Dynamic Resource Provisioning for the Cloud using Bin Packing Technique

R. Madhumathi[a], R. Radhakrishnan[b], S. Suresh Kumar[c]

[a,c]*Assistant Professor, Computer Science & Engineering, Sri Ramakrishna Engineering College, Coimbatore-641022, India*

[b] *Principal, Vidhya Mandir Institute of Technology, Perundurai - 638051, India*

Abstract: **The biggest challenge in cloud computing environment is resource allocation, which in turn should be managed effectively in order to optimize the task execution. The cloud providers let their customers to access the resources in the form of virtual machines in their data centers and charge them over a period. Resource allocation must ensure effective utilisation and meeting the customer needs. Also, resources need to be reallocated in case of failures or load maximization problems. Usually utmost care should be taken in maintaining the capacity of total number of virtual machines without exceeding the capacity of the physical machines. Therefore, the load of resources that exceeds the capacity decides the VM migration. A practical online bin packing algorithm called the Variable Item Size Bin Packing allocates data center resources dynamically through live VM migration. However, the Service Level Agreement parameters are not considered while migrating the VMs to the other PMs. To overcome this, Enhanced Variable Item Size Bin Packing technique is proposed in our work. Here, the CPU usage is considered as the SLA parameter. The experimental result proves that the proposed methodology provides better result than the existing methodology.**

I. INTRODUCTION

Cloud computing is an attractive technology in the field of computer science. It makes use of the internet and provides services to the external users with secure, quick, convenient data storage and computing power. Virtualization, distribution and dynamic extendibility are the basic characteristics of cloud computing [1]. Resource allocation [2] is a technique works on achieve efficient resource utilization and maximum throughput by distributing workload across several computers, computer clusters, network links and central processing units. The problem of task mapping in heterogeneous systems is finding proper assignment of tasks to processors in order to optimize some performance metrics such as system utilization, load balancing and minimum execution time. A suitable scheduling algorithm is needed to overcome this restriction.This paper proposes a new resource allocation scheme called Enhanced Variable Item Size Bin Packing (EVISBP) technique. Bin packing problem is a combinatorial NP-hard problem. It requires packing a set of objects into a finite number of bins of capacity V in a way that minimizes the number of bins used. Virtual machine (VM) migration is done in existing scenario, if the load of resources are exceeds the capacity.There are two varieties of bin packing techniques namely online and offline. In online Bin-Packing [9], items arrive one at a time (in unknown order), which are put in a bin sequentially, before considering the next item. In offline Bin-Packing, all items are given upfront. Due to the dynamic nature of cloud, online bin-packing algorithms are used in storage, scheduling and resource allocation preferably.

The rest of the paper is organized as follows. Related work is described in section 2. Section 3 gives an overview of our approach. Section 4 describes enhanced variable item size bin-packing algorithm. Experimental results are discussed in section 5. Section 6 concludes the paper.

II. RELATED WORK

Li et al. have proposed a variant of the DBP problem, namely, the MinTotal DBP problem which targets at minimizing the total cost of the bins used over time [3,4]. The competitive ratios of the various bin-packing problems are studied in this paper. Cheema et al. have implemented resource allocation models for SaaS application deployments over CC platforms considering the cost and user requirements [5]. Xiao et al. proposed a system that uses a virtualization technology to allocate datacenter resources dynamically based on application demands and support green computing by optimizing the number of servers in use [6]. Here, Skewness is used to measure the unevenness in the multidimensional resource utilization of a server.

Warneke et al. have proposed Nephele, which is a data processing framework to exploit the dynamic resource provisioning and mentioned the challenges and opportunities for efficient parallel data processing in cloud environment [7]. Espadas et al. have given a tenant based model for overcoming over and underutilization when SaaS platforms are deployed over cloud computing infrastructures [8]. Lee et al. have proposed a resource allocation scheme based on performance analysis for efficient allocation of virtual machines on the cloud infrastructure [9]. Di et al. have proposed dynamic optimal proportional – share (DOPS) resource allocation method, which redistributes available resources among the running tasks dynamically, such that the maximum capacity of each resource in a node is utilized well [10].

Rahman et al. have proposed a novel scheme for virtual resource allocation with three key contributors, optimization of task's resource allocation under user's budget, maximized resource utilization based on PSM and Lightweight resource query protocol with low contention [12]. Chunlin et al have proposed multi-layer optimization in cloud computing to optimize the utility function, subject to resource constraints of IaaS, SaaS providers and end users [14]. Abdelkader et al have proposed a dynamic task scheduling algorithm with load balancing for heterogeneous computing system [15]. It achieves load balancing and better execution time than HEFT algorithm and triplet cluster algorithm.

III. OVERVIEW

Figure1 represents a typical virtualized data center where our resource allocation model is used. The components are VM scheduler, virtual machine monitor (VMM) and physical machines (PMs). The following inputs are received by the VM scheduler.

- The resource demand history of VMs
- The capacity and load history of PMs
- The current assignment of VMs to PMs

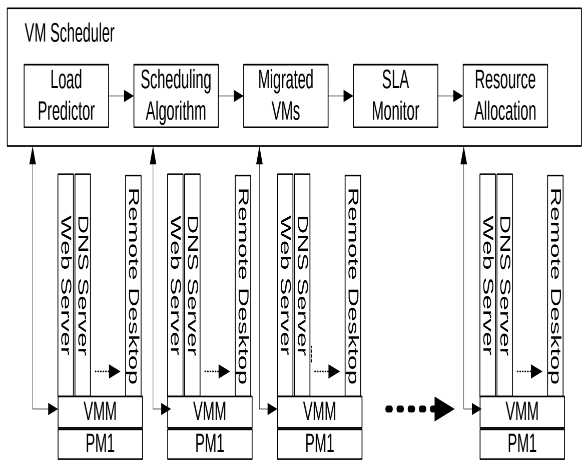

Fig. 1.Overview of the system architecture

VM scheduler consists of three modules. The load predictor calculates the load requirements of each VM. The scheduling algorithm then allocates the resources and the migrated VM list is generated. The SLA monitor checks during the migration of VMs to check whether the SLA parameters are correctly followed. Then the, allocation of appropriate nodes to VMs is done by the resource allocator based on the load predicted by the load predictor. Data centre applications provide significant opportunities for multiplexing server resources. Virtualization technology makes it easy to move running application across physical machines.

Few applications such as web server, remote desktop, DNS mail server, Map/Reduce, etc., run inside the VMs. The VMM collects the usage statistics of resources on corresponding PM such as CPU utilization, memory consumption, network sends and receives, etc. These statistics are forwarded to a centralized VM scheduler, the key component that is responsible for load balance and green computing by adjusting the mapping of VMs to PMs. The VMScheduler is invoked periodically and receives the inputs.

IV. RESOURCE ALLOCATION AS BIN PACKING

4.1 Resource estimation

Here, four bins with varying size are considered, namely S, M, L and SM. S denotes small size bin, M denotes medium size bin, L denotes large size bin and SM denotes small-medium size bin.

The load for each VM is estimated using,

$$E(t) = \alpha * E(t-1) + (1-\alpha) * O(t), 0 \le \alpha \le 1$$

Where, $O(t)$ denotes observed load at time t.

The utilization of the host $Util(host)$ is calculated as the sum of the utilization of individual VMs. That is,

$$Util(host) = \sum_{i=1}^{n} Util(VM_i)$$

The upper threshold value of the host T_{upper} is calculated as,

$$T_{upper} = 1 - (((Puu * Sqr) + sum) - ((Pul * Sqr) + sum))$$

where, Puu denotes preserve amount of CPU capacity by upper probability limits, Pul denotes preserve amount of CPU capacity by lower probability limits, sum denotes utilization of individual VMs and Sqr denotes the square root of the sum. That is, $Sqr = \sqrt{sum}$

The CPU will be considered overloaded when the utilization is above this value, so we migrate some of the VMs. The node is considered to be underutilized when the CPU utilization is below this value, so all VMs are migrated to other node. We thought that if the CPU usage is above 30%, lower threshold (T_{lower}) is always 0.3.The lower threshold value of the host T_{lower} is calculated as follows

$$T_{lower} = sum - (Pl * Sqr)$$

Where, Pl denotes probability limit of lower threshold of CPU.

4.2 EVISBP Algorithm

This scheduling algorithm checks whether every PM has enough resources for the appointed demands. If so, the VMM is able to adjust resource allocation locally. Otherwise, the algorithm performs overload avoidance by migrating away some of the VMs on the overloaded PMs. It also consolidates VMs on the under-utilized PMs, so that some of the PMs become idle and can be set into standby mode to save energy. The proper VM is selected to optimize the allocation during VM migration. We have considered placing of VM as a bin packing type of problem.

Migrate(VM)

```
for host (i to n)
    {       if($H_i$ != current host)
            {       util = util($VM_i$) +  util($H_i$);
                    if($T_{upper}$ < util > $T_{lower}$)
                            assign the VM to $H_i$          }
            else
            i++;
    }
```

Allocate(VM)

Input : Access the available resources at different PMs say {PM_1,PM_2,PM_3,…PM_m} and measure the demand of the VMs or cloud users say {VM_1,VM_2,VM_3,….VM_n}. Based on capacity and MIPS of PMs they are classified into four type's names S, M, L and SM.

```
for VM (i to n)
    for host (i to n)
    check if (capacity($H_i$) < capacity ($VM_i$)
            allocate the VM to the $H_i$
    else
            i++ (till all the VMs are handled)
calculate (util) for each VM
calculate upper and lower threshold value
for host (i to n)
    {       check if (util($H_i$) > $T_{upper}$)
            for VM  ( i to n)
            {       check (util($VM_i$) > $T_{upper}$ )
                    {       call migrate($VM_i$)
                    }
            }
            check if (util ($H_i$) < $T_{lower}$)
            {       for each VM (i to n)
                    {       call migrate($VM_i$)
                    }
            }               }
stop
```

V. RESULTS

The proposed model is simulated using CloudSim toolkit. Experiments are conducted by varying the number of VMs and tasks for obtaining average number of active PMs, accommodating the given request and to calculate the average number of migrations occurred. The number of PMs, VMs and its properties are gathered as input. The results obtained clearly reveal that the performance of the proposed algorithm is better against the VISBP algorithm.

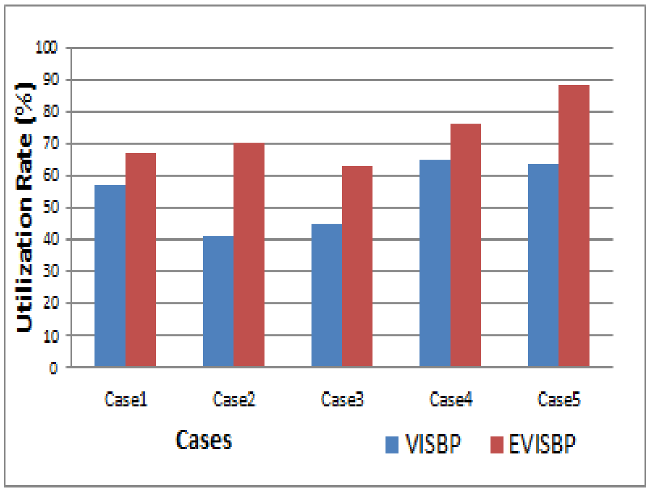

Fig. 2 Utilization rate of PMs

The figure 2 shows the overall utilization of PMs by considering five different cases. In case 1 five PMs and twenty VMs are considered. Similarly case 2 has ten PMs and fourty VMs, case 3 has fifteen PMs and sixty VMs, case 4 has twenty PMs and eighty VMs and case 5 has twenty five PMs and hundred VMs. The overall utilization rate is calculated from the utilization of individual PM and the number of live machines. Our main objective is to handle more number of tasks using less number of VMs. The figure 3 depicts that the proposed algorithm uses less number of VMs than the existing algorithm.

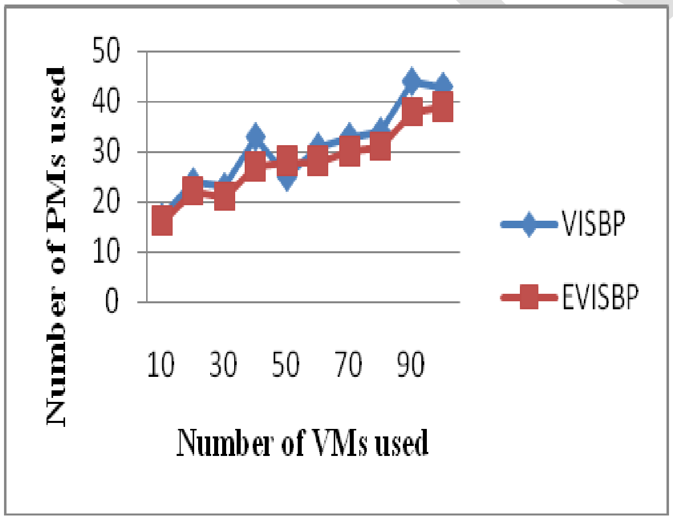

Fig. 3 VMs Vs PMs

VI. CONCLUSION

In this paper, we proposed an efficient algorithm that follows a best fit Bin-Packing technique for the allocation of virtual machines to the physical nodes. The migration of VMs takes place based on the threshold values of the host. This has increased the total utilization rate of physical nodes used and decreased the PMs to VMs ratio. When compared with the existing algorithm, the utilization rate of proposed algorithm was fairly high in the experiments conducted with different cloud users. This method is suits well for specific application performance model and in future memory de-duplication techniques can be employed to improve the ratio of VM to PM.

ACKNOWLEDGEMENT

The authors would like to thank the Management, Director & principal of Sri Ramakrishna Engineering College and Head of the Department of Computer Science & Engineering, for their support in completing this work.

REFERENCES

[1]. Rajkumar Buyya et al. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as 5th utility. Journal of Future Generation Computer Systems.2009; 25:599-616.
[2]. Linlin Wu et al. SLA-based Resource Allocation for Software as a Service Provider (SaaS) in Cloud Computing Environments. IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. 2011. pp. 195-204
[3]. Yusen Li et al. On Dynamic Bin Packing for Resource Allocation in the Cloud. In: SPAA 2014 Proceedings of 26thACM symposium on Parallelism in algorithms and architectures. 2014. pp. 2-11.
[4]. Yusen Li et al. Dynamic Bin Packing for On-Demand Cloud Resource Allocation. IEEE Transactions on Parallel and Distributed Systems, 2015.
[5]. Ibrahim A. Cheema et al. Cloud based Cost Effective Resource Allocation Model for Software-as-service Deployments. International Journal of Computers and Technology. 2013; 9:1068-1079.
[6]. Zhen Xiao et al. Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment. IEEE Transactions on Parallel and Distributed Systems. 2013; 24:1107-1117.
[7]. Daniel Warneke, Odej Kao. Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud. IEEE Transactions on Parallel and Distributed Systems, 2011; 22: 985-997.
[8]. Javier Espadas et al. A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures. Journal of Future Generation Computer Systems. 2013; 29: 273-786.
[9]. Hwa Min Lee et al. Performance analysis based resource allocation for green cloud computing. Journal of Supercomputing. 2013. pp. 80-93.
[10]. Sheng Di, Cho-Li Wang. Dynamic Optimization of Multiattribute Resource Allocation in Self-Organizing Clouds.

IEEE Transactions on Parallel and Distributed Systems. 2013; 24: 464-477.

[11]. L. Epstein, M. Levy. Dynamic Multi-Dimensional Bin Packing. Journal of Discrete Algorithms. 2010; 8:356–372.

[12]. Md. Mahfuzur Rahman et al. Differential Time-Shared Virtual Machine Multiplexing for Handling QoS Variation in Clouds. In: Proceedings of 1st ACM Multimedia International workshop on Cloud-Based Multimedia Applications and Services for e-Health. 2012. pp. 3–8.

[13]. Clark, K. Fraser et al. Live Migration of Virtual Machines. In: Proceedings of 2nd International conference on Networked Systems Design and Implementation. 2005. pp.273–286.

[14]. Li Chunlin, Li Layuan. Multi-Layer Resource Management in Cloud Computing. Journal of Network and Systems Management. 2013; 22: 100-120.

[15]. Doaa M. Abdelkader, Fatma Omara. Dynamic task scheduling algorithm with load balancing for heterogeneous computing system. Egyptian Informatics Journal. 2012; 13:135-145.

[16]. Rodrigo N. Calheiros et al. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environment and Evaluation of Resource Provisioning Algorithm. 2011; 41: 23-50.