Virtual Machines Placement Problem Based on a Look-Ahead Workload Window over Distributed Cloud Gaming Infrastructure

Eya Dhib, Khaled Boussetta, Nawel Zangar and Nabil Tabbane

November 11, 2019

Virtual Machines placement problem based on a look-ahead workload window over distributed Cloud Gaming infrastructure

1. Eya DHIB, 3. Nawel ZANGAR
and 4. Nabil TABBANE
MEDIATRON Laboratory
Higher School of Communication of Tunis
Tunisia

2. Khaled BOUSSETTA
L2TI Laboratory
Institut Galilée, Université Paris 13
France

***Abstract*—Based on a distributed Cloud infrastructure, geographic region, available resources, delays constraints, power consumption, etc are factors that involve in determining the best datacenter where to allocate virtual resources. Allocation cost will also be determined according to that choice. For example, datacenters, installed in cold area, offer lower cost because they need few cooling maintenance. However, small datacenters, called Cloudlets, installed closer to players, could impose higher cost because of their limited resources or high need to cooling maintenance. The present paper contributes on optimizing resources allocation cost by proposing intelligent resources placement over a distributed Cloud infrastructure for Massively Multi-players Online Gaming (MMOG) service. We propose a dynamic Virtaul Machines (VMs) placement algorithm that enable inter migration of VMs between datacenters in order to guarantee low cost while respecting constraints of capacity and response delay. Our placement algorithm is based on a look-ahead window that predicts future workload of the Cloud service with the aim of minimizing number of VMs migration. Experiments show effectiveness of our contribution in maintaining the balance between low cost objective and delay and VMs migration constraints.**

Index Terms_MMOG, Distributed Cloud infrastructure, VMs placement problem, resources allocation, multiple multi-dimensional knapsack problem.

I. Introduction

An important challenge of running large-scale Cloud services in a geo-distributed Cloud system is to minimize the overall cost. Cloud system consists of tens of geographically distributed datacenters interconnected with high-capacity WAN leased lines. As writing this paper, Amazon Web Service (AWS) deployed 33 datacenters distributed in 12 regions all over the world as shown in figure 1. It aims to extend its system to hold 42 datacenters worldwide distributed in 16 regions at the end of 2017 [1]. Cost may vary from Cloud provider to another. Besides, geographic region, available resources, delays constraints, power consumption, etc are keys factors that involve in deciding allocation resources cost. For example, running a VM with 2 virtual CPUs, 3.75 GB of memory and Linux OS on an AMAZON datacenter, installed in Sao Paolo (Brasil) is priced with 0.155 $/h. However, same characteristic of VM is priced as 0.1$/h if placed in an AMAZON datacenter installed in Ohio (Est-USA). Consecutively, the choice of datacenter, where to

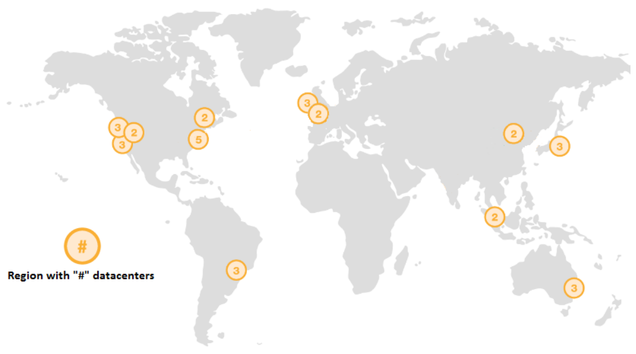

Figure 1. Worldwide distributed AMAZON Cloud infrastructure

allocate and place virtual resources, is a critical task. In principle, the problem of VMs placement can be classified into two categories: static VMs placement and dynamic VMs placement [2] [3]. Static placement of VMs can be conducted at the startup of a system, or where there are new VMs created which need to be placed to some Physical Machines (PMs) without moving any existing VMs, or all VMs would be shutdown and redistributed on PMs. However, in real systems, several issues make static method unreliable for VMs placement purposes. For example, varied available capacities in datacenters, unpredictable resources requirements of users especially in peak periods, etc, could distort VMs mapping decision. Therefore, dynamic placement is more suitable to achieve satisfactory performance. In fact, dynamic placement method adjusts resources in accordance with the service needs. The present paper discusses the dynamic VMs placement problem in order to minimize the resources allocation cost over a distributed data centers for large scale MMOG service. We propose a predictable dynamic VMs placement algorithm based on a look-ahead workload window. The predictable workload window helps in selecting and placing the adequate resources in the appropriate data center that minimize both delay and VMs migration metrics even for next periods. Results show effectiveness of our contribution in maintaining the balance between cost, delay and VMS migration constraints.
Our paper makes the following contributions:

- We formulate the problem of dynamic VMs placement under available capacity, response delay and minimal inter-migration VMS constraints in order to guarantee low resources allocation cost.

- We propose a predictable dynamic placement algorithm based on a look ahead workload window that predict future workload of the MMOG service in order to minimize number of VMs migration between datacenters. We evaluate our contribution comparing to static and dynamic placement algorithms.

The rest of the paper is structured as following: section II summarizes an overview of the related work. Section III express the system model for MMOG Cloud service and details the distributed Cloud Gaming architecture. Section IV discusses the predictable dynamic VMs placement problem. Section V analyzes experiments results. Last section concludes and suggests future works.

II. Related works

Dynamic placement approaches consist in suiting dynamically the mapping of resources to physical machines in order to balance workloads of PMs or for other reasons. Interested to that adaptable behavior, many existing studies gave attention to dynamic resources placement approach. We found that different mathematical formulations are used to model the placement problem as bin-packing or linear programming methods, etc. Authors in [4] propose a « Sandpiper » system that implements two algorithms of hotspot detection and mitigation. They use a greedy algorithm to determine a sequence of moves or swaps which iteratively migrates the highest loaded VM to the least loaded PM. Paper [5] discussed three major challenges faced by dynamic VMs placement which are constraints of multi-dimensional resource bounds, initial and legal intermediate states. They design a configuration-generation algorithm based on simulated annealing algorithm to solve the mapping optimization problem. Some existing researches implement two phase solutions to solve the dynamic VMs placement problem. The first phase consists in determining the minimum number of PMs needed. Second phase uses some heuristics to place VMs to PMs. Paper [6] presents an algorithm for dynamic VMs placement with minimizing cost goal. They calculate the minimum number of hosts required to support all VMs and remap them to PMs. The problem is mapped to a bin packing problem and is solved by using a heuristic approach based on the first-fit approximation. Authors in [7] used this two phases approach based on constraint programming for VMs consolidation and replacement problem. Enabling VMs migration, the first phase determines the necessary and minimal number of VMs. Thereafter, a reconfiguration plan with the lowest cost is chosen based on the decided configuration of virtual resources.

III. System model

The MMOG virtual world is a large-scale virtual game world populated with entities [8]. Entities could be either avatars representing players or objects describing the game scene, e.g walls, trees, weapons, etc. The virtual game world is partitioned into zones. As shown in figure 2, entities are distributed all over these zones. Zones are typically predefined and limited geographical virtual areas.

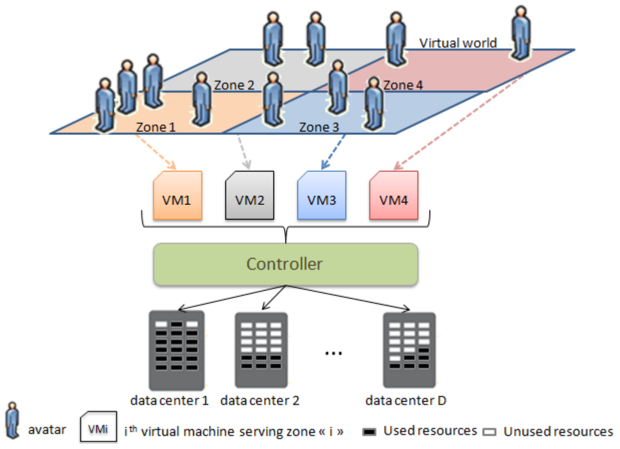

Figure 2. Distibuted MMOG architecture

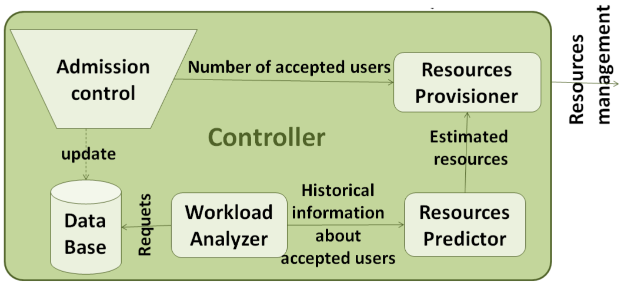

Figure 3. Submodels of the controller unit

Each zone is handeled independently by a separate VM. VMs are placed into physical servers according to the strategy of the game provider. All physical servers could be placed over a set of distibuted data centers. Due to mobility of avatars, MMOG zones have heterogeneous density in terms of entities. This heterogeneity makes configuration of correspondent VMs heterogeneous too. Thus, each zone may require different amount of resources such as CPU, memory, bandwidth, etc.

A cloud gaming platform is managed by a basic component called a controller [9] [10]. It is a software component responsible of resources management. Figure 2 focus on the controller role by decomposing it in submodules as following:

- *Admission control*: computes number of actually connected players. Accepted players are subject to application provisioning.
- *Data base*: data, sent by the *admission control*, are inserted into an SQL database.
- *Workload analyzer*: interacts with the historic data base to estimate futur service workload. Results are passed to *resources predictor* module.
- *Resources predictor*: decides capacity of expected resources to be allocated based on informations sent by *workload analyzer* module.
- *Resources provisioner*: allocates resources based on informations sent by *resources predictor* and *admission control*.

In a previous work [11], we have implemented the controller component with all sub-models in the aim of modeling the MMOG workload. We used as database,

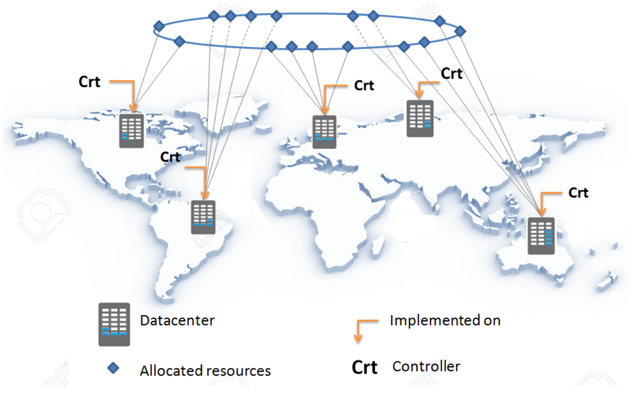

Figure 4. Cloud Gaming distributed architecture

Table I
USED SYMBOLS

Symboles	Description
M	Number of Data centers
V	Number of virtual machines
K	Number of resources type
$c_{m,v}$	Allocation cost corresponding to v^{th} virtual machine placed in m^{th} data center
$x_{m,v}$	Placement decision variable
$w_{m,v}^k$	Weight of k^{th} resource type corresponding to v^{th} virtual machine placed in m^{th} data center
p_v	Number of players served by v^{th} virtual machine
$D_v(p_v)$	Processing delay corresponding to v^{th} virtual machine when serving p_v players
$N_{m,p}$	Network delay corresponding to player p
Dt_{max}	Maximum threshold of total response delay
p	Player
v	vth virtual machine
m, m'	mth data center, m'th data center
k	kth resource type

real traces of "World of Warcraft" (WoW) game [12], a popular MMOG real game. Records describe a three-years play game from January 2006 to January 2009. As result, we have proposed the Seasonal Autoregressive Integrated Moving Average (SARIMA) model that generally fits the MMOG workload behavior. We suggest in this paper to use that predictive workload model for allocation and placement resources purposes.

IV. Predictable dynamic VMs placement problem based on look-ahead workload window

We model our predictable dynamic VMs placement problem using a multiple multidimensional knapsack one. Knapsacks are data centers. VMs are items to be placed in knapsacks. K types of resources can be required by an item (CPU, memory, bandwidth, space disk, etc). We define a vector weight $w_{m,v} = (w_{m,v}^1, w_{m,v}^2, ..., w_{m,v}^K)$ as a set of requirement resources of v^{th} item placed in m^{th} knapsack. Each knapsack has a vector capacity constraint $b_m = (b_m^1, b_m^2, ..., b_m^K)$ stating the maximum amount of each resource type. Formally, the problem is defined as follow:

$$minimize_{t=t_0}^{T}(\sum_{m=1}^{M}\sum_{v=1}^{V} c_{m,v}x_{m,v}) \quad (1)$$

$$(D_v(p_v) + 2 * N_{m,p})x_{m,v} \leq Dt_{max}, \forall m, \forall v, \forall p \quad (2)$$

$$\sum_{v=1}^{V} w_{m,v}^k x_{m,v} \leq b_m^k, m = 1..M, k = 1..K \quad (3)$$

$$min_t \sum_{v=1}^{V}\sum_{m \neq m'=1}^{M} x_{m,v}x_{m',v} \quad (4)$$

$$\sum_{m=1}^{M} x_{m,v} = 1 \quad (5)$$

$$x_{m,v} \in \{0,1\} \quad (6)$$

During a period of time T, we consider a set of V VMs to be placed in M datacenters. Each VM, noted v with $1 \leq v \leq V$, has a vector weight w_v composed by the amount of CPU $w_{m,v}^1$, memory $w_{m,v}^2$, bandwidth $w_{m,v}^3$, and storage space $w_{m,v}^4$. The variable $x_{m,v}$ is used to specify whether the v^{th} virtual machine would be placed in the m^{th} datacenter. In this case, $x_{m,v}$ takes 1 value; otherwise, zero value is filled instead. The objectif function, given by equation 1, aims to minimize the cost relative to the placement of VMs in datacenters. The pricing model adopted is for AMAZON EC2 [1]. Equation 2 express the delay constaint of our problem. In fact, the global response delay, which is the sum of network and processing delays, should not exceed a predefined threshold Dt_{max}. We have expressed and detailed network and processing delays functions in our previous work [16]. Equation 3 lists the capacity constraint of a knapscak problem for K types of resources. In fact, all allocated resource type, w_m^k, in the m^{th} datacenter should not exceed its coresponding capacity b_m^k. Equation 4 emphasizes minimization of migration number for each VM over time. Equation 5 ensures that each VM is placed in only one data center.

Since the objective function follows a linear aspect, we choose to solve the abovementioned problem by means of "linprog" solver. We have implemented our contribution on Matlab [13], a multi-paradigm numerical computing environment. We use N to denote all possible resources configurations offered by the Cloud provider, Z present number of zones that the MMOG world is partitioned to, VM_i is a variable that parses different resources configuration scenarios where the index i vary from 1 to N. T_w present the size of the look-ahead prediction window. As discussed previously, we employ the SARIMA(1,1,1) model to predict future workload of the MMOG service during the next T_w period. Hence, $V\hat{M}_{T_w}[z]$ denotes the estimated workload of z^{th} zone when looking T_w period ahead. As output, π_t denotes the placement vector of our algorithm at time t. π_t is defined as a placement sequence $\{\pi_t[1], \pi_t[2],..., \pi_t[z],..., \pi_t[Z]\}$ comprising placement of all zones of the MMOG world. At time t_0, algorithm1 initializes, in line 1, the first placement sequence π_{t_0} to NULL value. Line 2 sorts all configuration scenarios in ascending order based on their performances and allocation costs. In fact, the more the resources configuration is powerful, the high the cost will be [10]. Thereafter, line 3 initializes the window's size to look-ahead. As fourth step, Lines 4-19 iteratively find the adequat configuration

Algorithm 1 VMs placement algorithm based on look-ahead predictive workload window

```
Require: $N$: number of resources configuration scenarios,
    $Z$: number of zones of the MMOG world,
    $VM_i$: $i^{th}$ actual resources configuration scenario: number of VMs to allocate ,
    $T_w$: size of the look-ahead predictive workload window,
    $\hat{VM}_{T_w}[z]$: predicted workload relative to $z^{th}$ zone within next $T_w$ period,
    $\pi_t$: placement vector of decided VMs configuation of 1 .. Z zones,
    $\pi_t[z]$: $z^{th}$ element of the placement vector,
1: $\pi_{t_0}$ = NULL vector;
2: sort $VM_i$ with $i = 1..N$ depending on performances and allocation cost in ascending order;
3: set the size of the look-ahead predictive workload window $T_w$
4: while $t = t_0 : T_w : T$ do
5:   for z = 1 .. Z do
6:     if Jump == True then
7:       continue;
8:     else
9:       virtual resources allocation and placement ($t$,$z$,$\pi_t[z]$);
10:      Estimate future workload, noted $\hat{VM}_{T_w}[z]$ , of the $z^{th}$ zone within the next $T_w$ period;
11:      if $\pi_t[z]$ satisfy $\hat{VM}_{T_w}[z]$ then
12:        $pi_{t+T_w}[z]$ = $pi_t[z]$;
13:        Jump = True;
14:      else
15:        Jump = False;
16:      end if
17:    end if
18:  end for
19: end while
```

resources and their placement sequence for each zone over time.

Line 6 verifies if the previous configuration resources allocated for zone z could satisfy needs of the same zone at t. Verification decision is made in lines 11-16. If verified, our algorithm moves so to the next zone. In the other case, we call the allocation and placement resources function detailed in algorithm 2. This function fills out the sequence placement vector for a zone z at time t. Once decision is made, we predict workload for current zone z during next T_w period. Estimated workload, noted $\hat{VM}$, is given by the SARIMA(1,1,1) model as discussed in [11]. Line 11 tests if π_t could reply, also, future workload $\hat{VM}$ of zone z. To do so, the solver linprog tests if the configured resources of vector π_t at t satisfy delay, capacity and minimum migration constraints at $t + T_w$. On the base, Jump variable changes its value. If verification is proved, it takes 1 value, meaning that allocation and placement phase will be ignored and the same placement sequence will be kept for time $t + T_w$. If not, Jump variable takes zero value, meaning that we will reconfigured the sequence placement for next time slot.

We call the allocation and placement function at a given

Algorithm 2 virtual resources allocation and placement(t,z,$\pi_t[z]$)

```
Require: $N$: number of resources configuration scenarios,
    $VM_i$: $i^{th}$ actual resources configuration scenario: number of VMs to allocate ,
    $W_{z_t}$: workload of $z^{th}$ zone at time $t$
    $C(z, p_z, v)$: cost relative to $z^{th}$ zone when allocating $v^{th}$ resources configuration scenario,
    $D(z, p_z, v)$: response delay relative to $z^{th}$ zone when allocating $v^{th}$ resources configuration scenario,
    for i = 1 .. N do
2:    if $VM_i$ satisfy $W_{z_t}$ then
        Calculate the allocation cost $C(z, p_z, v)$ relative to the $VM_i$ configuration;
4:      Calculate the processing delay $D(z, p_z, v)$ relative to the $VM_i$ configuration;
        $\pi_t[z]$ = linprog($VM_i$,$C(z, p_z, v)$, $D(z, p_z, v)$);
6:      if $\pi_t[z]$ is NULL then
          upgrade to the next configuration scenario;
8:        continue;
        else
10:       break;
        end if
12:   end if
    end for
```

time t and for a particular zone z. At each iteration, algorithm 2 calculates the response delay and the cost corresponding to all configuration scenarios, then, calls the "linpog" solver to select the most suitable scenario that satisfies objective function and its constraints.

V. Experimental results

We compare in this section experiments results of three VMs placement approaches which are static, dynamic and predictable dynamic algorithms. Static heuristic keeps initial VMs placement without giving attention to workload variability over time. However, dynamic heuristic enable VMs re-allocation and migration over available datacenters if workload changes occurs. Predictable dynamic placement heuristic works like the simple dynamic algorithm with anticipating workload variations thanks to employing the SARIMA predictive model. This helps in allocating and placing resources which serves actual and future needs. Hence, number of future resources migration could be reduced. For our experiments, we consider 33 data centers, as a Cloud infrastructure, distributed all over the world [1]. For each data center, we assume 10 physicals servers having the following hardware characteristics [14]: 2048 Mb as memory space, 10^4 Mb as storage space, 100 Mbps as available bandwidth. Each unit has 4 processors with 100 MIPS as speed processor. Each physical server can host several VMs. Used VMs configurations are those offered by AMAZON EC2 platform for MMOG applications [1]. The number of zones and their population are extracted from the "World of Warcraft" (WoW) game [12], a popular MMOG real game. Referring to [15], we fixed the global response delay threshold D_{max} at 500 ms. In order to evaluate performances in terms of delay, cost

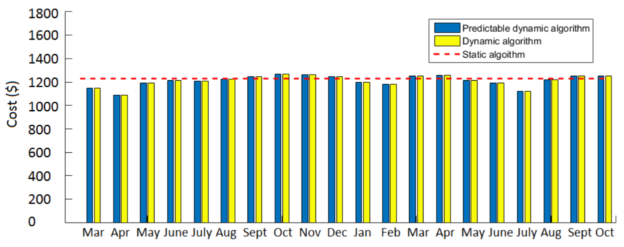

Figure 5. Allocation Cost

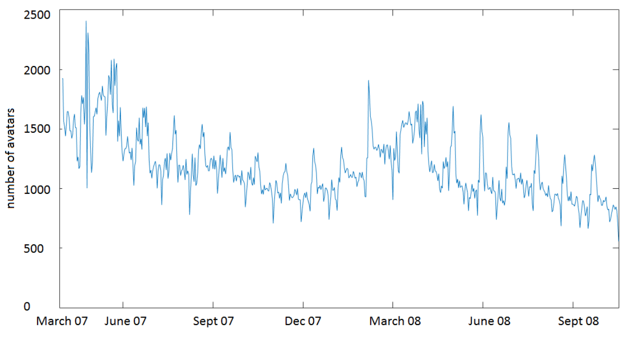

Figure 6. MMOG service workload

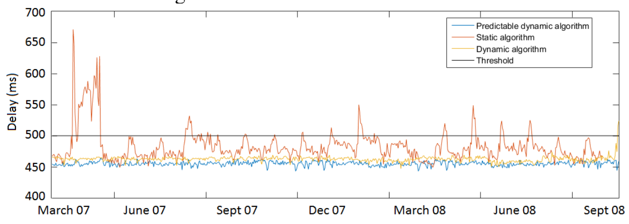

Figure 7. Global response delay

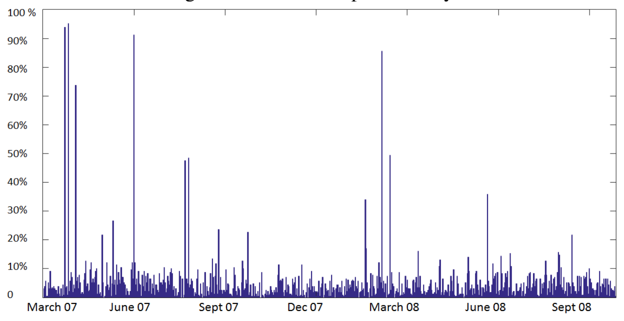

Figure 8. Difference of VMs migration between Dynamic and Predictable dynamic placement algorithms

and number of VMs migration of different algorithms, we developed and ran experiments using Matlab simulation tool [13].

Figure 5 shows costs results. The static heuristic allocates resources with a constant cost. Because both dynamic and predictable dynamic algorithms adjust allocated resources to the needs, their correspondent costs are lower or sometime equal to static cost. They realize up to 13% savings compared to static algorithm at high dense period (in March-April 2007).

As regards the delay constraint, figure 7 compares latency metric for above three heuristics. Static placement algorithm realizes longest delay that exceeds tolerate threshold in peaks periods. This could leads to a boring experience game. Because dynamic algorithm suits allocation and placement resources to service workload variation, we see its ability to maintain a quasi stable and shorter delay than the static one. Powerful VMs are allocated and placed in closer datacenters when the service presents high peak workload. Consecutively, processing and network delays would be as shorter as possible. In low peak workload, dynamic algorithm allows longer but acceptable delays by allocating powerless VMs and placing them in more far datacenters. However, predictive dynamic algorithm realizes shortest response delays comparing to others. At high peak rate (between March and July 2007), predictive dynamic placement heuristic optimizes delay by 23%comparing to static approach. Along all simulations period, both dynamics approaches realize closer delays with slight improvement marked by predictable dynamic placement algorithm.

A dual reading of two figures, 5 and 7, synthesizes that predictable dynamic algorithm makes better the deal between optimizing allocation cost and correspondent delay. Its look-ahead workload window helps in selecting and placing adequate resources in the appropriate data center that minimizes both delay and VMs migration metrics even for next periods. Configuration decision of VMs scenario to allocate depends imperatively on the density of players to serve. High density requires powerful VMs to guarantee short processing delay. Placement decision is based on distance between datacenter and players and on probability of non migration of resources during the look-ahead window time. Our contribution prefers placing resources in a datacenter that keeps global response delay under the predefined threshold during $[t, t + T_w]$ without VMs migration.

We define δ as the difference between number of VMs migration realized by the Predictable Dynamic algoithm, noted R_{PD} and number of VMs migration realized by the Dynamic algoithm, noted R_D. Fomally, $\delta = (R_D - R_{PD})/R_D$. Figure 8 show variation of δ variable over time. The simple dynamic algorithm realizes more VMs migration than our predictable dynamic algorithm. This emphasizes the role of the look-ahead window in selecting the placement that serves actual and future needs of the service. Figure 8 shows that our contribution saves up to of 10% of VMs migration in normal behavior service and up to 90% for high peak periods (March, June 2007, March 2008). This advantage could mitigates unnecessary migration signalisations for the system architecture.

VI. Conclusion

The present paper contributes to the improvement of resources management for a Cloud gaming service. For that, it gives attention to VMs placement problem over a distributed and heterogeneous Cloud infrastructure. Goal is to optimize the overall resources allocation cost under delay and VMs migration constraints. A multiple multi-dimensional knapsack model is used for problem formulation which is an NP hard one. Experimentally, our contribution realizes successfully the balance between these three compromises: allocation cost, delay and VMs migration. As future work, we plan to study the impact of VMs placement problem on the Quality of Experience

(QoE) of Cloud Gaming users. We aim also to propose a utility function or a load balancing mechanism to distribute fairly the workload over datacenters and to increase the utility rate of the allocated resources.

REFERENCES

[1] Amazon elastic compute cloud (Amazon EC2). http://aws.amazon.com/

[2] Z. A. MANN, *Allocation of Virtual Machines in Cloud Data Centers?A Survey of Problem Models and Optimization Algorithms*, ACM Comput. Surv. 48, 1, Article 11 (August 2015), 34 pages. DOI: http://dx.doi.org/10.1145/2797211.

[3] Z. Usmania, S. Singh, *A Survey of Virtual Machine Placement Techniques in a Cloud Data Center*, International Conference on Information Security & Privacy (ICISP2015), 11-12 December 2015.

[4] T. Wood, P. Shenoy, A. Venkataramani, M.Yousif, *Sandpiper:Black-box and gray-box resource management for virtual machines*. Comput.Netw.53,2923?2938, 2009.

[5] X. Liao, H.Jin, H.Liu, *Towards a green cluster through dynamic remapping of virtual machines*, Future Gener. Comput.Syst.28,469?477, 2012.

[6] N. Bobroff, A. Kochut, K. Beaty, *Dynamic placement of virtual machines for managing SLA violations*, In Proceedings of 10th IFIP / IEEE International Symposium on Integrated Network Management, pp.119?128, (May)2007.

[7] F. Hermenier, X. Lorca, J. Menaud, G. Muller, J. Lawall, *Entropy: a consolidation manager for clusters*, In Proceedings of the 2009 ACM SIGPLAN/SIGOPS international Conference on Virtual Execution Environments. Washington, DC, (March)2009.

[8] H.Kavalionak, E.Carlini, L.Ricci, A.Montresor and M.Coppola, *Integrating peer-to-peer and cloud computing for massively multiuser online games*, In: Peer-to-Peer Net- working and Applications, Volume 8, Issue 2, pp 301-319, March 2015.

[9] R. N. Calheiros, E. Masoumi, R. Ranjan and R.Buyya*Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications' QoS*, Cloud Computing, IEEE Transactions on (Volume:PP , Issue: 99), Aout 2014.

[10] R. N. Calheiros, R. Ranjany and R. Buyya, *Virtual Machine Provisioning Based on Analytical Performance and QoS in Cloud Computing Environments*, Parallel Processing (ICPP), 2011 International Conference on.

[11] *Impact of seasonal ARIMA workload prediction model on QoE for Massively Multiplayers Online Gaming*

[12] War of Warcraft avatar history dataset. (2011). [Online]. Available: http://mmnet.iis.sinica.edu.tw/dl/wowah/

[13] http://www.mathworks.com/products/matlab/

[14] Y. Gao, H. Guan, Z. Qi, Y. Hou and L. Liu,*A multi-objective ant colony system algorithm for virtual machine placement in cloud computing*,Journal of Computer and System Sciences, 79(8), pp.1230-1242., 2013.

[15] S.Wang, S.Dey, *Addressing Response Time and Video Quality in Remote Server Based Internet Mobile Gaming*, In 2010 IEEE Wireless Communication and Networking Conference.

[16] E.DHIB, K.BOUSSETTA, N.ZANGAR and N.TABBANE,*Modeling Cloud Gaming Experience for Massively Multiplayer Online Games*, Conference: 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), DOI: 10.1109/CCNC.2016.7444810