Indian Journal of Science and Technology, Vol 9(30), DOI: 10.17485/ijst/2016/v9i30/99011, August 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Log based Automated SMI Parameter Identification and Resource Recommendations in Cloud

K. S. Guruprakash[1] and S. Siva Sathya[2]

[1]Department of Computer Science and Engineering, TRP Engineering College, NH 45, Irungalur, Mannachanallur Taluk, Tiruchirappalli - 621105, Tamil Nadu, India; ks_guruprakash@yahoo.com
[2]Department of Computer Applications, Pondicherry University, Kalapet, Puducherry – 605014, Pondicherry, India; ssivasathya@gmail.com

Abstract

Objectives: Resource provisioning is the major requirement in cloud provisioning. The major objective is to provide effective resource provisioning can improve the utility of a cloud service at reduced costs. **Methods/Analysis:** This paper presents an effective method to identify the quality parameters for effective provisioning of cloud resources. User log files are used to identify the quality parameters. It is assumed that the user migrates from a web service, cluster based service or another cloud based service. The log files from these architectures are used to map the SMI parameters and the quality values are obtained by analyzing them. **Findings:** Experiments were conducted on an access log data with 4.4 million entries and 3 million independent users. The required QoS and provided QoS were plotted and it was observed that most of the points are situated either on the diagonal or in the top left. This exhibits the efficiency of our approach to appropriately identify the user requirements and provide appropriate allocations. The ratio between the time taken for the entire process to complete and the data size was also analyzed for identifying the scalability of the system. It could be observed that as the size of the data increases, the time taken also increases. Hence the time taken is observed to be linear. **Applications/Improvement:** Identification of quality parameters were never performed with such granularity. Hence the results obtained exhibits effective quality assignments appropriate to user's needs.

Keywords: Cloud Provisioning, Resource Recommendations, SMI Parameter Mapping, Log File Based Mapping, Workload Identification

1. Introduction

Digitalization and automation has increased the usage of computers to a tremendous extent in all the industries. Hence consumers are looking for better, cheaper and portable mechanisms to automate their systems. Cloud Services is one of the best solutions provided to the consumers to solve this problem. Due to the several varieties of services provided by cloud systems, several users are migrating towards cloud[1]. The services provided by cloud includes IaaS, PaaS and SaaS. Several other services such as XaaS (All as a Service) and NaaS[2] (Network as a Service) have also been included. Cloud computing is a business model that enables utilization of a set of shared resources that can be upscaled or downscaled according to the customers requirement. The major advantage for the consumers is that any cloud service can be utilized using a single PC, hence the cost of investment is negligible[4]. A single major requirement for utilizing a cloud is that the user should specify their requirements for appropriate allocation of cloud resources. The downside of this system is that most of the users are unaware of the specification to be provided in the initial configuration stage. This leads to frequent upscales, hence elevated costs for the consumers[5]. This paper presents an effective approach that can be used to identify the configuration details by utilizing the web or cluster or cloud based log files containing the user's access patterns.

Resource provisioning[6] remains to be one of the topics of major research in cloud. An agent-based resource pro-

*Author for correspondence

visioning system for cloud is presented in[3]. This method performs automated service composition, hence providing effective request processing and automated service composition. Another major advantage of this approach is that it also considers the cost of virtual machines prior to the allocations, thus making the allocation both effective and economical. Delay is another major aspect related to provisioning. Since downtimes are not tolerated, it becomes mandatory for the provisioning algorithms to perform computations faster[7].

Strategies in the area of cloud computing and methods to optimize the objective function for users and resource providers were presented in[8,9]. SLA based service level agreements and their optimization techniques were presented in[10]. A user preference based resource provisioning strategy is presented in[11]. This method uses the user's preference and service time to predict workloads in a cloud environment. A dynamic prediction strategy is followed; hence the flexibility to adapt to various workloads provides this method an added advantage. High throughput computing based cloud provisioning is presented in[13]. This method uses the job trace of an application and applies statistical methods to it to identify the influential features of a specific application. These values are used to configure virtual machines and also aid in their appropriate deployments. Another workload based application deployment is presented in[14]. This method also predicts the future workloads, hence providing an effective learning mechanism. A similar aspect based provisioning method is presented in[15]. A specific SaaS based provisioning system is presented in[16]. This method defines the communication architecture between the user and the application provider. A metaheuristic based resource allocation techniques is presented in[17]. This technique utilizes metaheuristic techniques for resource allocations, leading to near optimal and faster solutions. Similar techniques operating on multiple criteria for the decision making process during resource provisioning is presented in[18-20].

2. Log Based Automated SMI Parameter Identification and Resource Provisioning in Cloud

Cloud service providers require appropriate parameters in order to perform resource provisioning. Since cloud resources are shared, it becomes economical to assign appropriate resources to the requested users in order to avoid resource wastage. One major disadvantage of automatic resource provisioning is that the system automatically upscales resources according to the requirements, but downscaling is not automatically performed. This leads to wastage of resources. These problems can be alleviated if the users communicate their requirements appropriately. A cloud system model is presented in Figure 1.

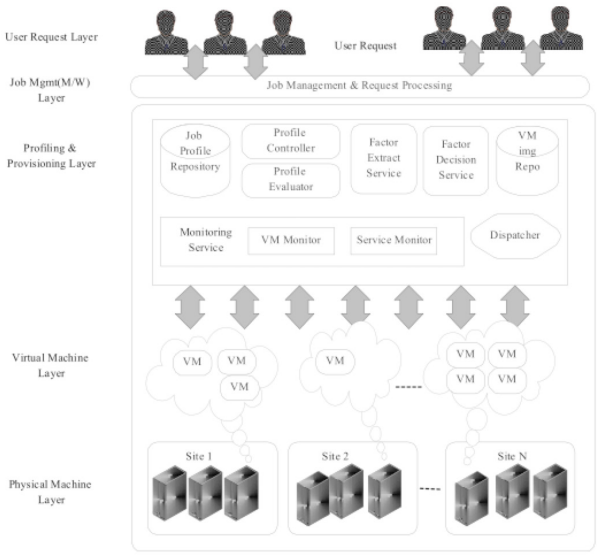

Figure 1. Cloud System Model.

The first layer is the user request layer, where users are required to provide their appropriate requests translated in terms of service requirements. The next layer processes all the received requests and manages the jobs submitted to the cloud. The third layer is the provisioning layer that profiles the requests and performs resource provisioning. The virtual machine layer manages the virtual machines and the physical machine layer corresponds to the actual hardware machinery and their management. Log based automated SMI parameter identification operates on the user request layer. This approach automates the process of resource requisition by identifying the SMI parameters from the user log files. Figure 2 presents the architecture of the log based SMI parameter identification system.

The current approach is based on an assumption that the user moves to the cloud service after either utilizing web based services or cluster based services or some other cloud based services. Hence the user has their corresponding log data. This data is used as the base for identifying

the requirements of the user. The user can also provide an additional requirement of workload details, if they are aware of the type of workload that is to be handled by the cloud. If the user does not provide log files, the workloads details become mandatory. Table 1 presents the workloads utilized for our approach and the SMI parameters that are required by the workloads. These two parameters form the mandatory entities for our approach.

Table 1. Workload Details

Workload	Details
Websites	Reliable Storage High Bandwidth High Availability
Technological Computing	Computing Capacity Reliable Storage
Online Transaction	Security High Availability Internet Accessibility Usability
E-Commerce	Variable Computational Load Customization
Storage and Backup	Reliability Persistence
Graphics Oriented	Network Bandwidth Latency Data Backup Visibility
Mobile Computing	High Availability Reliability Portability

The log files are then explored for inconsistencies and data cleaning is performed. The cleaned log entries are then passed on to the next phase for SMI parameter mapping. The entries in log files do not directly correspond to the SMI parameters. Data analysis is used to identify the corresponding parameters.

- *Bandwidth*: Network bandwidth refers to the number of bits transferred (sent/received) in a single workload per unit time (usually in seconds).

Network Bandwidth=Bits/ second (B/S) (1)

This information can be mined from the user logs by identifying the size of the data transferred and the time taken for the transfer can be identified using the timestamps associated with the log information. These data can be aggregated to identify the maximum and the minimum workload required by the application.

- *Availability*: Availability refers to the recovery level of the system in case of failure.

Availability=mean time to failure/(mean time to failure + mean time to repair) (2)

Success status codes correspond to 2xx, while failures status codes are represented by the series 4xx and 5xx. Time taken between a failure status and the first success status code (2xx) corresponds to the tolerable recovery time.

- *Computational Capacity:* Computational capacity corresponds to the ratio between the actual usage time and the expected usage time.

Computing Capacity=Actual Usage time of the Resource/Expected Usage time of the Resource. (3)

- *Usability:* Usability refers to the ratio of successful workload operations exhibited by the system.

Usability=no of successful operations in a workload/ (total operations available in the workload) (4)

- *Correctness*: Correctness defines the degree of accuracy provided to the cloud customers.

Correctness=total number of failed transmissions/ (total number of failed transmissions + total number of successful transmissions) (5)

- *Variable computing load*: It is the change in the load balance with respect to time. Calculating the variance of the workload can be used to identify this parameter.
- Reliability: Reliability refers to the time taken for the system to recover and operate successfully after a failure.

Reliability=mean time to failure + mean time to repair (6)

- Latency: Latency is the difference between the workload input and output times.

Latency = Time of output produced with respect to that Cloud workload - Time of input in a Cloud workload (7)

- Serviceability: Serviceability is the probability of the service being up and running

Serviceability= Service Uptime/ (Service Uptime+ Service Downtime) (8)

Apart from these quality parameters, other major requirements include appropriate indication of heavy and light workload times[21,22]. If the service is global, it also becomes mandatory to identify the density of virtual

machines in each geographic location. Regions exhibiting high access levels can be assigned separate VMs such that the traffic in specific regions does not interfere with the global traffic. Workload times are to be identified in order to perform effective scaling of resources. Upscaling and downscaling being major issues, identifying the regions and time of occurrence of dense or sparse traffic would help in better resource utilization[23-25].

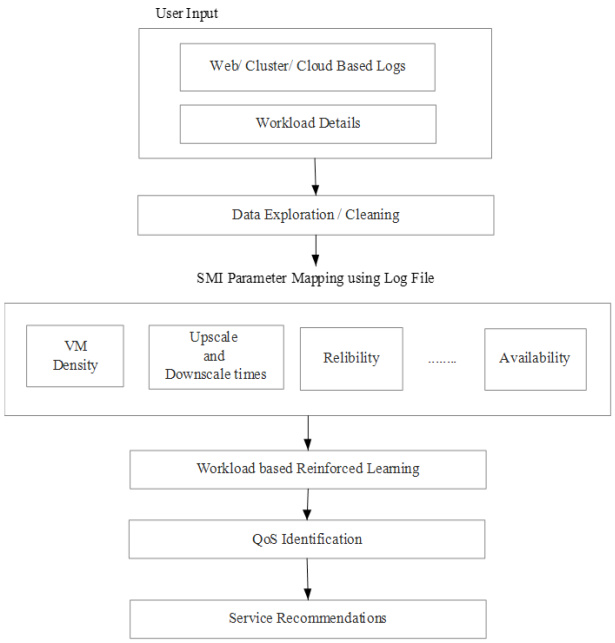

Figure 2. Log based Automated SMI Parameter Identification and Resource Provisioning in Cloud.

Distribution density and distribution location of VMs can be identified by clustering the logs according to their IP addresses and determining the workload and bandwidth details of specific clusters. Clusters depicting low workload levels are merged with similar clusters and are provided with VMs that can be shared by them. Every cluster is also checked for high and low operating times and upscaling and downscaling times are fixed accordingly. The results of this phase depict the requirements of the application, as used in the previous deployment. The current requirement might be similar or the user might have migrated for the need of better resources. Though the current usage levels can be obtained from the logs, additional requirements, if any cannot be directly obtained from the logs. Workload details are used to provide that information to the learning system. Any inputs given by the user as a probable workload[12] depicts their current requirement. This can correspond to the current logs; it might even depict a higher or a lower requirement scenario. Hence if workload information is provided, its corresponding SMI parameters are identified and the values exhibited by these parameters are provided priority over the parameters obtained from the log files. The advantage of using such a method is that a flexible and user based custom decision is made by the proposed approach. Hence custom parameters can be presented from the user request layer rather than selecting a predefined scheme from the cluster settings. This reinforcement phase has been observed to provide a more accurate and effective means of parameter identification.

The next phase deals with identifying the corresponding QoS values for the identified SMI parameters[26]. A value based parameter identification tends to result in continuous data. This data would be huge and direct mapping will not be possible. Hence fuzzy parameter settings are proposed, which uses three fuzzy distinctions (low, medium and high) of the parameter values. The service provider, depending on their resource capacities defines the distinctions and boundaries. After the identification of the fuzzy QoS requirements, the final SMI parameters are passed to the job management and request-processing layer for further processing. The algorithm for the log based automated SMI parameter mapping is presented below

Algorithm

Input :Web/ Cluster/ Cloud based Logs &Workload Details

Output : SMI Parameter Values

Let n be no.of SMI parameters

Begin

```
  //Data exploration and cleaning to eliminate
   inconsistencies and missing elements
Contents=readLine(WebLog)
  //To identify Virtual Machine Density
 Call RegionBasedCluster(Contents)
 //To identify Upscale and Downscale levels
  Call TimestampBasedCluster(Contents)
  //Combine subsequent heavy workloads
  //Identify the workload parameters
   For I =1 to n
     //Calculate other SMI parameter values using
       equations  (1) to (8)
   End For
 //Workload based reinforcement learning to modify
```

defined parameters on the basis of user's Requirements

```
//Final QoS identification and service recommendation to the user
End
Procedure Region Based Cluster (Contents)
Begin
  //Identify the number of transactions corresponding
    to the IP  in each cluster
    Ipcount= Cluster Contents on the basis of IP
   Print ipaddress : ipcount
  //Group clusters with low density values
 //Each cluster represents region based requirements
   of Virtual Machines
End
Procedure Timestamp Based Cluster (Contents)
Begin
// Identify the number of transactions corresponding to
    time
    scalecount= Cluster Contents on the basis of IP
                        address and timestamp
    Print ipaddress :( timestamp , scalecount)
End
```

```
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET / HTTP/1.1" 200 9157
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET /assets/js/lowpro.js HTTP/1.1" 200
10469
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET /assets/css/reset.css HTTP/1.1" 200
1014
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET /assets/css/960.css HTTP/1.1" 200
6206
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET /assets/css/the-associates.css HTTP/
1.1" 200 15779
10.223.157.186 - - [15/Jul/2009:15:50:35 -0700] "GET /assets/js/the-associates.js HTTP/
1.1" 200 4492
```

Figure 3. Sample Access Log Dataset.

3. Results and Discussion

Experiments were conducted by applying our algorithm on a web access log containing approximately 4.4 million entries (4,477,823) representing transactions of 333,924 individual users on various timelines. Figure 3 shows the sample access log dataset. It represents the IP addresses of the user's request, the timestamp, request type, file path, status code and size of the data being transferred. Individual users are identified using IP addresses and clustering is performed on them to identify the other required parameters.

Figure 4 presented the ratio between the required QoS and the Provided QoS value. Points concentrated on the diagonal indicate that the requested QoS perfectly aligns with the provided QoS. Points on the top left indicates that the QoS provided was higher than the requested QoS, while the points situated in the bottom left indicates that the provided QoS is lesser than the requested QoS. It could be observed that most of the points are situated either on the diagonal or in the top left. This exhibits the efficiency of our approach to appropriately identify the user requirements and provide appropriate allocations.

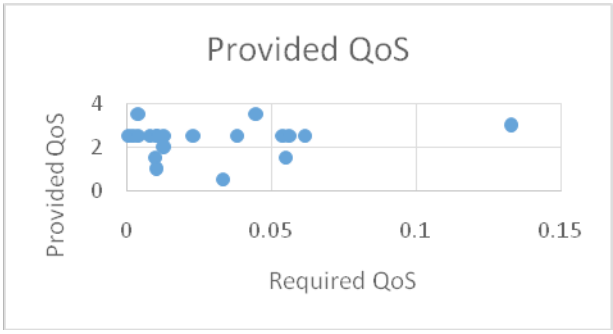

Figure 4. Requested vs. Provided QoS.

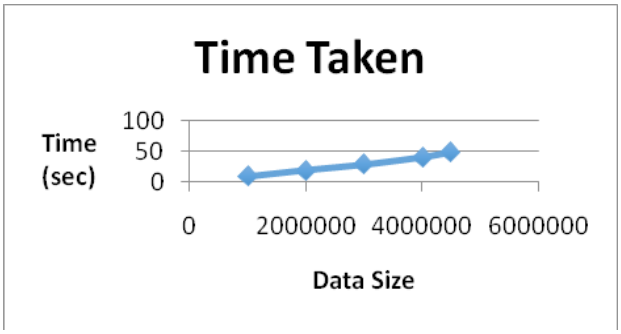

Figure 5. Time Taken vs. Data Size.

Figure 5 presents the ratio between the time taken for the entire process to complete and the data size. It could be observed that as the size of the data increases, the time taken also increases. Hence the time taken is observed to be linear.

4. Conclusion

Automatic identification of user requirements is one of the most challenging tasks for both the consumer and the cloud service provider. The consumer might not have complete knowledge about the parameters to be provided. However it is mandatory for the consumer to provide data in order to effectively allocate resources. Though a simpler method of selecting from predefined workloads exist, it is very abstract, hence mostly leads to inappropriate selections. This approach deals with identifying the SMI parameters using the customer log files from the previous

usage scenario. This method exhibited better predictions customized to the user's needs rather than selecting the abstract scenarios. Our current work has been applied on a web log to identify parameters. Our future work will utilize all the above-mentioned logs to identify the SMI parameters. The current approach does not handle a customer who does not have either the logs or workload information. Further extensions will also be based on this direction, to identify the customer's requirements by performing customer classification. Existing provisioning work are completely based on identifying the best cloud service package for the current user without considering the basic parameters. In future, our contributions will also be based on integrating Multi Criteria Decision Making (MCDM) to identify the best service parameters or packages using the parameters identified from the log records.

5. References

1. Kostoska M, Gusev M, Ristov S. A new cloud services portability platform. Procedia Engineering, 2014 Mar; 69:1268-75.
2. Huang J, Liu G, Duan Q. On modeling and optimization for composite network - Cloud service provisioning. Journal of Network and Computer Applications. 2014 Oct; 45:35-43.
3. Singh A, Juneja D, Malhotra M. A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing. Journal of King Saud University-Computer and Information Sciences. 2015 Nov.
4. Ezugwu AE, Buhari SM, Junaidu SB. Virtual machine allocation in cloud computing environment. International Journal of Cloud Applications and Computing (IJCAC). 2013 Apr; 3(2):47-60.
5. Fox A, Griffith R, Joseph A, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I. Department Electrical Engineering and Computer Sciences, University of California, Berkeley, Rep. UCB/EECS: Above the clouds: A Berkeley view of cloud computing. 2009 Feb.
6. Wu CM, Chang RS, Chan HY. A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters. Future Generation Computer Systems. 2014 Jul; 37:141-47.
7. Quang-Hung N, Thoai N, Son NT. Epobf, energy efficient allocation of virtual machines in high performance computing cloud. 2013 Oct.
8. Buyya R, Abramson D, Giddy J, Stockinger H. Economic models for resource management and scheduling in grid computing. Concurrency and computation: practice and experience. 2002 Nov; 14(13-15):1507-42.
9. Buyya R, Branson K, Giddy J, Abramson D. The Virtual Laboratory: a toolset to enable distributed molecular modelling for drug design on the World-Wide Grid. Concurrency and Computation: Practice and Experience. 2003 Jan; 15(1):1-25.
10. Li JZ, Chinneck J, Woodside M, Litoiu M. Fast scalable optimization to configure service systems having cost and quality of service constraints. Proceedings of the 6th international conference on Autonomic computing. ACM, 2009 Jul; p. 159-68.
11. Hu D, Chen N, Dong S, Wan Y. A user preference and service time mix-aware resource provisioning strategy for multi-tier cloud services. AASRI Procedia. 2013 Nov; 5:235-42.
12. Singh R, Sharma U, Cecchet E, Shenoy. Autonomic mix-aware provisioning for non-stationary data center workloads. Proceedings of the 7th International Conference on Autonomic computing. ACM 2010 Jun; p. 21-30.
13. Kim S, Kim JS, Hwang S, Kim Y. Towards effective science cloud provisioning for a large-scale high-throughput computing. Cluster Computing. 2014 Dec; 17(4):1157-69.
14. Wang XY, Lan D, Wang G, Fang X, Ye M, Chen Y, Wang Q. Appliance-based autonomic provisioning framework for virtualized outsourcing data center. ICAC'07, Fourth International Conference on Autonomic Computing. IEEE, 2007 Jun; p. 29.
15. Urgaonkar B, Shenoy P, Roscoe T. Resource overbooking and application profiling in a shared internet hosting platform. ACM Transactions on Internet Technology (TOIT). 2009 Feb; 9(1).
16. Li C. Optimal resource provisioning for cloud computing environment. The Journal of Supercomputing. 2012 Nov; 62(2):989-1022.
17. Madni SH, Latiff MS, Coulibaly Y. An Appraisal of Meta-Heuristic Resource Allocation Techniques for IaaS Cloud. Indian Journal of Science and Technology. 2016 Jan; 9(4):1-14.
18. Shyamala K, Rani TS. An analysis on efficient resource allocation mechanisms in cloud computing. Indian Journal of Science and Technology. 2015 May; 8(9):814-21.
19. Mahmoud AA, Zarina M, Nik WN, Ahmad F. Multi-Criteria Strategy for Job Scheduling and Resource Load Balancing in Cloud Computing Environment. Indian Journal of Science and Technology. 2015 Nov; 8(30):1-5.
20. Sheshasaayee A, Margaret TS. The Challenges of Business Intelligence in Cloud Computing. Indian Journal of Science and Technology. 2015 Dec; 8(36):1-6.
21. Randles M, Lamb D, Taleb-Bendiab A. A comparative study into distributed load balancing algorithms for cloud computing. 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA). IEEE, 2010 Apr; p. 551-56.

22. AVouk M. Cloud computing–issues, research and implementations. CIT. Journal of Computing and Information Technology. 2008 Dec; 16(4):235-46.
23. Srikantaiah S, Kansal A, Zhao F. Energy aware consolidation for cloud computing. Proceedings of the 2008 conference on Power aware computing and systems. 2008 Dec; p. 10-10.
24. Berl A, Gelenbe E, Di Girolamo M, Giuliani G, De Meer H, Dang MQ, Pentikousis K. Energy-efficient cloud computing. The Computer Journal. 2010 Sep; 53(7):1045-51.
25. Kim KH, Beloglazov A, Buyya R. Power-aware provisioning of cloud resources for real-time services. Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science. ACM, 2009 Nov.
26. Singh S, Chana I. Q-aware: Quality of service based cloud resource provisioning. Computers & Electrical Engineering. 2015 Oct; 47:138-60.