International Journal on Applications in Information and Communication Engineering
Volume 1: Issue 6: June 2015, pp 46-48. www.aetsjournal.com ISSN (Online) : 2394-6237

A Virtual Matching Resource Framework In The Cloud Computing

Alex David S, Durai. S ,Thanjai Vadivel M

***Abstract*—** A general resource management architecture that uses the constant matching framework to decouple policies from mechanisms when mapping virtual machines to physical servers. Secure, clients and operators are able to express a variety of distinct resource management policies as they deem fit and these policies are captured as preferences in the constant matching framework. The highlight of thisnew many-to-one constant matching theory that efficiently matches VMs with heterogeneous resource needs to servers, using both offline and online algorithms. Theoretical analyses show the convergence and optimality of the algorithm. Experiments with a prototype implementation on a 30-node server cluster, as well as large-scale simulations based on real-world workload traces, demonstrate that the architecture is able to realize a diverse set of policy objectives with better performance and practically.

Indexterms- CloudSharing, Resourcemanagement, constant matching , Virtual Machine placement.

I. INTRODUCTION

Cloud is a type of parallel and distributed system which consists of a collection of interconnected and virtualized computers. These computers are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements, which are established through negotiation between the service provider and consumers.

The computing resources can be allocated dynamically upon the requirements and preferences of user. The consumers may access applications and data on the Cloud from anywhere at any time, it is difficult for the cloud service providers to allocate the cloud resources dynamically and efficiently. Physical resource are Computer, Processor, disk, database, network, Bandwidth, scientific instruments and the logical resources are Execution, monitoring, communicate application and etc.

Dynamic allocation of tasks to computers is complicated in the cloud computing environment due to the complicated process of assigning multiple copies of the same task to different computers. Likewise, the resource allocation is also a big challenge in cloud computing. Cloud computing not only enables users to migrate their data and computation.remote location with minimal impact on system performance, but also ensures easy access to acloud computing environment to visit their data and obtain the computation at anytime and anywhere. Cloud computing is attempting to provide cheap and easy access to measurableand billable computational resources when compared withother paradigmssuch as Distribute Computing, Grid Computing, etc. In a cloud computing environment, the tasks are distributed across distinct computational nodes. In order to allocate cloud computing resources, nodes with spare computing power are detected and network bandwidth, line quality, response time, task costs, and reliability of resource allocation are analyze the quality of cloud resources such as network bandwidth, complete time, task costs, and reliability, etc.

The primary contributions of this work are:

• Introduce a Service-level-agreement (SLA) model that map application performance requirements to resource demand requirement. A systematic method to estimate the total amount of capacity for provisioning multiplexed VMs. The estimated aggregate capacity ensures that the SLAs for individualVMs are still preserved.

•VM selection algorithm that seeks to find those VMs with the most compatible demand patterns. The identified VM combinations lead to high capacity savings if they are multiplexed and provisioned together.

It illustrates effective and feasible applications of the proposed technique for capacity planning and for providing resource guarantees via VM reservations. Both applications can be easily employed in existing cloud and virtualization management infrastructures with minimal intrusion and substantial benefits in return.

A. System Model

Anchor, a new architecture that decouples policies from mechanisms for cloud resource management. This is analogous to the design of BGP, where ISPs are given the freedom to express their policies, and the routing mechanism is able to efficiently accommodate them. Anchor consists of three components: a resource monitor, a policy manager, and a matching engine, as shown in Both the operator and its clients are able to configure their resource management policies, based on performance, cost, etc., as they deem fit via the policy manager. When VM placement requests arrive, the policy manager polls information from the resource monitor, and feeds it with the policies to the matching engine. The matching mechanism resolves conflicts of interest among stakeholders, and outputs a matching between VMs and servers.

Alex David S is Assistant Professor, Department of CSE, Vel Tech University, (Email : adrtel@gmail.com)

Durai. S is Assistant Professor, Department of CSE, Vel Tech University, (Email : duraitrichy@gmail.com)

Thanjai Vadivel M is Assistant Professor, Department of CSE, Vel Tech University, (Email : thanjaivadivel@gmail.com)

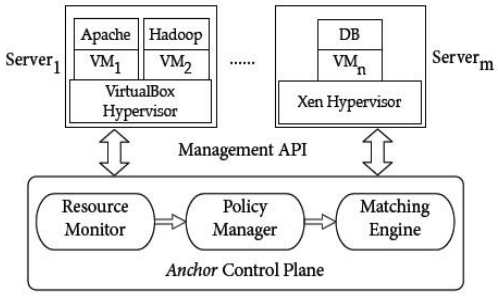

Fig.1 Anchor Architecture

The challenge of Anchor is then to design an expressive, fair, and efficient matching mechanism discussed. Second major contribution is a novel matching mechanism based on the stable matching framework from economics, which elegantly achieves all the design objectives.Rather than optimality, stability is used as the central solution concept to address the conflicts of interestamong stakeholders, fulfilling the fairness requirement. ts algorithmic implementations based on the classical deferred acceptance algorithm (DA) have been demonstrated to be practical in many real-world applications, fulfilling the efficiency requirement.

The novelty of our stable matching mechanism lies in a rigorous treatment of size heterogeneity specifically; classical stable matching theory cannot be directly applied here. Each VM has a different size corresponding to its demand for CPU, memory, and storage resources. General job-machine stable matching problem with size heterogeneous jobs.The ambiguity of the conventional stability propose a new stability concept, develop algorithms to efficiently find stable matchings with respect to the new definition, and prove convergence and optimality results.

B.Individual VM Capacity Requirements

Aggregate capacity requirements for Multiplexed VMs side, and the possibility of being unmatched .Preferences can be represented as rank order lists of the form p (m1) = w4, w2, . . . ,wi, meaning that man m1's first choice of partner is w4, second choice is w2 and so on, until at some point he prefers to be unmatched It is use to denote the ordering relationship of agent i. If i prefers to remain unmatchedinsteadof being matched to agent j, i.e. !i j, j is said to be unacceptable to it.

II. SYSTEM IMPLEMENTATION

The future resource needs of VMs. As said earlier, our focus is on Internet applications. One solution is to look inside a VMfor application level statistics, e.g., by parsing logs of pending requests. Doing so requires modification of the VM which may not always be possible. Instead make our prediction based on the past external behaviors of VMs. Our first attempt was to calculate an exponentially weighted moving average (EWMA) using a TCP- like scheme

$E(t) = \propto * E(t-1) + (1- \propto) * 0(t), 0 <= \propto <= 1,$

where E(t) and O(t) are the estimated and the observed load at time t, respectively. $\propto$ reflects a tradeoffbetween stability and responsiveness.EWMA formula to predict the CPU load on the DNS server load every minute and predict the load in the next minute. Each dot in the figure is an observed value and the curve represents the predicted values. Visually, the curve cuts through the middle of the dots which indicates a fairly accurate prediction. The parameters in the parenthesis are the values. W is the length of the measurement window

A job-machine pair (j, m) is a blocking pair if any of the two conditions holds:

$$\text{(a):} \quad c(m) \geq s(j), j \succ_m \emptyset, \text{ and } m \succ_j \mu(j), \qquad (1)$$

$$\text{(b):} \quad c(m) < s(j), c(m) + \sum_{j'} s(j') \geq s(j),$$

$$\text{where } j' \prec_m j, j' \in \mu(m), \text{ and } m \succ_j \mu(j). \qquad (2)$$

$E(t) = -|\propto| * E(t-1) + (1+ |\propto|)*O(t) = O(t)+|\propto|*(O(t)-E(t-1))$ (3)

A . Job Machine Stable Matching

A job-machine matching is strongly stable if it does not contain any blocking pair.

Table.1

stronglyconstant matching

p(j)	j	s(j)	c(m)	m	p(m)
A	a	2	2	A	c, a, b
A, B	b	1	1	B	b, c
B, A	c	1			

B.The Skewness Algorithm

Introduce the concept of skewness to quantify the unevenness in the utilization ofmultiple resources on a server. Let n be the number of resources and the utilization of the ith resource. Define the resource skewness of a server p skewness

$$(p) = \sqrt{\sum_{i=1}^{n} \left(\frac{r_i}{\bar{r}} - 1\right)^2}, \quad \rightarrow 1$$

Where r is the average utilization of all resources for server p.

I. Constant Matching

A large body of research in economics has examined variants of stable matching and references therein. Algorithmic aspects of stable matching have also been studied in computer science.The use of stable matching in networking is fairly limited.Uses the DA algorithm to solve the coupled placement of VMs in datacenters.Recent work advocate Constant matching as a general framework to solve networking problems.All these works assume a traditional uni-size job model.

II. VM Placement

Existing papers develop specifically crafted algorithms

and systems for specific scenarios, such as consolidation based on CPU usage energy consumption bandwidth multiplexing and storage dependence. They are thus complementary to Anchor, as the insights and strategies can be incorporated as policies to serve different purposes

III. .Hot and Cold Spots

Our algorithm executes periodically to evaluate the resource allocation status based on the predicted future resource demands of VMs. Define a server as a hot spot if the utilization of any of its resources is above a hot threshold. This indicates that the server is overloaded and hence some VMs running on it should be migrated away. Define the temperature of a hot spot p as the square sum of its resource utilization beyond the hot threshold:

$$\text{temprature}(p)=\sum_{r\in R}(r\text{-}rt)^2, \quad (4)$$

Where R is the set of overloaded resources in server p and rt is the hot threshold for resource r. (Note that only overloaded resources are considered in the calculation.) The temperature of a hot reflects

If a server is not a hot spot, its temperature is zero. .Define a server as a cold spot if the utilizations of all its resources are below a cold threshold. This indicates that the server is mostly idle and a potential candidate to turn off to save energy. However, do so only when the average resource utilization of all actively used servers (i.e., APMs) in the system is below a green computing threshold.

A server is actively used if it has at least one VM running. Otherwise, it is inactive. Finally, Define the warm threshold to be a level of resource utilization that is sufficiently high to justify having the server running but not so high as to risk becoming a hot spot in the face of temporary fluctuation of application resource demands.

Different types of resources can have different thresholds. For example,define the hot thresholds for CPU and memory resources to be 90 and 80 percent, respectively. Thus a server is a hot spot if either its CPU usage is above 90 percent or its memory usage is above 80 percent.

IV. Hot Spot Mitigation

The list of hot spots in the system in descending temperature. To eliminate all hot spots if possible. Otherwise, keep their temperature as low as possible. For each server p, first decide which of its VMs should be migrated away.Sort its list the resulting temperature of the server if that VM is migrated away. Aim to migrate away the VM that can reduce the server's temperature the most. In case of ties, select the VM whose removal can reduce the skewness of the server the most. For each VM in the list, see if can find a destination server to accommodate it.

The server must not become a hot spot after accepting this VM. Among all such servers, select one whose skewness can be reduced the most by accepting this VM. Note that this reduction can be negative.

III. GREEN COMPUTING

When the resource utilization of active servers is too low, some of them can be turned off to save energy. This is handled in our green computing algorithm. The challenge here is to reduce the number of active servers during low load without sacrificing performance either now or in the future. Need to avoid oscillation in the system. Our green computing algorithm is invoked when the average utilizations of all resources on active servers are below the green computing threshold.The list of cold spots in the system based on the ascending order of their memory size.The warm threshold is designed to prevent that. If multiple servers satisfy the above criterion, prefer one that is not a current cold spot. This is because increasing load on a cold spot reduces the likelihood that it can be eliminated. However, will accept a cold spot as the destination server if necessary. All things being equal, canselect a destination server whose skewness can be reduced the most by accepting this VM. If can find destination servers for all VMs on a cold spot, record the sequence of migrations and update the predicted load of related servers. Otherwise, do not migrate any of its VMs.

IV. CONCLUSION

The presented design implementation, and evaluation of a resource management system for cloud computing services. System multiplexes virtual to physical resources adaptively based on the changing demand. Use the skewness metric to combine VMs with different resource characteristics appropriately so that the capacities of servers are well utilized. Algorithm achieves both overload avoidance and green computing for systems with multiresource constraints.

REFERENCES

[1] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.
[2] L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, Oct. 2008.
[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," Proc. ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.
[4] "Amazon elastic compute cloud (Amazon EC2)," http://aws. amazon.com/ec2/, 2012.
[5] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," Proc. Symp. Networked Systems Design and Implementation (NSDI '05), May 2005.
[6] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," Proc. USENIX Ann. Technical Conf., 2005.
[7] M. McNett, D. Gupta, A. Vahdat, and G.M. Voelker, "Usher: An Extensible Framework for Managing Clusters of Virtual Machines,"