www.ijiarec.com

ISSN: 2348-2079

MAR-2015

International Journal of Intellectual Advancements and Research in Engineering Computations

DYNAMIC ALLOCATION METHOD FOR EFFICIENT LOAD BALANCING IN VIRTUAL MACHINES FOR CLOUD COMPUTING ENVIRONMENT

[1]B.Uvaraja, [2]Dr.N.Shanthi

ABSTRACT

This work proposes a Dynamic resource allocation method for Cloud computing. Cloud computing is a model for delivering information technology services in which resources are retrieved from the internet through web-based tools and applications, rather than a direct connection to a server. Users can set up and boot the required resources and they have to pay only for the required resources. Thus, in the future providing a mechanism for efficient resource management and assignment will be an important objective of Cloud computing. In this project we propose a method, dynamic scheduling and consolidation mechanism that allocate resources based on the load of Virtual Machines (VMs) on Infrastructure as a service (IaaS). This method enables users to dynamically add and/or delete one or more instances on the basis of the load and the conditions specified by the user. Our objective is to develop an effective load balancing algorithm using Virtual Machine Monitoring to maximize or minimize different performance parameters(throughput for example) for the Clouds of dSSifferent sizes (virtual topology de-pending on the application requirement). We develop a set of heuristics that prevent overload in the system effectively while saving energy used. It trace driven simulation and experiment results demonstrate that our algorithm achieves good performance.

INTRODUCTION

Hosting data-intensive query services in the cloud is increasingly popular because of the unique advantages in scalability and cost-saving. With the cloud infrastructures, the service owners can conveniently scale up or down the service and only pay for the hours of using the servers. This is an attractive feature because the workloads of query services are highly dynamic, and it will be expensive and inefficient to serve such dynamic workloads with in-house infrastructures [2]. However, because the service providers lose the control over the data in the cloud, data confidentiality and query privacy have become the major concerns. Adversaries, such as curious service providers, can possibly make a copy of the database or eavesdrop users' queries, which will be difficult to detect and prevent in the cloud infrastructures. While new approaches are needed to preserve data confidentiality and query privacy, the efficiency of query services and the benefits of using the clouds should also be preserved. It will not be meaningful to provide slow query services as a result of security and privacy assurance. It is also not practical for the data owner to use a significant amount of in house resources, because the purpose of using cloud resources is to reduce the need of maintaining scalable in-house infrastructures.

Therefore, there is an intricate relationship among the data confidentiality, query privacy, the quality of service, and the economics of using the cloud. For example, the crypto-index [12] and Order Preserving Encryption (OPE) [1] are vulnerable to the attacks. The enhanced crypto-index approach [14] puts heavy burden on the in-house infrastructure to improve the security and privacy. The New Casper approach [24] uses cloaking boxes to protect data objects and

Author for Correspondence:
[1]PG Scholar, Department of CSE, Nandha Engineering College, Erode, India, Email: Bsku.cse@gmail.com.
[2]Dean & Professor, Department of CSE, Nandha Engineering College, Erode, India. Email: shanthimoorthi@yahoo.com.

queries, which affects the efficiency of query processing and the in-house workload. We have summarized the weaknesses of the existing approaches in Section 7. We propose the Random Space Perturbation (RASP) approach to constructing practical range query and k-nearest-neighbour (kNN) query services in the cloud. The proposed approach will address all the IEEE four aspects of the CPEL criteria and aim to achieve a good balance on them. We also introduce the framework for constructing the query services with the RASP perturbation. In Section 4 we describe the algorithm for transforming queries and processing range queries. In Section 5, the range query service is extended to handle kNN queries. When describing these two services, we also analyze the attacks on the query privacy. Finally, we present some related approaches in Section 7 and analyze their weaknesses in terms of the CPEL criteria.

RELATED WORKS

In this section, we discuss about VM Allocation and VM Load Balancing.Cloud computing is a paradigm in which IT (information technology) application provide as a service. Cloud Computing allows users to utilize the computation, storage, data and services from around the world in commercialize Manner. In cloud environment, scheduling is the major issue. Scheduling is responsible efficient utilization of the Resources. Here , a Scheduling model based on minimum network delay using Suffrage Heuristic coupled with Genetic algorithms for scheduling sets of independent jobs algorithm is proposed, the objective is to minimize the make Span.

Liang Sun et al. It presents a genetic algorithm with a penalty function for the job shop scheduling problem. In the context of proposed algorithm, a clonal selection based hyper mutation and a life span extended strategy is designed. During the search process, an adaptive penalty function is designed so that the algorithm can search in both feasible and infeasible regions of the solution space. Simulated experiments were conducted on 23 benchmark instances taken from the OR-library. The results show the effectiveness of the proposed algorithm.

Alam and Nithita Das etal.In multiprocessor systems, one of the main factors of systems' performance is task scheduling. The well The task be distributed among the processors the well be the performance. Again finding the optimal Solution of scheduling the tasks into the processors is NP-complete, that is, it will take a lot of time to find The optimal solution. Many evolutionary algorithms (e.g. Genetic Algorithm, Simulated annealing) are Used to reach the near optimal solution in linear time. We propose a heuristic for genetic Algorithm based task scheduling in multiprocessor systems by choosing the eligible processor on Educated guess. From comparison it is found that this new heuristic based GA takes less computation Time to reach the suboptimal solution.

Rashmi, etal.Cloud computing, undoubtedly, has become the buzzword in the IT industry today. Looking at the potential Impact it has on numerous business applications as well as in our everyday life, it can certainly be said that This disruptive technology is here to stay. Many of the features that make cloud computing attractive, have Not just challenged the existing security system, but have also revealed new security issues. It Provides an insightful analysis of the existing status on cloud computing security issues based on a detailed Survey carried by the author. It also makes an attempt to describe the security challenges in Software as a Service (saas) model of cloud computing and also endeavors to provide future security research directions.

ALGORITHM

Cloud computing infrastructure is the massive deployment of virtualization tools and techniques as it has an extra layer i.e. Virtualization layer that acts as a creation, execution, management, and hosting environment for application services. The modeled VMs in the above virtual environment are contextually isolated but still they need to share computing resources -processing cores, system bus etc. Hence, the amount of hardware resources available to each VM is constrained by the total processing power i.e. CPU, the memory and system bandwidth available within the host.

CONTEMPORARY VM LOAD BALANCERS

Virtual machine enables the abstraction of an OS and Application running on it from the hardware. The interior hardware infrastructure services interrelated to the Clouds is modeled in the CloudSim simulator by a Datacenterelement for handling service requests. These requests are

application elements sandboxed within VMs, which need to be allocated a share of processing power on Datacenters host components. Datacenter object manages the datacenter management activities such as VM creation and destruction and does the routing of user requests received from User Bases via the Internet to the VMs. The Data Center Controller uses a VmLoadBalancer to determine which VM should be assigned the next request for processing. The contemporary Vmloadbalancer are Round Robin, throttled and active monitoring load balancing algorithms.

A) **ROUND ROBIN LOAD BALANCER (RRLB):** In this, the datacenter ID controller assigns the requests to a list of VMs on a rotating basis. The first request is allocated to a VM-picked randomly from the group and then the Datacenter controller assigns the subsequent requests in a circular order. Once the VM is assigned the request, the VM is moved to the end of the list.

B) **THROTTLED LOAD BALANCER (TLB):** The TLB maintains a record of the state of each virtual machine (busy/ideal). If a request arrived concerning the allocation of virtual machine, the TLB sends the ID of ideal virtual machine to the datacenter controller and datacenter controller allocates the ideal virtual machine.

C) **ACTIVE MONITORING LOAD BALANCER (AMLB):** The AMLB maintains information about each VMs and the number of requests currently allocated to which VM. When a request to allocate a new VM arrives, it identifies the least loaded VM. If there are more than one, the first identified is selected. ActiveVmLoadBalancer returns the VM id to the Data Center controller. The data Center Controller sends the request to the VM identified by that id. DataCenter Controller notifies the ActiveVmLoadBalancer of the new allocation and cloudlet is sent to it.

Modules

- Datacenter
- Datacenter Broker
- Host
- VM
- Cloudlet

Module Description

- Datacenter

  Datacenter is composed of a set of hosts and it is responsible for managing virtual machines (VMs) (e.g., VM provisioning). It behaves like an IaaS provider by receiving requests for VMs from brokers and creating the VMs in hosts.

- **DATACENTERBROKER**

  This class represents a broker acting on behalf of a user. It modifies two mechanisms: the mechanism for submitting VM provisioning requests to data centers and the mechanism for submitting the tasks to VMs.

- **HOST**

  Host executes actions related to management of VMs (e.g., creation and destruction) and update task processing to VMs. A host has a defined policy for provisioning memory, processing elements, and bandwidth to virtual machines. A host is associated to a datacenter. It can host virtual machines.

- **VM**

  It represents a software implementation of a machine that executes applications called virtual machine (VM) which works like a physical machine. Each virtual machine divides the resources received from the host among tasks running on it.

- **CLOUDLET**

  A cloudlet class is also known as a task. CloudSim represents the complexity of an application in terms of its computational requirements. This class is managed by the scheduling policy which is implemented in Datacenter Broker Class.

PROPOSED SECHEM

QUERY SERVICES IN THE CLOUD

This section presents the notations, the system architecture, and the threat model for the RASP approach, and prepares for the security analysis in later sections. The design of the system architecture keeps the cloud economics in mind so that most data storage services and computing tasks will be done in the cloud. The threat model makes realistic security assumptions and clearly defines the practical threats that the RASP approach will address.

DEFINITIONS AND NOTATIONS

First, we establish the notations. For simplicity, we consider only single database tables, which can be the result of denormalization from multiple relations. A database table consists of n records and d searchable attributes. We also frequently refer to an attribute as a dimension or a column, which are exchangeable in the paper. Each record can be represented as a vector in the multidimensional space, denoted by low case letters. If a record x is d-dimensional, we say x ∈Rd, where Rd means the d-dimensional vector space. A table is also treated as a d × n matrix, with records represented as column vectors. We use capital letters to represent a table, and indexed capital letters, e.g., Xi, to represent columns. Each column is defined on a numerical domain. Let Tbe a table and Xi, Xj , and Xk be the real valued attributes in T , and a and b be some constants. Take the counting query for example. A typical range query looks like *select count(*) from Twhere*Xi ∈[ai, bi] *and* Xj∈(aj, bj) *and* Xk =ak, which calculates the number of records in the rangedefined by conditions on Xi, Xj , and Xk. Range queries may be applied to arbitrary number of attributes and conditions on these attributes combined with conditional operators "and"/"or". We call each part of the query condition that involves only one attribute as a *simple condition*. A simple condition like Xi ∈[ai, bi] can be described with two half space conditions Xi ≤ bi and −Xi ≤ −ai. Without loss of generality, we will discuss how to process half space conditions like Xi ≤ bi in this paper. A slight modification will extend the discussed algorithms to handle other conditions like Xi < bi and Xi = bi. kNN query is to find the closest k records to the query point, where the Euclidean distance is often used to measure the proximity. It is frequently usedin location-based services for searching the objects close to a query point, and also in machine learning algorithms such as hierarchical clustering and kNN classifier. A kNN query consists of the query point and the number of nearest neighbours.

SYSTEM ARCHITECTURE

We assume that a cloud computing infrastructure, such as Amazon EC2, is used to host the query services and large datasets. The purpose of this architecture is to extend the proprietary database servers to the public cloud, or use a hybrid private-public cloud to achieve scalability and reduce costs while maintaining confidentiality. There are two clearly separated groups: the trusted parties and the untrusted parties. The trusted parties include the data/service owner, the in-house proxy server, and the authorized users who can only submit queries. The data owner exports the perturbed data to the cloud. Meanwhile, the authorized users can submit range queries or kNN queries to learn statistics or find some records. The untrusted parties include the curious cloud provider who hosts the query services and the protected database. The RASP-perturbed data will be used to build indices to support query processing. There are a number of basic procedures in this framework:

(1) F(D) is the RASP perturbation that transforms the original data D to the perturbed data D′;

(2) Q(q) transforms the original query q to the protected formq′ that can be processed on the perturbed data;

(3) H(q′,D′) is the query processing algorithm that returns the result R′.

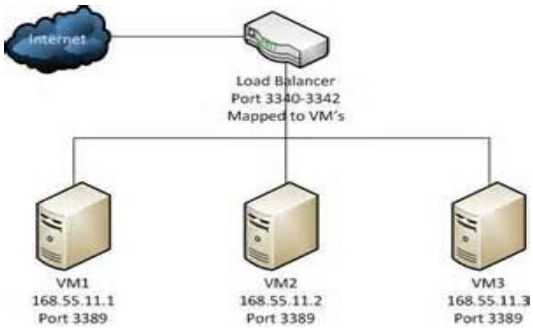


Figure1. Architecture of VM load balancing

When the statistics such as SUM or AVG of a specific dimension are needed, RASP can work with partial homomorphic encryption such as Paillierencryption to compute these statistics on the encrypted data, which are then recovered with the procedure G(R′).

CLOUDSIM ARCHITECTURE

Figure 3 shows the multi-layered design of the CloudSim software framework and its architectural components. Initial releases of CloudSim used SimJava as the discrete event simulation engine that supports several core functionalities, such as queuing and processing of events, creation of Cloud system entities (services, host, data center, broker, VMs), communication between components,and management of the simulation clock. However in the current release, the SimJava layer has been removed in order to allow some advanced operations that are not supported by it. We provide finer discussion on these advanced operations in the next section.

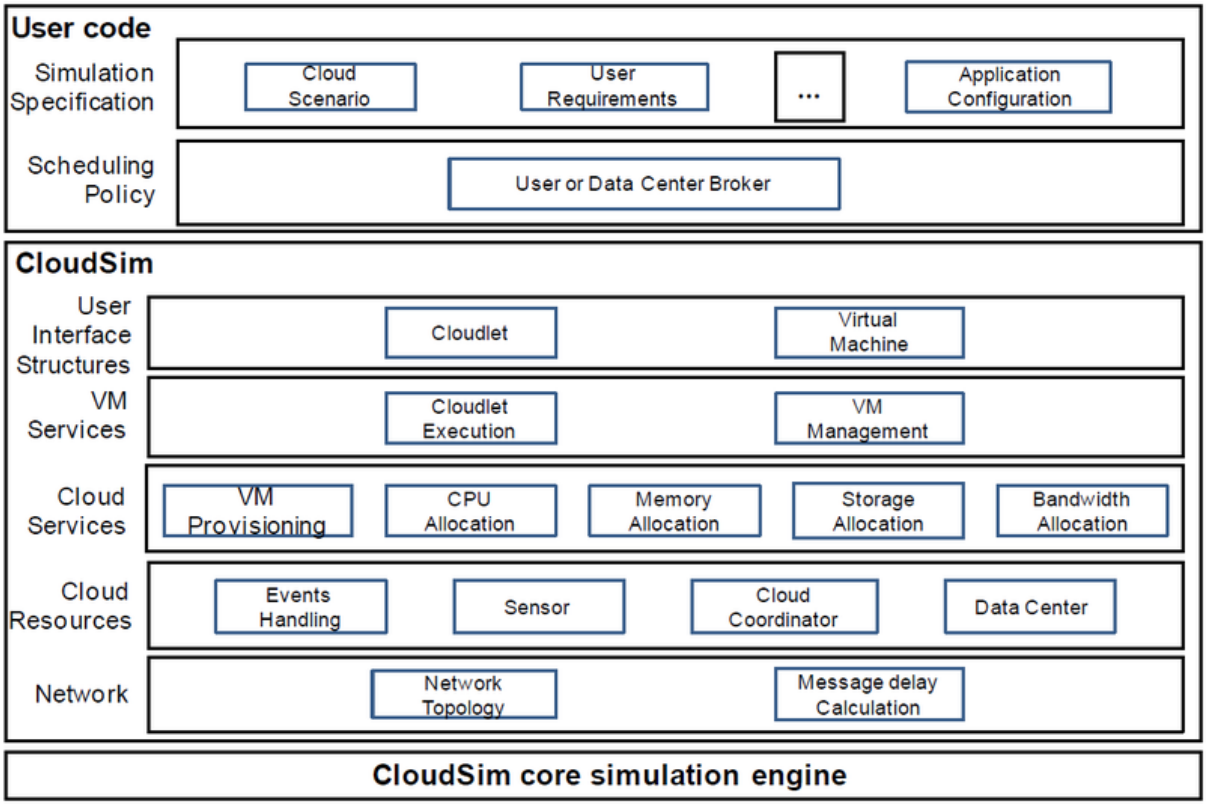


Figure 3.1 Architecture of CloudSim

The CloudSim simulation layer provides support for modeling and simulation of virtualized Cloud-based data center environments including dedicated management interfaces for VMs memory, storage, and bandwidth. The fundamental issues, such as provisioning of hosts to VMs, managing application execution, and monitoring dynamic system state, are handled by this layer. A Cloud

provider, who wants to study the efficiency of different policies in allocating its hosts to

VMs (VM provisioning), would need to implement his strategies at this layer. Such implementation can be done by programmatically extending the core VM provisioning functionality. There is a clear distinction at this layer related to provisioning of hosts to VMs. A Cloud host can be concurrently allocated to a set of VMs that execute applications based on SaaS provider's defined QoS levels. This layer also exposes the functionalities that a Cloud application developer can extend to perform complex workload profiling and application performance study. The top-most layer in the CloudSim stack is the User Code that exposes basic entities for hosts (number of machines, their specification, and so on), applications (number of tasks and their requirements), VMs, number of users and their application types, and broker scheduling policies. By extending the basic entities given at this layer, a Cloud application developer can perform the following activities: (i) generate a mix of workload request distributions, application configurations; (ii) model Cloud availability scenarios and perform robust tests based on the custom configurations; and (iii) implement custom application provisioning techniques for clouds and their federation.

As Cloud computing is still an emerging paradigm for distributed computing, there is a lack of defined standards, tools, and methods that can efficiently tackle the infrastructure and applicationlevel complexities. Hence, in the near future there will be a number of research efforts both in the academia and industry toward defining core algorithms, policies, and application benchmarking based on execution contexts. By extending the basic functionalities already exposed to EventsHandlingCloudSim, researchers will be able to perform tests based on specific scenarios and configurations,thereby allowing the development of best practices in all the critical aspects related to Cloud Computing.

MODELING THE CLOUD

The infrastructure-level services (IaaS) related to the clouds can be simulated by extending the data center entity of CloudSim. The data center entity manages a number of host entities. The hosts are assigned to one or more VMs based on a VM allocation policy that should be defined by the Cloud service provider. Here, the VM policy stands for the operations control policies related to VM life cycle such as: provisioning of a host to a VM, VM creation, VM destruction, and VM migration. Similarly, one or more application services can be provisioned within a single VM instance, referred to as application provisioning in the context of Cloud computing. In the context of CloudSim, an entity is an instance of a component. A CloudSim component can be a class (abstract or complete) or set of classes that represent one CloudSim model (data center, host).

A data center can manage several hosts that in turn manages VMs during their life cycles. Host is a CloudSim component that represents a physical computing server in a Cloud: it is assigned a pre-configured processing capability (expressed in millions of instructions per second—MIPS), memory, storage, and a provisioning policy for allocating processing cores to VMs. The Host component implements interfaces that support modeling and simulation of both single-core and multi-core nodes.

VM allocation (provisioning) is the process of creating VM instances on hosts that match the critical characteristics (storage, memory), configurations (software environment), and requirements (availability zone) of the SaaS provider. CloudSim supports the development of custom application service models that can be deployed within a VM instance and its users are required to extend the core Cloudlet object for implementing their application services. Furthermore, CloudSim does not enforce any limitation on the service models or provisioning techniques that developers want to implement and perform tests with. Once an application service is defined and modeled, it is assigned to one or more pre-instantiated VMs through a service-specific allocation policy.Allocation of application-specific VMs to hosts in a Cloud-based data center is the responsibility of a VM Allocation controller component (called VmAllocationPolicy). This component exposes a number of custom methods for researchers and developers who aid in the implementation of new policies based on optimization goals (user centric, system centric, or

both). By default, VmAllocationPolicy implements a straightforward policy that allocates VMs to the Host on a First-Come-First-Serve (FCFS) basis. Hardware requirements, such as the number of processing cores, memory, and storage, form the basis for such provisioning. Other policies, including the ones likely to be expressed by Cloud providers, can also be easily simulated and modeled in CloudSim. However, policies used by public Cloud providers (Amazon EC2, Microsoft Azure) are not publicly available, and thus a pre-implemented version of these algorithms is not provided with the CloudSim.

MODELING THE VM ALLOCATION

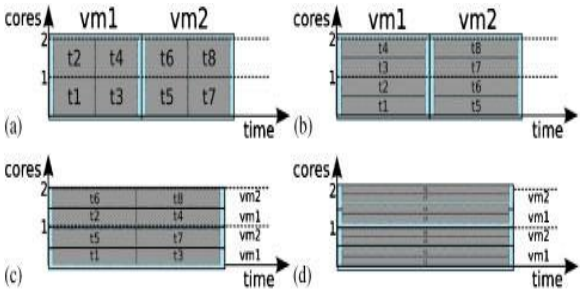
One of the key aspects that make a Cloud computing infrastructure different from a Grid computing infrastructure is the massive deployment of virtualization tools and technologies. Hence, as against Grids, Clouds contain an extra layer (the virtualization layer) that acts as an execution, management,and hosting environment for application services. Hence, traditional application provisioning models that assign individual application elements to computing nodes do not accurately represent the computational abstraction, which is commonly associated with Cloud resources.There is a requirement of concurrently instantiating two VMs on that host. Although in practice VMs are contextually (physical and secondary memory space) isolated, still they need to share the processing cores and system bus.Hence, the amount of hardware resources available to eachVMis constrained by the total processing power and system bandwidth available within the host. This critical factor must be considered during the VM provisioning process, to avoid creation of a VM that demands more processing power than is available within the host. In order to allow simulation of different provisioning policies under varying levels of performance isolation, CloudSim supports VM provisioning at two levels: first, at the host level and second, at the VM level. At the host level, it is possible to specify how much of the overall processing power of each core will be assigned to each VM.

At the VM level, the VM assigns a fixed amount of the available processing power to the individual application services (task units) that are hosted within its execution engine. For the purpose of this paper, we consider a task unit as a finer abstraction of an application service being hosted inthe VM.

At each level, CloudSim implements the time-shared and space-shared provisioning policies. To clearly illustrate the difference between these policies and their effect on the application service performance, in Figure 4 we show a simple VM provisioning scenario. In this figure, a host with two CPU cores receives request for hosting two VMs, such that each one requires two cores and plans to host four tasks' units. More specifically, tasks t1, t2, t3, and t4 to be hosted in VM1,whereas t5, t6, t7, and t8 to be hosted in VM2.

presents a provisioning scenario, where the space-shared policy is applied to both VMs and task units. As each VM requires two cores, in space-shared mode only one VM can run at a given instance of time. Therefore, VM2 can only be assigned the core once VM1 finishes the execution of task units. The same happens for provisioning tasks within the VM1: since each task unit demands only one core, therefore both of them can run simultaneously. During this period, the remaining tasks (2 and 3) wait in the execution queue. By using a space-shared policy, the estimated finish time of a task p managed by a VM i is given bywhere est(p) is the Cloudlet- (cloud task) estimated start time and rlis the total number of an

instructions that the Cloudlet will need to execute on a processor. The estimated start time depends Effects of different provisioning policies on task unit execution:



(a) space-shared with provisioning for VMs and tasks; (b) space-shared provisioning for VMs and time-shared provisioning for tasks; (c) time-shared provisioning for VMs, space-shared provisioning for

tasks; and (d) time-shared provisioning for VMs and tasks. On the position of the Cloudlet in the execution queue, because the processing unit is used exclusively (space-shared mode) by the Cloudlet. Cloudlets are put in the queue when there are free processing cores available that can be assigned to the VM. In this policy, the total capacity of a host having np processing elements (PEs) is given by: where cap(i) is the processing strength of individual elements.

MODELING THE CLOUD MARKET

Market is a crucial component of the Cloud computing ecosystem; it is necessary for regulating Cloud resource trading and online negotiations in a public Cloud computing model, where services are offered in a pay-as-you-go model. Hence, research studies that can accurately evaluate the costto-benefit ratio of emerging Cloud computing platforms are required. Furthermore, SaaS providers need transparent mechanisms to discover various Cloud providers' offerings (IaaS, PaaS, SaaS,and their associated costs). Thus, modeling of costs and economic policies are important aspects to be considered when designing a Cloud simulator. The Cloud market is modeled based on a multi-layered (two layers) design. The first layer contains the economic of features related to the IaaS model such as cost per unit of memory, cost per unit of storage, and cost per unit of used bandwidth. Cloud customers (SaaS providers) have to pay for the costs of memory and storage when they create and instantiate VMs, whereas the costs for network usage are only incurred in the event of data transfer. The second layer models the cost metrics related to SaaS model. Costs at this layer are directly applicable to the task units (application service requests) that are served by the application services. Hence, if a Cloud customer provisions a VM without an application service (task unit), then they would only be charged for layer 1 resources (i.e. the costs of memory and storage). This behavior may be changed or extended by CloudSim users.

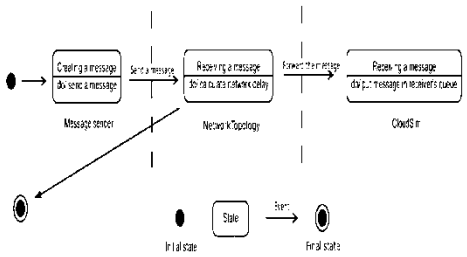


Figure 3.2. Network communication flow.

MODELING DYNAMIC WORKLOADS

Software developers and third-party service providers often deploy applications that exhibit dynamic behavior in terms of workload patterns, availability, and scalability requirements.Typically, Cloud computing thrives on highly varied and elastic services and infrastructure demands. Leading Cloud vendors, including Amazon and Azure, expose VM containers templates to host a range of SaaS types and provide SaaS providers with the notion of unlimited resource pool that can be leased on the fly with requested configurations. Pertaining to the aforementioned facts, it is an important requirement that any simulation environment supports the modeling of dynamic workload patterns driven by application or SaaS models. In order to allow simulation of dynamic behaviors within CloudSim, we have made a number of extensions to the existing framework, in particular to the Cloudlet entity. We have designed an additional simulation entity within CloudSim, which is referred to as the Utilization Model that exposes methods and variables for defining the resource and VM-level requirements of a SaaS application at the instance of deployment.

Another important requirement for Cloud computing environments is to ensure that the agreed SLA in terms of QoS parameters, such as availability, reliability, and throughput, are delivered to the applications.Lack of intelligent methodologies for VM provisioning raises a risk that all VMs deployed on a single host may not get the adequate amount of processor share that is essential for fulfilling the agreed SLAs. This may lead to performance loss in terms of response time, time outs, or failures in the worst case. The resource provider must take into account such behaviors and initiate necessary actions

to minimize the effect on the application performance. To simulate such behavior, the SLA model can either be defined as fully allocating the requested amount of resources or allowing flexible resource allocations up to a specific rate as long as the agreed SLA can be delivered (e.g. allowing the CPU share to be 10% below the requested amount). CloudSim supports modeling of the aforementioned SLA violation scenarios. Moreover, it is possible to define particular SLA-aware policies describing how the available capacity is distributed among competing VMs in case of a lack of resources. The number of SLA violation events as well as the amount of resource that was requested but not allocated can be accounted for by CloudSim.

EXPERIMENTS

DATASETS

Three datasets are used in experiments. (1) A synthetic dataset that draws samples from uniform distribution in the range . (2) The Adult dataset from UCI machine learning database5. We assign numeric values to the categorical values using a simple one to-one mapping scheme.

COST OF RASP PERTURBATION

The major costs can be divided into two parts: the OPE and the rest part of RASP. We implement a simple OPE scheme by mapping original column distributions to normal distributions. The OPE algorithm partitions the target distribution into buckets. Then, the sorted original values are proportionally partitioned according to the target bucket distribution to create the buckets for the original distribution. With the aligned original and target buckets, an original value can be mapped to the target bucket and appropriately scaled.

PERFORMANCE OF TWO-STAGE RANGE QUERY PROCESSING

In this set of experiments, we study the performance aspects of polyhedron-based range query processing. The block size is 4KB and we allow each block to contain only 20 entries to mimic a large database with many disk blocks. Samples from the original databases indifferent size (10,000 − 50,000 records, i.e., 500-2500data blocks) are perturbed and indexed for query processing. Another set of indices is also built on the original data for the performance comparison with non-perturbed query processing. We will use the number of disk block accesses, including index blocks and data blocks, to assess the performance to avoid the possible variation caused by other parts of the computer system. In addition, we will also show the wall-clock time for some results.

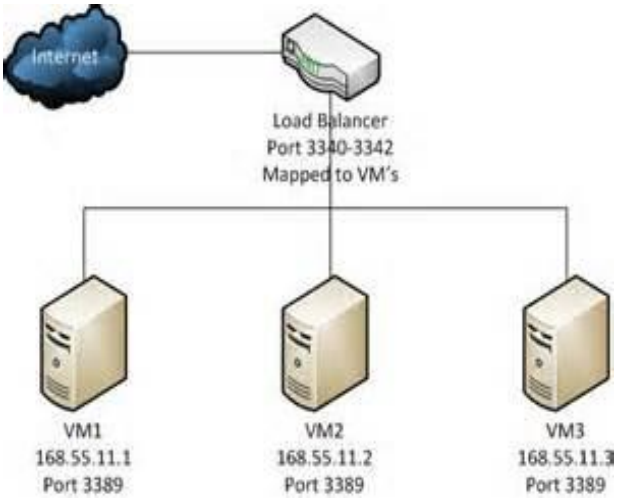


Figure 5.1. Architecture of VM load balancing

Recall the two-stage processing strategy: using the MBR to search the indexing tree, and filtering the returned result with the secured query in quadratic form. We will study the performance of the first stage by comparing it to two additional methods:

(1) the original queries with the index built on the original data, which is used to identify how much additional cost is paid for querying the MBR of the transformed query;

(2) the linear scan approach, which is the worst case cost. Range queries are generated randomly within the domain of the datasets, and then transformed with the to observe the effect of the scale of the range to the performance of query processing.

CONCLUSION

We propose the RASP perturbation approach to hosting query services in the cloud, which gratify the CPEL criteria: data Confident, query Privacy, Efficient query processing, and Low in-house workload. The goal of managing multiple virtualization platforms and multiple virtual machine migrations across physical machines without disruption method is achieved. We discuss that ensure load balance when multiple virtual machines run on multiple physical machines.We also conduct several sets of experiments to show the efficiency of query processing and the low cost of in-house processing.

REFERENCES

[1] TarunGoyal&AakankshaAgrawal ”Host Schudeling Using Genitic Algorithm In Cloud Computing Environment” International Journal Of Rearch In Enineering& Tecnology.,Vol.1,Issue1,June 2013

[2] Liang Sun,Xiaochun Cheng ”Solving Job Scheduling Problem Using Genitic Algorithm With Penalty Function”International Journal Of Intelligent Information Processing., Vol1,No2,December 2010

[3] ProbirRoy,Md.MejbahUiAlam And Nithita Das ”Huristic Based Task Scheduling In Multiprocessor Systems With Genetic Algorithm By Choosing The Eligible Processor” International Journal Of Distributed And Paralle Systems.,Vol.3,No.4,July 2012

[4] Rashmi, Dr.G.Sahoo, Dr.S.Mehfuz ”Securing Software As A Service Model Of Cloud Computing: Issues And Solutions., International Journal On Cloud Computing: Services And Architecture,Vol.3,No.4,August 2013

[5] S.StephenVaithiya And S.MarySairaBhanu “Zone Based Job Scheduling In Mobile Grid Environment” International Journal Of Grid Computing & Application.,Vol.3,No.2,June 2012.

[6] Pinky Rosemarry,RavinderSingh,PayalSinghal And DilipSisodia “Grouping Based Job Scheduling Algorithm Using Priority Queue And Hybrid Algorithm In Grid Computing” International Journal Of Grid Computing & Application., Vol.3,No.4,December 2012

[7] VandanaChoudhary,SaurabhKacker “An Approach To Improve Task Scheduling In A Decentrelized Cloud Computing Environment” International Journal Computer Technology& Applications.,Vol.3,Jan-Feb 2012.

[8] ArabiE.Keshk, Ashraf Ei-Sisi, Medhata.Tawfeek, F.A.Torkey “Intelligent Strategy Of Task Scheduling In Cloud Computing For Load Balanceing” International Journal Of Emerging Trends & Technology In Computer Science.,Vol 2,Issue 6,November-December 2013.

[9] ArabiE.Keshk, Ashraf Ei-Sisi, Medhata.Tawfeek, F.A.Torkey “Intelligent Strategy Of Task Scheduling In Cloud Computing For Load Balanceing” International Journal Of Emerging Trends & Technology In Computer Science.,Vol 2,Issue 6,November-December 2013.

[10] AmandeepKaur Sidhu1, Supriya Kinger2, “A Sophisticated Approach For Job Scheduling In Cloud Server” International Journal Of Computer Trends And Technology (Ijctt) – Volume 4,Issue 7–July 2013.

[11] Pardeep Kumar*, AmandeepVerma, “ Independent Task Scheduling In Cloud Computing By Improved Genetic Algorithm” Independent Task Scheduling In Cloud Computing By Improved Genetic Algorithm.,Volume 2, Issue 5, May 2012

[12] JaliyaEkanayake, Student Member, Ieee, ThilinaGunarathne, Student Member, Ieee, And Judy Qiu “Cloud Technologies Forbioinformatics Applications” IEEE Transactions On Parallel And Distributed Systems, Vol. 22, No. 6, June 2011.

[13] David Carrera, Member, Ieee, MalgorzataSteinder, Member, Ieee, Ian Whalley, Jordi Torres, Member, Ieee, And Eduard Ayguade “Autonomic Placement Of Mixed Batch And Transactional Workloads”

IEEE Transactions On Parallel And Distributed Systems, Vol. 23, No. 2, February 2012.

[14] Luiz F. Bittencourt, Edmundo R. M. Madeira, And Nelson L. S. Da Fonseca, University Of Campinas "Scheduling In Hybrid Clouds" IEEE Communications Magazine September 2012.