International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 8, No.4 (July-2019)

http://dx.doi.org/10.12785/ijcds/080401

Managing Performance Interference Effects for Intelligent and Efficient Virtual Machines Placement based on GWO Approach in Cloud

Hedi HAMDI[1], Sabrine AMRI[2] and Zaki Brahmi[3]

[1] Jouf university Sekaka, Kingdom of Saudi Arabia
[2]University of Monteral Montral, QC H3T 1J4, Canada
[3]Taibah University Al-Ola, Kingdom of Saudi Arabia

Received 22 Feb. 2019, Revised 6 May 2019, Accepted 28 May 2019, Published 1 July 2019

Abstract: Cloud computing paradigm has been a trend in the computational world. Thus, many service providers today are competing to enhance their features to attract more customers as they are offering them a bunch of features through a pay-as-you-go pricing model. However, despite their huge fame, cloud environments still suffer from some issues that are being studied by researchers from various perspectives. One of the controversial cloud issues nowadays is interference among virtual machines (VMs) sharing the same hardware platform called also physical machine (PM). This problem occurs due to contention on shared resources (e.g. CPU, disk, memory, network I/O, etc.) between co-hosted VMs which results in a performance degradation. The co-hosting of VMs on the same PM, emerges from the ambition of server consolidation that cloud providers aim to reach in order to improve power efficiency and optimize resource utilization. Furthermore, the Virtual Machine Placement (VMP) is one of the most challenging problems in cloud environments management and it is being studied from various perspectives. Therefore, the key factor of successful server consolidation is to minimize performance interference among co-located VMs. In this paper, we are going to review two closely related research lines (i.e. the inter-VM interference detection and/or prediction and the interference-aware virtual machine placement in cloud computing environments), give a comparative study between the reviewed approaches for each of them and propose our Swarm intelligence-based metaheuristic, named Grey Wolf Optimizer (GWO) approach, for interference aware Virtual Machine Placement Problem (VMPP).

Keywords: Cloud Computing, Virtual Machine (VM), Physical Machine (PM), VM Placement (VMP), Server Consolidation, Resource Utilization, Performance Interference, SLA violation, Grey Wolf Optimizer (GWO), Swarm Intelligence metaheuristic

1. Introduction

Virtual machines placement (VMP) in cloud data centers, is strictly considered among the NP-hard [1] multidimensional [2] problems similar to bin packing problems [3], where objects with a given area must be packed into a finite number of bins such that the minimum number of bins is used. Moreover, VMP are multi-objective problems. Thereby, aiming to reduce power consumption and resource wastage (i.e. through virtualization and server consolidation) in data centers, results in the birth of new challenges that must be seriously considered in order to define efficient VMP solutions. In other words, we must be aware of the multidimensional aspect of the virtual machines placement problem (VMPP). Hence, while trying to optimize energy consumption by provisioning multiple VMs on the Same PM and put as much as possible idle PMs on a power saving mode, we should not forget about resource usage balancing within each PM and resource contention between co-hosted VMs. Which results in a severe VMs performance degradation called also performance interference, and consequently violates the Service Level Agreement (SLA). The main challenge then, is how to combine a set of goals which are sometimes contradictory: 1) minimize used resources but 2) maximize resource balance of the server hosting multiple VMs and 3) minimize power consumption by collocating VMs on the same PM but 4) minimize inter-VMs interference threshold. Performance interference is a performance degradation [4] due to lack of effective performance isolation among co-hosted VMs [5]. In that case, the performance of one VM can still be affected by the behavior of another adversely one, both sharing same

Email: hhamdi@ju.edu.sa, sabrine3amri@gmail.com, eibrahmi@taibahu.edu.sa

systems physical resources. Moreover, studies on Amazon EC22 have shown that, disk I/O bandwidth can vary by 50% [6], and network I/O bandwidth of medium instances can vary by 66% [7], due to the contention on shared resources. In some cases, VM interference caused by resource contention may even lead a virtual machine to stop responding [8], [9], [10], [11]. Therefore, VM interference problem should be well considered when allocating VMs across PMs. This is why we insist on the fact that, the key factor of successful VM placement is to minimize performance interference among co-located VMs [12]. Many studies [13], [14], [15], [16], [17], [18], [19], [20], [21], have addressed this problem, some have tried to propose different but sometimes similar solutions to detect and calculate what they called a performance interference problem, while others [22], [23], [24], [25], [26], [27], have tried to propose approaches VMP taking into account the interference between co-located VMs. However, most of them are proposing solutions that seem to be tighten to a specific level of VMP problem dimension and do not treat the problem from all its perspectives. In a previous work [28] we examined two closely related research axes (the detection we examined two closely related research axes (the detection and / or prediction of interferences between VMs, as well as the placement of interference-sensitive virtual machines in cloud computing environments), we proposed taxonomies and comparative studies of the current achievements in each of them. In this paper, a Swarm intelligence-based metaheuristic Grey Wolf Optimizer (GWO) approach for interference-aware Virtual Machine Placement Problem (VMPP).

The rest of the paper is organized as follows. In section 2, we review the revealing interference detection and/or prediction approaches without considering any virtual machines placement solutions. Section 3 presents revealing works using interference-aware models to define virtual machines placement approaches. In Section 4, we present the Grey Wolf Optimizer (GWO) Approach. Section 5 introduces our proposed approach: Interference-Aware VMP based on Grey Wolf Optimizer (GWO) Approach. In Section 6, we give and discuss the experimental study. In Section 7, we provide the conclusions.

2. MANAGING INTER-VM PERFORMANCE INTERFERENCE

Performance interference is the contention and fight between co-hosted VMs over the storage and computational resources (i.e. Disk I/O,CPU, Memory, Network I/O, etc.) provided by the hosting PM. It can be considered as the mismatch [6] between the resource supply, provided by the hosting PM, and the resource demand, of co-located VMs (i.e. whether the resource demand can be satisfied by the resource supply).

A. Proposed inter-VM interference taxonomy

Fig.1 presents our taxonomy of inter-VM interference detection techniques. Thereby, we meta-model the studied works detailed in next section based on: (1) the studied level metric (i.e. VM, PM or application level), and (2) the adopted approach to identify it (i.e. learning based or queuing based approach).

B. Comparative study

We propose a comparison between the existing approaches that outline the most the performance interference issue among VMs based on the following criteria.

1) Comparison criteria: We emphasize distinct criteria that we adopted during our comparative study between the existing approaches. Those criteria come out to sculpt a better understanding of the similarities and mismatches between the proposed literature approaches upon performance interference issue.

- **Framework name (F.N):** indicates the name that authors choose to attribute to their proposed framework.

- **Target user (T.U):** specifies the cloud actor to whom is dedicated each study, in a way that he can deploy the proposed solution related to interference detection. The target user who benefits of each suggested solution can be, either the cloud service provider while offering his services to tenants, or the cloud customer while leasing the VM instances, or even both.

- **Studied level metrics (L.M):** indicates which level authors have explored to capture the interference problem. Was it the VM level performance metric (such as network bandwidth, CPU utilization and memory consumption), the PM level metric (such as clock cycles per instruction) or the Application level metric (such as response time).

- **Used resources (U.R):** lists the mentioned physical resources that researchers used to experience and benchmark interference (i.e. CPU, memory, network I/O etc.).

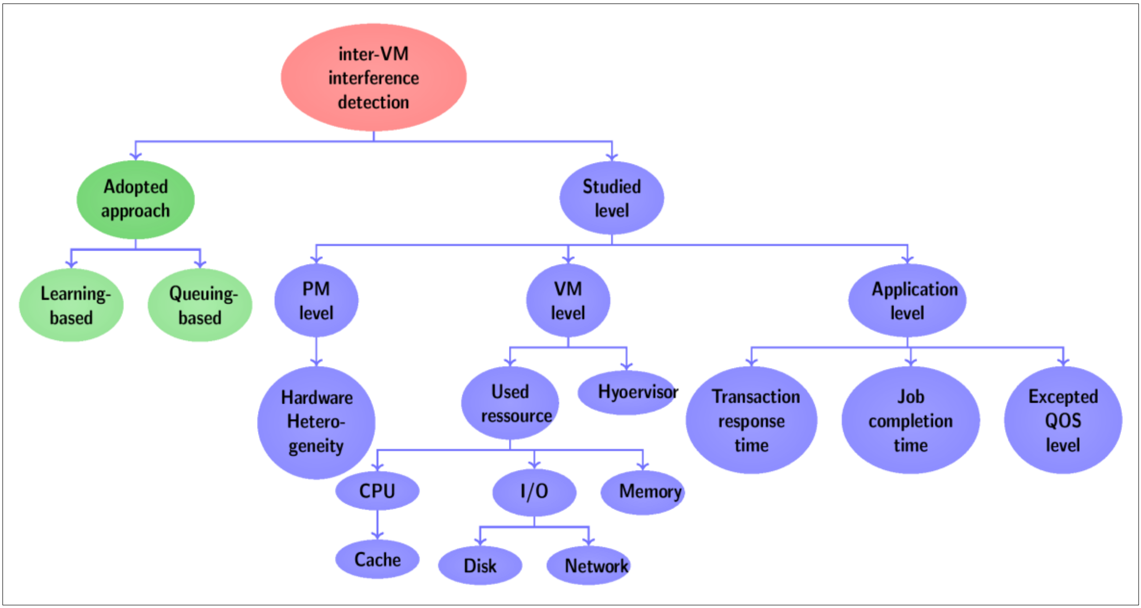

Figure 1. Inter-VM interference detection taxonomy

- **Interference identification manner (I.M):** refers the used manner to identify the interference problem. It can be either through predicting and anticipating it before it occurs (i.e. proactive), or detecting it simply based on historical information but without predicting future data changes (i.e. reactive).
- **Used approach (U.A):** used approach to identify interference can be either learning-based approach or queuing based approach.
- **Used techniques (U.T):** indicates the researchers adopted techniques to compute interference (e.g. Markov chain, least square method, Collaborative filtering).
- **Used metrics (U.M):** points the used parameters to express performance interference. Precisely, parameters to identify the performance degradation (e.g. QoS, response Time).
- **Target field (T.F):** this criterion emphasizes authors main interest while identifying the interference problem. Whether it was for proposing a solution for either VM placement (VM.P), resource provisioning (R.PV) or none of the previous ones and simply identifying the interference issue.

2) Emerging comparison: Table. I shows the comparative study for works, dealing with the performance interference problem, that we consider relevant and outline the reported literature. In what follows we build our comparative overview based on the set of criteria depicted in the previous section. The comparative study shows some similarities between the different approaches. Hence, even though authors are trying to explore performance interference issue from various perspectives using diverse metrics and techniques, yet they still adopting similar steps. [9] [9], [10], [12] study both VM and PM level metrics to detect the induced interference issue. Others, [8], [11] consider the application level performance only and fewer are those who explore the hardware level [6]. From another perspective, many approaches unify in the manner they predict interference. Thereby [6], [7], [11] and [12] use the proactive strategy while [8], [9], [10], [23] and [24] are adopting the reactive strategy.

3. MANAGING VIRTUAL MACHINES PLACEMENT

We reviewed research work dealing with virtual machine placement problem from the perspective of Energy-awareness regarding the inter-VM performance interference issue when placing VMs on PMs. As it is described in Figure 2, the interference aware VMP is one of the adopted aspects in researcher's approaches related

Table 1. COMPARATIVE STUDY OF PERFORMANCE INTERFERENCE APPROACHES

	F.N	T.U	L.M	U.R	I.M	U.A	U.T	U.M	TF	
									R.PV	VM.P
[6]	Heifer	C	PM, VM	CPU, Memory, Network, Disk I/O	P	L	Least square method	Resource utilization (demand vs supply), Task execution time	Y	N
[7]	CloudScope	S.P	VM	CPU, Disk, I/O, Network	P	L	Discrete-time Markov chain, Virtualization slowdown factor (V-slowdown)	Job completion time	Y	N
[8]	(CRE)	C	App		P	L	Collaborative filtering -Proxy concept. -Mean reference response time -Adjusted Weighted Sum – PCC (Pearson Correlation Coefficient)	Transaction response time	N	N
[9]		S.P, C	VM, App	CPU, Disk, I/O, Network	R	Q	Experiments on both virtualized and non-virtualized servers	Job completion time	Y	N
[10]	Q-Clouds	S.P, C	VM, App	CPU(cache)	P	L	-Static models -Closed loop controller –Discrete time multi-input multi-output(MIMO)-Page coloring to avoid interference -Head room	QoS levels (Q-states)	N	N
[11]		S.P	App	CPU, Network, Disk I/O	P	L	Application clustering, Weighted mean method, Principal component analysis (PCA), Linear regression analysis	Workload characteristics, Runtime characteristics	N	N
[12]	IC2 System	C	VM, App	CPU (cache), Memory bandwidth	P	L	Outlier detection method, Decision tree classifier	(MXC)MaxClients, (KAT)KeepaliveTimeout, pm.max child ren	N	N
[23]		S.P, C	VM	Network, CPU cache			Statistical regression techniques, Heuristic search algorithm	Request sizes, Response time	N	N
[24]		S.P	VM	Disk I/O	R	Q	Static Analysis technique	Mean queuelength, Response times, Number of disk read/write	Y	N

to the VMP research axes. In this study, we are focusing on surveying papers related to the mentioned VMP research axes.

1) Proposed VMP taxonomy

Based on studying the most relevant VMP existing research articles, we came up with classifying them according to two levels: (1) the studied aspect (i.e. Energy-aware or Security aware VMP) and (2) the used approach (game-theoretic, metaheuristic, FCA or Gossiping, based approaches) as shown in our proposed taxonomy in Figure2.

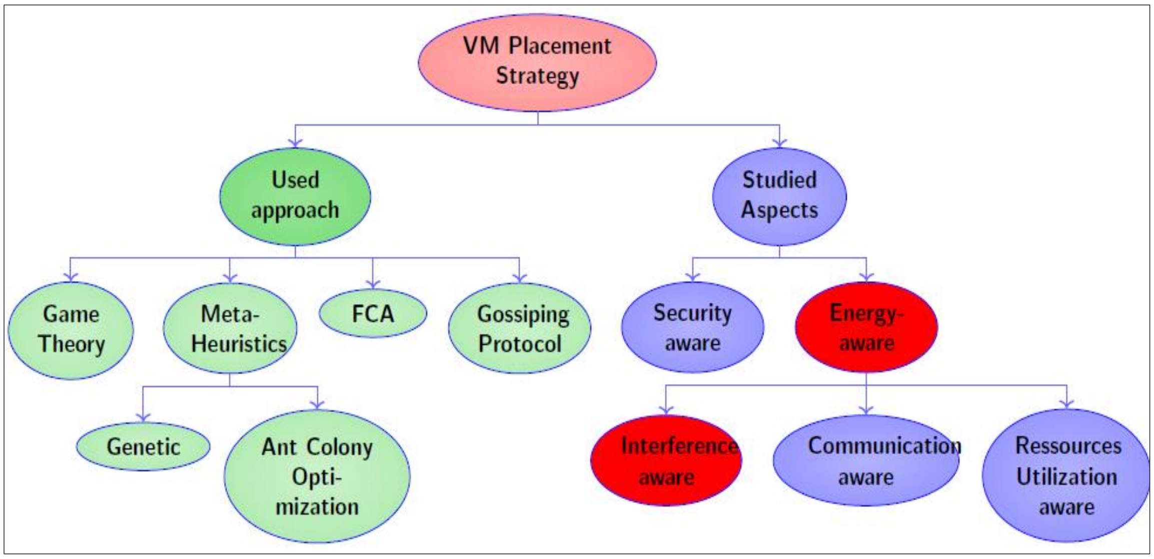

Figure 2.Taxonomy of the VMP works

B. Comparative study

We established a comparative study of the most relevant VMP approaches based on a set of criteria that we define below. This comparative study may clarify the similarities and the mismatches between the most relevant interference-aware virtual machine placement problem approaches.

1) Comparison criteria: Our comparative study is based on the following criteria:

- **Used resources:** VM defined profiles in each VMP approach. It consists on resources that present intensive utilization to experience interference (e.g. CPU, RAM, Disk I/O, etc.).
- **Interference metrics:** represents the used parameters to compute interference. Precisely, parameters to measure the performance degradation.
- **Placement metrics:** identifies the adopted parameters to verify by the placement module while mapping VMs to PMs.
- **Placement goal:** defines the main objective of server consolidation while placing the VMs (e.g. minimize active PMs Number, Minimize interference, etc.). Placement approach: specifies the authors adopted approach to resolve the VMP problem.

2) Emerging comparison: Table 2 shows the comparative study for works, dealing with the performance interference aware VMP problem, that we consider most relevant and outline the reported literature. In what follows we build our comparative overview based on the set of criteria depicted in the previous section. The comparative study shows that none of the surveyed approaches considers VMP problem from different levels of the cloud computing environments (i.e. VM level, PM level and Application level).

The previous review of the most relevant literature to inter-VM interference as well as virtual machine placement problem in cloud data centers, shows that the main challenges for managing the performance interference problem that need to be further investigated while placing VMs on PMs. In fact, we noted that the reported literature has neglected the following:

- Most of the studied approaches detect and/or predict interference through either, application or VM levels, however we rarely find someone who considers hardware level. Hence, it’s essential to skim all cloud levels (i.e.PM, VM, application, hypervisor, etc.) to overcome the interference issue, which is not trivial.

- Virtual machine instances are provided to tenants as on-demand computing resources. We assume then, that cloud-customer profiling must be seriously considered during the process of interference identification, since cloud-customer is the main trigger of VM and application behavior. So, we believe that its crucial to investigate interference changes according to users behavior. At this level, its critical to consider the real-time aspect.

Table 2. COMPARATIVE STUDY OF INTERFERENCE-AWARE VMP APPROACHES

	Used resources	Interference metrics	Placement metrics	Placement goal	Placement approach
[27]	CPU RAM Disk I/O	-CPU execution time percentage of increase -RAM and Disk I/O Throughput percentage decrease	Interference threshold	Minimize PM number	Heuristics (First Fit, Best Fit, Worst Fit)
[29]	CPU Memory Disk I/O	-slowdown(job1@job2):ratio (completion time of job1 when job2 is running on the Same PM To completion time of job1 when running alone on the PM).	Ratio of idle PMs -Ratio of SLA violation	Allocate interfering jobs on different cores of the same PM Minimize energy consumption in datacenters and, Reduce performance degradation between colocated jobs	Heuristics (Modified Best Fit)
[30]	CPU Memory	QoS in term of: -throughput -latency –Response time	Workload heterogeneity	Select the best workload host according to its internal interference level	Decision making techniques
[5]	CPU cache: (LLC) Memory bus	LLC misses and references	Interference intensity- Interference sensitivity	Minimize average performance degradation ratio of all applications	
[31]	CPU cache: (LLC)	Distance analysis	Cache interference intensity Cache pollution	Schedule VMs based on Identifying applications cache intensity	VM Scheduling based on Application classification
[32]	Network I/O	QoS violation: service time of network I/O request.	Resource demand of a VM Application QoS -VM interference	Reduce VM interference Fully exploit resource capacities of PMs Satisfy applications QoS requirements Maximize cloud providers profit	Integer Linear programming Polynomial time Heuristic
[33]	CPU Memory	Cycles Per Instruction (CPI): response time	lowest performance interference level	Online VM placement strategy	Heuristics- Machine Learning

- Cloud computing environments are very dynamic. A newly coming substantial unpredictable application can seek for computational resources any time, or even an existing running workload can instantly be resource intensive. Therefore, using historical data alone is not enough to predict interference. Thus, along with traces it's essential to consider prediction models that are flexible and dynamicity-aware. At this level, it's critical to consider the real-time aspect. Far from the traces, the real-time aspect is an attractive search area that attempts to consider the present front-end performance to anticipate the back-end conduct. To tackle this matter, we are chiefly motivated by a relatively young field: the data stream mining.

4. GREY WOLF OPTIMIZER (GWO) APPROACH

The Grey Wolf Optimizer (GWO) is a new meta-heuristic inspired by grey wolves (Canis lupus). The GWO algorithm mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. In order to design the models for both the hunting technique as well as the social hierarchy of grey wolves.

A. Mathematical models of grey wolves hunting process:

The grey wolves' social hierarchy is mathematically modeled as follows. The fittest solution is considered as the alpha (α). Consequently, the second and third best solutions are named beta (β) and delta (δ) respectively.

The rest of the candidate solutions are assumed to be omega (ω). In the GWO algorithm the hunting (optimization) is guided by α, β, and δ. The ω wolves follow these three wolves. Three main steps of hunting (i.e. searching for prey, encircling prey, and attacking prey), are implemented to perform optimization.

1) Encircling

Encircling prey is mathematically modeled by the following equations:

$$\vec{D} = \left|\vec{C} \times \vec{X}_p(t) - \vec{X}(t)\right| \quad (1)$$

$$\vec{X}(t+1) = X_p(t) - \vec{A} \times \vec{D} \quad (2)$$

Vectors $\vec{A}$ and $\vec{C}$ are calculated as follows:

$$\vec{A} = 2\vec{a}r_1 - \vec{a} \quad (3)$$

$$\vec{C} = 2\vec{r}_2 \quad (4)$$

Where:

$\vec{\boldsymbol{D}}$**:** victor distance between grey wolf and prey p.

t: current iteration.

$\vec{\boldsymbol{X}}$**p:** position vector of the prey (the best agent)

$\vec{\boldsymbol{X}}$**:** position vector of a grey wolf.

$\vec{\boldsymbol{A}}, \vec{\boldsymbol{C}}$ are coefficient vectors

$\vec{\boldsymbol{a}}$**:** The components of are linearly decreased from 2 to 0 over the course of iterations.

$\overrightarrow{\boldsymbol{r1}}, \overrightarrow{\boldsymbol{r2}}$ are random vectors in [0,1]

2) Hunting: The alpha leads the pack for hunting. Therefore, alpha is considered as the best solution. The beta and delta are considered as better solutions because they have better information about the probable location of the prey. Among the all obtained solutions, three best solutions are considered as alpha, beta, delta, and others have to update their positions according to the positions of best search agents. To mathematically simulate the hunting behavior of grey wolves, it is supposed that the alpha, beta and delta have better knowledge about the potential location of prey. Hence, first three best solutions obtained so far are saved and oblige the other search agents (including omegas) to update their positions according to the positions of best search agents. Following are the equations to implements the hunting mechanism:

$$\vec{D}_\alpha = \left|\overrightarrow{C_1} \times \vec{X}_\alpha - \vec{X}\right|, \vec{D}_\beta = \left|\overrightarrow{C_2} \times \vec{X}_\beta - \vec{X}\right|, \quad (5)$$
$$\vec{D}_\delta = \left|\overrightarrow{C_3} \times \vec{X}_\delta - \vec{X}\right|$$

$$\overrightarrow{X_1} = \vec{X}_\alpha - \vec{A}_1 \times (\vec{D}_\alpha), \overrightarrow{X_2} = \vec{X}_\beta - \vec{A}_2 \times (\vec{D}_\beta), \quad (6)$$
$$\overrightarrow{X_3} = \vec{X}_\delta - \vec{A}_3 \times (\vec{D}_\delta)$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (7)$$

$$\text{Where:}\begin{cases}\vec{\mathrm{D}}_\alpha : \text{distance between wolf}(\alpha)\text{and wolf }(\omega)\\ \vec{\mathrm{D}}_\beta : \text{distance between wolf}(\beta)\text{and wolf }(\omega)\\ \vec{\mathrm{D}}_\delta : \text{distance between wolf}(\delta)\text{and wolf }(\omega)\end{cases}$$

3) Attacking prey (exploitation): Grey wolves finally finish their hunt by attacking their prey after it stops moving. Approaching the prey, is mathematically modeled by decreasing the value of $\vec{a}$ and $\vec{A}$ from 2 to 0 over the course of iterations.

4) Search for prey (exploration): The search process starts with creating a random population of grey wolves (candidate solutions) in the GWO algorithm. Over the course of iterations, alpha, beta, and delta wolves estimate the probable position of the prey. Each candidate solution updates its distance from the prey. The parameter "a" is decreased from 2 to 0 in order to emphasize exploration and exploitation, respectively. Candidate solutions tend to diverge from the prey when $|\vec{A}|>1$, and converge towards the prey when $|\vec{A}|<1$. Finally, the GWO algorithm is terminated by the satisfaction of an end criterion.

B. GWO Algorithm

To see how GWO is theoretically able to solve optimization problems, some points may be noted:

- The proposed social hierarchy assists GWO to save the best solutions obtained so far over the course of iteration.

- The proposed encircling mechanism defines a circle-shaped neighborhood around the solutions which can be extended to higher dimensions as a hyper-sphere.
- The random parameters A and C assist candidate solutions to have hyper-spheres with different random radii.
- Exploration and exploitation are guaranteed by the adaptive values of a and A.
- The proposed hunting method allows candidate solutions to locate the probable position of the prey.
- The adaptive values of parameters a and A allow GWO to smoothly transition between exploration and exploitation. With decreasing A, half of the iterations are devoted to exploration |A|⩾1 and the other half are dedicated to exploitation |A|<1.

The GWO has only two main parameters to be adjusted (a and C). There are possibilities to integrate mutation and other evolutionary operators to mimic the whole life cycle of grey wolves.

```
Initialize the population of wolves $X_i, (i = 1,2, \dots, N_{max})$
Initialize parameters a, A, and c
Calculate the fitness of agent
$(X_\alpha) =$ the best search agent
$(X_\beta) = the\ second\ best\ search\ agent$
$(X_\delta) = the\ third\ best\ search\ agent$
While $(t < Max\ number\ of\ iterations)$
    for each search agent
        Update the position of the current search agent by the equation(7)
    end for
    Update a, A, and c
    calculate the fitness of all search agents
    Update $(X_\alpha)$, beta $(X_\beta)$ and $(X_\delta)$
End while
Return $X_\alpha$
```

5. INTERFERENCE-AWARE VMP BASED ON GREY WOLF OPTIMIZER (GWO) APPROACH

In this section we propose our interference-aware VMP based on a Swarm Intelligence metaheuristic, named as Grey Wolf Optimizer (GWO). First, we formalize our VMP problem based on our defined problem dimensions (i.e. resource use, resource balancing and performance interference) and define our emerging objective function. Then, we reveal our own interference-aware VMP approach inspired by GWO by defining our GWO-based interference-aware VMP algorithm (GWO-IVMP).

A. VMPP Formalization

Formally the VM Placement Problem VMPP can be modeled as the triple: *VMPP=<VM, PM, VPM>*. Where:

- $VM = \{V_1, \dots, V_n\}$: is the set of n virtual machines
- $PM = \{P_1, \dots, P_m\}$: is the set of m physical machines
- $\{V_i\} \times PM \rightarrow VPM$: where VPM is the Virtual machines Placement Matrix with n rows and m columns, that models VM-to-PM placement and is defined as follows.
- $VPM_{n,m} = \{x_{i,j} | i \in [1,n], and\ j \in [1,m]\}$
- $x_{i,j} = \begin{cases} 1, & if\ V_i\ is\ placed\ on\ P_j \\ 0, & otherwise \end{cases}$
- $x_{i,j} = 1$, means that the placement of *coupling* (V_i , P_j) = *true*

Each P_j $\in PM$, ($j \in [1,m]$), has:

- A set of hosted VMs given as: $V_{P_j} \in VM$, where $V_{P_j} = \{V_{ij}, i \in [1,n], and\ j \in [1,m]\}$, for $\forall\ x_{i,j} = 1$
- Resource Capacity Vector (RCV) given as a dimensional vector: $RCV_{P_j} = \langle C_{P_j}^{r_1}, C_{P_j}^{r_2}, \dots, C_{P_j}^{r_d} \rangle$, where: $C_{P_j}^{r_k}$ is the global capacity of resource r_k provided by PM number j, k is the index of resource r. Where, $k \in [1,d]$ and d is the number of different provided resources. $r_k \in R,\ where: R = \{\text{CPU}, \text{RAM}, \text{Disk I/O}\}$, in our model we consider 3-dimentional resources (CPU, RAM, Disk I/O).
- Resource Utilization Vector (RUV) in a P_j is given as a d-dimensional vector: $RUV_{P_j} = \langle U_{P_j}^1, U_{P_j}^2, \dots, U_{P_j}^{r_d} \rangle$, where:
    - $U_{P_j}^{r_k} = \sum_{i=1}^{v} D_{V_{ij}}^{r_k}$, for $\forall\ x_{i,j} = 1$, is the overall used quantity of resource r_k, by all hosted VMs, among the overall capacity of resource r_k provided by the PM number j.
    - v = $|V_{Pj}|$, is the number of VMs, $V_{ij} \in V_{Pj}$, hosted by $P_j \in$ PM .

- $D_{V_{ij}}^{r_k}$, is the quantity of resource demand r_k required by V_{ij}, as it is explained in the following virtual machines formalization.

- Resource Utilization Threshold Vector (RUTV): is a fixed d-dimensional Vector for each $P_j \in PM$ and must not be exceeded during the VM packing process (i.e.VM placement process) process. The RUTV vector is introduced in order to comply the SLA constraints.
  - $RUTV_{P_j} = \langle(\beta_{P_j}^{r^1}, \theta_{P_j}^{r^1}), (\beta_{P_j}^{r^2}, \theta_{P_j}^{r^2}), \ldots, (\beta_{P_j}^{r^d}, \theta_{P_j}^{r^d})\rangle$, where $\beta_{P_j}^{r^k}$, denotes the upper threshold that resource r_k, utilization percentage should not exceed. $\theta_{P_j}^{r^k}$, denotes the lower threshold that resourcer_k, utilization percentage should not be less than it.

Each V_i $\in VM$, i index of virtual machine, $i \in [1, n]$, has:

- Resource Demand Vector (RDV) given as a d-dimensional vector:
  - $RDV_{V_i} = \langle D_{V_i}^{r_1}, D_{V_i}^{r_2}, \ldots, D_{V_i}^{r_d}\rangle$, where: $D_{V_i}^{r_k}$ is the quantity of resource r_k required (demanded) by $V_i \in VM$.

1) Management of Performance Interference model:

Performance interference refers to the performance degradation percentage in a $P_j \in PM$, when placing on it a new $V_i \in VM$. Performance interference on each physical machine is given as:

$$\boldsymbol{interf}_{\mathrm{i,j}} = \frac{\sum_{k=1}^{d} \boldsymbol{interf}_{\mathrm{i,j}}^{r_k}}{\boldsymbol{d}} \quad (8)$$

The formula of (eq. 8), gives the average of interferences (i.e. caused by contention over all required resources ($r_k \in R$)) occurred while intending to place a given ($V_i \in VM$) into a given ($P_j \in PM$). We designed by $\boldsymbol{interf_{i,j}^{r_k}}$, the interference model obtained for each resource r_k using the interference models obtained through Machine Learning-based regression technique defined by [29]. According to [11], the following interference models was obtained after executing specific benchmarks that stress each computational resource separately, in order to identify how much interference percentage each type of shared resource produces:

- Interference model of CPU resource is given as:

$$interf_{i,j}^{cpu} : y = 6{,}5346\ln(x) - 4{,}4983 \quad (9)$$

- Interference model of RAM resource is given as:

$$interf_{i,j}^{RAM} : y = 34{,}398\ln(x) - 4{,}7183 \quad (10)$$

- Interference model of Disk I/O resource is given as:

$$interf_{i,j}^{DiskI/O} : y = 35{,}347\ln(x) - 7{,}2785 \quad (11)$$

Where, y is the percentage of interference, and x in the number of VMs $\in V_{P_j}$ seeking for resource r_k.

2) Resource management models: Other than performance interference minimization heuristic, our VM packing process must not overload the PM and violate the SLA (Service Level Agreement) constraint. In our proposed solution we adopted the two following heuristics as well:

- Each VM should be placed on the PM that has the minimum unused resources without exceeding resources use threshold. The purpose of our heuristic is to maximize physical resources exploitation.
- Each VM should be placed on to the PM that decreases further the imbalance rate of resources use. This is feasible by using magnitude imbalance vector

a) Management of Resource utilization (RU) model: In order to control the resource usage in each PM, we will compute the amount of resource supply in each PM through defining two vectors (i.e. Resource Supply Vector (RSV) and New Resource Supply Vector (NRSV)), that store respectively the available quantity of resources before and after the virtual machine, $V_i \in VM$, placement on the physical machine $P_j \in PM$.

- Resource Supply vector (RSV), defines the available quantity of resource offered by $P_j \in PM$ respecting its Resource Utilization Threshold Vector (RUTV), and is calculated as follows: $RSV_{P_j} = \langle S_{P_j}^{r^1}, S_{P_j}^{r^2}, \ldots, S_{P_j}^{r^d}\rangle$, where: $S_{P_j}^{r^k}$ denotes the available quantity of resource r^k, respecting its threshold $\beta_{P_j}^{r^k}$, and is given by $S_{P_j}^{r^k} = \beta_{P_j}^{r^k} \times C_{P_j}^{r^k} - U_{P_j}^{r^k}$
- New Resource Supply vector (NRSV), which denotes, the new remaining quantity of resources that a P_j will have if V_i is placed on it, and is calculated as

follows: $NRSV_{P_j} = \langle NS_{i,j}^{r^1}, NS_{i,j}^{r^2}, \dots, NS_{i,j}^{r^d} \rangle$, such that:

$$NS_{i,j}^{r^k} = S_{P_j}^{r^k} - D_{V_i}^{r_k} \qquad (12)$$

b) Management of Resource Balancing (RB) model: In fact, capturing the measure of overall resources utilization across multiple resource types is one of the most important factors, saturation of only one resource type can lead to no further improvement in utilization while leaving other types of resources underutilized [28], [29], [30], [31].

- Resource Imbalance Vector (RIV), is a d-dimensional vector, it computes the degree of imbalance in the current utilization of a given P_j, when placing a new incoming V_i , on it. And it's defined as follows: $RIV_{i,j} = \langle (NS_{i,j}^{r^1} + M), (NS_{i,j}^{r^2} + M), \dots, (NS_{i,j}^{r^d} + M) \rangle$ where $= \sum_{k=1}^{d} \frac{D_{v_i}^{r_k}}{d}$, in our case, $d = 3$ (3 types of resources).

In order to quantify the RIV vector, we will calculate its length which is in fact vectors magnitude. The magnitude of a vector is computed based on the Pythagorean Theorem and it defines the difference between the d-dimensions of supply vector and the average value. The magnitude is always a positive value, since it's a length value.

- The magnitude of RIV is given as follows:

$$mag(RIV_{i,j}) = \sqrt{(NS_{i,j}^{r^1} + M)^2 + \cdots + (NS_{i,j}^{r^d} + M)^2} \qquad (13)$$

When selecting a P_j to host a V_i, the couple (V_i, P_j) that has the smaller magnitude of RIV is the one that mostly balances the resources utilization of the server across different dimensions.

B. *Objective function*

In this section, we give our objective function relative to the Virtual Machines Placement Problem (VMPP) in cloud data centers. Initially, when we take a global look at our goals while placing VMs on PMs, it seems to be a multi-objective problem since we have multiple goals to achieve to resolve the VMPP:

- **First goal**: Optimize resource utilization, (i.e. maximize resource use, and consequently minimize the quantity of unused resources on a given PM), which is modeled by (***NS**$_{i;j}$*).
- **Second goal**: Optimize resource balancing, (i.e. all resources of a given PM are preferred to be equally used, and none of them should be stressed more than the others), which is modeled by (***mag(RIV**$_{i;j}$***)).
- **Third goal**: Minimize inter-VM performance interference, (i.e. minimize performance degradation. In that case, the performance of one VM should not be affected by the behavior of another adversely one, both sharing same physical resources), which is modeled by (***interf**$_{i;j}$*).

Nevertheless, when we closely examine all the mentioned goals, we clearly notice that all of them attempt to minimize something (i.e. quantity of unused resources, resource imbalance and performance interference). Therefore, the virtual machines placement problem (VMPP) for each VMP couple (V_i; P_j) ∈ VMPS, can be modeled by a Mono-objective function $F_{i;j}$ which is the sum of all our obtained goals models, when placing V_i on P_j , such that:

$$F_{i,j} = \left(NS_{i,j} + mag(RIV_{i,j}) + interf_{i,j}\right) \qquad (14)$$

Thereafter, in order to reach a better solution for the VMPP, our goal is to minimize the global objective function F. F is the objective function of a given virtual machines placement solution (VMPS) and is defined as:

$$F = \sum_{1}^{d} F_{i,j} , (\forall\, (V_i , P_j) | x_{i,j} = 1) \qquad (13)$$

Subject to the constraints:

- $\forall i \; \sum_{j=1}^{m} x_{i,j} \leq 1$, this constraint ensures that each V_i is allocated to at most only one P_j , though one host P_j can host multiple VMs, $V_{P_j} \in VM$.
- $\theta_{P_j}^{r^k} \times C_{P_j}^{r_k} \leq U_{P_j}^{r_k} \leq \beta_{P_j}^{k} \times C_{P_j}^{r_k}; k \in [1, d]$, This constraint ensures that the overall PM utilization must be lower than the fixed threshold. (i.e. must be lower than the fixed upper threshold and upper than the lower threshold of resource r^k utilization percentage).
- $U_{P_j}^{r_k} \geq \theta_{P_j}^{k} \times C_{P_j}^{r_k}; k \in [1, d]$, This constraint ensures that the overall PM utilization must be higher than the fixed lower threshold.
- $D_{V_{i,j}}^{r_k} < C_{P_j}^{r_k}$, or also: $U_{P_j}^{r_k} \leq C_{P_j}^{r_k}$, this constraint ensures that the load on each host machine is not greater than its capacity.
- $\theta_{P_j}^{r^k} \times C_{P_j}^{r_k} < NS_{i,j}^{r^k}$, the new supply of the required resources, should not be negative, or

else the coupling (V_i, P_j) violates the SLA constraint.

C. GWO-based interference-aware VMP (GWO-IVMP)

1) Mapping between our VMP Problem formulation and GWO algorithm: In order to apply GWO algorithm in VMP problem we present in table V-C1 a mapping between GWO and our problem formulation parameters.

2) GWO-IVMP: GWO-based interference-aware VMP algorithm:

In order to apply the GWO in our VMP algorithm, we will adjust the GWOs steps using our problems parameters respecting the mapping in Table V-C1. The steps description is as follows.

- **Step1:** Initial population:
  In this Step we are going to initialize the set of incoming VMs, PM, seeking to be hosted and the set of the candidate hosting PMs, PM. Then, our grey wolves initial population is going to be the set of randomly obtained VMPS (i.e. VMs to-PMs placement solutions obtained by randomly assigning each incoming $V_i \in VM$, to a random $P_j \in PM$ to be its host). Any needed parameters can be also initialized in this step (e.g. a, A, and C vectors).
- **Step2**: Fitness of wolves:

At this level, we will be computing the fitness of each wolf according to eq.15 from the generated initial population.

- **Step 3:** Find alpha, beta, and delta:

Sort the wolves in a decreasing dominance level based on their fitness values. Alpha (α), beta (β), and delta (δ) will be the three best suitable arrangements of wolves that minimize the most their objective functions. The remaining applicant arrangements are regarded to be the omega (ω). Let F_α be the first-best fitness solution, F_β the second-best fitness solution and F_δ the third-best fitness solution.

- **Step 4:** Update position of wolves:
  Update position of omega wolves based on Eq.17.
- **Step 5:** Update alpha, beta, and delta:
  After performing movements of all group members, the (new) alpha, beta, and delta must be found. Repeat steps 45 until the stopping condition is met after maximum number of iterations It_{max}.

In what follows we give our interference aware GWO-based VMP algorithm (GWO-IVMP).

Table 1. MAPPING BETWEEN GWO AND OUR PROBLEM FORMULATION

GWO Algorithm	**Problem Formulation**
A Grey wolf	A solution which is a set of initial (V_i,P_j) placement: VMPS
Fitness function	The objective function F given by Eq.15.
α, β, δ and ω	The three best suitable arrangements of wolves that minimize the most their objective function F given by Eq.15. i.e. best three VMPS having best three fitness values.
X_α: position of best search agent	$F_{i;j}{}^{\alpha}$: first best value of fitness solution
X_β: position of best search agent	$F_{i;j}{}^{\beta}$: second best value of fitness solution
X_δ: position of best search agent	$F_{i;j}{}^{\delta}$: third best value of fitness solution
Encircling prey	
$\vec{D} = \lvert \vec{C} \times \vec{X}_p(t) - \vec{X}(t) \rvert$	$\vec{D} = \lvert \vec{C} \times \mathrm{F}(t+1) - \mathrm{F}(t) \rvert$ (16)
$\vec{X}(\mathrm{t}+1) = \mathrm{Xp(t)} - \vec{A} \times \vec{D}$	$\mathrm{F(t+1) = Fp(t)} - \vec{A} \times \vec{D}$ (17)
$\vec{A} = 2\vec{a}r_1 - \vec{a}, \vec{C} = 2r_2$	$\vec{A} = 2\vec{a}r_1 - \vec{a}, \vec{C} = 2r_2$
Hunting	
$\overrightarrow{D_\alpha} = \lvert \overrightarrow{C_1} \times \vec{X}_\alpha(t) - \vec{X} \rvert$	$\overrightarrow{D_\alpha} = \lvert \overrightarrow{C_1} \times \mathrm{F}^{\alpha}_{i,j} - \mathrm{F} \rvert$
$\overrightarrow{D_\beta} = \lvert \overrightarrow{C_2} \times \vec{X}_\beta(t) - \vec{X} \rvert$	$\overrightarrow{D_\beta} = \lvert \overrightarrow{C_2} \times \mathrm{F}^{\beta}_{i,j} - \mathrm{F} \rvert$
$\overrightarrow{D_\delta} = \lvert \overrightarrow{C_3} \times \vec{X}_\delta(t) - \vec{X} \rvert$	$\overrightarrow{D_\delta} = \lvert \overrightarrow{C_3} \times \mathrm{F}^{\delta}_{i,j} - \mathrm{F} \rvert$
$\overrightarrow{X_1} = \vec{X}_\alpha - \overrightarrow{A_1} \times (\overrightarrow{D_\alpha})$	$\overrightarrow{X_1} = \mathrm{F}^{\alpha}_{i,j} - \overrightarrow{A_1} \times (\overrightarrow{D_\alpha})$
$\overrightarrow{X_2} = \vec{X}_\beta - \overrightarrow{A_2} \times (\overrightarrow{D_\beta})$	$\overrightarrow{X_2} = \mathrm{F}^{\beta}_{i,j} - \overrightarrow{A_2} \times (\overrightarrow{D_\beta})$
$\overrightarrow{X_3} = \vec{X}_\delta - \overrightarrow{A_3} \times (\overrightarrow{D_\delta})$	$\overrightarrow{X_3} = \mathrm{F}^{\delta}_{i,j} - \overrightarrow{A_3} \times (\overrightarrow{D_\delta})$
$\overrightarrow{X}(t+1) = \frac{\overrightarrow{x_1} + \overrightarrow{x_2} + \overrightarrow{x_3}}{3}$	$F(t+1) = \frac{\overrightarrow{x_1} + \overrightarrow{x_2} + \overrightarrow{x_3}}{3}$

```
Input: VM =$V_1,V_2, ...., V_n$, PM =$P_1,P_2, ...., P_m$
Output: VMPS = ϕ /* A VM Placement Solution */
begin
        S = initialize Set of Solutions(VM, PM);
        initialize Parameters(a, A, C);
        /* compute the fitness of initial population*/
        for each ($S_i \in S$) do
        $S_i$.Ftitness = Fitness($S_i$);
        end
        (α, β, δ) = Find Alpha Beta Delta(S);
        /* Update position of each wolf based on Eq.17
                */
        while (t < $It_{max}$ ) do
            for each (search agent $S_i \in S$ {α, β, δ}) do
                Update Position($S_i$);
            end
            Update(a, A, C);
            (α, β, δ) = **Find Alpha Beta Delta**(S*);
            t ← t +1 ;
        end
        return VPMS;
end
```

Algorithm 2: GWO-based Interference-aware VMP Algorithm (GWO-IVMP)

6. EXPERIMENTAL STUDY

This section emphasizes the description of test beds scenarios used to evaluate the performance of our GWO-based interference-aware VMP approach (GWO-IVMP), and results given by another random VMP approach that we will define as well.

A. Used test beds

Since the virtual machines placement problems size depends heavily on the number of both virtual machines and physical machines, we can observe the variations of the completion time when the number of virtual machines increases while the number of physical machines remains fixed and also when we change the number of physical machines and keep the number of virtual machines fixed. For that, we have prepared two groups of scenarios of test beds (as presented in Table IV):

- In a first test scenario, the number of physical machines was fixed as 1000, while the number of virtual machines varies with a step of 500.
- In a second scenario, the number of virtual machines was fixed as 3500, while the number of physical machines varies with a step of 200.

The two groups of randomly generated scenarios were also used to evaluate the performance and efficiency of our solution.

Table 2. SCENARIOS OF TEST BEDS

Scenario 1		
Test (ID)	**Number of VMs**	**Number of PMs**
1	1500	1000
2	2000	1000
3	2500	1000
4	3000	1000
5	3500	1000
6	4000	1000
7	4500	1000
8	5000	1000
9	5500	1000
10	6000	1000

Scenario 2		
Test (ID)	**Number of VMs**	**Number of PMs**
11	3500	800
12	3500	1000
13	3500	1200
14	3500	1400
15	3500	1600
16	3500	1800
17	3500	2000
18	3500	2200
19	3500	2400
20	3500	2600

B. Comparison basics

We used the comparison of our GWO-IVMP solution, based on Swarm intelligence, with one other solution. For the rest of the job, all the test problems are applied to two policies:

- Our proposed GWO-based interference-aware VMP policy.
- A random VMP policy.

These two policies are developed at the CloudSim simulator level that provides a simulated cloud environment. Before we begin implementing the proposed policies for allocating virtual machines, it is essential to know where the change can be made, since CloudSim offers several classes that support the simulation of the cloud environment. Thus, in order to implement a new virtual machine placement policy, it is essential to have knowledge of the existing policies and classes that support these allocation strategies. The “VMMAllocation-Policy” class is the component of CloudSim where the virtual machine allocation policies are performed. So, at this class level, we are developing our GWO-IVMP solution based on Swarm intelligence and another random VMP solution.

1) Implementation of the random VMP policy: At the ”VMMAllocationPolicy” class of the CloudSim package, we have developed a second virtual machine placement policy that is based on a random choice of a physical machine to host the virtual machine. For each virtual machine in the simulated environment, the system chooses a random physical machine, if it meets the requirements of the virtual machine then the allocation will take place, otherwise the system chooses another physical machine randomly, and so on.

2) Implementation of our GWO-based VMP policy: Still at the ”VMMAllocationPolicy” class, we have developed our decentralized virtual machine placement solution based on Swarm intelligence, where each virtual machine in the system individually decides which physical machine to migrate based on its obtained fitness value. The size of initial population was set to 20 and the maximum number of iterations is fixed to 50. We’ve chosen these values based on observing the ratios of best results given by our solution.

C. Results and discussion

In this section, we will identify the factors influencing the performance of all virtual machine placement solutions. Then, based on these factors we will present the simulation results by applying the different scenarios on both virtual machine allocation policies previously mentioned (i.e. our GWO-IVMP solution and the random VMP solution).

D. Evaluation metrics

In order to analyze the quality of proposed solutions, we must evaluate the performance, efficiency and scalability of each of them:

- The performance evaluation is performed by comparing the quality of the results generated by different solutions for a set of randomly generated test beds.
- The evaluation of efficiency is performed by comparing the computation time (completion time) of various solutions for a series of test problems of different sizes and complexities.
- Scalability is tested by studying how its compute time (completion time) increases as the size of the test problem increases.

However, in cloud data centers, a virtual machine placement solution is reported effective and efficient if it respects the two major constraints, namely:

- Minimal energy consumption.
- Maximized and balanced resource use in all dimensions.

E. Results

The simulation is conducted through the 20 previously defined test beds in Table IV, and each simulation has been repeated several times, the results being generated using the average. The results are calculated according to the evaluation metrics mentioned above: the number of active physical machines, the efficiency of the reached packing efficiency PE, the quantity of unused resources UNR and the completion time.

1) Number of active PMs: Figure 3, show curves of obtained results after testing the two previously defined VMP policies (i.e. our GWO-IVMP solution and the random VMP solution) respecting both previously fixed scenarios in Table IV, in term of the evaluation criterion, number of active physical machines. Results of obtained number of active PMs in term of varying number of VMs while fixing number of PMs, as it is defined in first scenario, are shown in Figure 3. It shows also the results of second scenario, in which number of active PMs is captured while varying PMs number and fixing VMs' number. In Figure 3, we have varied the number of of virtual machines between 1500 and 6000, with a step of 500. Our GWO-IVMP solution used only about 200 to 600 physical machines to place all virtual machines. In the same Figure, we also varied the number of PMs while fixing the number of VMs to 3500, our GWO-IVMP solution used almost the same number of physical machines that was less than 500. However, for both scenarios the random VMP solution was almost using all physical machines in order to allocate the given virtual machines, regardless the number of VMs (i.e. whether it was fixed or varied).

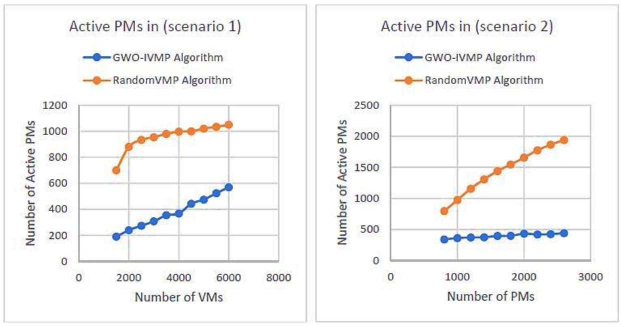

Figure 3.Number on active PMs in term of VMs (scenario1 and scenario2)

2) Packing efficiency PE: This section evaluates the effectiveness of the packing within each solution when simulating twenty test beds. Packaging efficiency (PE), is a critical factor in assessing the quality of any virtual machine placement solution. The higher the PE value, the better the solution is. For the twenty different test beds, our GWO-based interference aware virtual machines placement algorithm has reached the highest packaging efficiency value. From graphs in Figure 4, we can deduce that as the number of virtual machines in the system increases, the efficiency increases. However, when the number of virtual machines is invariant and the number of physical machines increases, packaging efficiency decreases. But anyway, our solution has guaranteed an efficiency value greater than 8.

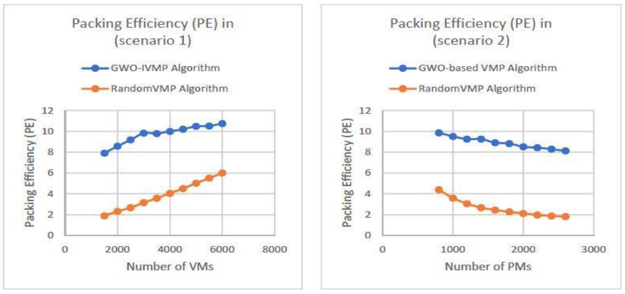

Figure 4. Packing Efficiency (PE) in term of VMs (scenario 1 and scenario 2)

3) Quantity of unused resources UNR: During the simulation of the twenty test beds, we took the amount of unused resources produced by each allocation solution. The results are summarized in Figure 5. The amount of unused resources is represented as a percentage of the total capacity of the active physical machines. For the twenty different simulations, our GWO-IVMP policy was wasting less than 10% of available resources, while the random solution had high resource wastage up to 85%.

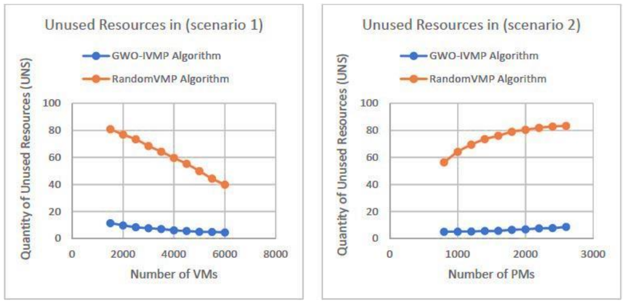

Figure 5. Quantity of UNused Resources (UNS) in term of VMs (scenario 1 and scenario 2)

3) Completion time: In order to verify the scalability of our Swarm intelligence-based algorithm, we have plotted a first completion time curve in Figure 6 for the first scenario where the number of virtual machines varies between 1500 and 6000 with a step of 500 while the number of physical machines is set to 1000. We have also plotted a second completion time curve, for the second scenario where the number of physical machines increases from 800 to 2600 with a step of 200 while the number of virtual machines has been set at 3500. As shown in Figure 6, the execution time of our GWO-IVMP solution increases too slowly as the size of the problem increases; i.e. either by increasing the number of virtual machines or the number of physical machines.

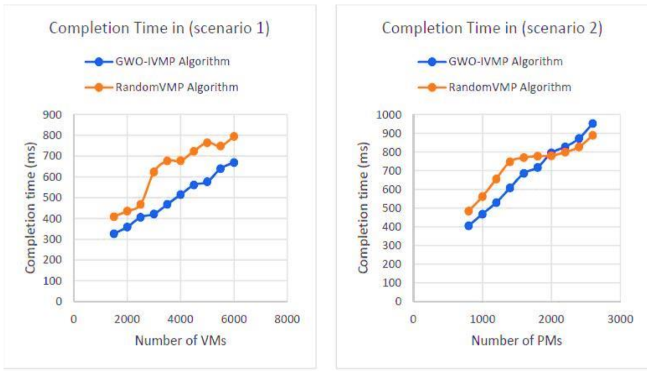

Figure 6. Completion time in term of VMs (scenario 1 and scenario 2)

For large scale problems, the execution time of our algorithm has not exceeded 1000(ms). Same thing for the execution time of the random VMP algorithm, except that the random solution is based on a simple assignment of the first available physical machine without taking into consideration either the maximization or the balance of the use of available resources.

F. Discussion

Results of the experiments clearly show that our solution based on Swarm intelligence surpasses the other proposed random VMP solution on all performance measures, namely: the number of active physical machines, the efficiency of packing, the amount of unused resources and the completion time. Our virtual machine placement algorithm was able to guarantee a better reduction in power consumption while ensuring a low number of active physical machines (servers) for the twenty different test beds. The highest packing efficiency values were produced by our GWO-IVMP solution. Regarding the amount of unused resources in physical machines, our algorithm shows the lowest percentages. For the twenty different simulations, the amount of unused resources in our GWO-IVMP algorithm was less than 10%, while the unused resource quantities of the random solution were between 40% and 90%. We can conclude that our virtual machine placement solution has strongly respected the constraint of maximizing the exploitation of resources. From the execution time graphs in Figure 6, we have been able to deduce that the completion time of our GWOIVMP solution has increased almost linearly as the size of the problem increases. The completion time of our solution has had the same rate whether while varying the number of virtual machines or the number of physical machines.

Therefore, we can conclude that our solution based on Swarm intelligence is highly scalable. In other words, the evaluation of the results of the 20 test beds applied to both proposed VMP policies (i.e. our GWO-IVMP algorithm and the random VMP algorithm), showed that our GWO-IVMP algorithm generates better results compared to the other random VMP algorithm.

7. CONCLUSION

In this work we proposed a Swarm intelligence metaheuristic-based Grey Wolf Optimizer (GWO) approach for interference-aware virtual machine placement in cloud datacenters. Our approach mainly focuses on energy optimization and efficiency of resource utilization while placing VMs on PMs in data centers. Experimental results have shown that our virtual machine placement algorithm consistently outperforms the random VMP algorithm in terms of performance criteria. It ensured a minimum number of active physical machines and a minimum amount of unused resources, high packing efficiency and reliable computing time. The results also showed that our Swarm intelligence based solution is highly scalable by its shorter completion time and slowly increasing with the complexity of the problem. The evaluations carried out validate the robustness and the efficiency of the algorithm object of this work. As a future work we will be concerned by two main levels for enhancing our interference-aware VMP solution: Front-end and back-end. The former exhibits the set of users that use cloud services and the latter is materialized by the set of servers where VMs are hosted. We also point out three main challenges in interference detection and prediction.

REFERENCES

[1] Chekuri, C., & Khanna, S. (2004). On multidimensional packing problems. SIAM journal on computing, 33(4), 837-851.

[2] Wood, T., Shenoy, P. J., Venkataramani, A., & Yousif, M. S. (2007, April). Black-box and Gray-box Strategies for Virtual Machine Migration. In proc. Of NSDI (Vol. 7, pp.17-17).

[3] Mishra, M., & Sahoo, A. (2011, July). On theory of vm placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach. In proc. Of IEEE International Conference on Cloud Computing (CLOUD), (pp. 275-282). IEEE.

[4] Tickoo, O., Iyer, R., Illikkal, R., & Newell, D. (2010). Modeling virtual machine performance: challenges and approaches. ACM SIGMETRICS Performance Evaluation Review, 37(3), 55-60.

[5] Koh, Y., Knauerhase, R., Brett, P., Bowman, M., Wen, Z., & Pu, C. (2007, April). An analysis of performance interference effects in virtual environments. In proc of. IEEE International Symposium on Performance Analysis of Systems & Software, 2007 (pp. 200-209). IEEE.

[6] Barker, S. K., & Shenoy, P. (2010, February). Empirical evaluation of latencysensitive application performance in the cloud. In Proceedings of the first annual ACM SIGMM conference on Multimedia systems (pp. 35-46). ACM.

[7] Shieh, A., Kandula, S., Greenberg, A. G., Kim, C., & Saha, B. (2011, March). Sharing the Data Center Network. In NSDI (Vol. 11, pp. 23-23).

[8] Matthews, J. N., Hu, W., Hapuarachchi, M., Deshane, T., Dimatos, D., Hamilton, G., & Owens, J. (2007, June). Quantifying the performance isolation properties of virtualization systems. In Proceedings of the 2007 workshop on Experimental computer science (p. 6). ACM.

[9] Padala, P., Zhu, X., Wang, Z., Singhal, S., & Shin, K. G. (2007). Performance evaluation of virtualization technologies for server consolidation. HP Labs Tec. Report, 137.

[10] Xavier, M. G., Neves, M. V., Rossi, F. D., Ferreto, T. C., Lange, T., & De Rose, C. A. (2013, February). Performance evaluation of containerbased virtualization for high performance computing environments. Proc. of 21st Euromicro International Conference Parallel, Distributed and Network-Based Processing (PDP), 2013 (pp. 233-240). IEEE.

[11] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw are2014; 69:4661.

[12] [12] Xu, F., Liu, F., & Jin, H. (2016). Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud. IEEE Transactions on Computers, 65(8), 2470-2483.

[13] Chen, X., Rupprecht, L., Osman, R., Pietzuch, P., Franciosi, F., & Knottenbelt, W. (2015, October). Cloudscope: Diagnosing and managing performance interference in multi-tenant clouds. Proc. Of the IEEE 23rd International Symposium on Modeling,Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS),(pp. 164-173). IEEE.

[14] Amannejad, Y., Krishnamurthy, D., & Far, B. (2015, May). Detecting performance interference in cloud-based web services. In proc. Of IFIP/IEEE International Symposium on Integrated Network Management (IM), 2015 (pp. 423-431). IEEE.

[15] Yuan, Y., Wang, H., Wang, D., & Liu, J. (2013, June). On interferenceaware provisioning for cloud-based big data processing. In Proc. Of IEEE/ACM 21st International Symposium on Quality of Service (IWQoS), 2013 (pp. 1-6). IEEE.

[16] Nathuji, R., Kansal, A., & Ghaffarkhah, A. (2010, April). Q-clouds: managing performance interference effects for qos-aware clouds. In Pro. Of the 5th European conference on Computer systems (pp. 237-250). ACM.

[17] Koh, Y., Knauerhase, R., Brett, P., Bowman, M., Wen, Z., & Pu, C. (2007, April). An analysis of performance interference effects in virtual environments. In proc of. IEEE International Symposium on Performance Analysis of Systems & Software, 2007 (pp. 200-209). IEEE.

[18] Maji, A. K., Mitra, S., Zhou, B., Bagchi, S., & Verma, A. (2014, December). Mitigating interference in cloud services by middleware reconfiguration. In Proceedings of the 15th International Middleware Conference (pp. 277-288). ACM.

[19] Noorshams, Q., Bruhn, D., Kounev, S., & Reussner, R. (2013, April). Predictive performance modeling of virtualized storage systems using optimized statistical regression techniques. In Proc. of the 4th ACM/SPEC International Conference on Performance Engineering (pp. 283-294). ACM.

[20] Casale, G., Kraft, S., & Krishnamurthy, D. (2011, June). A model of storage I/O performance interference in virtualized systems. In Proc. of 31st International Conference Distributed Computing Systems Workshops (ICDCSW), 2011 (pp. 34-39).IEEE.

[21] [21] Kim, S. G., Eom, H., & Yeom, H. Y. (2013). Virtual machine consolidation based on interference modeling. Journal of Supercomputing, 66(3), 1489-1506.

[22] Jersak, L. C., & Ferreto, T. (2016, April). Performance-aware server consolidation with adjustable interference levels. In Proc. of the 31st Annual ACM symposium on applied computing (pp. 420-425). ACM.

[23] Shim, Y. C. (2015). Inter-VM Performance Interference Aware Static VM Consolidation Algorithms for Cloud-Based Data Centers. In proc. Of The International Conference on Communications and Computers, pp-18.

[24] Moreno, I. S., Yang, R., Xu, J., & Wo, T. (2013, March). Improved energy-efficiency in cloud datacenters with interference-aware virtual machine placement. In Autonomous Decentralized Systems (ISADS), 2013 IEEE Eleventh International Symposium on (pp. 1-8). IEEE.

[25] Jin, H., Qin, H., Wu, S., & Guo, X. (2015). CCAP: a cache contentionaware virtual machine placement approach for HPC cloud. International Journal of Parallel Programming, 43(3), 403-420.

[26] Lin, J. W., & Chen, C. H. (2012, June). Interference-aware virtual machine placement in cloud computing systems. International Conference In Computer & Information Science (ICCIS), 2012 (Vol. 2, pp. 598-603). IEEE.

[27] Caglar, F., Shekhar, S., & Gokhale, A. (2011). Towards a performance interference aware virtual machine placement strategy for supporting soft real-time applications in the cloud, In proc of 3rd IEEE International Workshop on Real-time and distributed computing in emerging applications.

[28] Sabrine Amri, Hamdi H´edi and Zaki Brahmi, Inter-VM Interference in Cloud Environments : A Survey, In Proceedings of 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications.IEEE Computer Society, page 154159 - 2017.

[29] Shim, Y. C. (2015). Inter-VM Performance Interference Aware Static VM Consolidation Algorithms for Cloud-Based Data Centers. In proc. Of The International Conference on Communications and Computers, pp-18.

[30] Moreno, I. S., Yang, R., Xu, J., & Wo, T. (2013, March). Improved energy-efficiency in cloud datacenters with interference-aware virtual machine placement. In Autonomous Decentralized Systems (ISADS), 2013 IEEE Eleventh International Symposium on (pp. 1-8). IEEE

[31] Jersak, L. C., & Ferreto, T. (2016, April). Performance-aware server consolidation with adjustable interference levels. In Proc. of the 31st Annual ACM symposium on applied computing (pp. 420-425). ACM.

[32] Lin, J. W., & Chen, C. H. (2012, June). Interference-aware virtual machine placement in cloud computing systems. In Computer & Information Science (ICCIS), 2012 International Conference on (Vol. 2, pp. 598-603). IEEE.

[33] Caglar, F., Shekhar, S., & Gokhale, A. (2011). Towards a performance interferenceaware virtual machine placement strategy for supporting soft real-time applications in the cloud, In proc of 3rd IEEE International Workshop on Real-time and distributed computing in emerging applications.

Dr. Hedi HAMDI received his PhD in Computer sciences from The University of Bordeaux France in December 2009. Currently, he is an Assistant Professor at Computer Science Department, College Of Computer and information Science, Jouf university where he is conducting research activities in areas of cloud computing, information security, software defined network, Network Functions Virtualization. Dr. Hedi HAMDI is a member of RIADI-GLD Laboratory (ENSI-Manouba University).

Amri Sabrine received his master Thesis in Computer sciences form sousse university in December 2017. She currently a PHD Student. His main research contributions concern: Machine learning, VM consolidation and cloud computing.

Dr. Brahmi Zaki received his PhD in Computer sciences from The Faculty of Mathematical, Physical and Natural Sciences of Tunis-Manar University (Tunisie) in December 2010. He is currently an Associate Professor at Computer Science Department, College Of Science And arts at Al-Ola, Taibah University. He is member at RIADI-GLD Laboratory (ENSI-Manouba University). His main research contributions concern: Web services composition, intensive workflow scheduling in cloud computing and data stream mining for outlier and workload detection in Cloud computing.