ACTIVE RESOURCES ALLOCATION FOR CLOUD VIRTUAL ENVIRONMENTS USING EASJSA

Mrs.S.ANITHA, MCA., M.Phil.,[1]
ASSISTANT PROFESSOR,
selvianithas@gmail.com,

S.GAYATHRI,[2]
M.PHIL FULL-TIME RESEARCH SCHOLAR,
Gayushanmugam2@gmail.com, 9677665858.

DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS,
VIVEKANANDHA COLLEGE OF ARTS AND SCIENCES FOR WOMEN, TAMILNADU, INDIA.

Abstract— Cloud computing allows business customers to scale up and down their resource usage based on needs. Many of the touted gains in the cloud model come from resource multiplexing through virtualization technology. This project presents a system that uses virtualization technology to allocate data center resources dynamically based on application demands and support green computing by optimizing the number of servers in use. The existing system is “skewness” model introduced to measure the unevenness in the multidimensional resource utilization of a server. By minimizing skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources. It develops a set of heuristics that prevent overload in the system effectively while saving energy used. In addition, this paper proposes an Enhanced Adaptive Scoring Job Scheduling Algorithm (EASJSA) for the cloud environment. Compared to other methods, it can decrease the completion time of submitted jobs, which may compose of computing-intensive jobs and data-intensive jobs.

Keywords— Cloud Computing, Virtualization Technology, Data-intensive jobs, EASJSA, Multidimensional Resource Utilization

I. INTRODUCTION

Studies have found that servers in many existing data centers are often severely underutilized due to over provisioning for the peak demand. The cloud model is expected to make such practice unnecessary by offering automatic scale up and down in response to load variation. Besides reducing the hardware cost, it also saves on electricity which contributes to a significant portion of the operational expenses in large data centers. Virtual machine monitors (VMMs) like Xen provide a mechanism for mapping virtual machines (VMs) to physical resources [1]. This mapping is largely hidden from the cloud users. Users with the Amazon EC2 service, for example, do not know where their VM instances run. It is up to the cloud provider to make sure the underlying physical machines (PMs) have sufficient resources to meet their needs.

VM live migration technology makes it possible to change the mapping between VMs and PMs while applications are running. However, a policy issue remains as how to decide the mapping adaptively so that the resource demands of VMs are met while the number of PMs used is minimized. The paper considers a system which introduces the concept of “skewness” to measure the unevenness in the multidimensional resource utilization of a server [2]. By minimizing skewness, it can combine different types of workloads nicely and improve the overall utilization of server resources. We develop a set of heuristics that prevent overload in the system effectively while saving energy used.

The rest of this paper is organized as follows. Section 2 presents the related work followed by the main contribution dynamic resource allocation as well as the problem definition in Section 3. Section 4 gives a brief introduction to Enhanced Adaptive Scoring Job Scheduling algorithm while and explains the proposed approach. Finally, Section 5 presents the evaluation of the algorithm followed by the conclusions and future work described in Section 6 and Section 7.

II. RELATED WORK

Data centers server farms that run networked applications—have become popular in a variety of domains such as web hosting, enterprise systems, and e-commerce sites. Server resources in a data center are multiplexed across multiple applications each server runs one or more applications and application components may be distributed across multiple servers [3]. Further, each application sees dynamic workload fluctuations caused by incremental growth, time-of-day effects, and flash crowds. Since applications need to operate above a certain performance level specified in terms of a service level agreement, effective management of data center resources while meeting SLAs is a complex task [7]. One possible approach for reducing management complexity is to employ virtualization. In this approach, applications run on virtual servers that are constructed using virtual machines, and one or more virtual servers are mapped onto each physical server in the system.

Virtualization of data center resources provides numerous benefits [4]. It enables application isolation since malicious or greedy applications can not impact other applications co-located on the same physical server. It enables server consolidation and provides better multiplexing of data center resources across applications. Perhaps the biggest advantage of employing virtualization is the ability to flexibly remap physical resources to virtual servers in order to handle workload dynamics. Data center energy savings can come from a number of places: on the hardware and facility side, e.g., by designing energy efficient servers and data center infrastructures, and on the software side, e.g., through resource management [8]. In this paper, we take a software-based approach, consisting of two interdependent techniques: dynamic provisioning that dynamically turns on a minimum number of servers required to satisfy application specific quality of service, and load dispatching that distributes current load among the running machines [5].

III. MAIN CONTRIBUTIONS

The main contribution for solve problem of existing system is develops a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used. It introduces the concept of “skewness” to measure the uneven utilization of a server. By minimizing skewness, it can improve the overall utilization of servers in the face of multidimensional resource constraints [6]. It designs a load prediction algorithm that can capture the future resource usages of applications accurately. The algorithm can capture the rising trend of resource usage patterns and help reduce the placement churn significantly. It looks inside a VM for application level statistics, e.g., by parsing logs of pending requests. Doing so requires modification of the VM which may not always be possible.

- To split the Jobs are into sub tasks and assign them to more cloud nodes.
- To take in to account both dependent tasks and independent task scheduling.
- To consider job replication strategy.
- To decrease completion time of jobs.

IV. PROPOSED PROTOCOL

The proposed system covers all the existing system approach. In addition, among all the cloud nodes, Enhanced Adaptive Scoring Job Scheduling algorithm (EASJS) is applied for cloud nodes resource scheduling so that the given job is split into „N‟ tasks along with Replication Strategy.

Enhance Adaptive Scoring Job Scheduling (EASJSA) aims to decrease job”s completion time. It considers not only the computing power of each resource in the grid but also the transmission power of each cluster in a grid system. It defines the computing power of each resource, the product of CPU speed and available CPU percentage. The transmission power of each cluster is defined as the average bandwidth between different clusters. It should use the status of each resource in the grid as parameters to initialize the cluster score of all clusters.

- Each job is considered as sub tasks.
- A single job is given to a selected multiple clusters since jobs are split into tasks.
- Cluster score values are recalculated even during the job is partially completed. This is achieved when a particular sub task is finished.
- Storage capacity of cluster resources is taken into account.
- Multiple α and β and γ values are calculated for each sub task and so cluster assignment is effective than existing system.
- Replication method assists in faster job completion.

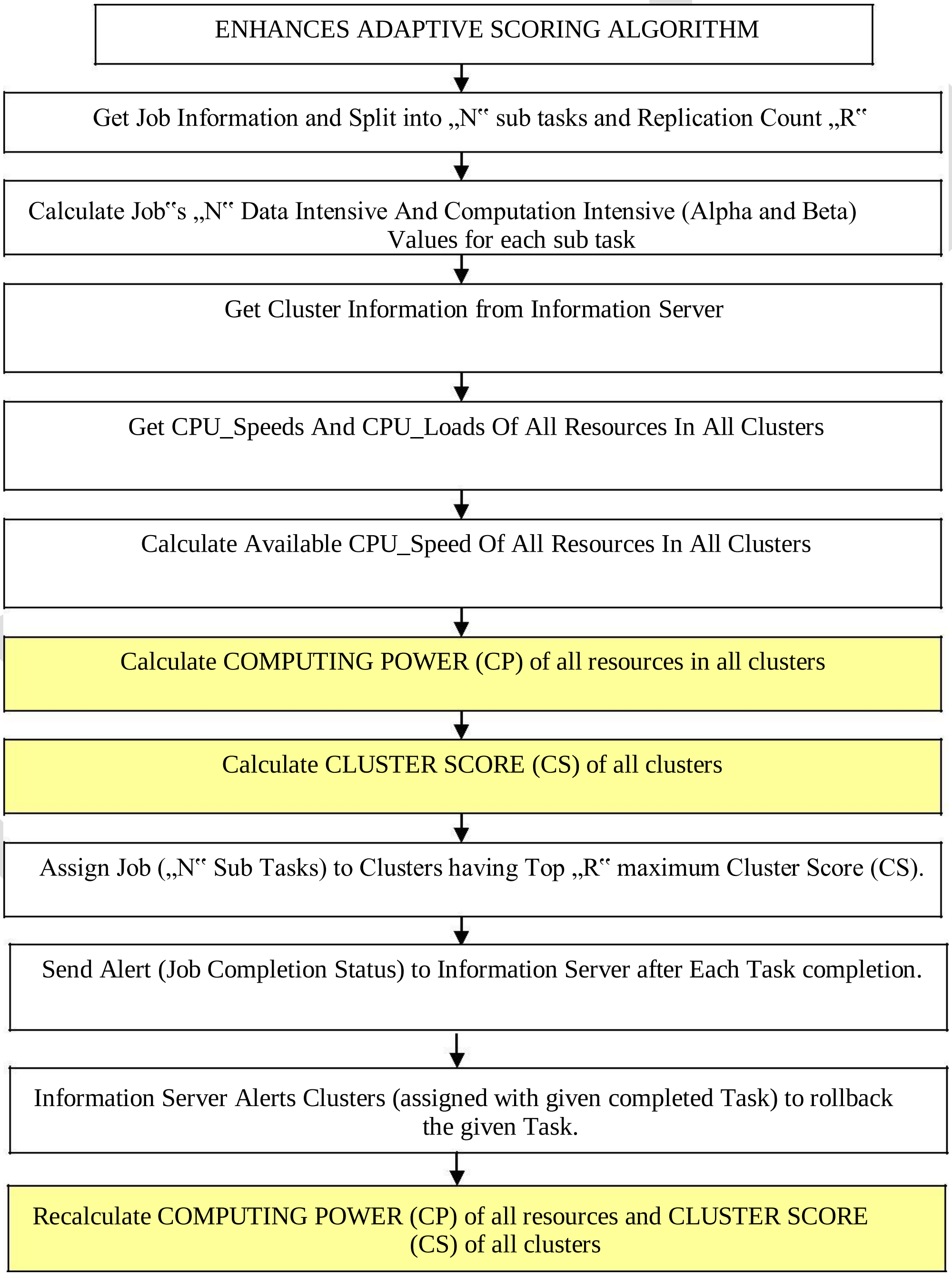
ENHANCES ADAPTIVE SCORING ALGORITHM
Get Job Information and Split into „N" sub tasks and Replication Count „R"
Calculate Job"s „N" Data Intensive And Computation Intensive (Alpha and Beta) Values for each sub task
Get Cluster Information from Information Server
Get CPU_Speeds And CPU_Loads Of All Resources In All Clusters
Calculate Available CPU_Speed Of All Resources In All Clusters
Calculate COMPUTING POWER (CP) of all resources in all clusters
Calculate CLUSTER SCORE (CS) of all clusters
Assign Job („N" Sub Tasks) to Clusters having Top „R" maximum Cluster Score (CS).
Send Alert (Job Completion Status) to Information Server after Each Task completion.
Information Server Alerts Clusters (assigned with given completed Task) to rollback the given Task.
Recalculate COMPUTING POWER (CP) of all resources and CLUSTER SCORE (CS) of all clusters

FIG 1.1 SYSTEM ARCHITECTURE

Algorithm Work:

In this module, the cluster score is calculated based on the following formula.

$$\mathbf{CS_i = \alpha \,.\, ATP_i + \beta \,.\, ACP_i + \gamma}$$

where CS_i is the cluster score for cluster i, a and b are the weight value of ATP_i and ACP_i respectively, the sum of and is 1, ATP_i and ACP_i are the average transmission power and average computing power of cluster i respectively. ATP_i means the average available bandwidth the cluster i can supply to the job and is defined as:

$$ATP_i = \frac{\sum_{j=1}^{m} Bandwidth\_available_{i,j}}{m-1}, \quad i \neq j$$

where $Bandwidth\_available_{i,j}$ is the available bandwidth between cluster i and cluster j, m is the number of clusters in the entire grid system.

Similarly, ACP_i means the average available CPU power cluster i can supply to the job and is defined as:

$$ACP_i = \frac{\sum_{k=1}^{n} CPU\_Speed_k * (1 - load_k)}{n}$$

where CPU_speed_k is the CPU speed of resource k in cluster I, load is the current load of the resource k in cluster i, n is the number of resources in cluster i. Also let

$$CP_k = CPU\_Speed_k * (1 - load_k)$$

CP_k indicates the available computing power of resource k.

Because the transmission power and the computing power of a resource will actually affect the performance of job execution, these two factors are used for job scheduling. Since the bandwidth between resources in the same cluster is usually very large, we only consider the bandwidth between different clusters. Local update and global update are used to adjust the score. After a job is submitted to a resource, the status of the resource will change and local update will be applied to adjust the cluster score of the cluster containing the resource. What local update does is to get the available CPU percentage from Information Server and recalculate the ACP, ATP and CS of the cluster. After a job is completed by a resource, global update will get information of all resources in the entire grid system and recalculate the ACP, ATP and CS of all clusters.

V. EXPERIMENTAL RESULTS

The following result finding for our experimental works, they are

- It is found that the cluster selection is efficient if the job is split into sub tasks.
- Resources are effectively utilized and waiting time is less in scheduling next successive job in queue.
- Resources with limited values are also having the chance for job allocation if the job is split into sub tasks.
- Instead of calculating the right cluster after each job completion, the proposed system calculates the clusters availability at regular intervals so that any new job can be assigned even during the execution of current job.
- Overall efficiency of the grid is more compared to existing system.
- Better suitable for jobs which can be split based on RAM, CPU speed and storage location.
- The experimental results show that EASJSA is capable of decreasing completion time of jobs and the performance of ASJS is better than other methods.
- Inter dependant jobs are not combined in the proposed system which may be future work.

Studying and improving EASJSA for such kinds of jobs may be carried out in the future

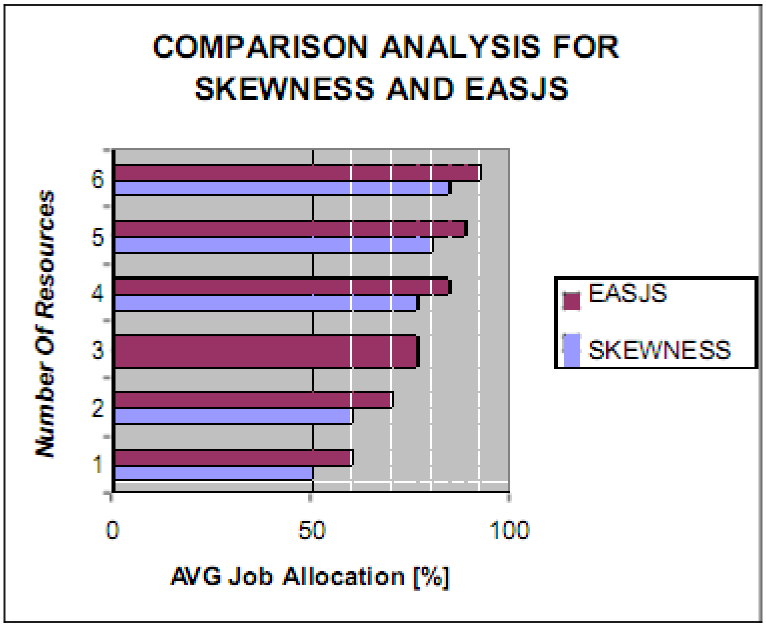

Fig 1.2 Comparisons of Skewness and EASJSA Algorithm

ACKNOWLEDGMENT

I express my deep gratitude and sincere thanks to my supervisor **Mrs.S.Anitha,MCA.,M.Phil., Assistant Professor, Department of Computer Science** at **Vivekanandha college of Arts and Sciences for women** for her valuable, suggestion, innovative ideas, constructive, criticisms and inspiring guidance had enabled me to complete the paper present work successfully.

VI.CONCLUSION

The proposed Enhanced adaptive scoring method to schedule jobs in cloud environment. EASJS selects the fittest resource to execute a job according to the status of resources. Local and global update rules are applied to get the newest status of each resource. Local update rule updates the status of the resource and cluster which are selected to execute the job after assigning the job and the Job Scheduler uses the newest information to assign the next job. Global update rule updates the status of each resource and cluster after a job is completed by a resource. It supplies the Job Scheduler the newest information of all resources and clusters such that the Job Scheduler can select the fittest resource for the next job. The experimental results show that ASJS is capable of decreasing completion time of jobs and the performance of EASJS is better than other methods

VII. FUTURE ENHANCEMENTS

In future, EASJS can be applied to real grid applications. This paper focuses on job scheduling. The paper can be modified to consider division of file and the replica strategy in data-intensive jobs. Jobs are independent in this project, but they may have some precedence relations in real-life situation. Studying and improving EASJS for such kinds of jobs may be carried out in the future.

REFERENCES

1) 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 ystems Principles (SOSP “03), Oct. 2003.

2) C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual

3) Machines," Proc . Symp. Networked Systems Design and Implementation (NSDI "05), May

4) M. Nelson, B.-H. Lim , and G . Hutchins , "Fast Transparent Migration for Virtual Machines," Proc. USENIX Ann. Technical

5) Conf., 2005

6) T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-Box and Gray-Box Strategies for Virtual Machine Migration," Proc. Symp. Networked Systems Design and Implementation (NSDI "07), Apr. 2007.

7) Chen, H. Wenbo, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-Aware Server Provisioning and Load Dispatching for Connect ion-Intensive Intern et Services," Pro c. USENIX Symp. Networked Systems Design and Implementation (NSDI "08), Apr. 2008.

8) P. Padala, K.-Y. Hou, K.G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, and A. Merchant, "Automated Control of Multiple Virtualized Resources , " Proc. ACM European conf. Computer Systems (EuroSys "09), 2009.

9) J.S. Chase, D.C. Anderson, P.N. Thakar, A.M. Vahdat, and R.P. Doyle, "Managing Energy and Server Resources in Hosting Centers," Proc. ACM Symp. Operating System Principles (SOSP "01), Oct. 2001

10) Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A Scalable Application Placement Controller for Enterprise Data Centers," Proc. Int"l World Wide Web Conf. (WWW "07), May 2007

11) M. Zaharia, A. Konwinski, A.D. Joseph, R.H. Katz, and I. Stoica, "Improving Map Reduce Performance in Heterogeneous Environments," Proc. Symp. Operating Systems Design and Implementation (OSDI "08), 2008.

12) M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, "Quincy: Fair Scheduling for Distributed Computing Clusters," Proc. ACM Symp. Operating System Principles (SOSP "09), Oct. 2009