

Multi-Core Aware Virtual Machine Placement for IaaS Cloud DataCenters

Nagadevi.S¹, Dr.S.V.Kasmir Raja²

¹Research Scholar, Department of Computer Science and Engineering, IQAC, SRM Institute of Science and Technology, Chennai, India,

²Adjunct Professor, IQAC, SRM Institute of Science and Technology, Chennai, India,

Abstract

Cloud computing provides elastic computational resources that have been widely deployed in data centres to provide Infrastructure as a Service (IaaS). Virtualization is a key technology in cloud computing for resource sharing. In the virtualization technology the most challenging issue is Virtual Machine Placement Problem (VMPP). Virtual Machine Placement is a significant process executed as a branch of VM migration, which involves placing a Virtual Machine (VM) on a suitable physical machine (PM) in order to improve the resource utilization efficiently. Many existing VMP algorithms considered the overall CPU capacity of a PM, without considering the number of cores available in a PM and core-CPU capacity of a PM. i.e. VMs are mapped onto PMs, if the CPU capacity of the VM is less than or equal to the total CPU capacity of a PM. Such an allocation results in core overload which leads to performance degradation and violation of Service Level Agreement. However, in the real scenario, PM and VM consists of multiple cores. So, to place VM on PM, the core- CPU capacity of a VM must be mapped to a core-CPU capacity of a PM. i.e., the core of a VM should be mapped to a core of a PM. In this paper, we have proposed a Multi-Core Aware Virtual Machine Placement Algorithm (MCA-VMP). In MCA-VMP, the number of cores available in a PM and core-CPU capacity of a PM is considered instead of total CPU capacity of a PM. We used, Google Cluster Traces to generate the virtual machine configurations. Based on Google Cluster Traces the dataset is generated. Monte Carlo Simulation method is used to produce Google Cloud Jobs (GoCJ). Our simulation results shows that MCA-VMP is efficient than traditional non-Core Aware VMP algorithms. Our proposed algorithm MCA-VMP improves the performance of a DataCenter in terms of resource utilization, PM overload and resource wastage.

Keywords: IaaS, Virtual Machine Placement, Core Aware, CPU Utilization, multi-core

1. INTRODUCTION

Cloud computing paradigm is based on the model of pay for what you use. The infinite availability of resources at the back end leads to the cloud computing model. The efficiency and flexibility of cloud paradigm is due to readily available virtualized environments and machines.

Nowadays, organizations move their business into their own data centres. Data centres provide the environment for the

user's applications to run. Efficient utilization of data centre resources is the key issue of every organization. With the help of virtualisation concepts, organizations have virtualised their physical resources to service many requests. User gives the request to Cloud in to run their applications. The Cloud Service Provider (CSP) processes the users request. Then the request is either accepted or rejected by the CSP based on the availability of the resources in the DataCenter (DC). If the users request is accepted, then the VM satisfying the user request is created and assigned to them by CSP.

One key issue in deploying VM on PM is virtual machine placement. i.e. placing the VM on an appropriate PM, so that the resources used efficiently. VM placement is an important aspect of data centre resource management. One way to address the VM placement problem is by evaluating the resource utilization level of VMs and PMs. Resource utilization is one of the metric used to evaluate VMP algorithms.. To deploy a vm on a pm, a pm must have a sufficient resource capacity in all given dimensions.

Many researchers have designed virtual machine placement algorithms by considering the overall CPU capacity of a PM without considering the number of cores available in a PM and core CPU capacity of a PM. i.e. VMs are mapped onto PMs, if and only if the total CPU capacity of the VM is less than or equal to the total CPU capacity of a PM .Such an allocation results in core overload which leads to performance degradation and violation of Service Level Agreement. However, in the real scenario, PM and VM consists of multiple cores. So, to place VM on PM, the core CPU capacity of a VM must be mapped to a core CPU capacity of a PM. i.e., the core of a VM should be mapped to a core of a PM. So, VM to PM mapping must be done based on core mapping and not on machine mapping. For example, to place a VM with single core, 500MIPS, we may consider a PM with two cores and 250 MIPS each. According to non-multi-core aware virtual machine placement, the PM having a total of 500 (2cores* 250) MIPS is sufficient to place a VM with 500MIPS. But, a single core VM cannot take advantage of a dual core PM. The VM will only make use of a single core of a PM not both. Here, in our proposed system we have considered a multi-core aware VMP.

2. RELATED WORK

In [1], VM Placement decisions were made based on the following: Reservation, On-Demand and Spot Market. On-Demand VM placement decisions are based on initial

VM Placement and migration of VMs. Both of the above steps can be considered for power conservation, SLA, revenue maximization and reliability.

In [2], Classifiers, the resource allocation problem in heterogeneous data centre are categorized into 3 types: based on VM, based on PM, and based on application. Their study indicates the proactive resource allocation by predicting the future workload enhances the performance of resource management. It also confirms that finding the min. no of active PMs to satisfy the resource demand is the optimal solution to reach max. resource utilization.

In [3], authors have designed multi-objective VMP (MOVMP) algorithm using multi-dimensional Resource Usage Model. MOVMP uses Resource Utilization Factor (RUF) to determine the target physical machine to place virtual machine. The goal of the algorithm is to improve resource utilization in a balanced manner and also result in minimizing resource wastage, power consumption and SLA violations of Cloud data centres.

In [4], author used multi-core processors, to place the core of multi-core VMs. Using constraint programming; proposal algorithm shows better results to balance solution quality and time compared to non-multicore-aware heuristic. Algorithm considers only the CPU as a single dimension, achieved 25-60%; out performs traditional techniques.

In [5], authors have presented a publicly available cloud dataset to reflect a real workload from Google cluster traces Using Monte Carlo Simulation. The dataset is generated as Google Cloud Jobs (GoCJ) dataset. Size of jobs in the dataset is represented by MI. The GoCJ dataset consists of jobs of different types as small, medium, large, extra-large, huge in varying compositions.

In [6], the authors have proved that the stochastic characteristics of resource usages and proposed a solution, probabilistically, to handle resource overload during migration. The proposed stochastic load balancing algorithm used multi-dimensional resource request to measure migration cost by considering network topology. Also minimizes the migration overhead

In [7] reallocation of VMs performed to minimize the number of physical nodes. Idle nodes are switched-off to decrease the power consumption. MBFD is used for VM provisioning and VM placement. VMs are selected for migration based on MBFD.

In [8] initial VM placement is done in an unloaded DataCenter to reduce resource wastage (CPU, Memory, BW), power consumption and to minimize SLA violation(MIPS). ACO (Ant Colony Optimization) is used over GA and Heuristic method. Genetic Algorithm doesn't use feedback information, searches with blindness, large search space, do a lot of redundant iteration, convergence speed and efficiency of optimum solution is slow and low.

In [9] the authors have developed algorithm to increase

resource utilization, to meet SLA requirement and to reduce number of PMs. Pearson Correlation Coefficient between a pair of VMs is considered. As a result CPU utilization increased, SLA violation decreased , number of active servers reduced. [10] Mapping VMs to PMs is called VMP.VMP is part of VM Migration. Goal is to minimize Energy by shutting down services.

In [11], designs an online VMP algorithm to increase cloud providers revenue for the multi dimensional resources.

In most of the above mentioned VMP algorithms except [4], authors considered the overall CPU capacity of a PM, without considering the number of cores available in a PM and core-CPU capacity of a PM. i.e. VMs are mapped onto PMs, if the CPU capacity of the VM is less than or equal to the total CPU capacity of a PM. Such an allocation results in PM overload which leads to performance degradation and violation of Service Level Agreement.

3. PROPOSED SYSTEM MODEL

In our proposed system, we assumed a DataCenter with heterogeneous PMs. That means, PMs are having different configurations. i.e. each PM is considered to be having varying capacities in the dimensions CPU, number of cores and RAM . The number of PMs is fixed. In each PM multiple VMs could be created. Each PM is represented as PM<core-mips,,ram,cores>.'core-mips' represents the CPU capacity of a single core ,ram represents RAM capacity of a PM and cores represents the number of cores of a PM. The cores gives the total number of virtual machines that could be created within a PM. We also assumed that in each core only one VM could be created. This is simple multi-core model that we used to show the difference between the core-aware and non-core aware VM placement. VMs are represented as VM<core-mips,ram>. VMs are assumed to have single core.The proposed system architecture is shown in Fig.1.

When the request to place a new VM arrives, then the VMs CPU capacity is compared with the PMs core-CPU capacity of all PMs. Then, the suitable PM is selected according to the MCA-VMP algorithm. Cloud users submit their request to the IaaS cloud. VM Request Generator generates the request in the form of a VMs. The CSP maintains a pool of resources as PMs in a DC. Now, the requirement is to create the requested VM in an appropriate PM in the DataCenter. VM Request generated by the VM Request Generator is submitted to the VM Management System. Then the VM Placement Manager finds a suitable host for the requested VM based on the proposed MCA-VMP algorithm.

In our algorithm to map a VM onto a PM, the configurations mentioned in section .4 are used. To calculate the resource utilization of a proposed algorithm, we considered only the CPU capacity. Resource utilization must be calculated for a physical machine not for a single core of the machine. So, the CPU utilization of a single PM is calculated as,

$$\text{CPU Utilization} = \frac{\text{Total Number of MIPS used by all Active Cores of a PM}}{\text{Total Number of Active Cores * Core CPU MIPS}}$$

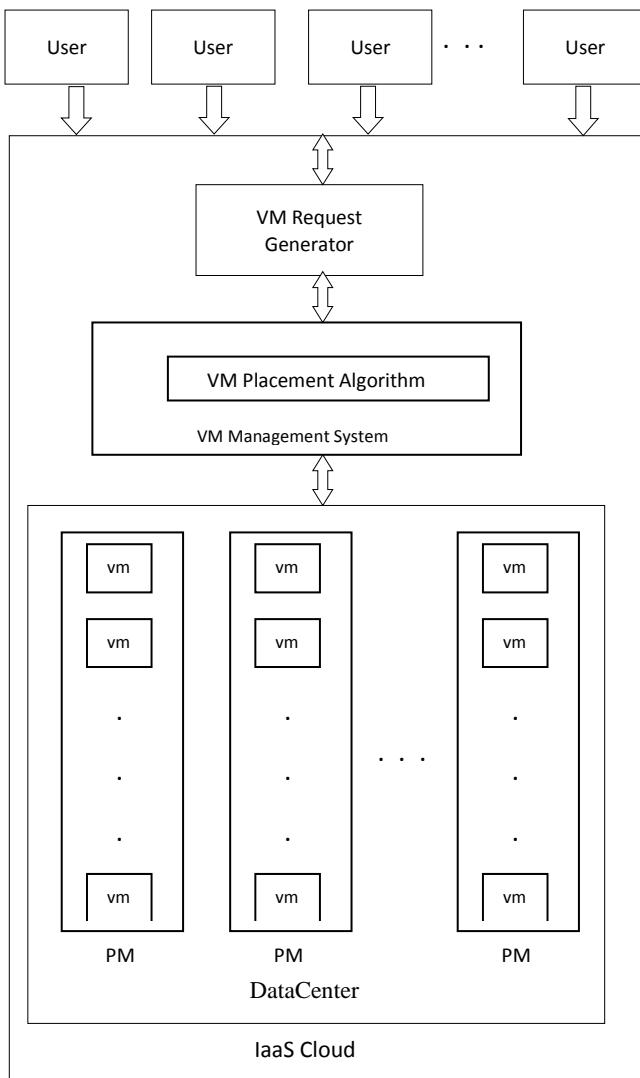


Fig. 1. Proposed System Architecture

3.1. Non-Core Aware Virtual Machine Placement Algorithm:

Input: number of vms, n and number of pms m,

Output: allocatedpm

allocatedpm=null;

For each vm in 1....n

 For each pm in 1....m

 If vm-mips <= pm-mips && vm-ram<= pm-ram
 then

 If vm-mips<=pm-core-mips then
 allocatedpm =pm;
 break;
 else
 continue;
 End if

 End if

End for

allocate vm on allocatedpm;

pm-mips = pm-mips - vm-mips;

pm-ram = pm-ram - vm-ram;

End for

In non-core aware VMP, the target PM to place a VM is searched based on the total capacity of a PM. If enough resources available in a PM, to place a VM, then VM is placed on that PM, without checking the number of free cores of a PM . This kind of VM placement results in PM overload and core overload, results in performance degradation of the PM.

3.2. Multi-Core Aware First-Fit Virtual Machine Placement Algorithm:

Input: number of vms n and number of pms m,

Output: allocatedpm

For each vm in 1....n

For each pm in 1....m

//Check if core is available in a physical machine

 Read the number of free cores in the pm, as nFreeCore,

 If free core is not available in a pm, then skip the pm from further process and continue with the next pm ;

 Else

 If (vm-mips<= pm-core-mips && vm-ram<=pm-ram) then,

 allocatedpm=pm;

 create vm in the first free core of the pm;

 nFreeCore=nFreeCore-1;

 pm-ram= pm-ram - vm-ram;

 break;

 //no need to update the CPU capacity of a PM since it is core-based
 //Either RAM is not available or there is no core available with the requested CPU capacity.

 Else

 Exit;

 End if;

End if;

If no pm is having the free core to place a vm, then exit;

End for

End for

In multi-core aware VMP, the new vm-mips is checked against the core-mips of a PM, so that, the PM does not get overloaded. So, the overall performance is improved when compared to non-core aware virtual machine placement.

4. SIMULATION DATA

The CPU capacity is measured in terms of MIPS (Million Instructions per Second). Each core of a PM carries the equal CPU capacity. For example, if a PM is having 4 cores and each core-CPU capacity is 500 MIPS. Then, the total CPU capacity of a PM is $4 \times 500 = 2000$ MIPS. In other words, if CPU capacity of a 4-core PM is 2000 MIPS, then each core has the core-CPU capacity $2000/4 = 500$ MIPS. But, the RAM capacity is applicable for the entire PM and RAM capacity of a PM is shared by all the cores of a PM in different capacity.

Table 1: Physical Machine Configurations

PM Type	Number of PMs	Number of Cores	Core-MIPS	RAM (MB)	Total Number of cores
1	12	4	1860	4096	12*4cores=48
2	2	2	2660	4096	2*2cores=4 Total=52 cores

To reflect the real workload behaviour of a cloud environment, we used the Google Cluster Traces to generate the virtual machine configurations. Based on Google Cluster Traces Monte Carlo Simulation method is used to generate Google Cloud Jobs (GoCJ) as shown in Table 2.

Table 2: GoCJ dataset

VM Type	MIPS Range	Composition (%) of jobs
Small	150 – 550	20
Medium	590 – 990	40
Large	1010 – 1350	30
Extra-Large	1500 - 5250	4
Huge	5250 – 9000	6

Each row in a GoCJ dataset consists of numerical value representing the size of a job in terms of MIPS. The GoCJ sizes vary on the basis of MIPS as shown in Table. Here, the job sizes are equal to the VM MIPS. Based on the GoCJ dataset, the simulation is executed for 50 jobs. Each VM is considered to be having a single core and 512MB RAM. The VM configurations considered for our proposed work is shown in Table 3.

Table 3: Virtual Machine Configurations

Total No. of VMs	VM Type	Number of Cores in each VM	RAM (MB)	VM MIPS
50	As in GoCJ	1	512	As generated by GoCJ

5. SIMULATION ANALYSIS

We have simulated our model in Cloudsim Tool. We have used 14 PMs and 50 VMs for our algorithm. Our algorithm results are compared with non-core aware First-Fit algorithm. Our algorithm is proved to be efficient in terms of amount of CPU utilization and CPU overload.

Here, we considered the VM and PM configurations as given in section 4. GoCJ dataset is used for the count of 50VMs. The VM placement is done on the basis of non-core aware First-Fit algorithm. VM is created on PM if PM is having the MIPS. As long as the PM RAM and PM MIPS are available; the VM is being allocated to that PM, without considering the number of free cores available in a PM. Such an allocation leads to a suboptimal solution. In the table given below, the assignment of vm-4, vm-5, vm-6 and vm-7 is possible on Host0 because mips and RAM is available. Though the number of cores of a Host0 is 4, 8 VMs are created on it. Such an allocation leads to undesirable solution. Additionally created VMs cannot run applications immediately; rather it

has to wait in a queue for the other VMs to finish which increases the waiting time of a VM. So, the SLA is violated and performance degraded. Three VMs, vm-10 (MIPS 5250), vm-28 (MIPS 9000) and vm-46 (7125) are not created in any of the data centres due to MIPS unavailability.

Table 4: Non-Core aware VMP simulation results

PM	VMs Allocated	VMs not created
Host0	vm-0,vm-1,vm-2,vm-3, vm-4,vm-5,vm-6,vm-7	vm-10,vm-28,vm-46
Host1	vm-8,vm-9,vm-11,vm-12, vm-14,vm-15,vm-16,vm-17	
Host2	vm-18,vm-19,vm-20,vm-21, vm-22,vm-23,vm-24,vm-27	
Host3	vm-25,vm-26,vm-29,vm-30, vm-31,vm-32,vm-33,vm-34,	
Host4	vm-35,vm-36,vm-37,vm-38, vm-39,vm-40,vm-41,vm-42,	
Host5	vm-43,vm-44,vm-45, vm-47,vm-48,vm-49	
Host6	Not Activated	
Host7	Not Activated	
Host8	Not Activated	
Host9	Not Activated	
Host10	Not Activated	
Host11	Not Activated	
Host12	vm-13	
Host13	Not Activated	

In Table 4 each of the PM is created with at most 8 VMs. But, PM is having only 4 cores. So, the creation of 5th, 6th, 7th and 8th VM is overloads the PM and leads to infeasible solution. It is not practical that the MIPS of a VM are distributed among the cores of a PM. But, vm-13 has been created in Host12, not on other Hosts, since the allocation is based on core MIPS, not on PM MIPS. Since vm-13 requires 1900MIPS, it may be allocated to Host12 or Host13, but not on other Hosts. The vm-13 is allocated to the next available PM, using First Fit algorithm which is having the core capacity more than or equal to 1900MIPS.

In multi-core aware First-Fit virtual machine placement algorithm, the PMs are considered with cores. The VMs are created if the PM is having the free cores. If free core is not available then VM cannot be created on that PM. Even when the RAM and MIPS are available in a PM, it is not possible to create VMs on the PM if free core is not available. Most of

the earlier VM placement algorithm does not consider the cores. They considered only the available capacity of PM in terms of available MIPS for the VM placement but not cores.

In this MCA-VMP, VMs are created on a PM is equal to the number of cores of that PM. If core is not available in a PM, then even when the MIPS and RAM is available, it is not possible to create a VM. As in the previous case, vm-10 (MIPS 5250), vm-28 (MIPS 9000) and vm-46 (MIPS 7125) are not created in any of the PMs, due to MIPS unavailability. In this method all the cores of a PM are utilized and also, the created VMs could run applications on it.

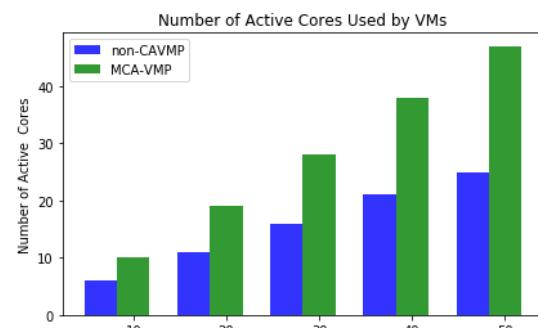
In MCA-VMP, the PM selection is based on number of free cores of a PM and VM allocation is based on core-MIPS. In non-core aware VMP, the PM selection is based on available MIPS and VM allocation is based on core-MIPS. Table.5 shows that each of the PM is created with at most 4 VMs. Since, the VM allocation is core based. Similar to non-core aware, in MCA-VMP, vm-13 has been created in Host12. It shows that the MCA-VMP is also based on core-MIPS.

Table 5: Multi-Core aware VMP simulation results

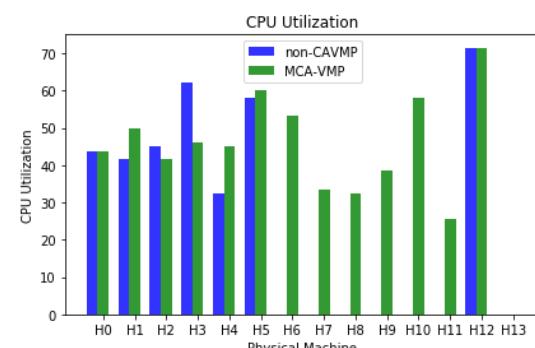
PM	VMs Placed	VMs not Placed
Host0	vm-0,vm-1,vm-2,vm-3,	vm-10, vm-28, vm-46
Host1	vm-4,vm-5,vm-6,vm-7,	
Host2	vm-8,vm-9,vm-11,vm-12,	
Host3	vm-14,vm-15,vm-16,vm-17	
Host4	vm-18,vm-19,vm-20, vm-21	
Host5	vm-22,vm-23,vm-24,vm-25	
Host6	vm-26,vm-27,vm-29,vm-30	
Host7	vm31, vm32,vm33,vm34	
Host8	vm-35,vm-36,vm-37,vm-38	
Host9	vm-39,vm-40,vm-41,vm-42	
Host10	vm-43,vm-44,vm-45,vm-47,	
Host11	vm-48,vm-49,	
Host12	vm-13	
Host13	Not activated	

5.1. Simulation Evaluation

Our simulation results using MCA-VMP, compared with the non-CAVMP using First-Fit algorithm. The performance metrics that we use here is Number of Active Cores used, Core Utilization and Resource Wastage. Our simulation results, shows that the proposed MCA-VMP improves the number of active cores, Core Utilization and minimizes the Resource Wastage.



(a)



(b)

Fig.2. Simulation results of MCAVMP Using GoCJ dataset of size 50vms. a) Number of Active Cores b) CPU Utilization

6. CONCLUSION

In this paper, we have argued that core awareness is necessary in the process of VM placement in cloud DataCenters. The non-core aware VM placement algorithms lead to sub-optimal results and PM overload. We designed a new algorithm based on cores called MCA-VMP. Our algorithm used a real GoCJ dataset to reflect the real scenario. Our simulation results show that MCA-VMP is efficient in terms of resource utilization, resource wastage and PM overload compared to traditional non-CAVMP algorithm. Our proposed MCA-VMP algorithm helps to improve the total amount of active cores used, core utilization and resource wastage. In our future work, we would like to extend out proposed work for the other resource dimensions like Memory and Storage

REFERENCES

- [1] K.Mills,J.Filliben and C.Dabrowski,"Comparing VM-Placement Algorithms for On-Demand Clouds", Third IEEE International Conference on Cloud Computing Technology and Science, 2011
- [2] Xiang Sun, Nirman Ansari, Ruopeng Wang,Optimizing Resource Utilization of a Data Centre, IEEE Communications Surveys and Tutorials, Vol. xx.No.xx. xx 2016 01.10.1109/COMST.2016.2558203.

- [3] Madesh K Gupta, Ankit Jain, Tarachand Amgoth, Power and resource- aware virtual machine placement for IaaS Cloud, Sustainable Computing: Informatics and Systems, 19(2018) 52-60.
- [4] Zoltan Adam Mann, Muticore-aware virtual machine placement in cloud data centre, Transactions on Computers, 0018-9340,2015.
- [5] Altaf Hussain and Muhammad Aleem, GoCJ Google Cloud Jobs Dataset for Distributed and Cloud Computing Infrastructures, Data MDPI 2018,3,38; doi:10.3390/data 3040038
- [6] Lei Yu, Liu hua chen, Zhipeng Cai, Haiying shen, Yi Liang, Yi Pan,Stochastic Load Balancing for Virtual Resource Management in Data centres, IEEE Transactions on Cloud Computing, Vol, N., Nov 2014.
- [7] Anton Beloglazov and Rajkumar Buyya," Energy Efficient allocation of Virtual machines in Cloud Data Centers", 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010.
- [8] Fei MA,Feng LIU,Zhen LIU, Multi-objective Optimization for Initial Virtual Machine Placement in cloud Data Center, Journal of Information & Computational science 9: 16 (2012) 5029-5038
- [9] Zar Lwin Phyo and Thander Thein, Correlation Based VMs Placement Resource Provision, International Journal of Computer science & Information Technology(IJCSIT) vol 5, no 1 , February 2013
- [10] Rajeev Kumar Gupta and R.K.Pateriya, " Survey on Virtual machine Placement Techniques in Cloud Computing Environment", International journal on Cloud Computing: Services and Architecture(IJCCSA), Vol 4, No.4,August 2014
- [11] Laiping Zhao,Liangfu,Zhou Jin, and Ce Yu," Online Virtual machine placement for Increasing Cloud Providers Revenue " IEEE Transactions Services Computing,2015.