Original Research Paper

An Efficient Resource Provisioning Technique in Inter-Cloud using Peer-to-Peer Approach

[1]Manbir Kaur, [2]Kiranbir Kaur and [3]Lohit Kapoor

[1,2]*Computer Engineering and Technology, Guru Nanak Dev University, India*
[3]*Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology, India*

Article history
Received: 24-03-2017
Revised: 13-04-2017
Accepted: 24-05-2017

Corresponding Author:
Manbir Kaur
Computer Engineering and Techonology, Guru Nanak Dev University, India
Email: kaurmanbir2423@gmail.com

Abstract: Resource discovery is the process of computing systems; it supports resource management and scheduling of applications. It searches for the apposite variety of resource that satisfies the user's requirements. Resource discovery in computing system is perplexing because of conventional centralized approaches and resilient availability of resources. This scenario becomes more complex in inter-clouds owing to the dynamism and heterogeneity of clouds involved. The inspiration driving this paper is to utilize peer-to-peer approach to circumvent the problem of resource provisioning and resource discovery that are predominantly associated with centralized approach. This paper introduces a system model of Resource discovery in Inter-Cloud and mechanism to joining Remote Resource Manager using reputation based algorithm. We attempt to minimize the data traffic and balance the load partly by using the reputation based algorithm.

Keywords: Clouds, Inter-Clouds, Resource Provisioning, Resource Discovery
Categories and Subject Descriptors: [Cloud Computing]: Clouds, Inter-Cloud
General Terms: Inter-Cloud

Introduction

With the advancement of technology, it is necessary to provide constant services to user with minimum cost and maximum performance. A large number of physical machines are present to fulfill the users' requests in inter-cloud. Therefore, load on various nodes need to be balanced at any instant of time. The load can be the CPU load, the storage capacity or the network availability. For balancing the load, we need to discover resources to maintain the workload. Inter-Clouds have number of Cloud Service Providers where each Cloud Service Provider has a pool of resources and resources allocated to users depend on various parameters like QoS, latency time, user requirement and availability of resources (Grozev and Buyya, 2014). Broker in inter-cloud facilitates the Cloud Service Provider with the accessibility of resources, execution of user's request and performance constraints for both the user and cloud provider. A broker maintains the provisioning of the resources and the available services. It can be deduced that workload directly or indirectly maintained by broker. Due to improper allocation, some of the machines may become overloaded while other machines are idle or doing very little work. This reduces the system performance. We use peer-to-peer approach in which Remote Resource Manager (RRM) replaces broker. Therefore, RRM keeps track of other Data Centers locations, availability of resources and sends this information to the nearby Cloud Service Provider. If there, is a scarcity of resources in any Cloud Service Provider then that Cloud Service Provider checks the information provided by RRM and selects the Cloud Service Provider, which fulfill it request without violating Service Level Agreement.

In this study, a reputation-based algorithm is being utilized for the discovery of resources and selects the Cloud Service Provider with least cost and maximum performance for dispensing resources to another Cloud Service Provider to fulfill user demands by using Peer-to-Peer approach. We use a word reputation for the selection of cloud service provider, because request made by user for cloud service provider depends on the performance and reputation of cloud i.e. its latency time, cost and how its performance differs from other clouds. One of the vital challenges is to determine the number and kind of resources that are required to satisfy client prerequisite without violating Service Level Agreement. Resource provisioning and efficient resource discovery is a part of our entire work.

Background and Related Work

Brokering service proposed by (Xiao and Wang, 2012) results better for on demand allocation of resources but it does not respond anything except on-demand resources. Therefore, there is need to allocate resources by considering various parameters like response time, performance, QoS etc. (Salama and Shawish, 2014) discussed that resource allocation influences the performance specifically or in a roundabout way as load balancing, QoS, control and optimization of energy indirectly. Nathani *et al.* (2012) depicted policies for resource allocation in IaaS cloud namely best effort and intermediate policy. In the former policy, a request made by a user is satisfied if resource is available else, the request put in FIFO line while in later policy, the request fulfilled when resource is available present otherwise and the request is rejected. Additionally, (Guo and Bu, 2012) likewise utilized approach of access control to discuss hierarchal algorithm for handling the problem of workload and response time. To remove the brokering service we use a peer-to-peer approach to improve performance and minimize cost.

Zaman and Grosu (2012) proposed a mechanism in which the information about Virtual Machine presence to satisfy user request is finished for a given period and sent to the user for a specific resource. As a result, the user remains updated about the request status whether it is going to be completed or rejected. Fajjari *et al.* (2012) proposed the backtracking technique to optimize resource usage among data centers. Xiao and Wang (2012) proposed a resource allocation algorithm based on priority to boost the benefit of data centers when available resources are insufficient to fulfill the demands of the user with allocating resources in dynamic mode rather than in static mode. Thus, (Maguluri *et al.*, 2012) gave a portrayal of "Traffic Optimal allocation algorithm" for scheduling and workload. This algorithm cut down the traffic problem by allocating resources according to the arrival of requests for resources.

In the event that Resource distribution not done appropriately, it brings about overutilization or underutilization of resources, which prompts service interruption. To manage this, a Resource Allocation Strategy (RAS) developed by (Patel and Patel, 2013) for apportioning the resources as per demand. It additionally determined the category and quantity of resources required and we make use of reputation-based algorithm for allocation of resources to fulfill user demands. Pop *et al.* (2009) proposed a genetic scheduling algorithm in which execution of task is autonomous from another task execution. Choi *et al.* (2013) proposed an algorithm in which resource provisioning depends on least cost. A framework presented by (Mechtri *et al.*, 2013) which permits the broker to take decision for the best cloud from availible clouds where the decision relies on performance, execution time, response time and so on. This framework is somehow similar to our schematic model but difference is that we use peer-to-peer approach for the selection of clouds. Like (Mechtri *et al.*, 2013), which permits broker for allocation, (Choi *et al.* 2013) build up a framework in which allocation of resources is done by networking manager where networking manager communicates with broker, interconnecting different cloud providers. In our approach, remote resource manager and regional resource manager do allocation of resource. Papagianni *et al.* (2103) proposed another broker-based model where a broker allotted to every user. In this model, when user sends a request, the broker checks the prerequisite and the availability of resources and allocates resources as per specification and decrease complexity. Liang *et al.* (2012) built another broker-based mechanism in which allocation of resources done by broker in the light of the fact that the broker purchase resources from other cloud providers. Brokers allocate resources as per Service Level Agreement marked between the user and the cloud provider.

The resources allocated to the user as indicated by QoS requirements and results in high performance in the model proposed by (Wu *et al.*, 2014). This model is appended it to the European Internet test bed named FEDERICA. Considerable research is going on to build up an efficient centralized approach and usage of broker used frequently to serve this purpose. Broker is the focal element that decides which Cloud Service Provider is appropriate for allocating resources to clients based on various constraints like QoS, response time and execution time and so on. A broker maintains the provisioning of resources and already available services. Centralized approach is usually leveraged for provisioning of resources as a part of which broker is presented as a intermediary. In this study, we proposed a peer-to-peer approach in which there is no broker, Cloud Service Provider specifically speak with different Cloud Service Providers. In the case of peer-to-peer network, user directly communicates with Cloud Service Provider when there is no centralized approach and the user has to choose the kind and number of resources required to fulfill the request. One of the important challenges is to determine the number and type of resources that are required to fulfill customer requirement without violating Service Level Agreement. In our peer-to-peer approach, each Cloud Service Provider has data about idle resources of other Cloud Service Provider because of which it turns to be simple for Cloud Service Provider to fulfill the request made by user. Remote Resource Manager replaces broker and Regional Resource Manager monitor other Data Center locations, accessibility of resources and sends this information to the nearby Cloud Service Provider. In this manner, when there is a scarcity of resources in any Cloud

Service Provider then Cloud Service Provider checks the information gave by RRM and select the Cloud Service Provider which satisfy its demand without violating Service Level Agreement. Peer-to-peer approach results better than centralized approach. In view of resource allocation and scheduling, we attempt to build up a reputation-based algorithm, which selects the Cloud Service Provider with least cost and maximum performance for dispensing resources to another Cloud Service Provider to fulfill user demands.

System Model

The Schematic view of Resource discovery in Inter-Cloud shown in Fig. 1 in which the Remote Resource Manager (RRM) from a specific Regional Resource Manager initially tries to serve inside the Local Group (LG) where all local groups are placed inside a circle in Fig. 1 where circle represents the cloud service provider and corresponding RRM. Each RRM has stored the Resource Availability (RA) advertisements from different Regional Resource Managers inside the LG. In Case that the assets accessibility inside the LG is not met, the request for RRM sends a Remote Resource Request (RRR) to Super Group. The assets asked for in the RRR are accessible at a specific RRM, the RRM sends the details of the Resource Managers to request for the availability RRM. In the event that none of the RRMs inside a LG meet the requested services, the RRR is spread further inside the Super Group (SG) until the solicitation is met or all choices are depleted.

An obvious challenge in all resource discovery strategies is to manage the trade-off between resource cost and latency. The cheapest resources could be located the farthest and latency adds its own costs in terms of data transfer costs and communication overheads. This choice needs to be made by the requesting RRM.

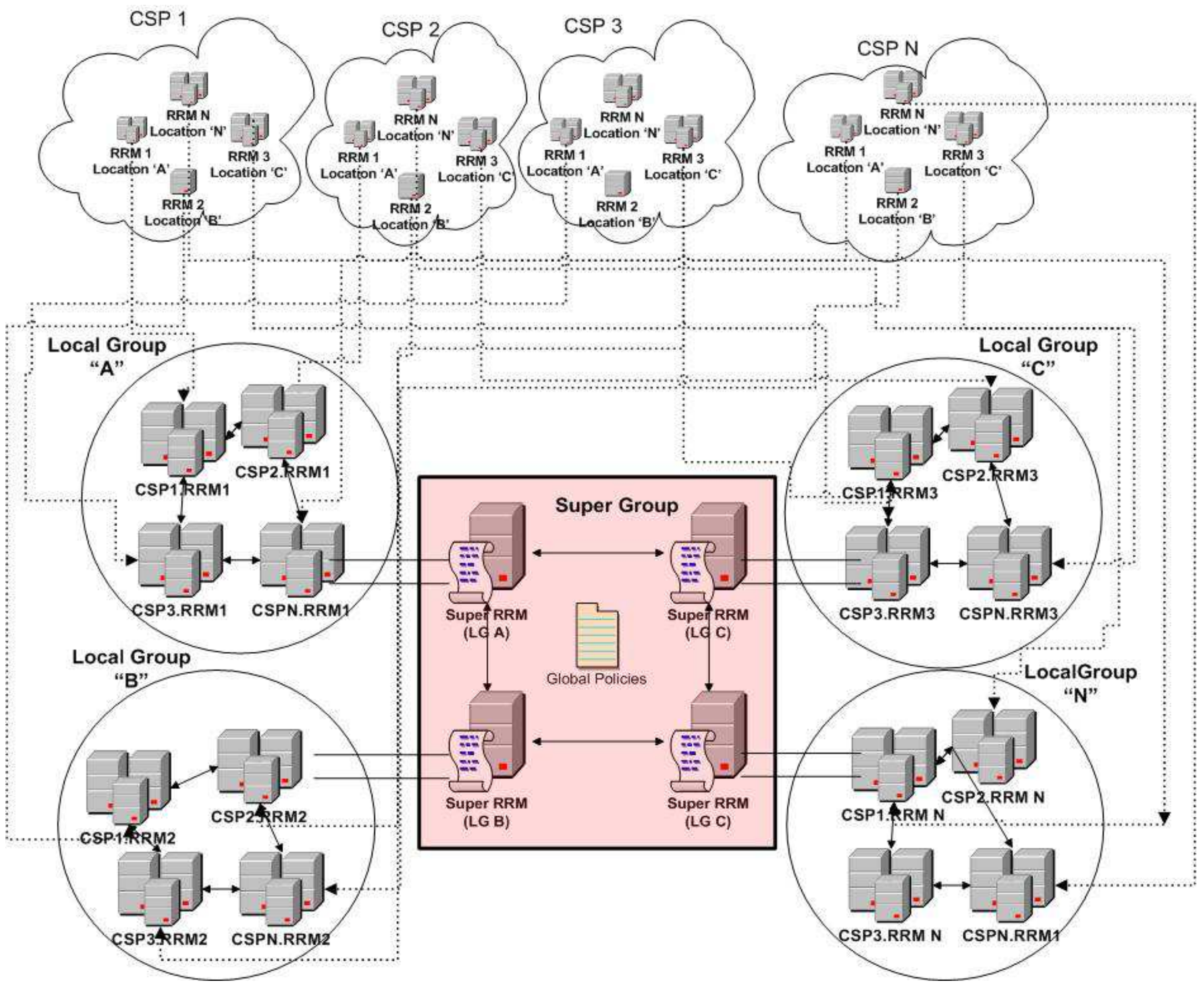


Fig. 1. Schematic view of Resource discovery in Inter-Cloud

Every data center inside the federation puts out a Resource Availability (RA) status intermittently as promotions. The RA normally communicated as far as possible Resources (RES) and their related cost (C), where every asset can be a virtual machine, stage or administration. The information of running resources is updates by Resource Availability status from different Regional Resource Managers inside the LG. This status shows whether resource is currently running or wait for allocation. Resource Availability status sends information to Super Group if resource leaves the system or being used by another Cloud Service Provider. So that request made by user for same resource gets the information whether the resource is idle or being used. In this way, RA is the arrangement of assets, expense tuples publicized by each RRM inside the Local Group and reserved by the super Remote Resource Manager that takes part in the SG:

$$RA = \{(RES_{1,}C_1),(RES_2,C_2)\ldots\ldots.(RES_N,C_N)\}$$

where, X is the quantity of assets offered for the remote use by a specific data center at a specific time. X differs in view of the asset requested at the data center. Therefore, different RRMs need to store just the last RAs issued by RRMs of other server farms since these precisely depicts the condition of accessible resources and the expenses related to the assets is likewise a part of the RA. It makes possible to insert new resources from other cloud service provider by sending request to another local group at the same time when resources is being used.

Other data centers that are willing to avail services within the federation put out a Resource Request (RR) advertisement agai in terms of required RES and desired cost:

$$RR = \{(RES_{1,}C_1),(RES_2,C_2)\ldots\ldots.(RES_K,C_K)\}$$

where, K is the number of resources required by the requesting RRM.

The objective for the requesting RRM is to locate another RRM such that:

$$RR_K \approx RA_X$$

where, $K <= X$, so that the number of resources available at the prospective partner RRM is more than or equal to the number of resources requested.

The RR from a specific RRM initially tried to be serviced inside the LG. Each RRM would have till now stored the RA advertisements from various RRMs present in the LG. If the asset accessibility inside the LG is not meet, then the request for RRM sends a Remote Resource Request (RRR) to the Super Group. In the event that the assets requested for in the RRR are accessible at a specific RRM, the RRM sends the details of the RMs to the requesting RRM. In the event that none of the RRMs inside a LG meets the requested services, the RRR is spread further inside the SG until the solicitation is met or all choices are depleted.

For instance, a high-priority user service request with stated Service Level Agreements serviced by choosing a RRM with the lowest latency i.e., closest to the requesting RRM, which also meets the cost criteria. On the other hand, a low priority user request serviced by a best-fit approach in which cost is given more importance over latency to maximize the profit of the requesting RRM. The RR or the RRR requests issued can reflect the relevant priority of cost or latency. Further, to handle scenarios where an RRM may not wants its data to be processed at a particular geographical location, a conditional RRR can be issued which prevents the query being forwarded to the excluded locations.

Sequence of Operations

Joining Process for New RRM

If the RRM is the first in the network, it assumes the role of the Super RRM. Since RRMs are related to data centers, they are assumed to be accessible at all times. For ensuring joins of RRM's the request responded by the Super RRM. With the expansion in the size of LG, request gets cached on all delegate RRMs that they go through. Thus, RRM join times are in this way brought down or lowered. The recently entered RRM is currently able to get RA and send RR advertisement from/to different RRMs. Algorithm 1. Depicts that RRMs joining the LG in Peer Clouds

Super RRM Selection

Choice of RRM as a Super RRM made on first come and first serve basis i.e., the first RRM to join a LG designates itself as Super RRM for a specific region. The consecutive RRMs hold their joining rank in the LG. The Super RRM becomes a Gateway to the SG by ordering asset promotions from various RRMs present in the LG and share it with the present collection of super RRMs. In the worst situation, that the Super RRM comes up short, the following position of RRM assumes control as the Super RRM. This procedure starts if the Super RRM does not convey a unique status message within predefined time. Each RRM persistently produces a RA (asset accessibility) status message every 5 min, which holds the present status of assets and their related cost and circlulate it inside the LG. Every RA message has a period to-live parameter associated with it to guarantee that more stale messages do not remain available for use. The RA status messages reserved by different RRMs in the LG and used to start an agreement negotiation with them in the light of future administration

Algorithm 1. RRMs joining the LG in Peer Clouds

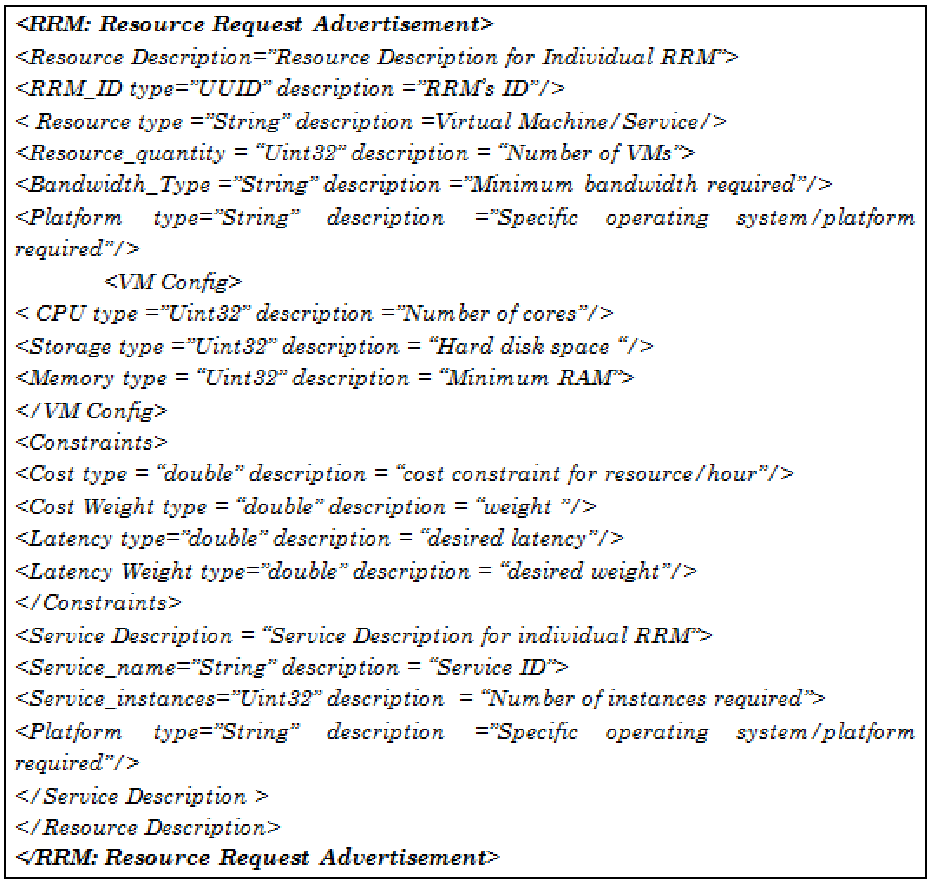
```
For each RRM ε Peer Cloud do
if (RRM does not ε to LG_i)
locateSuperRRM(myRRMID, regionID)//find SuperRRM for my region
if (!superRRM) // no SuperRRM found
newSuperRRMID = becomeSuperRRM(myRRMID)
LG = createLG(myRegionID)
joinSG(newSuperRRMID, regionID)
else // SuperRRM exists in my region
registerWithSuperRRM(myRRMID)
joinLG (LG)
Exit
```

Resource Discovery

The procedure of assets disclosure is done by two types of constraints (a) costor (b) asset particular, which are a part of the asset demand advertisements. If specific request for the resources are not serviceable inside the LG because of absence of assets or not meeting cost limitations, then they placed out in the SG for required asset provisioning. The Super RRM proliferates the solicitation to different Super RRMs in the SG that spread the solicitations further inside their individual LGs. RRMs t satisfy the asset criteria indicated in the advertisement contact the promoting RRM specifically. The algorithm for asset provisioning is given in Algorithm 2. while an example of resource advertisement is delineated in Fig. 2. Selection of resources by any RRM is performed on the basis of "latency" or "price" or both.

Algorithm 2. Resource lookup and provisioning in PeerClouds at each RRM

```
advertiseResources(resourceVector, costVector)//RA
advertiseRequirements(resourceVector,
constraintsVector) //RR
processResponse (responseVector)
for each response in responseVector
rankResponse(response)
selectedRRM = getTopRRM()
sendConfirmation(selectedRRM)
processRequest (request)
If (evaluateRequest(request))
sendConfirmation(request.getRRM())
Exit
```



```
<RRM: Resource Request Advertisement>
<Resource Description="Resource Description for Individual RRM">
<RRM_ID type="UUID" description ="RRM's ID"/>
< Resource type ="String" description =Virtual Machine/Service/>
<Resource_quantity = "Uint32" description = "Number of VMs">
<Bandwidth_Type ="String" description ="Minimum bandwidth required"/>
<Platform type="String" description ="Specific operating system/platform
required"/>
        <VM Config>
< CPU type ="Uint32" description ="Number of cores"/>
<Storage type ="Uint32" description = "Hard disk space "/>
<Memory type = "Uint32" description = "Minimum RAM">
</VM Config>
<Constraints>
<Cost type = "double" description = "cost constraint for resource/hour"/>
<Cost Weight type = "double" description = "weight "/>
<Latency type="double" description = "desired latency"/>
<Latency Weight type="double" description = "desired weight"/>
</Constraints>
<Service Description = "Service Description for individual RRM">
<Service_name="String" description = "Service ID">
<Service_instances="Uint32" description = "Number of instances required">
<Platform type="String" description ="Specific operating system/platform
required"/>
</Service Description >
</Resource Description>
</RRM: Resource Request Advertisement>
```

Fig. 2. Sample RRM resource request advertisement

Experimental Setup, Simulations and Results

The first experiment measured the startup time for 10 to 50 participating RRMs with one designated Super RRM in a Local Group. The aim of the experiment is to observe the cumulative time for the initial configuration and organization of a Local Group. It is clear from Fig. 3. that as the number of participating RRMs increases the overall startup time per RRM reduces from 9.3 sec./RRM (for 10 RRMs) to 8.2 sec./RRM (for 50 RRMs). This is due to the impact of super RRM startup time and resource aggregation on the overall time gets averaged out. The startup time includes the JXTA initialization time per peer/RRM as well.

A variety of timing measurements for two different types of operations and resource discovery queries within the test setup were obtained for varying topology sizes.

Figure 4 provides the time taken for a new RRM to join the existing setup. Average time ranges from 770 to 860 ms for topologies with 10 to 50 RRMs within a LG. The join process for a new RRM comprises initialization time plus JXTA peer join time plus the time taken for the RRM to connect with the Super RRM.

To evaluate the performance of resource queries following parameters are used (Table 1).

Figure 5 and 6 present the Request Service Rate (RSR) and Response Time (RT) within an LG for varying number of RRMs. It observed that the RSR remains linear with varying number of queries. The size of the LG has a direct impact on RSR. Thus, a larger size of LG results in lower number of resource queries that forwarded to the SG.

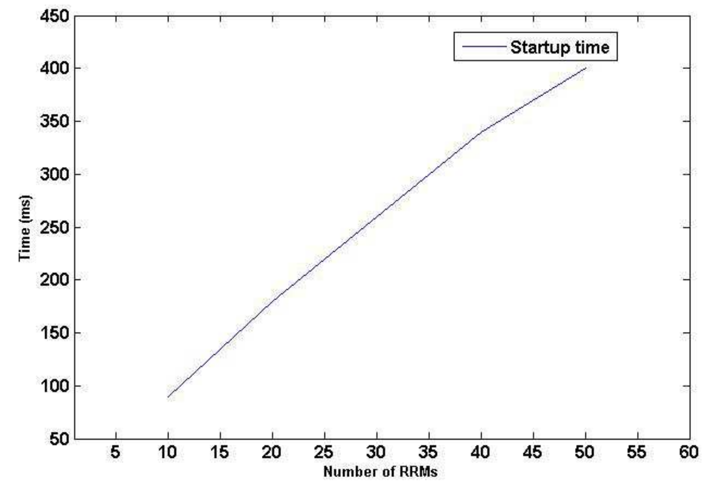


Fig. 3. Startup time with varying number of RRMs

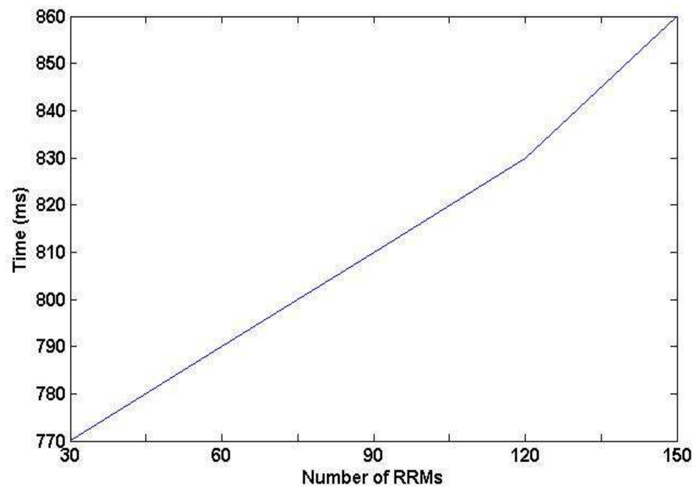


Fig. 4. Average join time for a new RRM as a function of LG size

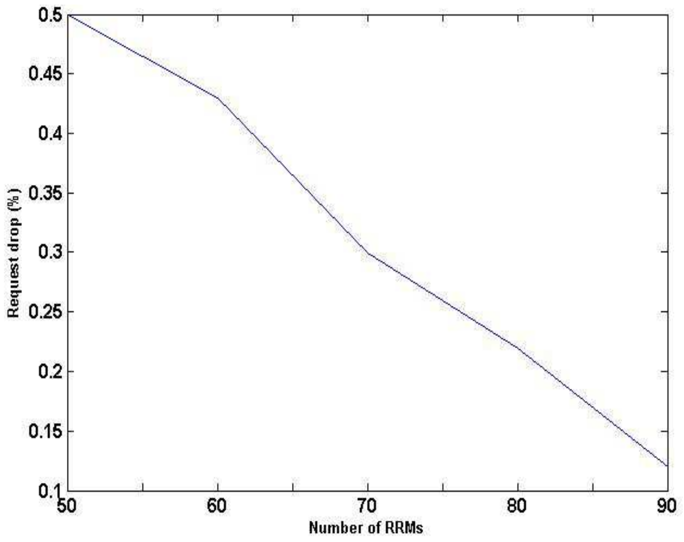


Fig. 5. Request service rate

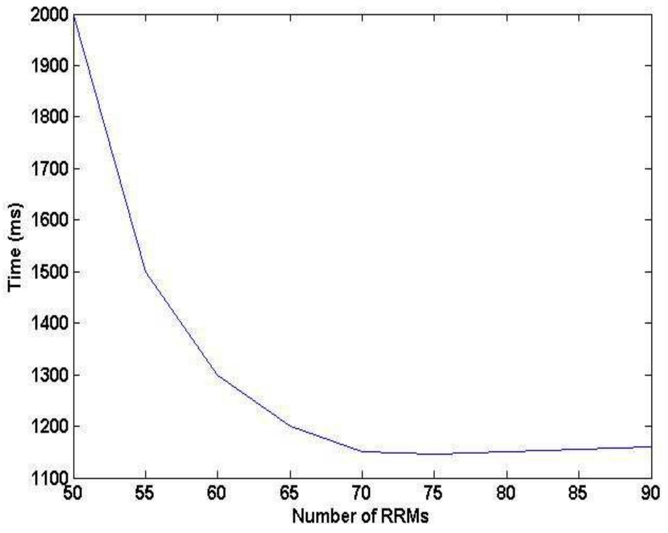


Fig. 6. Response time

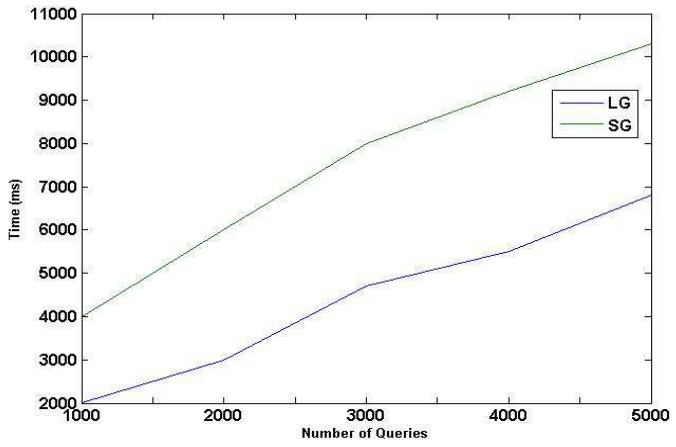


Fig. 7. Average resource query response time for varying number of queries (LRQ)

Next, we evaluated resources query responses from LG and SG under following preferences set by resource query generator/user:

- Latency based Resource Query (LRQ): In this type of resource query there is an attempt to look out for resources, which fall under pre-defined latency
- Cost based Resource Query (CRQ): In this type of resource query there is an attempt to look out for resources which falls under pre-defined cost
- Hybrid Resource Query (HRQ): It attempts to find resources, which fall under the threshold response time while maintaining the requested costs

For LRQ, about 7% of the queries were serviced by the SG and 93% of the queries were serviced by the LG. Further there is an average increase of 41% in response time when the responses come from SG as compared to LG owing primarily to communication delays as shown in Fig. 7.

Table 1. Cloudlets/queries parameters

Parameters name	Ranges
Cloudlet Length (the length or size (in mips) of this cloudlet to be executed per Virtual Machine)	(1000 to 5000 mips)
pes Number (CPU cores per Virtual Machine)	1
Resource request frequency (The number of requests per unit time)	2~5 per min
Duration of resource usage (Time to hold a resources)	30~60 min
Flash-crowd scenario frequency (Peak hour time)	once every 3 h
Flash-crowd scenario duration (Time duration of peak hour)	10 min
Flash-crowd resource request frequency (The number of requests per unit time)	15~20 per min
resCost (Cost requested per resource)	0.20-0.40$ per h

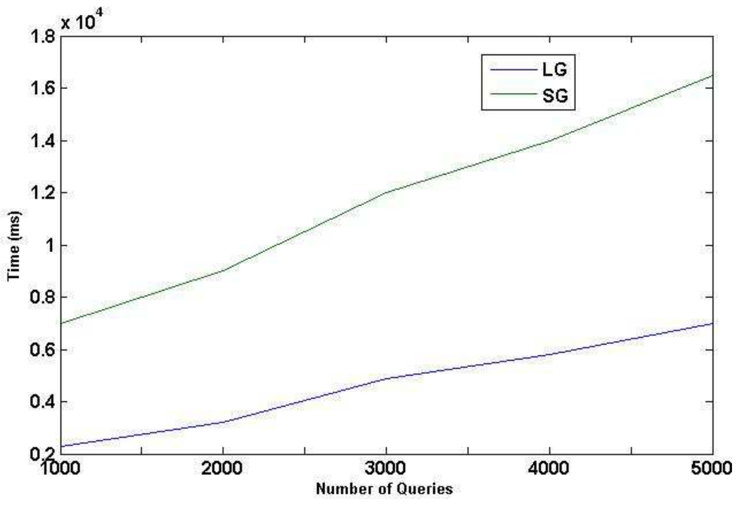


Fig. 8. Average resource query response time for varying number of queries (CRQ)

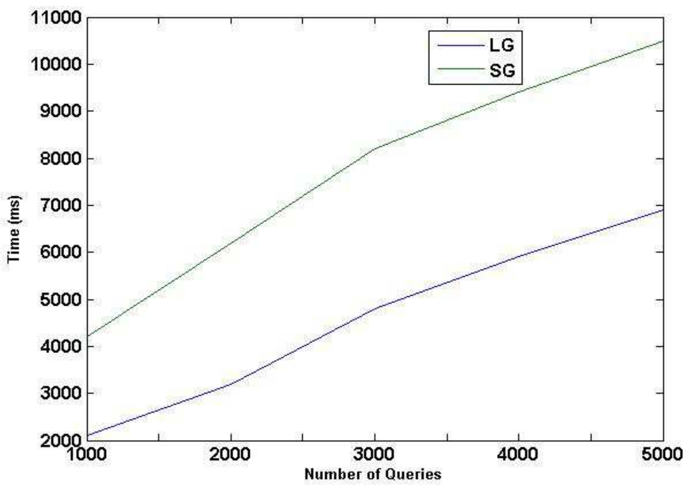


Fig. 9. Average resource query response time for varying number of queries (HRQ)

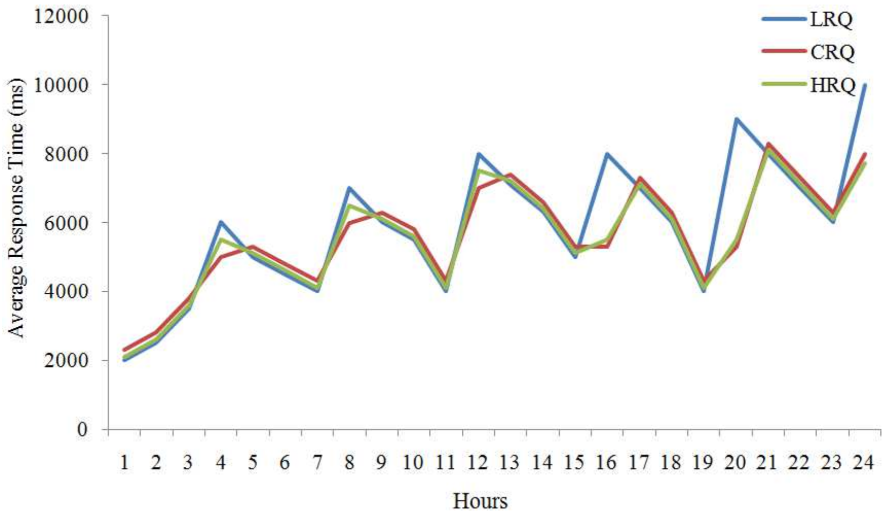


Fig. 10. Comparative view of CRQ, LRQ and HRQ

For CRQ, about 43% of the queries were serviced by the SG and 57% of the queries were serviced by the LG. As shown in the Fig. 8, the queries serviced by SG suffers very high overhead (communication delay), resulting in high response time.

Further, for HRQ as shown in Fig. 9, 93% of queries were serviced within LG and 7% from SG and the resultant response time remains marginal high to LRQ and below CRQ.

In Fig. 10, a complete 24 h result is displayed where we can observe that during flash crowd scenario (i.e., after every 3 h) CRQ responded in lowest time followed by HRQ and then LRQ. This is due to the reason that in CRQ, 43% of requests are serviced by SG which hold sufficient resources for the requests, while in the case of LRQ 93% requests are serviced in LG which are insufficient during peak hours resulting in high waiting time for the requests. However in normal conditions LRQ serviced the requests in lowest time when compared to CRQ and HRQ.

Discussion

The paper discusses that solitary cloud vendor is unable to allocate resources of different varieties to user. A lot of research is going on to establish an efficient centralized approach and implementation of broker used frequently to serve this purpose. Broker is the central entity that decides which Cloud Service Provider is appropriate for allocating resources to clients depends on various constraints like QoS, response time, execution time etc. If provisioning of resources does not well organized then it results in underutilization and overutilization of resources.

As Centralized approach is commonly used for provisioning of resources where broker is introduced as an intermediary A broker only manages allocation of resources and already available services. Due to improper allocation done by broker, a situation may arise where some of the machines overloaded while other machines are idle or doing very little work, which reduces system performance.

To improve the performance we used a peer-to-peer approach for the replacement of broker, Cloud Service Provider directly communicate with other Cloud Service Provider. In the case of peer-to-peer network, user directly communicates with Cloud Service Provider when there is no centralized approach and user has to choose the kind and number of resources required to fulfill the request. By doing comparative analysis of Latency based Resource Query, Cost based Resource Query and Hybrid Resource Query we realize that peer-to-peer approach is somewhere results better than centralized approach.

We realize that Peer to peer approach results better than centralized approach.

Conclusion

In this study, we utilized a peer-to-peer approach for resource provisioning and resource discovery in inter-cloud. The proposed reputation based algorithm is formulated using DHT. Resource provisioning and efficient resource discovery is a piece of our entire work. We built up an algorithm that distinguish what number of resources are idle in other Cloud Service Providers and served by a specific Cloud Service Provider to another Cloud Service Providers. The proposed algorithm is simulated using CloudSim simulator. We have discussed about the allocation of resources to the different cloud vendor inside a Cloud Data Center. It distributes the resources to needed cloud vendors, which will reduce the overall processing time and waiting time for allocation of resources. This algorithm helps the cloud vendor to select resources from nearby cloud vendors by tracking cost and performance. It selects the resource with appropriate performance and processing time. We additionally talked about the significance of proficient allocation of resources to solve the delay in response time, balance resource utilization and partly load balancing. The proposed technique is advantageous for cloud provider and also for cloud users for saving costs and enhancing performance. The proposed algorithm is evaluated by using the CloudSim simulator toolkit and the run time challenges were attempted to be tackled by implementing the proposed algorithms in cloud environment. This all implemented in peer-to-peer environment. The evaluation for the response time, balance utilization of resources of a host and load distribution on all the hosts are just in the light of the reputation-based technique. So utilizing this strategy for underutilized host and over utilized host, in future work, it can give the quantifiable enhancements on load balancing and provides the server consolidation.

Acknowledgment

The authors would like to thank the reviewers for their constructive comments and suggestions on improving this paper. They also wish to acknowledge the comments and contributions provided by reviewers that greatly strengthened the manuscript.

Author's Contributions

Each author participated sufficiently in the work to take public responsibility for appropriate portions of the content.

Manbir Kaur: Contribute to conception, design, acquisition of data, Analysis and interpretation of data.

Kiranbir Kaur: Contribute in drafting the article or reviewing it critically for significant intellectual content.

Lohit Kapoor: Give final approval of the version to be submitted and any revised version.

Ethics

Author 1,2 and 3 declares that they have no conflict of interest.

References

Nathani, A., S. Chaudharya and G. Somani, 2012. Policy based resource allocation in IaaS cloud. Futur. Gener. Comput. Syst., 28: 94-103. DOI: 10.1016/j.future.2011.05.016

Choi, T.S., Y. Kim and S. Yang, 2013. Graph clustering based provisioning algorithm for optimal inter-cloud service brokering. Proceedings of the 15th Asia-Pacificpp Network Operations and Management Symposium, Sept. 25-27, IEEE Xplore Press, pp: 1-6.

Fajjari, I., N. Aitsaadi, G. Pujolle and H. Zimmermann, 2012. An optimised dynamic resource allocation algorithm for cloud's backbone network. Proceedings of the IEEE 37th Conference on Local Computer Networks, Oct. 22-25, IEEE Xplore Press, pp: 252-255. DOI: 10.1109/LCN.2012.6423621

Grozev, N. and R. Buyya, 2014. Inter-Cloud architectures and application brokering: Taxonomy and survey. ACM Comput. Surv., 47: 369-390.

Liang, H., L.X. Cai, D. Huang, X. Shen and D. Peng, 2012. An SMDP-based service model for interdomain resource allocation in mobile cloud networks. IEEE Trans. Veh. Technol., 61: 2222-2232. DOI: 10.1109/TVT.2012.2194748

Maguluri, S.T., R. Srikant and L. Ying, 2012. Heavy traffic optimal resource allocation algorithms for cloud computing clusters. Proceedings of the 24th International Teletraffic Congress, Sept. 04-07, ITCP, Krakow, Poland, pp: 20-39.

Mechtri, M., D. Zeghlache, E. Zekri and I. Marshall, 2013. Inter-cloud networking gateway architecture. Proceedings of the IEEE 5th International Conference on Cloud Computing Technology and Science, Dec. 2-5, IEEE Xplore Press, pp: 188-194. DOI: 10.1109/CloudCom.2013.124

Papagianni, C., A. Leivadeas, S. Papavassiliou, V. Maglaris and C. Cervello-Pastor *et al.*, 2013. On the optimal allocation of virtual resources in cloud computing networks. IEEE Trans. Comput., 62: 1060-1071. DOI: 10.1109/TC.2013.31

Patel, R. and S. Patel, 2013. A survey on resource allocation strategies in cloud computing. Int. J. Eng. Res. Technol.

Guo, P. and L.L. Bu, 2012. The hierarchical resource management model based on cloud computing. Proceedings of the IEEE Symposium on Electrical and Electronics Engineering, Jun. 24-27, IEEE Xplore Press, pp: 471-474. DOI: 10.1109/EEESym.2012.6258695

Pop, F., C. Dobre, V. Cristea, N. Bessis and F. Xhafa *et al.*, 2009. Reputation-guided evolutionary scheduling algorithm for independent tasks in inter-clouds environments. Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops, Mar. 25-28, IEEE Xplore Press, pp: 1-17. DOI: 10.1109/WAINA.2013.206

Salama, M. and A. Shawish, 2014. A QoS-oriented inter-cloud federation framework. Proceedings of the IEEE 38th Annual Computer Software and Applications Conference, Jul. 21-25, IEEE Xplore Press, pp: 642-643. DOI: 1109/COMPSAC.2014.51

Wu, Q., Q. Zhu, X. Jian and F. Ishikawa, 2014. Broker-based service level agreement-aware composite service provisioning. J. Syst. Software, 96: 194-201. DOI: 10.1016/j.jss.2014.06.027

Xiao, J. and Z. Wang, 2012. A Priority based scheduling strategy for virtual machine allocations in cloud computing environment. Proceedings of the International Conference on Cloud Computing and Service Computing, Nov. 22-24, IEEE Xplore Press, pp: 50-55. DOI: 10.1109/CSC.2012.16

Zaman, S. and D. Grosu, 2012. An online mechanism for dynamic virtual machine provisioning and allocation in clouds. Proceedings of the IEEE 5th International Conference on Cloud Computing, Jun. 24-29, IEEE Xplore Press, pp: 253-260. DOI: 10.1109/CLOUD.2012.26