A Learning Technique for VM Allocation to Resolve Geospatial Queries

Jaydeep Das[1], Arindam Dasgupta[1], Soumya K. Ghosh[2], and Rajkumar Buyya[3]

[1] Advanced Technology Development Centre
Indian Institute of Technology Kharagpur, India
[2] Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur, India
[3] Cloud Computing and Distributed Systems (CLOUDS) Lab
School of Computing and Information Systems
University of Melbourne, Australia
jaydeep@iitkgp.ac.in,adgkgp@gmail.com,skg@iitkgp.ac.in
rbuyya@unimelb.edu.au

Abstract. Provisioning of virtual machines for efficient geospatial query management on cloud is an interesting and challenging work. The aim of this paper is to distribute workloads of different types of spatial queries into suitable virtual machine efficiently. To increase the effectiveness of the system serving geospatial queries, we use real-time geospatial query pattern learning methodology. This methodology is used to train the application specific properties and the system will learn which type of the geospatial query should be allocated what type of virtual machine automatically. The learning methodology gives knowledge about the resource required by each type of geospatial query. Using this understanding, various geospatial query templates are stored in the query template repository for further assistance. This way fast and robust assignment of virtual machine for the geospatial queries is possible which reduces their waiting time.

Keywords: Geospatial Query, VM Allocation, Cloud Computing

1 Introduction

With the major advances of various sensor technologies and wireless techniques, a substantial amount of heterogeneous data with geospatial context has been accumulated by different organizations. Using these geospatial data, the organizations generate meaningful information by processing them. Besides, the use of geospatial information with the advent of smart phone technology has been drastically increased among common people. Thus, in order to facilitate such requirement, a scalable infrastructure is always in demand. In geospatial query operation, a huge amount of geospatial data is required to be accessed which are processed through multiple geospatial operations. For any geospatial query, the amount of processing varies from one location to another. Therefore, for the same query template, the computational resource requirement is always variable.

Cloud computing paradigm provisions such computing and storage resources as and when required to meet Quality of Service (QoS) requirements of applications and users varying with time. The cloud computing platforms provides cost effective, scalable, and minimal management strategy [1, 2]. It is the technology which is used in various geospatial domains. However, the provisioning of cloud resources may not be utilized in efficient way. As a result, the cost of geospatial query becomes high. Therefore, the cloud resources should be assigned for resolving a geospatial query in such a way, so that the cost of the query can be minimized. In any geospatial query, a lot of geospatial services are needed in particular order at different instance of time as the services have been provided by different organizations. Whenever, a geospatial query is requested by a user, the query resolver generates a parse tree to know the required data services and processing applications. Again, if same query is requested by other users, from different geographical area of interest, then different type of workload has been raised.

In most of the previous work, the virtual machine (VM) utilization cost is optimized by considering any one of strategy such as query scheduling [3, 4], resource provisioning [5] and deployment of the query [6, 7] in particular type of VM. However, to optimize the query performance with budget constraints all the strategy must be considered in judicious manner. In this work, a framework has been developed which distributes the cloud resources based on incoming geospatial queries. A learning technique has been applied for selection of VM and query provisioning. A VM allocation has been developed to learn the geospatial service composition pattern. It extracts the most effective heuristic strategy for executing the geospatial query with optimized cost.

2 Related Work

Virtual Machine scheduling for workload management is a mature problem. Many works had been done for VM scheduling to serve query resolution. In literature workloads are dealt with resource provisioning, query placement and query scheduling [8] to minimize the cost. Some authors propose query response time minimization to achieve optimal VM scheduling.

As a weighted profitable within deadline scheduling problem is NP-complete [9], Yun Chi et al.[3] proposed a greedy scheduling algorithm based on a service level agreement tree (SLA-Tree) which helps for scheduling, dispatching and capacity planning. According to SLA, profits of each query are varied based on the query response time. The algorithm is built-in assuming that scheduled queries will happen in near future. The amount of profit will lose or gained after postponement or expedition of actual scheduled query is measured through SLA-Tree.

Cost based scheduling (CBS) [9] and Incremental cost-based scheduling (iCBS) [4] algorithms are based on continuous changing of priority score of queries and execute queries accordingly. iCBS governs the cost depending upon query response time and follows piecewise linear SLA. For many piecewise linear SLAs,

the time complexity in iCBS reduces to O(log N) in contrast to O(N) time complexity of CBS.

In [6], two approximation algorithms of SLA violation penalty for uniform and non-uniform query processing are proposed. where approximation algorithm with dynamic programming is used to improve the quality and reduce the cost of query processing.

3 System Architecture

In order to resolve the spatial query in cloud environment, it is needed to identify the relevant services and consequently selects the suitable VMs. Most of the geospatial queries coming from general users are very specific in nature. Only the data services can be varied in different context. Same types of geospatial queries are grouped together and make a specific template and stored in a repository. Geospatial queries are parsed by query parse tree generator. From the parse tree, system can understand which types of geospatial services are required to resolve those geospatial queries. Geospatial service registry provides the appropriate VM details information where the geospatial query can get execute. A VM allocator has been developed to learn the service patterns of the queries. In this framework, a VM pool contains multiple VMs with different configurations. According to geospatial query from users, the appropriate VMs will be dispatched to resolve the query. In Fig.1 the framework for geospatial query resolution on cloud has been depicted. The major component of the framework has been described below.

- *Spatial query service resolver:* The purpose of this component is to receive geospatial query from users in SQL like format through geospatial query interface. It feed the geospatial query to query parse tree generator and get back parsed query tree. It access the geospatial web service registry to find the location of producer of those services. It also generates a service chain which describes the accessing logic of the geospatial web services to resolve the query.
- *Query parse tree generator:* It generates a geospatial service query tree to identify the spatial feature services and processing services.
- *Query template repository:* Several types of geospatial query templates are deposited here. Templates are basically a different combinations of geospatial services.
- *Spatial service registry:* A spatial service registry is a software component. It supports the run-time discovery and evaluation of resources such as geospatial services, geospatial data-sets, and geospatial application schemes. It is often associated with managed repositories. Spatial query service resolver gets information about available spatial services from spatial service registry.
- *VM allocator:*It schedules and allocates VMs according to availability of geospatial services which helps to resolve the geospatial queries. VM allocator analyses and captures the prevalent service patterns from the historical log.

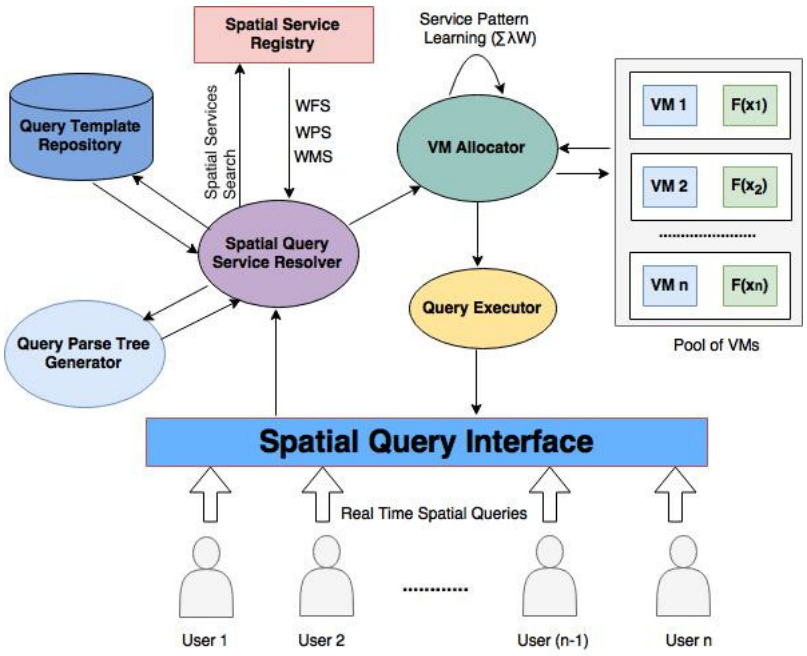

Fig. 1. Spatial query resolution framework

After using optimization algorithm, VMs are allocated for geospatial query resolution. It takes the weighted service requests from the query solver, i.e $< \lambda, w >$, where the (λ) term denotes the time-stamp value when the request is logged and w denotes the weight vector, which represents the computing resources requirement based on the applications. After receiving geospatial queries from the spatial query resolver, it assigns VMs by analysing service patterns. After dispatching VMs to query executor, it checks the VM pools for the availability of VMs, and dispatches if feasible. Otherwise, it stores the VM schedules in a queue and again after any completion of process check for availability. It is observable that sometimes VM scheduling is not possible according to the VM scheduler. It happens because in many real life scenarios, it may take more time to complete a task than the expected time. A feedback function $F(x)= F(x_1)$, $F(x_2)$, ... $F(x_n)$ is proposed to take the real time stamp values and provide feedback to the scheduler to improve the scheduling decisions.

– *VM executor :* After getting the details of allocated VMs and processing schedule geospatial queries, it executes geospatial queries with the help of spatial services in one or several pre-allocated VMs. Results of the geospatial queries return back to the users through spatial query interface.

4 VM Assignment Methodology

In algorithm 1, initially assign a XLarge VM for a new query. If XLarge VMs are not available, geospatial query send into a waiting queue. After execution of a new geospatial query, a feedback contains storage, memory and processing unit details, will create a new geospatial query template and post it into query template repository. If incoming geospatial query is matched with existing geospatial query template, then it assigns to same type of VM for execution. Otherwise, geospatial queries are sent into a waiting queue for further execution.

Algorithm 1: Algorithm for VM assignment of spatial query

Input : SpQry, An incoming spatial query
Output: SpVM, The selected VM for spatial query
$m \leftarrow checkQueryType(SpQry)$;
if *(*$m = null$*)* **then**
  $VM \leftarrow searchVM(XLarge)$;
  if $(VM = null)$ **then**
  waitingQueue.add(SpQry);
  while $((VM \leftarrow searchVM(XLarge)) = null)$;
else
  $VMType \leftarrow searchVMType(m)$;
  $SpVM \leftarrow searchVM(VMType)$;
end
if $(SpVM = null)$
$waitingQueue.add(SpQry)$;
while$((SpVM \leftarrow searchVM(VMType)) = null)$;
return (SpVM);

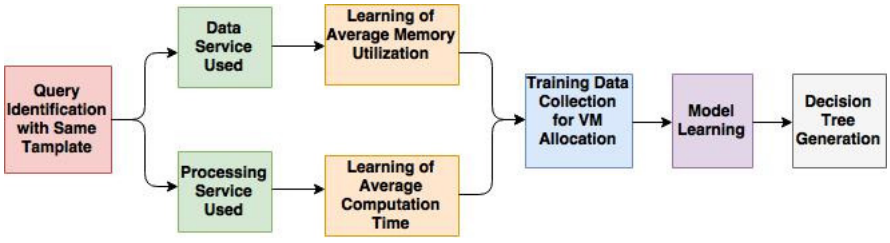

Fig. 2. Decision tree generation technique

– *Resource utilization data capturing :* A geospatial query template is generated with storage amount, processing cores and memory utilization after execution of each geospatial query. This template is getting stored into query template repository. When similar kind of geospatial query will come for execution then it will be easy to allocate a suitable VM for that geospatial query execution.

– *Decision tree generation :* After getting the information i.e. storage amount, processing cores and memory utilization, a decision tree will generate to take the decision about the VM type (XLarge, Large, Regular, Small, XSmall) allocation. This is done by the VMType function in algorithm 1 and it is shown in Fig.2.
– *VM selection :* After generation of a decision tree, an appropriate VM is allocated for new type of geospatial query. If existing geospatial query template is available then VM allocator allocates VM accordingly for geospatial query execution.

5 Performance Evaluation

We have done our experiment in Meghamala[4], private cloud of IIT Kharagpur, with different types of VMs. In order to evaluate the performance of the framework, a prototype of geospatial services along with query resolution component has been implemented in the cloud environment. The data services provide the geospatial data such as road network, land use/land cover, drainage, and settlements. The processing services provide various data processing features such as shortest path, buffer, nearest neighbour and geospatial set operations. These services have been accessed after parsing the geospatial queries. A set of sample geospatial queries based on different geospatial query templates has been created. A program has been deployed to trigger these geospatial queries continuously to the query resolver. For each geospatial query, the trigger time and the response time has been considered to calculate the waiting time through that programme and collected in a file. The number of geospatial query trigger has been increased gradually from 20 to 500. Whereas waiting time is increased to 1150 milliseconds and later it decreased to 640 milliseconds and it varies in between 650 to 700 milliseconds. From Fig. 3, we observe initially geospatial query waiting time is increased, and later it is decreased by increasing the number of geospatial queries.

6 Conclusions and Future Work

In this paper, we proposed a learning based algorithm for VM allocation to resolve geospatial queries with the help of geospatial query templates in cloud platform. From our experiment result, we observe that initially waiting time increases because the geospatial queries are new to the system and VM assignment takes more time. After learning the geospatial queries and generating the query templates, it becomes easy to assign VMs for similar kind of geospatial queries. In future work, we would like to include the priority value which will depend on the frequency of each geospatial query, and monetary cost which will depend on types of VM. It may improve the efficiency of VM allocation technique for geospatial query. We would like to perform these experiments in various public cloud platform such as Amazon EC2, Microsoft Azure, IBM Bluemix.

[4] http://www.sit.iitkgp.ernet.in/Meghamala/

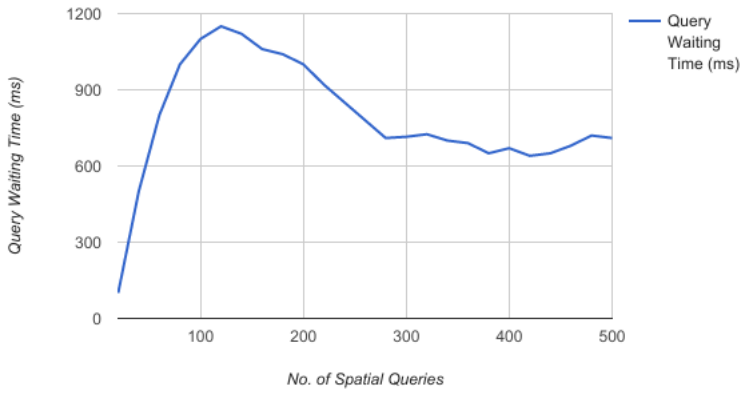

Fig. 3. No. of Spatial Query vs Query Waiting Time

References

1. Dastjerdi, A.V., Buyya, R.: An autonomous time-dependent SLA negotiation strategy for cloud computing. The Computer Journal **58**(11) (2015) 3202–3216
2. Mansouri, Y., Toosi, A.N., Buyya, R.: Cost optimization for dynamic replication and migration of data in cloud data centers. IEEE Transactions on Cloud Computing (2017)
3. Chi, Y., Moon, H.J., Hacigümüş, H., Tatemura, J.: SLA-tree: a framework for efficiently supporting SLA-based decisions in cloud computing. In: Proceedings of the 14th International Conference on Extending Database Technology, ACM (2011) 129–140
4. Chi, Y., Moon, H.J., Hacigümüş, H.: iCBS: incremental cost-based scheduling under piecewise linear slas. Proceedings of the VLDB Endowment **4**(9) (2011) 563–574
5. Jalaparti, V., Ballani, H., Costa, P., Karagiannis, T., Rowstron, A.: Bridging the tenant-provider gap in cloud services. In: Proceedings of the Third ACM Symposium on Cloud Computing, ACM (2012) 10
6. Liu, Z., Hacıgümüş, H., Moon, H.J., Chi, Y., Hsiung, W.P.: PMAX: tenant placement in multitenant databases for profit maximization. In: Proceedings of the 16th international conference on extending database technology, ACM (2013) 442–453
7. Das, J., Dasgupta, A., Ghosh, S.K., Buyya, R.: A geospatial orchestration framework on cloud for processing user queries. In: IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), IEEE (2016) 1–8
8. Marcus, R., Papaemmanouil, O.: WiSeDB: a learning-based workload management advisor for cloud databases. Proceedings of the VLDB Endowment **9**(10) (2016) 780–791
9. Peha, J.M., Tobagi, F.A.: Cost-based scheduling and dropping algorithms to support integrated services. IEEE Transactions on Communications **44**(2) (1996) 192–202