ADAPTIVE MULTI-TENANCY POLICY FOR ENHANCING SERVICE LEVEL AGREEMENT THROUGH RESOURCE ALLOCATION IN CLOUD COMPUTING

MasnidaHussin, AbdullahMuhammed and NorAsilahWatiAbd Hamid

Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, University of Putra Malaysia, 43400 UPM Serdang, Selangor, Malaysia

ABSTRACT

The appearance of infinite computing resources that available on demand and fast enough to adapt with load surges makes Cloud computing favourable service infrastructure in IT market. Core feature in Cloud service infrastructures is Service Level Agreement (SLA) that led seamless service at high quality of service to client. One of the challenges in Cloud is providing heterogeneous computing services for the clients. With the increasing number of clients/tenants in the Cloud, unsatisfied agreement is becoming a critical factor. In this paper, we present an adaptive resource allocation policy which attempts to improve accountable in Cloud SLA while aiming for enhancing system performance. Specifically, our allocation incorporates dynamic matching SLA rules to deal with diverse processing requirements from tenants.Explicitly, it reduces processing overheadswhile achieving better service agreement. Simulation experiments proved the efficacy of our allocation policy in order to satisfy the tenants; and helps improve reliable computing.

KEYWORDS

Resource allocation, Cloud computing, Service Level Agreement (SLA), Adaptive Service Agreement

1. INTRODUCTION

Cloud computing provides efficient resource sharing in tackling large-scale problems through ‘pay-as-you-use’ model. The scale of Cloud computing and the diversity of requirements from clients/tenants put accountability a high priority performance metric. The continuous growth of number of tenant in Cloud has led to complex agreement procedure from handling large tenants’ workloads to resource heterogeneity [1-5]. Furthermore, each tenant is associated with payment scheme that accordance with its service usage. Such issue required better resource management in order to monitor, schedule and allocate a right service agreement for the suitable demand.

Specifically, the resource management in Cloud supports for compute, storage and communication resources as well as for hosted workloads that dynamically arrive.Satisfaction on Cloud computing is an important indicator that reflect quality of resource management system [6-8]. Due to the advancement in the Cloud computing, resource capability (processing and communication) is not such a big issue nowadays. Cloud able to provides its services at anywhere and anytime. However, the issues concerning resource management have changed from mere availability to accountability [9-10]. In response to accountable computing, the tenants demand for the secure and reliable services. Hence, the service agreement between Cloud (Cloud provider) and tenants need to be comprehensive and consistent. The agreement is essentially to fulfil the provider profit with high service satisfaction by tenants.

DOI: 10.5121/ijcnc.2016.8404

Basically, well-designed Service Level Agreements (SLAs) can significantly contribute to reducing causes of potential satisfaction conflict. A Cloud Service Agreements (CSAs) serves as a means of formally documenting the service(s), performance expectations, responsibilities and limits between Cloud service providers and their tenants [6, 11-13]. However, it is hard to determine such information prior in dynamic computing environment. Most of the time, information given by the tenants is difficult to understand and translated into service agreement during negotiation process. In addition, network management systems provide only the most elementary information to guide on resource scheduling process. In many cases, if not all, there is limited information that available for the resource manager/scheduler to assist in the resource allocation [6].

This motivated us to design an adaptive Cloud service agreement (CSA) hat defined as a clear understanding of the offered services including enter and exit clauses of current hosted services contracts. With the diverse demands from the tenants, we classified group of services that offered by the provider based on (the frequent) demands from the tenants. Our resource allocation approach required the resource manager/scheduler to continuously monitor the valuation of matching while ensuring the execution and transaction processes are not costly in terms of overhead and latency.

The proposed approach has been evaluated in a simulated large-scale computing environment. It significantly contributes to providing good-quality allocation decisions while meeting “at-scale” processing requirements. The reminder of this paper is organized as follows. A review of related work is presented in Section II. In Section III we describe the models used in the paper. Section IV details our adaptive resource allocation for CSAs. Experimental settings, performance metrics and the experimental results are presented in Section V. Finally, conclusions are made in Section VI.

2.RELATED WORK

Despite of decades of research advances, resources allocation keeps posing challenging research questions due to ever-increasing workload variety and scale, and increasing diversity of resources and network domains. Hence, there has been increasing interest and important in developing better Service Level Agreement (SLA) in resource allocation (e.g., [2, 10, 14-16]) for performance optimization. The SLAs in resource allocation include processing power, memory/storage space, network bandwidth, high availability, data security etc. The problem of optimal resource allocation is further challenging due to the diversity present in the tenants’ applications that hosted by Cloud.

There is guidance of Cloud Service Agreements (CSAs) by Cloud Standards Customer Council in [11]. Their guidelines emphasized on interrogations that the tenants should taking into account in order to design and negotiate the service agreement with the Cloud providers. There are ten steps where for each step; there is section to describe the range of guarantees in the CSAs. The guideline also provides recommendations for tenants on matters that they should ask/negotiate with the provider. The CSAs guideline presented does not have a direct solution on allocation because of wide-ranging negotiation approach to understand and evaluate CSAs. The authors in [1]proposed the adaptive agreement negotiation, brokering and service deployment using virtualization. In their negotiation phase, both entities i.e., provider and users determine the definition and measurement of QoS parameters, the rewards and penalties for meeting and violating the execution process, respectively. Our work has same agreement approach in the paper where the tenants needed to resolve negotiation before committing to service agreement.

In Cloud computing, the level of tenant satisfaction is crucial, making the CSAs significantly important in such computing environment. Therefore, there are many SLA frameworks have been developed for Cloud. In[17], the authors proposed the users' requirements according to functional and non-functional requirements. Interestingly, there are several service specifications in the non-functional requirements such as availability, cost calculation, scalability, configuration and security that been highlighted during negotiation process. When the SLA document is ready, Cloud users reviewed the SLA terms and responded by signing the SLA. The users are also can initiates for either renegotiating or terminating the negotiation process. However, their negotiations approach slightest hard to be operated in dynamic and large-scale computing environment. The authors in [18] proposed the Cloud service agreement lifecycle that includes negotiation, deployment, monitoring, management and termination. The CSAs lifecycle can be extended, however, such approach required high-level guidance from humans that need to decide which steps need to be done to keep the agreement process stable. The hierarchical-based frameworks for managing service agreement process are proposed in several works (e.g., [19, 20]). The fundamental concepts in their work are similar where each layer necessitates communicating and interoperating each other's. These solutions are implicitly encountered computational complexity where the co-operate decisions with different layers consumed addition processing time.

The CSAs are almost exclusively work within resource management system. That means it can be tackled using resource management procedure; for example monitoring and scheduling. In[4] proposed admission control to improve service agreement. They considered the penalty compensation clause in SLAs with IaaS provider and enforce SLA violation. Also, the authors take into account slack time during scheduling for preventing risk of processing failure. The authors in [21] also proposed admission control and scheduling in order to effectively allocate resources for users' applications. In prior to admissions control, they used Artificial Neural Network (ANN) based forecasting model for determining the most suitable pattern of resource usage. Hence, the allocation decisions are based on the prediction list. Our resource allocation policy in this work explicitly take into account computational complexity while adaptively matching the most suitable SLA for tenants that aims for performance optimization.

3. THE MODELS

In this section, we describe the application and system models used in our study.

3.1 Application Model

The tenants are considered autonomous that distributed over many distinct networks. They requested and submitted their demands to resource schedulers; where the demands stayed first at the scheduler for scheduling process (matching and assigning). Each demand from tenant is associated with the set of parameters as shown below,

$$T\text{-}dem_i = (dur, req, bget) \qquad (1)$$

where *dur* refers to required duration for leasing the service, *req* is service requirement descriptions (e.g., storage or CPU) and *bget* is budget limit that sets by tenant for paying the service, respectively. The tenant's satisfaction varied according to the performance of the service that it received. Specifically, tenant *i* is satisfied with the service when the service requirement and budget limit fulfilled. The tenant's satisfaction level identified based on the combination between *req* and *bget* of tenant *i*.

A preliminary investigation is conduced to identify the importance of the tenant's satisfaction towards service supply from provider. Such satisfaction is investigated in the first place by considering both tenant's parameters i.e., *req* and *bget*. It means that higher percentage refers to better satisfaction towards the services. In particular, it can be given by,

$$SR\_tenant_i = \begin{cases} 1.0 & \text{if meets } req\text{AND } bget \\ 0.75 & \text{if meets } req\text{BUT NOT}bget \\ 0.5 & \text{if not meets } req\text{BUT meets}bget \\ 0 & \text{if not meets } req\text{AND}bget \end{cases} \tag{2}$$

3.2 System Model

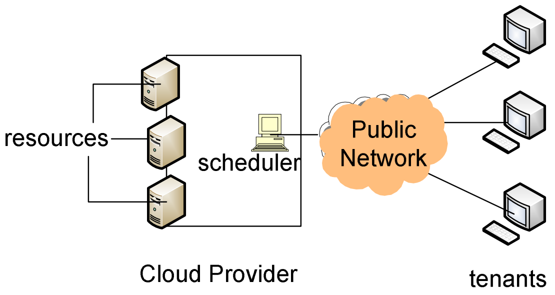


Fig.1. The System Model

The target system used in this work (Figure 1) consists of several number of tenant given asT_i where $i = \{1, 2\ldots, t\}$ which are independently operated by different administrative domains. Each has a separate application and submitted varies demand to the Cloud provider through public network. Each tenant requested for Cloud services either for storage, CPU or bandwidth where each is associated with a set of parameters, given in Eq. 1 It is assumed that the services that requested by the tenant are always available.

The Cloud service provider aims to lease its services i.e., storage, bandwidth and CPU to tenants. Each service has its own profile (e.g., type of service, parameters and history performance). The service provider has its target service profit, given as

$$P\text{-}profit = (c, pf) \tag{3}$$

Where c is an initial service cost, *maxpf* refers to maximum margin between service cost and rental costs, respectively. The provider intentions to maximize the profit while supplying the services to tenants. The Cloud services in this study are charged by the provider based on a service value. The service value is not necessary in dollar ($) where it can represents in variable instances price e.g., time, volume and reward.

There is a resource scheduler in the Cloud provider that acts as resource administrator where the communication between provider and tenants happens. It is assumed that the scheduler has complete knowledge of services that offer by the provider. In this work, the scheduler directly communicates with tenant to receive and submit the demands from/to the tenants, in the sense that there is a communication link between them. The link represents connection between provider and it might be an actual link of cable or a virtual link of the Internet. The scheduler is given the authority to keep track of its resources' details; where the scheduler deals with tenants in parallel.

4.ADAPTIVE CLOUD SERVICE AGREEMENT

This section begins by briefly describing Cloud service agreement. Later, we give the details of our resource allocation that based on availability and capacity, in order to provide multi-tenancy demands.

4.1 Discovery of Cloud Service Agreement

Service Level Agreements (SLAs) are formal documents that define a set of service level objectives and agreed by participants (i.e., tenants and provider). The objectives may concern availability, performance, security and compliance or privacy. In [11], the Cloud agreement are decomposed into three major artifacts: "Customer Agreement," "Acceptable Use Policy" and "Service Level Agreement". In general, the analyzed Cloud SLAs focused solely on availability and capacity. The standardized solutions or service specification are based on predefined platforms and applications. Such specifications are characterized by on-demand self-service, broad network access, resource pooling or rapid elasticity. Specifically, service documentations are designed to maintain currency with latest industry technology. It is one of disadvantages in Cloud agreement where the standard documentation focusing more towards the service offerings, not to individual customer requirements.

There are several key issues highlighted in service specifications, for examples service level objective, support policy, system resiliency and disaster recovery service policy and privacy policy. In this work, we merely take into account agreement for provisioning and management processes (such as discovery and allocation). The resource allocation takes into account both computation and communication factors to determine resource capability. It is significant to identify accurate information of resources' states in resource allocation for better performance.

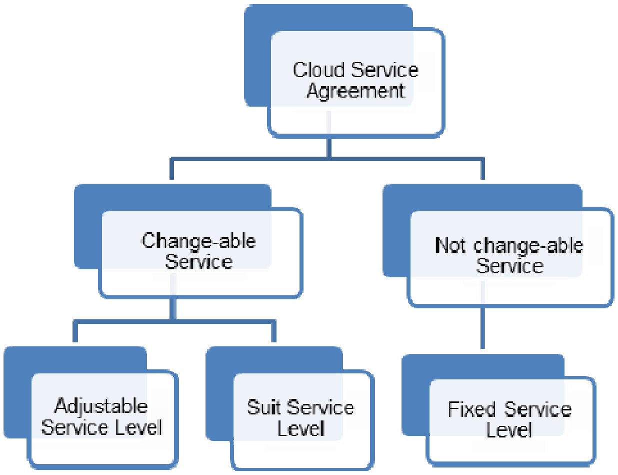


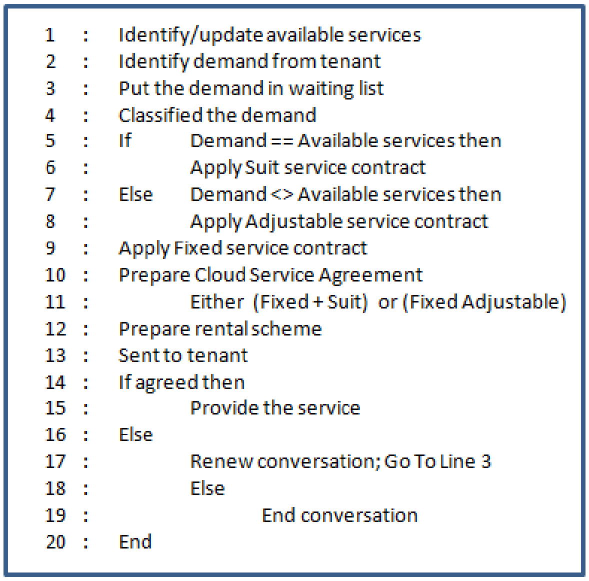
Fig.2 . Adaptive CSAs Stack

In response to the issue, our Cloud Service Agreements (CSAs) takes into account two different service agreements (Figure 2), (i) change-able service and (ii) non-changeable service agreements. These two service agreements are different in terms of type of service specification that offered by the Cloud provider. For the change-able service, such as resource capacity and bandwidth, the tenants permitted to plan their own standard contract; provided that the services are available by the provider. In the changeable service, the tenants are initially matched their requests with the provider's available services. If the tenants complied with the payment scheme, the standard contract of the change-able service agreement grants to be adjusted and modified at any time. On other hand, the non-changeable service for example security and audit policies is not a substitute for adoption since it solely managed internally by the Cloud provider. Hence, there is no conversation or negotiation procedure between tenants and provider for this type of service. It is because the tenants need to comprehend and respect to the service policy that designed prior by the provider.

Due to we focus on multi-tenancy resource sharing, both type of service agreements have significant on performance optimization. The tenants are able to evidently understand the service agreement that well-suited designed based on their demands and provider's service supply. Leverage Cloud services according to its criteria provide thorough monitoring and scheduling process; hence it enhances allocation decisions. Such scenario improves satisfaction level in both participants in regards service performance and profit.

4.2 CSA Resource Allocation

Specifically, the change-able service agreement is more complicated than the non-change-able service agreement. It is due to the changeable mode of the agreement desires more dynamic and robust real-time computing. In Cloud computing, the tenants obviously required the provider to fulfil their demands either for processing or communication requirements. In order to improve service satisfaction over multi-tenant computing platform, we introduce an adaptive resource allocation to reflect the importance of satisfaction in CSAs. There are several agreement steps in our allocation policy in order to satisfy diverse requirements. At each resource allocation step, our scheduler simultaneously evaluated resource availability; this including processing nodes and link communication.



```
1  :  Identify/update available services
2  :  Identify demand from tenant
3  :  Put the demand in waiting list
4  :  Classified the demand
5  :  If      Demand == Available services then
6  :          Apply Suit service contract
7  :  Else    Demand <> Available services then
8  :          Apply Adjustable service contract
9  :  Apply Fixed service contract
10 :  Prepare Cloud Service Agreement
11 :          Either (Fixed + Suit) or (Fixed Adjustable)
12 :  Prepare rental scheme
13 :  Sent to tenant
14 :  If agreed then
15 :          Provide the service
16 :  Else
17 :          Renew conversation; Go To Line 3
18 :          Else
19 :                  End conversation
20 :  End
```

Fig.3. Adaptive Service Agreement Algorithm

Meanwhile, all demands from the tenants are stored in waiting list at the scheduler for matching and negotiating processes. Such procedure aims to construct the most suitable agreement between tenant and provider. It happens in prior to the payment scheme been endorsed to tenants.

In specific, there are several steps in designing CSAs through our resource allocation approach. First, the tenants must identify type of Cloud deployment that they are ordered. It is because there is a different SLA consideration for each of the Cloud service models (i.e., SaaS, PaaS, IaaS). Second, the tenants must decide which measurement or performance are most critical to their specific demands and ensure those measures are included in the CSA. For example, the performance goals of Cloud computing are directly related to efficiency and accuracy of service delivery such as availability, response time, transaction rate, processing speed, but can include many other performance and system quality perspectives. Then, the mix-and-match policy is endorsed into the agreement where either it categorized into *Adjustable Service Level* or *Suit Service Level*. The *Adjustable Service Level* is considered to be offered if the performance is expected higher than available capacity of running services in the provider. Note that, this technique aims to strive for efficient resource allocation by providing dynamic and rapid provisioning. Such technique improves flexibility that comes from the Cloud computing while allowing the provider to upgrade to existing services. Such policy is suitable if the tenants have substantial allocated budget. Meanwhile, if the tenants' demands able to suit within the current available services in the provider, it then constructs the agreement contract under *Suit Service Level*. Both type of agreement required to maintain tenant satisfaction, in this work it denotes as higher satisfaction rate in Eq.(1).In response to reduce computing overhead, the tenants are avoided to exchange the term and condition in their service level until the rental period ends. The detail descriptions of allocation procedure are given in Figure 3.

5.Performance Evaluation

In this section, we first describe the experiment configuration. Then, experimental results are presented. We study the performance of our resource allocation approach; name *CSA allocation policy*(*CSA-policy*) for system performance that is compared with three other allocation approaches, which are *on-demand policy (OD-policy)* and *fit-available policy (FA-policy)*. In the *on-demand policy*, the Cloud services are supplied to tenants according merely to the tenants requirements, but not guaranteed on the budget that determined prior by the tenants. In the *fit-available policy*, the tenants' demands are allocated and mapped to any available services. In such policy, the tenants do not have right to request for more or extra services from provider. Performance metrics used for the experiment are satisfaction rate (Eq. 4) and processing overhead (Eq. 5), given by:

$$SU = {}^{\sum SR\_tenant}/_{T} \tag{4}$$

$$PO = {}^{agreed_t}/_{num} \tag{5}$$

Where, *SR_tenant* is identified from Eq. 2, *agreed*$_t$ denotes length time taken in negotiation and *num* refers number of complete agreement in the observation period, respectively. Both *agreed*$_t$ and *num*are evaluated through simulation program.

5.1 Experimental Settings

We have evaluated our resource allocation approach via simulation with a Cloud provider and the tenants are set between 20 and 100 which each submitted varies number of demands. Inter-arrival times (*iat*) of the submitted demands follow a Poisson distribution with a mean of five time units. Note that, *iat* satisfies with the allocation policy without explicitly increasing a delay in the waiting list. For a given demand, the duration *dur*is selected randomly from the following set:{2.4, 5.0, 25.5, 65.5, and 120.5} and *bget*arer and only generated from a uniform distribution ranging from 20 to 500. The initial service cost of provider *c* is uniformly distributed within the range of 10% to 50% less than *bget*. Note that the provider always aims to increase its profit through the rental cost.

5.2 Results

Experimental results are presented based on satisfaction rate and processing overhead to investigate the impact of allocation policy on satisfaction. As shown in Figure 4, the proposed adaptive allocation approach outperformed others in terms of satisfaction rate. The superior performance of our approach is primarily achieved by allocation policy that takes into account both demand and supply factors. It also observes that *OD-policy* is comparable with our approach. It is due to the fact that both strategies considered tenants' demand during negotiation process for optimal performance. However, it indicates that *CSA-policy* works more than 40% better in the case of more tenants in the system.

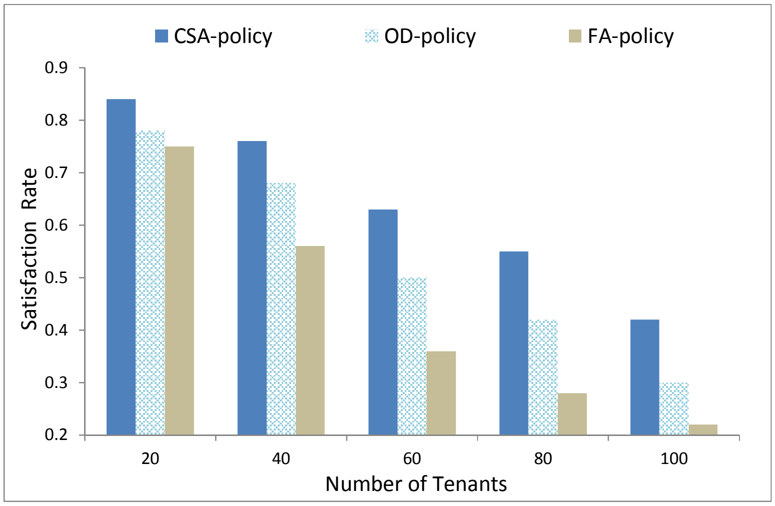


Fig.4.Service Satisfaction with Different Allocation Policies

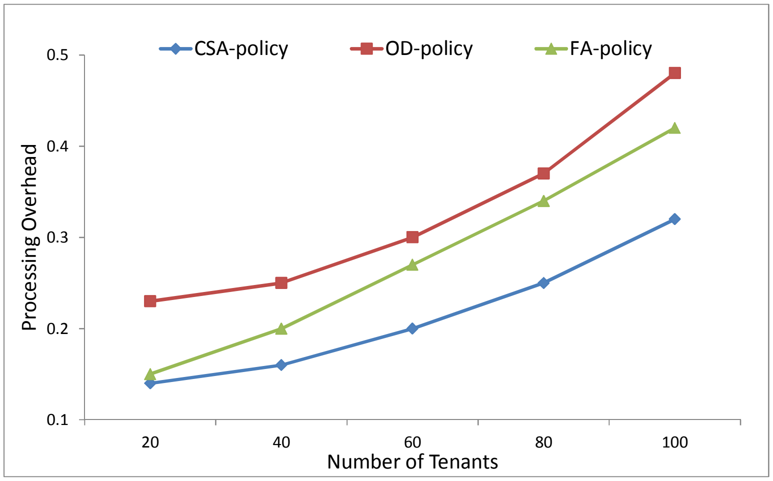


Fig.5 .Processing Overhead with Different Allocation Policies

Figure 5 shows the average processing overhead that is plotted against number of tenants, respectively. Our approach obtains appealing processing overhead compared to other approaches with reduced considerably, about 60% on average. Our *CSA-policy* benefits for handling various demands in dynamic computing environment.

We extend the analysis of *CSA-policy* corresponding to different settings in service requirement (Table 1). This setting aims to analyze the effect of diversity in tenants' requirement towards SLA.

Table 1. Distribution of Different Types of Tenant Demands

Type	**Budget**	**Service Requirement**
A	Low	Low
B	Low	High
C	High	Low
D	High	High

From Figure 6, we can see that the benefit of adaptive agreement in heterogeneous system with better satisfaction rates. It demonstrates comparable results among each scenario that the differences by merely 5%. We also observed that the satisfaction rate of *CSA-policy*with Type A is comparable with Type D. Meanwhile, the figure also shows that Scenario B is comparable with that in Scenario C. This is because the distributed of tenants' requirements pattern in those type of tenants—considering the budget and service requirement relatively similar. We can conclude that the *CSA-policy* plays its best role in agreement procedure when the tenants have similar interest of service.

There is different graph pattern that shows in Figure 7 in regards the processing overhead. It indicates that Type B and C works 20% better in the observation period. This performance can be explained by the reduction in processing time during agreement process where it takes small amount of time to meet a certain agreement level in both tenant and provider. This also indicates that the *CSA-policy*sustains performance optimization with low processing overhead.

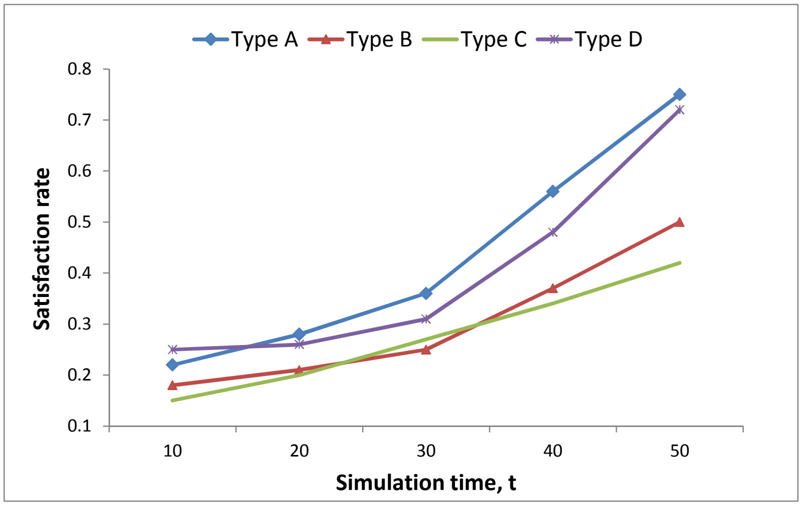


Fig.6. Satisfaction Rate in Different Type of Tenant Demands

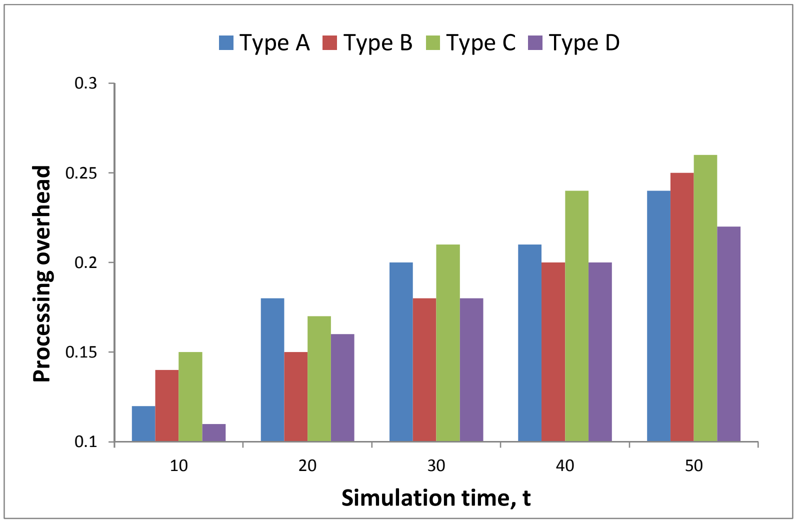


Fig.7. Processing overhead in Different Type of Tenant Demands

6.CONCLUSION

In Cloud computing, different Quality of Service (QoS) constraints have to be guaranteed to satisfy tenant's request. Effective Cloud service agreements (CSAs) are used as a formal contract between Cloud provider and tenant to ensure QoS. The diversity of CSAs options is limited for tenants; hence, they are restricted to specific preferences that meet their request. Therefore, our proposed allocation policy in Cloud computing is to provide dynamic CSAs negotiation between multi-tenant and provider for ensuring satisfaction between them are met. We designed adaptive CSAs that are variable and flexible to personalize service qualities by tenants' plans. Due to the negotiation process in our allocation policy consumed less computational effort, optimal performance is achieved. Optimistically, Cloud able to achieve better service agreement satisfaction when there is clear CSAs guideline in negotiating process.

ACKNOWLEDGEMENT

This work is supported by a Department of Higher Education Malaysia Grant 08-02-13-1363FR.

REFERENCES

[1] A. Kertesz, G. Kecskemeti, and I. Brandic, "An interoperable and self-adaptive approach for SLA-based service virtualization in heterogeneous Cloud environments," Future Generation Computer Systems, vol. 32, pp. 54-68, 2014.

[2] S. Son, G. Jung, and S. C. Jun, "An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider," Journal Supercomputing, vol. 64, pp. 606-637, 2013.

[3] M. Hussin, A. Abdullah, and S. K. Subramaniam, "Adaptive Resource Management for Reliable Performance in Heterogeneous Distributed Systems," in Lecture Notes in Computer Science. vol. 8286, R. Aversa, J. Kolodziej, J. Zhang, F. Amato, and G. Fortino, Eds., ed: Springer International Publishing, 2013, pp. 51-58.

[4] Linlin Wu, Saurabh Kumar Garg, and R. Buyya, "SLA-based admission control for a Software-as-a-Service provider in Cloud computing environments," Journal of Computer and System Sciences, vol. 78, pp. 1280-1299, 2012.

[5] R. da Rosa Righi, V. F. Rodrigues, C. A. da Costa, G. Galante, L. C. E. de Bona, and T. Ferreto, "Autoelastic: Automatic resource elasticity for high performance applications in the cloud," IEEE Transactions on Cloud Computing, vol. 4, pp. 6-19, 2016.

[6] C. Wu and R. Buyya, Cloud Data Centers and Cost Modeling: A complete guide to planning, designing and building a cloud data center,: Morgan Kaufmann, 2015.

[7] J. Barr. (2015). Cloud Computing, Server Utilization, & the Environment.

[8] Rafael Moreno-Vozmediano, Rubén S. Montero, and I. M. Llorente. (2013) Key Challenges in Cloud Computing: Enabling the Future Internet of Services. IEEE Internet Computing. 18-25.

[9] Y. C. Lee and A. Y. Zomaya, "Rescheduling for reliable job completion with the support of clouds," Future Generation Computer Systems, vol. 26, pp. 1192-1199, 2010.

[10] S. F. Jalal and M. Hussin, "Multi-Level Priority-based Scheduling Model in Heterogenous Cloud," Journal of Computer Science, vol. 10, p. 2628, 2014.

[11] C. S. C. Council, "Public Cloud Service Agreements: What to Expect and What to Negotiate," 2013.

[12] S. Singh and I. Chana, "QoS-aware autonomic resource management in cloud computing: a systematic review," ACM Computing Surveys (CSUR), vol. 48, p. 42, 2016.

[13] A.-F. Antonescu and T. Braun, "Simulation of SLA-based VM-scaling algorithms for cloud-distributed applications," Future Generation Computer Systems, vol. 54, pp. 260-273, 2016.

[14] Y. Awano and S.-i. Kuribayashi, "Proposed Joint Multiple Resource Allocation Method for Cloud Computing Services with Heterogeneous QoS," presented at the The Third International Conference on Cloud Computing, GRIDs, and Virtualization, Nice, France, 2012.

[15] A. Nathani, S. Chaudhary, and G. Somani, "Policy based Resource Allocation in IaaS Cloud," Future Generation Computer Systems, vol. 28, pp. 94-103, 2012.

[16] D. Serrano, S. Bouchenak, Y. Kouki, F. A. de Oliveira Jr, T. Ledoux, J. Lejeune, J. Sopena, L. Arantes, and P. Sens, "SLA guarantees for cloud services," Future Generation Computer Systems, vol. 54, pp. 233-246, 2016.

[17] M. Alhamad, T. Dillon, and E. Chang, "Conceptual SLA Framework for Cloud Computing," in 4th IEEE International Conference on Digital Ecosystems and Technologies 2010, pp. 606-610.

[18] M. Torkashvan and H. Haghighi, "CSLAM: A Framework for Cloud Service Level Agreement Management Based on WSLA," in 6'th International Symposium on Telecommunications (IST'2012), 2012, pp. 577-585.

[19] H. Goudarzi and M. Pedram, "Multi-dimensional SLA-based Resource Allocation for Multi-tier Cloud Computing Systems," in IEEE 4th International Conference on Cloud Computing, 2011.

[20] A. Mosallanejad, R. Atan, M. A. Murad, and R. Abdullah, "A Hierarchical Self-Healing SLA for Cloud Computing," International Journal of Digital Information and Wireless Communications (IJDIWC) vol. 4, pp. 43-52, 2014.

[21] Saurabh Kumar Garg, Srinivasa K. Gopalaiyengar, and R. Buyya, "SLA-Based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter," in ICA3PP 2011, Part I, LNCS 7016, Y. X. e. al., Ed., ed: Springer-Verlag Berlin Heidelberg, 2011, pp. 371-384.