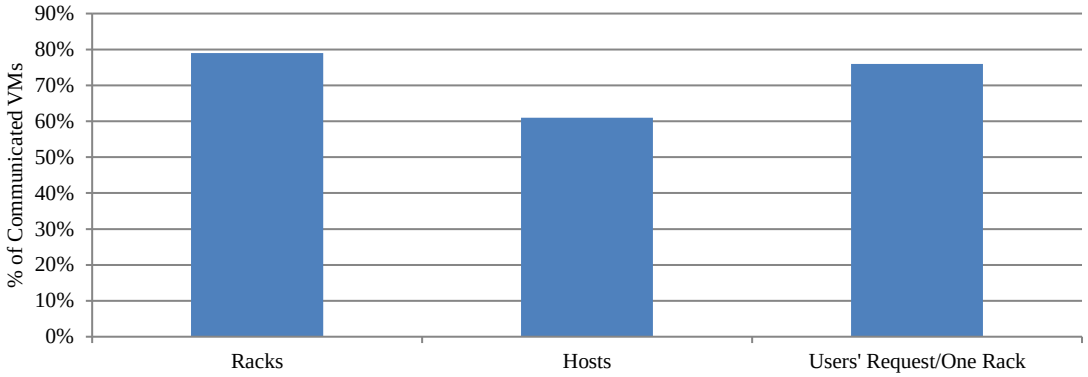

Fig.13: Genetic Optimality

8. CONCLUSION AND FUTURE WORK

VM placement is an important cloud computing problem that affects the energy consumption of DCs. Dependencies among VMs, when existed, must be taken into account during the placement process to ensure energy-efficient network resource utilization and minimize traffic between distant VMs hosts in different DCs. This paper introduced CAEE, a communication-aware energy-efficient VM scheduling model that consolidated communicated VMs to the minimum number of PMs and placed them into nearby hosts. The proposed algorithm aims to minimize DC energy consumption, intra-DC network traffic, and optimize DC computing resources utilization. It attains the tradeoff between VMs consolidation (to minimize the number of active servers) and intra-DC network traffic optimization (through hosting heavy communicated VMs to PMs that are located close to each other). The simulation results for a multi-tier application showed the importance of the proposed algorithm to minimize the energy consumption of DCs.

For the future work, our plan is to investigate the scalability of the proposed CAEE VM placement model under different network topologies and traffic patterns. Further, we aim to evaluate the proposed model in a real cloud platform, such as OpenStack software, to explore the significance of CAEE model by modern IT infrastructure.

REFERENCES

- Abdi, S., PourKarimi, L., Ahmadi, M., & Zargari, F. (2017). Cost minimization for deadline-constrained bag-of-tasks applications in federated hybrid clouds. *Future Generation Computer Systems*, 71, 113-128.
- Abohamama, A. S., & Hamouda, E. (2020). A hybrid energy–Aware virtual machine placement algorithm for cloud environments. *Expert Systems with Applications*, 150, 113306.
- Al-Fares, M., Loukissas, A., & Vahdat, A. (2008). A scalable, commodity data center network architecture. *ACM SIGCOMM Computer Communication Review*, 38(4), 63-74.
- Amazon (2019). https://aws.amazon.com/about-aws/global-infrastructure/
- Cao, G., Zhang, C., & Liu, W. (2017, December). Fast communication-aware virtual machine dynamic consolidation for cloud data center. In *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)* (pp. 237-244). IEEE.
- Dias, D. S., & Costa, L. H. M. (2012, December). Online traffic-aware virtual machine placement in data center networks. In *2012 Global Information Infrastructure and Networking Symposium (GIIS)* (pp. 1-8). IEEE.
- Fan, X., Weber, W. D., & Barroso, L. A. (2007). Power provisioning for a warehouse-sized computer. *ACM SIGARCH Computer Architecture News*, 35(2), 13-23.
- Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., ... & Sengupta, S. (2009, August). VL2: a scalable and flexible data center network. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data communication* (pp. 51-62).

- Guo, C., Wu, H., Tan, K., Shi, L., Zhang, Y., & Lu, S. (2008, August). Dcell: a scalable and fault-tolerant network structure for data centers. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data communication* (pp. 75-86).
- Han, Z., Tan, H., Wang, R., Chen, G., Li, Y., & Lau, F. C. M. (2019). Energy-efficient dynamic virtual machine management in data centers. *IEEE/ACM Transactions on Networking*, 27(1), 344-360.
- Irteza, S. M., Bashir, H. M., Anwar, T., Qazi, I. A., & Dogar, F. R. (2018). Efficient load balancing over asymmetric datacenter topologies. *Computer Communications*, 127, 1-12.
- Kurdi, H. A., Alismail, S. M., & Hassan, M. M. (2018). LACE: A locust-inspired scheduling algorithm to reduce energy consumption in cloud datacenters. *IEEE Access*, 6, 35435-35448.
- Mendiboure, L., Chalouf, M. A., & Krief, F. (2019). Edge computing based applications in vehicular environments: Comparative study and main issues. *Journal of Computer Science and Technology*, 34(4), 869-886.
- Meng, X., Pappas, V., & Zhang, L. (2010, March). Improving the scalability of data center networks with traffic-aware virtual machine placement. In *2010 Proceedings IEEE INFOCOM* (pp. 1-9). IEEE.
- Najm, M., & Tamarapalli, V. (2019, January). A cost-aware algorithm for placement of enterprise applications in federated cloud data center. In *Proceedings of the 20th International Conference on Distributed Computing and Networking* (pp. 510-510).
- Networking, C. V. (2016). Cisco global cloud index: Forecast and methodology, 2015-2020. white paper. Cisco Public, San Jose.
- Rawas, S., & Zekri, A. (2018). Location-Aware Energy-Efficient Workload Allocation in Geo Distributed Cloud Environment. *J. Comput. Sci.*, 14(3), 334-350.
- Rawas, S., Itani, W., Zekri, A., & Zaart, A. E. (2017, March). ENAGS: energy and network-aware genetic scheduling algorithm on cloud data centers. In *Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing* (pp. 1-7).
- Rawas, S., Zekri, A., & El Zaart, A. (2018). Power and Cost-aware Virtual Machine Placement in Geo-distributed Data Centers. In *CLOSER* (pp. 112-123).
- Sarkar, S., Chatterjee, S., & Misra, S. (2015). Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Transactions on Cloud Computing*, 6(1), 46-59.
- Shah, S. A., Nazir, B., & Khan, I. A. (2017). Congestion control algorithms in wireless sensor networks: Trends and opportunities. *Journal of King Saud University-Computer and Information Sciences*, 29(3), 236-245.
- Shrivastava, V., Zerfos, P., Lee, K.-W., Jamjoom, H., Liu, Y.-H., & Banerjee, S. (2011). Application aware virtual machine migration in data centers. In *Proceedings of INFOCOM*, 2011 (pp. 66–70). IEEE. doi:10.1109/INFCOM.2011.5935247
- Son, J., Dastjerdi, A. V., Calheiros, R. N., Ji, X., Yoon, Y., & Buyya, R. (2015, May). Cloudsimsdn: Modeling and simulation of software-defined cloud data centers. In 2015 *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (pp. 475-484). IEEE.
- Toosi, A. N., Qu, C., de Assunção, M. D., & Buyya, R. (2017). Renewable-aware geographical load balancing of web applications for sustainable data centers. *Journal of Network and Computer Applications*, 83, 155-168.
- Vidal, J, (2019). http://www.climatechangenews.com/2017/12/11/tsunami-data-consume-one-fifth-global-electricity-2025/.
- Yin, Z. Y., Jin, Y. F., Shen, S. L., & Huang, H. W. (2017). An efficient optimization method for identifying parameters of soft structured clay by an enhanced genetic algorithm and elastic–viscoplastic model. *Acta Geotechnica*, 12(4), 849-867.