Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Improving Server consolidation using physical consolidation concept

Alexander Ngenzi [1], *, R. Selvarani [2], R. Suchithra [3]

[1]*Department of Computer Science and Engineering, Jain University, Bangalore, India*
[2]*Department of Computer Science and Engineering, Alliance University, Bangalore, India*
[3]*Department of Master of Information and Technology, Jain University, Bangalore, India*

ARTICLE INFO

Article history:
Received 25 April 2016
Received in revised form
28 June 2016
Accepted 29 June 2016

Keywords:
Physical consolidation
Server consolidation
Best fit
First fit and load

ABSTRACT

Creating enough space for resources by minimizing number of servers in datacenters is a prerequisite for server consolidation. This requirement can be characterized by Bin packing algorithm, which is known to be NP hard problem. There are different categories of bin packing problem like First fit decreasing, Best Fit, and load balancing techniques. Our paper applies Best Fit algorithm to address Bin packing problem. The results show that this technique increases performance as far as server consolidation is concerned.

1. Introduction

One of the stages of server consolidation is Physical consolidation which involves moving a large number of source (existing) servers to a small number of high performance target servers. The target of our paper is Physical consolidation as an engine of improving server consolidation. The growth of Physical consolidation is due to server virtualization technology that enables multiple virtual machines, to which existing servers are moved, to share the physical resources of a computer. However, two contradictory requirements on shared resources must be satisfied at the same time. On the other hand, sufficient resources to avoid degradation in performance from performance management point of view. Resource utilization is then a prime indicator because the sum of the utilizations of virtual machines running on a destination server must not exceed the threshold prescribed for that server. On the other hand, the number of destination servers has a significant impact on the cost of server consolidation. The number of destination servers is required to be small enough to reduce the cost. This minimization can be formalized as a vector packing problem, which a variant of bin packing well known in the field of operation research. Given the items of different sizes in the bin packing problem and asked to pack them into a minimum number of bins with a given capacity. Items for server consolidation are the existing servers, Item sizes are the resource utilizations, bins are the destination servers, and the bin capacity is the utilization threshold of the destination servers. There are at least four kinds of utilizations for major physical resources, the CPU, the memory, and the network of computers. The utilizations of these resources are summed and compared to thresholds independently of other resources. The kinds of resource utilizations are dimensions in this vector packing problem.

2. Related work

Cloud datacenters need efficient resource management that is able to orchestrate bulks of different resources. In current public cloud datacenters, there is a mismatch between the requirements of tenant requests and resource utilization. Multiple resource types in datacenters make the situation even more complex (Gao et al., 2013). As virtual machines dynamically enter and leave a cloud system, it becomes necessary to relocate virtual machines among servers. However, relocation of virtual machines introduces run-time overheads and consumes extra energy, thus a careful planning for relocation is necessary. A virtual machines can be deployed in any physical server that supports virtualization, therefore the deployment becomes very flexible, and easy to manage (Ho et al., 2011). Due to the highly dynamic heterogeneity of resources on cloud computing platform, virtual machines must adapt to the cloud computing environment dynamically so as to achieve its best performance by fully using its service and resources.

* Corresponding Author.
Email Address: ngenzialex@gmail.com (A. Ngenzi)

But in order to improve resource utility, resources must be properly allocated and load balancing must be guaranteed (Gu et al., 2012). Consolidating multiple underutilized servers into a fewer number of non-dedicated servers that can host multiple applications is an effective tool for businesses to enhance their returns on investment. The problem can be modeled as a variant of the bin packing problem where items to be added are the servers being consolidated and bins are the target servers (Gupta et al., 2008). Resources like CPU, Memory and Hard disk need to be kept minimum to avoid unnecessary costs. Autonomic resource management could lead to efficient resource utilization and fast response in the presence changing workloads. The objective is to switch on those servers which are required and turn off those not required. Live migration of VMs is an essential tool for the management of Cloud Computing resources (Devi, 2016). Cloud computing resource covers all useful entities which can be used through the cloud platform, including computer software, computer hardware, equipment, instrument and so on. Cloud computing resources is decided by the characteristics of cloud computing resource management system, which should have functions and features like: hiding the heterogeneity of cloud computing resource, providing users with the unified access interface, Shielding the dynamic of cloud computing resources, evaluating and estimating the performance of each resource, guaranteeing to meet the service quality of the user request (QoS), a careful review of the user's request of cloud computing and ensuring the security of cloud computing (Yuan and Liu, 2011). As the prevalence of Cloud computing continues to grow, the need for resource management within the infrastructure layer also increases. The provisioning of the Cloud infrastructure in the data center (DC) is a fundamental prerequisite. This is followed by how well resources are allocated, migrated and managed. Also, dynamic usage patterns of the user, location and geographical distribution of DC, availability of Internet and Cloud service adaptation are all key factors in Cloud resource management (Younge et al., 2010). As the prevalence of Cloud computing continues to grow, the need for resource management within the infrastructure layer also increases. The provisioning of the Cloud infrastructure in the data center (DC) is a fundamental prerequisite. This is followed by how well resources are allocated, migrated and managed. Also, dynamic usage patterns of the user, location and geographical distribution of DC, availability of Internet and Cloud service adaptation are all key factors in Cloud resource management. Resource management is one of the main services that both providers and customers must ensure (Grant and Eluwole, 2013). The minimization of resources, both in CPU power and memory usage, brings benefits for both actors. Minimizing the amount of hardware resource and power consumption in use is one of the main services that such a cloud infrastructure must ensure. This objective can be done either by the customer at the application level (by dynamically sizing the application based on the workload), or by the provider at the virtualization level (by consolidating virtual machines based on the infrastructure's utilization rate) (Xia et al., 2010). There are many ways to provide resource management controls in a cloud environment. VMware's Distributed Resource Scheduler (DRS) for example has a rich set of controls providing the services needed for successful multi-resource management while providing differentiated QoS to groups of VMs, albeit at a small scale compared to typical cloud deployments (Song et al., 2009).The modeling of the relocation problem as a modified bin packing problem and proposing a new server consolidation algorithm that guarantees server consolidation with bounded relocation costs can also be taken into consideration. The conduction of a detailed analysis on the complexity of the server consolidation problem, and giving an upper bound on the cost of relocation is also important; finally, the conduction of simulations and comparison of server consolidation algorithm with other relocation methods, like First Fit and Best Fit method (Ho et al., 2011). The characteristics of bin packing application are the necessity to pack or fit a collection of objects into well-defined regions so that they do not overlap. From engineering point of view the problem is normally one of making efficient use of time or space (Lodi et al., 2010).

3. Server consolidation concept

Several constraints have been taken into considerations: First, we have considered the issues related to live migration like re-allocation of tasks, migrations and so on. The live migration requires compatible utilization of simulation software like CloudSim through Java programming. This will help us to come up with configurable resources like CPU and Memory. Second, it is necessary to limit the upper bound of our resources not reaching 100% utilization threshold. This is to prevent performance degradation. Again, live migration technology consumes high CPU cycles and high server power which would result into low performance and low throughput. So, keeping CPU utilization below threshold value would result into a certain level of throughput. Choosing the right value for CPU or memory is important, as a very high threshold means that the performance of tasks running on a particular server may drop significantly, while very low threshold lowers effectiveness of consolidation. There is no fixed amount of how much percentages should be used about the optimal CPU or memory threshold values for consolidation. Hence, our experiment will take the range between 50% to 80%. Furthermore, there is a need to migrate the task from the server only if this results in releasing this server. Server consolidation modeling (Fig. 1) has some properties of Bin packing algorithm, First-First and Best-Fit. However, First-Fit and Best-Fit are targeted only at minimizing number of servers while server consolidation modeling is targeted minimizing the number of migrations as well. As it is seen, after migration servers#2 and #3 are turned off to minimize power consumption as well as resources which may be wasted.

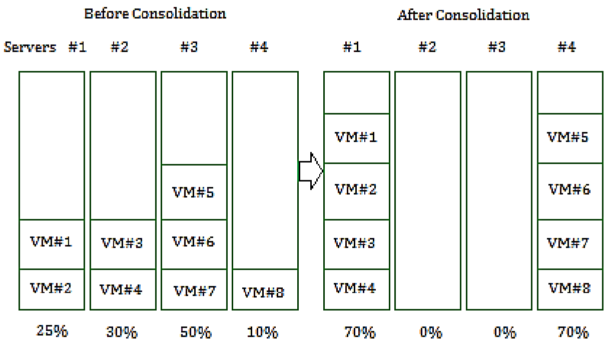

Fig. 1: Server consolidation modeling

This is aimed at minimizing the number of migrations, but in the case of migrating without releasing the server, it does not help us to achieve our objective. Rather, the first thing is to determine which tasks have to be moved to which servers, and after execution of algorithm then migrations are accomplished as per the results attained. In this research work, two dimensions can be used to characterize the task and server. Bin packing algorithm considers the first number as its effectiveness metrics. Server consolidation modeling should minimize the first and third numbers. If the target is the first number then a better result can be achieved by re allocating many tasks. This is what the bin packing problems do. However, this is not desirable due to the cost of live migration described earlier. Hence, the target possibly is the optimal result that considers the third number.

4. Proposed algorithm

The algorithm uses the best-fit concept to minimize the server cost. The algorithm iterates over a list of tasks sorted based on the CPU requirement in the descending order. So in the first iteration the algorithm gets the task with highest CPU requirement. Let's call the task in hand as current task. Once current task is decided, the algorithm tries finding the server on to which the task can be fitted exactly, means the server with exactly equal free CPU available as of the CPU requirement of the current task. If the algorithm fails to find the exact match, then it goes on to find the closest match, meaning that the algorithm will pick a server from the sorted server list that has the closest free CPU available with respect to CPU requirement of the current task. Once a closest match is found, the algorithm allocates the current task on to server. The algorithm continues by choosing the next task from the sorted task list. The algorithm starts again with finding out the exact match followed by finding out the closest match. Once there's no server that can accommodate the current task, the algorithm hands over the control to migration algorithm, which migrates the tasks across the servers to consolidate the running tasks. As the algorithm is choosing the best possible server for allocation, it prevents the wastage of free CPU available on the servers. The algorithm iterates over a list of servers sorted based free CPU available in the ascending order. Initially it chooses the first server from the sorted server list i.e., the server with least free CPU available or in other words, the server which has utilized the most of its CPU. Let's call the selected server as the target server. Once the target server is selected, the algorithm makes a list of running tasks on all servers leaving the tasks from the target server. Then algorithm sorts this list in ascending order based on the CPU requirement of the tasks. A task from this sorted task list is chosen as the task to be migrated. Once the task is chosen, the algorithm checks whether the task can be migrated to target server? If it can be migrated, then it checks whether the migration results in making enough free CPU available to allocate the current task from waiting task list? If the migration results in making enough free CPU available for allocation of current task, then the actual migration is made i.e., the task selected from sorted task list is moved to the target server and the current task allocated on to the server from where the task is moved. If the migration of selected task is not possible or if it's yielding enough free CPU for allocation of current task, then the algorithm goes on to choose the next from the sorted task list to try with migration. Like this the algorithm iterates over all the tasks from the sorted task list to find a feasible migration that results in allocation of the current task. If no feasible migration is possible, then it's concluded that there's no way to allocate the current task.

Algorithm set up

```
Input: C, S, T
where
C <- Configurable Parameters like memory, CPU
Ck <- $k^{th}$ Parameter in C
k <- 0
S <- List of all servers
T <- Task List
Ti <- $i^{th}$ Task in T

BEGIN
allocated <- false
sorted serverlist <- sort S by Ck descending
target <- Find the server with maximum CPU available
i.e., the first server in sorted serverlist
tasklist <- List all the tasks except from target
sorted tasklist <- sort T By Ck descending
foreach selectedtask belongs to sorted tasklist
if (total available Ck > Ck of selectedtask)
then if(Ck of selectedtask = Ck of selectedtask+1)
then if(Ck+1 of selectedtask > Ck+1 of selectedtask+1)
then selectedtask <- selectedtask
else selectedtask <- selectedtask+1
end if
end if
then foreach task belongs to target
```

```
if (task is migratable and Ck on target > Ck of selectedtask)
then migrate the task
allocate the selected task on to target
allocated <- TRUE
break (foreach loop)
end if
end if
end foreach

END
```

5. Discussion of results

In this experiment, we consider four servers in which tasks may be allocated or migrated. Here, Bin packing algorithm can be used to come up with configurable resources like CPU and Memory. It is necessary to limit the upper bound of our resources not reaching 100% utilization threshold. This is to prevent performance degradation. As it is explained in section IV, there is no fixed amount of how much percentages should be used about the optimal CPU or memory threshold values for consolidation. Hence, for our experiment we take the range between 40% to 70% before migration and between 70% to 100% after migration. Total utilization values have been chosen randomly for example; 70% and 40%, .This can be explained in details in the Fig. 2.

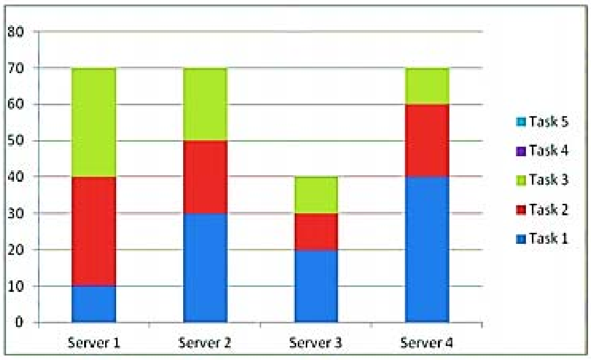

Fig. 2: Graphical presentation before migration

Experiment environment is shown in Table 1 above. We implemented simulation software using Java NetBeans programming. This software basically performs operations including generating random values according to total utilization in experiment setup and analyzing data, by displaying tables, graphs or statistics.

Table 1: Before migration

Servers	Task 1	Task 2	Task 3	Task 4	Task 5	Total
1	10	30	30	0	0	70
2	30	20	20	0	0	70
3	20	10	10	0	0	40
4	40	20	10	0	0	70

The snapshots of the simulation software can be seen in both Fig. 2 and Fig. 3. In this software, we can specify the number of tasks and the experiment values described in section V, list of tasks and servers can also be generated in both Table 1 and Table 2.

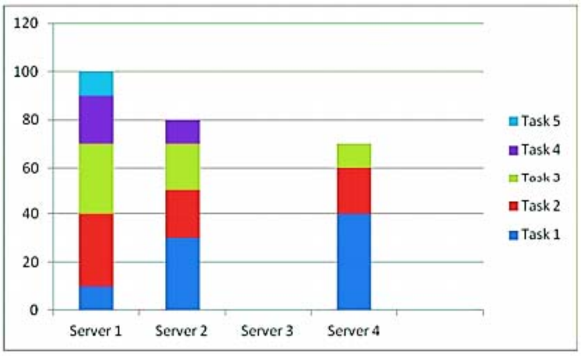

Fig. 3: Graphical presentation after migration

Table 2: After migration

Servers	Task 1	Task 2	Task 3	Task 4	Task 5	Total
1	10	30	30	20	10	100
2	30	20	20	10	0	80
3	0	0	0	0	0	0
4	40	20	10	0	0	70

As it is shown in Fig. 3 above, the resulting line graph shows allocation after migration. It is showing that server 3 is not utilized and we are able to manage with the three servers though there is a fourth server available. Our objective is to use fewer servers with efficient migration as we are doing migration we are able to fit the tasks in three servers without using the fourth. When there is no free space in servers we were migrated tasks thereby allocating resources efficiently without turning on new servers.

7. Conclusion and future findings

We applied Best fit algorithm compared to classical First Fit algorithm. We also allocated the resources to their respective bins. Furthermore, we were migrated the resources from fully utilized bins to low utilized bins and maintained load balancing. The experiments show that best fit algorithm is a suitable algorithm for packing servers with low utilizations not for high utilizations. The algorithm demonstrated that we can improve our performance more than 1000 servers sufficiently packed in their respective bins. We employed Cloudsim tool using Java programming to allocate and migrate resources. The next level of implementation will focus on how much time is required to migrate resources from one bin to another. We shall try to calculate the time delay between the servers which are the bins actually.

References

Devi AJ (2016). Green Technologies for the Energy-optimized Clouds. Asian Journal of Research in Social Sciences and Humanities, 6(6): 165-178.

Gao Y, Xue Y and Li J (2013). Utilization-Aware Allocation for Multi-Tenant Datacenters. In 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops, IEEE: 82-87.

Grant AB and Eluwole OT (2013). Cloud resource management—Virtual machines competing for limited resources. In AFRICON, 2013, IEEE: 1-7.

Gu J, Hu J, Zhao T and Sun G (2012). A new resource scheduling strategy based on genetic algorithm in cloud computing environment. Journal of Computers, 7(1): 42-52.

Gupta R, Bose SK, Sundarrajan S, Chebiyam M and Chakrabarti A (2008). A two stage heuristic algorithm for solving the server consolidation problem with item-item and bin-item incompatibility constraints. In Services Computing, 2008. SCC'08. IEEE International Conference, IEEE, 2: 39-46.

Ho Y, Liu P and Wu JJ (2011). Server consolidation algorithms with bounded migration cost and performance guarantees in cloud computing. In Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference, IEEE: 154-161.

Lodi A, Martello S, Monaci M and Vigo D (2010). Two-Dimensional Bin Packing Problems. Paradigms of Combinatorial Optimization, 2nd Edition: 107-129.

Song Y, Wang H, Li Y, Feng B and Sun Y (2009). Multi-tiered on-demand resource scheduling for VM-based data center. In Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. IEEE Computer Society: 148-155.

Xia B and Tan Z (2010). Tighter bounds of the First Fit algorithm for the bin-packing problem. Discrete Applied Mathematics, 158(15): 1668-1675.

Younge AJ, Von Laszewski G, Wang L, Lopez-Alarcon S and Carithers W (2010). Efficient resource management for cloud computing environments. In Green Computing Conference, 2010 International, IEEE: 357-364.

Yuan Y and Liu WC (2011). Efficient resource management for cloud computing. In System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2011 International Conference, IEEE, 2: 233-236.