Elasticity of Web Application for Cloud Computing Services

Sushil Deshmukh
Department Of Information Technology
RMD. Sinhgad School of Engineering
Pune, India
E-mail: sushild.may1989@gmail.com

Ms. Sweta Kale
Department Of Information Technology
RMD. Sinhgad School of Engineering
Pune, India
E-mail: swetakale30@sinhgad.edu

***Abstract* — The huge number of web application can gain advantage from elasticity property in their use of resources. As well as need to provide a fairness distribution of the resources use in cloud computing services. Now every web application instance are packed inside the virtual machine (VM) and use virtualization methodology to provide fault isolation. The modified Class Constraint Bin Packing problem can achieve good demand satisfaction ratio with developing semi-online color set algorithm as well as when load is become very low it conserve the energy by decreasing the number of server. Now further need to provide fairness distribution in differentiated services when allocating the resources across the application so model "Equivalence class or classes" for server and try to provide fairness distribution of services to premium costumer and normal user also run modified Class Constraint Bin Packing within each classes and balance the load across the server cluster adaptively as well as check that to extend the algorithm to pack the application with complimentary bottleneck resources together for different dimensions of server resource can adaptively utilized**

Keywords- CCBP, Cloud Computing, Auto Scaling, Green Computing, virtualization.

I. INTRODUCTION

Internet is the fastest and easiest source of information that can be used the huge number of user. They are using at a time and need to fastest response from server. So to provide customer service satisfaction many corporate and government industry are using cloud computing services for ex. banking sector, big MNCs and many more business organization. Now the elasticity of resource allocation is one of the cited benefits of cloud computing service. As many business customers having scale up and down their application resources usage the customer to buy as many virtual machines instance as they want to operate as like his physical hardware. Now a day cloud provide services as the user still need to decide how much resource are necessary and for how long. so that's is auto-scaling property of cloud where their resources can scale up and down automatically by cloud services will replicate the application which is uploaded by the user on single server onto more and fewer server as the user demands come and goes so user are charged only for what they actually use as called as "pay as you go" facility model.

A simple architecture of cloud computing services consist the data centers servers for the web application as well as a switch whose function is balancing the loud and distribute load to set of application server also having set of backend storage server. Switch is normally 7 layer switch whose capture application level information in web request from user and forward to them to application servers with corresponding application running. The switch sometimes runs in redundant pair for fault tolerance. As each server machine can host multiple application so it is important that application should be stateless because every application store their state information in backend storage servers, so that is why they can be replicated safely but it may cause storage servers becomes overloaded but the focus of this work is on application tire presenting a architecture is representative of a large set of internet services hosted in the cloud computing environment even through providing infinite capacity on demand.

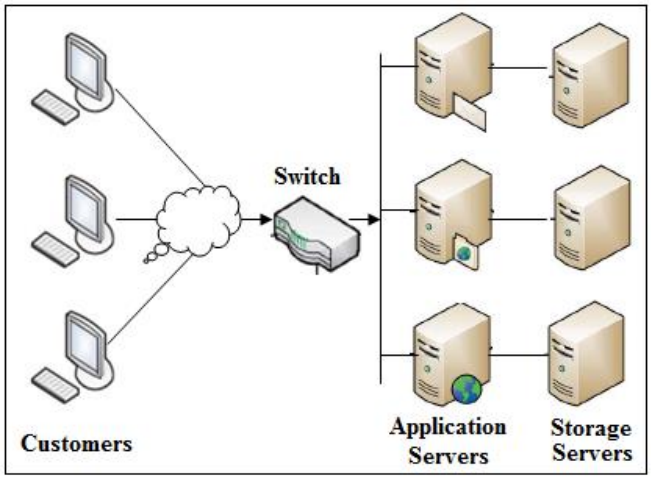

Figure 1. Architecture of Web Application in Cloud Computing.

There are no of way cloud computing provide services these are Software as a service (SaaS), Platform as a service (PaaS), Infrastructure as a service (IaaS)

Mainly the cloud provider is the company that offers the infrastructure and tool for cloud customer to host and maintain their performance of application in cloud. When we talk about scalability and performance management of web application running in software as a service (SaaS) model and platform as a service (PaaS) are the provider responsibility on the other hand when web application running on infrastructure as a service (IaaS) model are completely customer responsibility so when talk about automatic scaling of web application is completely based on SaaS model. The data center capacity in real world is finite but when large number of application access their peck demand around the same time, the available

resources in cloud becomes restricted and some of the demand satisfaction ratio or increase percentage of application demand that satisfy successfully. It defines new algorithm and modify CCBP problem and provided higher potential capacity for satisfying application demand. On the other hand when a load or demands of application is low, it is important to save the energy by reducing the number of server used.

In cloud three way of scaling is done horizontal scaling, vertical scaling, auto scaling. Auto scaling is the ability to scale up and scale down the application server's capacity automatically according to customer defines. To maintain the performance when demand is huge it increases the number of instance and decrease automatically when demand reduces to minimize cost.

The auto scaling having number of features

- It has scale out instances automatically and consistent when demand is increase.
- It covers unnecessary cloud instances automatically and save money when demand is collapse.
- It changes the delicate and impassable instances to maintain higher availability of cloud resources of your application.

It runs at on demand or spot instance.

II. LITERATURE SURVEY

The traditional bin packing problem has been deeply studied. When packing items into a minimum number of bins the vector bin packing problem considers multi-dimensional constraints [3]. It consider the memory and CPU demand requirement of web application as individual elements in the vector and use vector bin packing to solve the problem. Unfortunately, the memory requirement of web applications has to be satisfied as a whole vector, a major portion of the memory is acquired always even when the application receives small load. This is true for Java applications whose memory allocation is depend on the past load due to garbage collection. Hence, don't divide the memory requirement and satisfy by in a practical manner across the servers. There have two problem CCBP and CCMK. The CCMK is wish to maximize the total number of placed items in m bins (server) are m>1 [5]. And the CCBPs goal is to pack all item in minimum no of bins but under the restriction that each knapsack has a limited capacity and a bound on the number of different types of items it can hold. Unlike CCBP, it does not attempt to minimize the number of knapsacks used. So, it does not support green computing when the system load is low. In past a maximum number of approximation algorithms have developed for CCBP problem [1]. They are offline algorithms and online algorithm. Offline algorithm which do not support item departure. The strict online algorithms which do not allow movements of already packed items. That is case of item departure, if departed item is removed but the rest of the items in the bins are not re-packed. In this departure of item is associated with a color set, these algorithms do not maintain the property that there is one unfilled color set in the system. So the each color set is packed independently due to this can degrade the performance severely because it shown that the existing color set algorithms perform poorly in the practical of frequent item departure [2]. It cannot be applied in a cloud computing environment where the application demands change dynamically. Resource supply for Web server farms has been explained in [4]. Some clustered web form denoted that a number of different web site shared pooled resources they actually share common front end dispatcher to way perform load control and distribute user request. It as well as share back end bandwidth to return request result of user but the drawback is in cluster computing provide pooled resources which is shared all servers as it was waste of resource when load low.

In the all existing methods user goals not get satisfied because all methods having some limitations in them. So have proposed novel method to find the user goals and cluster the URLs according to the user goals.

In cloud providing services of web application the virtualization plays important role for fault isolation. It is one of the key enabling technology for cloud computing the main goal of virtualization is to improve the utilization of instance, enable fault tolerance when instance event failure, and easy to dispensation. Virtualization in computer technology is creation of virtual rather than actual. Here it create virtual machine instances for resources allocation. The virtualized resources can be accessed by devices, application, operating system, and by users. Resource virtualization can be categorized into servers, storage, and operating system.

The storage virtualization is allows transparent provisioning storage capacity and simplifies data flexibility and management. The server virtualization is using virtual machine monitor (VVM) layer running between operating system and hardware. The operating system virtualization used abstraction of operating system resource using virtualization layer and that does not runs directly on hardware. [6]This third virtualization implies higher overhead as compare to server virtualization. Due to this it can be used only for application testing but not for production environment. Here for web application of cloud it use server virtualization and also storage virtualization it will use in data centers for fault isolation. When the fault is occur during running process of application servers virtualization pin points the actual cause and location and replace VM instance by creating another VM instance and recovered failed machine quickly.

It is nothing but the bin packing algorithm solution. There are number of online bin packing algorithm CCBP is one of them it having class constraint bin packing. The class represents application which is divided into number of constraint and the bin is used for the server. Before CCBP there are any fit and next fit. Any fit algorithm can be divided into best-fit, first-fit, worst-fit, Almost-worst-fit. These entire algorithms are used for resource allocation and application placement. The best-fit allocation algorithm place applications on the servers

which has smallest block of memory in which it was fit. The idea behind this to use already loaded servers, when possible thus reduce other one for future request and therefore avoid the splitting. But this approach has negative impact on load distribution.

The first-fit allocation algorithm has advantage of using minimum time to selection or detecting the best resource to use. If there are lots of applications that can have request of resources it analyzing all it may take considerable time to choose this algorithm does not search for the best of available servers for the application allocation it chose first one that it finds. so the first-fit algorithm consider server according to order in which they opened and placed each application in the first possible bin.

In worst-fit allocation algorithm it solves the problem that found in previous two algorithms. If there are block of resources to choose particular request requirement it will use best-fit, or first-fit. If block will not match most likely to requested resources perfectly, then after allocation of application very small block of resources are left as unused this block is very small for other request thus it will goes to fragmentation and fragmentation is done in to the CCBP that each application will divided into number of class as constraint and packed it to the servers or bin. In worst-fit it packs every item in the least filled bin [7].

And the Average-worst-fit is close to worst –fit. If the current application constraint fits in more than one open bin then AWF choose the second least filled bin. Otherwise work like WS. Here the goal is to maximize the demand satisfaction ratio, and minimize the placement change frequency as well as minimize energy consumption. The only difference is the CCBP problem does not solve the "minimize placement change frequency" goal so there it has newly developed modified the CCBP model to give a support for minimizing placement change frequency and it provide new online semi approximation algorithm.

Now in previous CCBP the class constraint represent partial limit of number of application, and capacity of server represent amount of resources available at application server. And here size of item represents an amount of load for particular application. But most online CCBP algorithm does not support for item departure.

Now the key feature of new modified CCBP problem is support for item departure which one is essential to maintain good performance in cloud computing environment where resource demands of web application can change dynamically. Mainly CCBP problem is NP-hard problem there are number of approximation algorithm have been developed. But these entire algorithms assume that the entire input sequence of item is known in advance. But in new environment the demands or request of user can change dynamically and it will change unexpectedly

III. PROPOSED SYSTEM

Now in modified CCBP mainly focus on the two key point's one is application placement and other one is load distribution. Based on observation develop a semi-online approximation algorithm for CCBP which packs the recent item without any knowledge of any sub sequential item in list of input sequence. Here in this scenario color set algorithm is used for the label each class of item with color and arrange them in to color set as per they arrive input sequence.

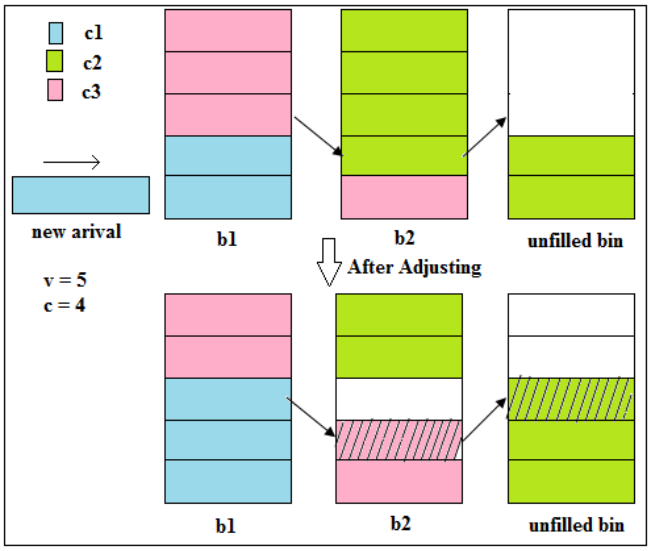

Figure 2 : Modified CCBP at arrival of new item

Item from different color set are packed independently. For packing each item in color set use greedy algorithm it is mathematical process which solve multistep problem by decide which next step will provide most obvious benefit. And item are packed into current bin until the capacity is reached. Here each color set has one unfilled bin so when new item from a specific color set arrives it packed into corresponding unfilled bin.

And if suppose all bin are full with color set then new bin is opened to board the item. Actually here the algorithm attempts to make space for new item in currently full bin by shifting some of item into unfilled bin. If the application load increase as the arrival of new item with the corresponding color, if the unfilled bin does not exist that color already then new color is to add in to bin as shown in fig 2. These movement of item is hypothetical and used only to calculate new load distribution the shifting of item one by one is make chain if we cannot find such a chain, the new color is add to unfilled bin which is starting new application instance if color set has no unfilled bin then new bin allocated.

Now the application load decrease is modeled as the departure of previously packed items. Main note is departure event is associated with number of specific color, not with specific item. Main advantage is the algorithm has freedom to select which item of particular color is to remove. And challenge is to maintain property that every color set has one unfilled bin departure working as follows.

- If in the color set does not present unfilled bin then remove any item of that color and the output bin becomes unfilled bin
- If the unfilled bin contains the departing color, so corresponding item removed directly
- In this case if need to remove an item from currently full bin then fill the hole with an item in form somewhere else

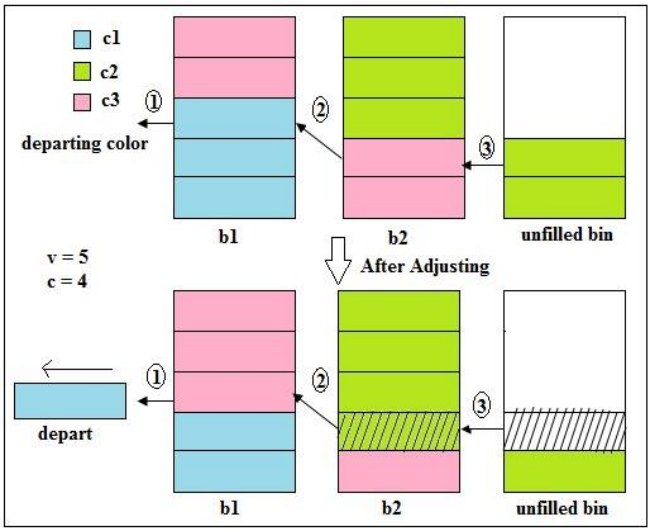

Figure 3 : Modified CCBP at departure of item

And finally last item of particular color leaves that selected color can be removed from its color set this is nothing but the closing down the last instance of an application when load reduce to zero. And color set become unfilled the challenge is here is to maintain property that there is at most one unfilled color set in the system.

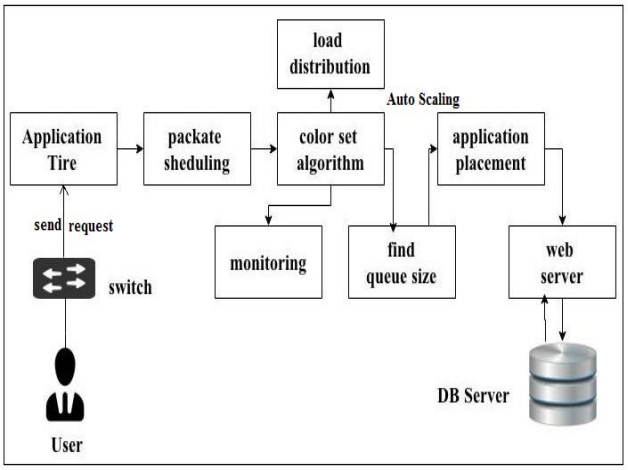

Figure 4 : System Architecture

Now in system architecture propose a packet scheduling algorithm that gives us a fairness distribution of services. First user request to service provider it get in application tire then it go application scheduler plugin in that packet scheduling algorithm is used by using this it gives fairness in distribution in job allocation or application placement. In that application placement at every instance color set algorithm is running for load distribution after all this monitoring concept completed application is going to placement at the server for processing in that processing need to known queue size in packet allocation. After allocation auto scaling is used and managed huge load by scale up and scale down resources and servers.

So the goals are to maximize the demand satisfaction ratio, minimize the placement change frequency and minimize energy consumption. Our optimization objectives can be expressed as follows.

$$(i)\quad \text{Maximize} \quad \sum_{a\in A}\sum_{s\in S} L_{a,s},$$

$$(ii)\quad \text{Minimize} \quad \sum_{a\in A}\sum_{s\in S} |P_{a,s} - P^*_{a,s}|,$$

$$(iii)\quad \text{Minimize} \quad E * |\{s \mid \sum_{a\in A} P_{a,s} > 0\}|,$$

Now S be the consider as server set on which we need to run set of application (denoted as A). The CPU capacity of server s ($s \epsilon S$) is C_S the maximum number of applications instance which can run on server s simultaneously according to memory factor M_S, and CPU demand of application a ($a \epsilon A$) is c_a. Let P be the application placement matrix ($P_{a,s}$ =1 means that application a as instance running on server s, otherwise $P_{a,s}$ = 0) and L to be application load ($L_{a,s}$ is CPU resources allocated on server s for application a) E is the energy consumption of an active server during design interval. Then current application placement matrix P^*, the predicted demand of each application (C_a) and CPU and memory resource capacity of each server (C_s and M_s) The output contains new application matrix P and load distribution matrix L.

IV. MODULE EVALUATION

A.Load Dispature

The load increase of an application is modelled as the arrival of items with the corresponding colour. Naive algorithm is to always pack the item into the unfilled bin. To make room for the new item in a currently full bin by shifting some of its items into the unfilled bin. The load decrease of an application is modelled as the departure of previously packed items

Increase the load of one application dramatically to emulate a “flash crowd” event while keeping the load of the other applications steady initially, the load in the system is low and only a small number of servers are used

B. Load Scheduler

To invoke our algorithm periodically or when the load changes cross certain thresholds. Highly efficient and can scale to tens of thousands of servers and applications. The amount of load change during a decision interval may

correspond to the arrivals or departures of several items in a row. A large load unit reduces the overhead of our algorithm because the same amount of load change can be represented by fewer items.

C. Class Constrain Bin Packing

Physical memory is typically the bottleneck resource that limits how many applications a server can run simultaneously. Need to run enough applications to drive the server busy. Aggregate load of applications in a color set is high. CPU demand relative to the memory consumption is high. Convert a block storage outline to an aggregate storage outline, and create aggregate storage application to contain the converted database and outline. Create an aggregate storage application and database. The aggregate storage outline is created automatically when you create the database.

D. Server Equivalence

Servers in large data centers are often acquired in large batches where each batch contains a large number of servers with identical hardware. Divide the servers into "equivalence classes" based on their hardware settings and run our algorithm within each equivalence class assumes that the items have the same unit size. This is not a restriction in practice because the item size is used as the load unit to represent a certain percentage of server capacity.

E. VM Specification

Internet applications can benefit from an automatic scaling property where their resource usage can be scaled up and down automatically by the cloud service. Encapsulate each application instance inside a virtual machine (VM) and use virtualization technology to provide fault isolation

V. ALGORITHMS

A.Auto Scaling Algorithm

Managing cloud computing elasticity is typically a per-application task and it implies mapping performance requirements to the underlying available resources. This process of adapting resources to the on-demand requirements of an application, called scaling, can be very challenging. Resource under-provisioning will inevitably hurt performance and create SLO violations, while resource over-provisioning can result in idle instances, thereby incurring unnecessary costs. The first thought could lead us to plan capacity for the average load or for the peak load. When planned for the average load, there is less cost incurred, but performance will be a problem if peak s of load occurs. Bad performance will discourage customers, and revenue will be affected. On the other hand if capacity is planned for peak workload, resources will remain idle most of the time

B. Semi Online Algorithm

The new algorithm is based on the algorithm for online scheduling from. In this section we present the algorithm and the proof of its optimality with emphasis on the issues that need to be handled differently in the more general semi-online setting. And then this algorithm is used to rectify the CCBP Problem.

D. Naive Algorithm

A naive Bayes classifier considers each value of particular features to contribute independently on the probability regardless of the presence or absence of the other features. It gives the class variable for some types of probability models, naive Bayes classifiers can be explained very efficiently in a supervised learning setting. In many practical applications, estimation of a parameter for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

E. Packet Scheduling

This packet scheduling algorithm is used for fairness distribution of application to server.

C. Greedy Algorithm

This algorithm is used to overcome traffic.

VI. CONCLUSION

For the load distribution and application placement developed a color set algorithm. High satisfaction ratio of application demand even in the high load. And when load is low it save energy by reducing the instances which is running on servers. If resources are become full. The provider may want to give their services to premium customers with a higher demand satisfaction ratio than other normal customers. So need to extend the system to support differentiated services but also consider fairness when allocating the resources across the applications by using packet scheduling algorithm. CCBP problem is well working at the time of load aggregation of applications in a color set is high. Future work is to develop a supporting algorithm for the distribution of incoming requests from user among the set of equivalence classes as well as balance the load across those servers adaptively.

ACKNOWLEDGMENT

First and foremost, I would like to express my sincere gratitude to my guide Ms. Sweta Kale, for her continuous support and head of the department Ms. D.T. Kurian in completion of my project stage-I. I will always be grateful for her patience, motivation, enthusiasm, and immense knowledge. This work is an extension of the previous research efforts of Zhen Xiao, Qi Chen, and Haipeng Luo. I am very much indebted to them for their inspiring work which is a boost for future researchers. Finally, I would like to pay my respect and love to my parents, for their encouragement throughout my career.

REFERENCES

[1] L. Epstein, C. Imreh, and A. Levin, Class constrained bin packing revisited, Theor. Comput. Sci., vol. 411, no. 3436, pp. 30733089, 2010.

[2] H. Shachnai and T. Tamir, Tight bounds for online class constrained packing, Theor. Comput. Sci., vol. 321, no. 1, pp. 103123, 2004

[3] C. Chekuri and S. Khanna, On multidimensional packing problems, SIAM J. Comput., vol. 33, no. 1, pp. 837851, 2004.

[4] J. L. Wolf and P. S. Yu, On balancing the load in a clustered web farm, ACM Trans. Internet Technol., vol. 1, no. 2, pp. 231261, 2001.

[5] H. Shachnai and T. Tamir, On two class-constrained versions of the multiple knapsack problem, Algorithmica, vol. 29, no. 3, pp. 442467, 2001.

[6] M.Kriushanth1, L. Arockiam2 and G. Justy Mirobi3 Research Scholar, Department of Computer Science, St. Joseph"s College (Autonomous) Tiruchirappalli, Tamilnadu1 Associate Professor in Computer Science, St. Joseph"s College (Autonomous), Tiruchirappalli, Tamilnadu, India "Auto Scaling in Cloud Computing: *An Overview",* International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 7, July 2013

[7] Leah Epstein1, Lene M. Favrholdt2, and Jens S. Kohrt2 1 Department of Mathematics, University of Haifa, Israel, 2 Department of Mathematics and Computer Science, University of Southern Denmark, Odense, Denmark, "Comparing Online Algorithms for Bin Packing Problems".