MODELING OF LOW COST, HIGH EFFICIENT CLOUD DATA CENTER BY EFFECTIVE COMBINATION OF REMOTE CUSTOMER PROVIDED RESOURCES

M.Ebenezer Chatterji,
M.E Scholar (M.E Communication and Networking), Cape Institute of Technology
Email: Ebenchatt@gmail.com

ABSTRACT- Distributed computing is a method for giving registering as a support of its clients. Numerous association, for example, Amazon, Microsoft, and Google and so on are demonstrating Cloud Computing administration to its customers. These days this datacenter are giving great figuring assets to their costumers to execute even an exceptionally intricate and logical errand remotely on the Cloud server. Despite the fact that the Cloud Computing is giving exceptionally productive processing framework, to getting to the great cloud administration purchaser need to pay a considerable measure to their Cloud administration suppliers. Progressively even a shopper is having great processing assets locally however in the business Cloud Computing Environment, administration suppliers don't know about nearby costumer assets. This venture concentrate on the neighborhood client assets and configuration a structure for leasing the nearby client assets to the Cloud Service supplier so that cloud client can likewise win from cloud furthermore can utilize their assets all the more effectively. This undertaking use Spot Cloud idea to coordinate the costumer neighborhood assets into the cloud, in order to empower the client to offer, purchase, and utilize this assets all the more productively. The most complex thing in this undertaking is that the asset accessible from various shopper is not indistinguishable that is the accessible assets are heterogeneous so this need an exceptionally proficient scaling calculation to make this assets to use effectively. The two potential difficulties face by this undertaking is the way the costumer can be roused to contribute or to use this assets. By what means can a high administration accessibility be guaranteed out of the dynamic assets. To guarantee proficient administrations, the ideal asset provisioning calculation is utilized.

Keywords: Cloud Computing, Spot Cloud, Cluster Computing, Virtualization.

I. INTRODUCTION

Late advances in distributed computing offers a proficient means for giving processing as a type of utility. Such undertaking cloud suppliers as Amazon, Google, and Microsoft have appreciated noteworthy increment of their client populaces, upholding them to always update and grow their datacenter infrastructures1. Yet, the current venture cloud limit is still a small amount of the need when we consider the quick development of the clients' interest. Late studies propose that the client experience of big business mists, for example, Amazon EC2, is for sure decreasing2. Then again, the clients' neighborhood processing assets are still quickly advancing. Today's propelled customer CPUs, similar to the Intel's Core i7, is no slower than huge numbers of the server CPUs a couple of years back; this CPU is significantly speedier than the majority of the medium or even some huge occasions in today's undertaking cloud.

The amassed calculation, stockpiling, and system assets accessible at cloud clients are without a doubt more than that at a run of the mill datacenter. At the end of the day, the cloud as a tricky stage is not just because of the inexhaustible assets accessible at a remote area; yet taking care of asset demand is a key variable to the expense of keeping up datacenters and giving cloud administrations, and the subsequent administration costs regularly prevent clients from relocating to the cloud3. There have been late studies on brilliant administration dividing that keeps certain errands neighborhood [1]. We however imagine a more broad arrangement that consistently incorporates the clients' neighborhood assets into the cloud stage, empowering them to offer, purchase, and use these assets, in this manner offering a more adaptable and less cloud. In this paper, we make a first stride

towards the possibility and the framework configuration of empowering customerprovided assets for distributed computing. We display the point by point outline of SpotCloud4, a genuine working framework that empowers clients to flawlessly contribute their assets to the general cloud. Since its arrangement in November 2010, SpotCloud has pulled in clients around the world. In this paper, we diagram the SpotCloud plan and, through traceanalysis, exhibit it as a promising supplement to the datacenter-based cloud. Specifically, it offers cloud administration with adaptable and moderately bring down expense but then equivalent execution to best in class venture mists. Given that these nearby assets are exceptionally heterogeneous and dynamic, we nearly look at two basic difficulties in this new setting:
(1) *How can the customers be motivated to contribute or utilize such resources?*and
(2) How can high service availability be ensured out of the dynamic resources?

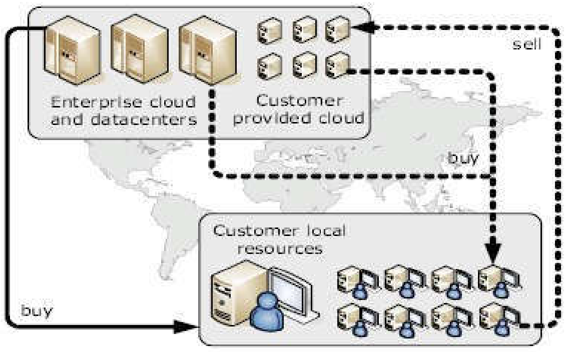


Fig. 1: Overview of Framework

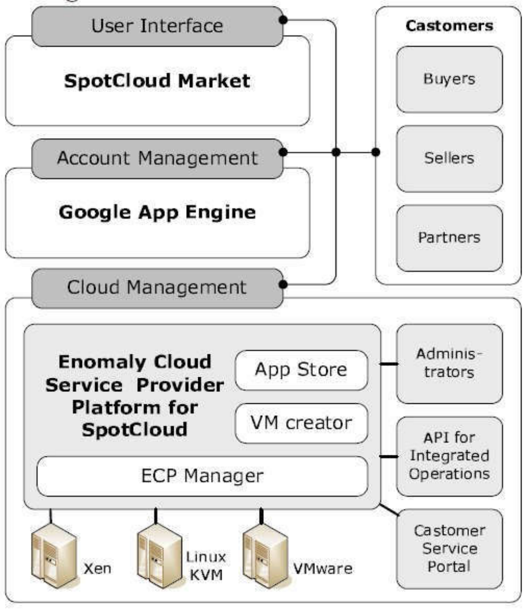


Fig. 2: Software module design

We demonstrate a distributed market for potential sellers to flexibly and adaptively determine their resource prices through a repeated seller competition game. We also present an optimal resource provisioning algorithm that ensures service availability with minimized lease and migration costs. We further examine the lease and migration costs in the presence of dynamic resource availability in the real deployment, and highlight the trade-offs therein. The rest of this paper is organized as follows: In Section

2, we present the related works. Section 3 discuss the framework design as well as the challenges. After that, we examine the pricing problem and the resource provisioning problem in Section 4 and 5, respectively. Section 6 presents the SpotCloud design and its performance result. We further investigate the cost and availability issues in the system in 7. Finally, Section 8 concludes the paper.

II. RELATED WORKS

The notable components of distributed computing have tempted various organizations to enter this business sector [2][3][4][5][6] and have likewise pulled in critical consideration from the scholarly world [7][8][1][9]. There have been a progression of estimation and correlation of the assorted cloud stages. Garfinkel et al. contemplated the execution of Amazon's Simple Storage Service (S3) and portrayed the involvement in moving an application from devoted equipment to S3 [10]. Walker et al. examined the execution of investigative applications on Amazon EC2 utilizing both large scale and miniaturized scale benchmarks [11]. A late study by Li et al. [12] further displayed CloudCmp, a precise comparator for the execution and expense of cloud suppliers in today's business sector. These studies have mostly centered around cloud empowered by big business datacenters. They have shown the force and advantage of such undertaking mists, additionally uncovered a hefty portion of their shortcomings. Specifically, Ward [13] demonstrated that the virtualization method in Amazon EC2 can leadto emotional hazards in system throughput and idleness, regardless of the possibility that the datacenter system is just gently stacked. To rebalance the workloads crosswise over physical machines, VMmigration is likewise broadly proposed [14].

A late study by Shrivastava et al. [15] presented an application-mindful movement approach which can incredibly decrease the system activity amid the VM relocation. There have been late studies on the parcel and portion of administrations to the datacenters and the neighborhood PCs, individually [16][17][18][19]. Our work varies from them through consistent provisioning the client nearby assets to the general cloud. There have been a lot of related deals with limit provisioning in PC bunches and lattices [20][21][22]. The traditional criticism control hypothesis has additionally been utilized to demonstrate the bottleneck level in web applications [23][24]. In light of these studies, Sharma et al. [17] further proposed a cost-mindful provisioning calculation for cloud clients. Using client gave assets, be that as it may, introduces new difficulties given their more grounded elements. Our work is identified with distributed, which additionally looks to use nearby client assets. Yet the vast majority of the distributed frameworks have concentrated solely on substance sharing [25], while a cloud stage is relied upon to offer assorted assets for a wide range of administrations. In addition, however certain motivation components have been created for distributed frameworks to punish free-riders, there still needs clear business/evaluating models to empower undertaking level administrations, also those requesting high accessibility as we are focusing on. Note that our framework is a business cloud stage. It is unique in relation to such instructive stages as Planet-lab [26] which scarcely be gotten to by the overall population

III. ENABLING CUSTOMER RESOURCES FOR CLOUD

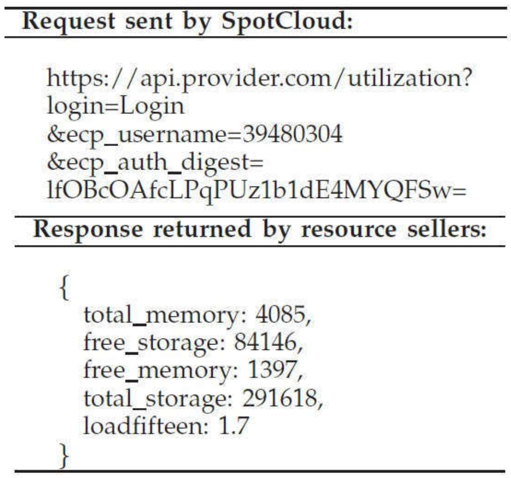
3.1 Framework Design

We now offer an overview of our framework that enables customer-provided resources in the cloud. We will then address the key design issues in this framework, and present a practical system implementation, namely Enomaly’s Spot Cloud, along with its performance measurement. In Figure 1, we outline the relation between the cloud providers and their customers. The solid lines illustrate the business model for the existing cloud, with the customers being pure resource-buyers. As or exclusively used for each individual’s local tasks, which are known to be ineffective. Aiming at mitigating this gap between centralized datacenter resources and the distributed local resources, our framework enables individual cloud customers to contribute their otherwise exclusively owned (and idled) resources to the cloud and utilize others’ resource if needed, as illustrated by the dotted lines in the figure.

3.2 SpotCloud: A Practical System

Implementation

In this part, we will present a real-world system implementation, namely Enomaly's SpotCloud, which further addresses a series of practical challenges. As shown in Figure 2, SpotCloud consists of three key modules: Cloud management, Account management, and User interface. The cloud management module supports a variety of common *hypervisors* (also known as *virtual machine managers*) including Xen, KVM and VMware as well as a highly fault tolerant and distributed *Extensible Messaging and Presence Protocol* (XMPP) with built-in failover capabilities. It also works with our resource provisioning algorithm for VM provision and migration. The account management, built on the Google App engine, allows the customers to create Google accounts for the SpotCloud marketplace. This marketplace is provided by the user interface module to let the potential sellers post Request sent by SpotCloud:



Request sent by SpotCloud:

```
https://api.provider.com/utilization?
login=Login
&ecp_username=39480304
&ecp_auth_digest=
lfOBcOAfcLPqPUz1b1dE4MYQFSw=
```

Response returned by resource sellers:

```
{
  total_memory: 4085,
  free_storage: 84146,
  free_memory: 1397,
  total_storage: 291618,
  loadfifteen: 1.7
}
```

Fig 3: An example of message format for utilization monitoring

https://api.provider.com/utilization?
login=Login
&ecp username=39480304
&ecp auth digest=lfOBcOAfcLPqPUz1b1dE4MY.

Response returned by resource sellers:
{
total memory: 4085,
free storage: 84146,
free memory: 1397,
total storage: 291618,
loadfifteen: 1.7
}
Fig. 3: An example of message format for utilization monitoring and update their configurations and prices for the contributed resources. Typical applications running in SpotCloud are as follows:

- Testing & Development Platforms
- Regional Load Balancing
- Image & Video Processing
- Scientific Research Data Processing
- Financial Modeling & Analysis
- Content Distribution

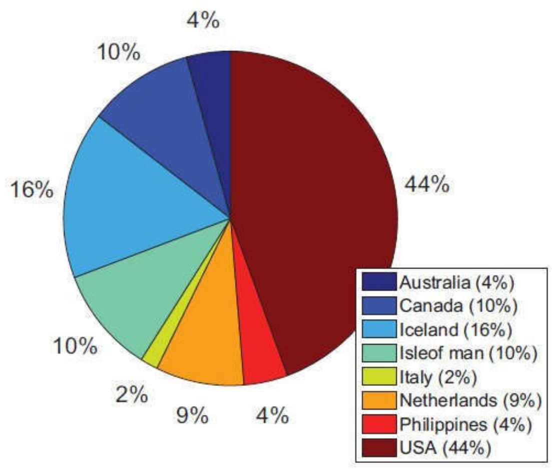


Fig:4 Locations of SpotCloud Resources

Figure 5 shows a simplified finite-state machine (FSM) in the SpotCloud system, where the *Authentication* state is managed by the account management module; the *Wait*, *Open* and *Billing* states are managed by the user interface module, and the rest of states are managed by cloud management module. The dark circles refer to the states that are used to communicate with the sellers.

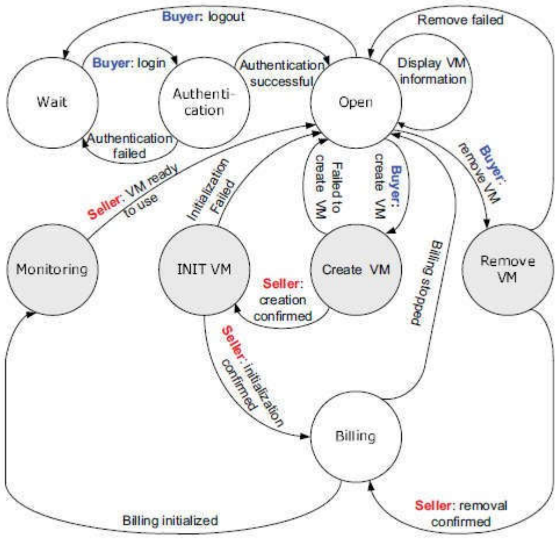


Fig:5 Finite-state machine in SpotCloud

Specifically, SpotCloud utilizes an arrangement of RESTful (sit tight for venders' data/answer to go to the following state) HTTP-based APIs for such interchanges. Figure 3 demonstrates a case of the message design when SpotCloud sends a HTTP usage screen solicitation to a merchant, where loadfifteen field incorporates the normal burden in the course of recent minutes for the vender. More subtle elements can be found in our API and Third Party Provider Integration Guide [27]. As appeared in Figure 4, SpotCloud stage has as of now pulled in the Internet clients around the world. We now look at an example set of 116 run of the mill clients for an essential comprehension of the framework execution and effectiveness. We first check the quantity of CPUs in the Spot-Cloud VMs. As appeared in Figure 6, it is anything but difficult to see that most SpotCloud assets (> 75%) have under 4 virtual centers. Each virtual center gives the comparable CPU limit of a 1.0 − 1.2 GHz 2007 Opteron processor. This is not shocking subsequent to a large portion of the client gave assets are not as capable as those from big business datacenters 5. Yet, there are likewise some moderately effective VMs; for instance, a client gave VM has 16 virtual centers with 2 calculation units in every center, which is equipped for running certain CPU serious errands. We additionally demonstrate the memory sizes on the VMs in Figure 7. We can see that most VMs (80%) in SpotCloud have a memory under 5GB, which is not additional colossal but rather is suitable to run the majority of this present reality errands.

It is significant that, the bends of SpotCloud VMs are entirely smooth, showing the presence of more adaptable alternatives to meet the heterogeneous requests from clients. Not quite the same as big business servers that are known not high accessibility, the asset accessibility in SpotCloud is for the most part contingent upon the merchants. Figure 8 demonstrates the online accessibility of the assets for one month. It is anything but difficult to see that the occasion accessibility in SpotCloud is tireless: 40% VMs have an online accessibility beneath 20%, that is, under 6 days in the 30-day estimation period. This accessibility is adequate for transient undertakings going on for a couple of hours or days. For more errands, SpotCloud needs to deliberately help its purchaser to pick legitimate VMs in view of our proposed assets provisioning calculations. It is significant that before a purchaser can truly utilize a cloud occurrence, there is a postponement because of the essential instatement forms in any cloud plat-from Amazon and Rackspace.

For instance, the AWS Management Console [28] demonstrates that it by and large needs 15 to 30 minutes to instate a Windows occasion on Amazon EC2 before a client can truly associate with it. For SpotCloud, as appeared in Figure 9, we can see that most VMs (more than 60%) can be instated inside of 10 minutes, and the greatest introduction deferral is under 27 minutes. This is significantly lower than that of Amazon EC2. The reason is that the framework/client profiles of SpotCloud VMs are as of now incorporated into purchasers' VM machines. Note that VMs' operation frameworks can likewise be customized by the purchasers in SpotCloud. Yet, in the event that purchasers would prefer not to choose the OS sort, Linux (Ubuntu 10.10) is set as a default, which to be sure has even lower introduction delay. We advance examine the VM throughput. As appeared in Figure 10, we can see that the throughput of more than half VMs are more than 10 Mbps, which is adequate to convey clients' substance to the cloud servers in ordinary cases. One imperative component of cloud administrations is that the clients pay just for what they have utilized. In a large portion of the cloud frameworks, this cost comprises of two noteworthy parts: The expense of renting VMs, and the expense of information exchange. Their valuing model is registered/chosen deliberately by the undertaking administration suppliers. As we have examined in the past area, the cost of SpotCloud VMs, in any case, is redone by individual merchants who give/offer their cloud limits. As appeared in Figure 11, we can see that the SpotCloud VMs are for the most part exceptionally modest. Besides, this bend is additionally entirely smooth showing that the purchasers have high adaptability in selecting VMs in this customerprovided cloud stage

IV. PRICING WITH HETEROGENEOUS RESOURCE PROVIDERS

Our follow examination demonstrates the proficiency of our framework model configuration. Specifically, SpotCloud has pulled in numerous clients to contribute their nearby assets in our commercial center. It is anything but difficult to see that one basic test is to sufficiently offer impetus for a client to contribute her/his assets or use others'. The issue is further entangled given that the clients are profoundly heterogeneous, making the coarse-grained evaluating model utilized by the current cloud suppliers barely working. In our framework outline, we address this issue through a conveyed asset showcase that permits the clients to choose the quality, amount, and valuing of their nearby assets to be contributed.

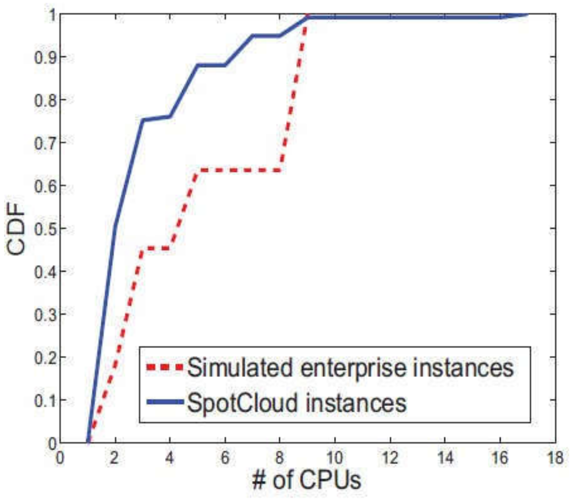


Fig.6: # of CPUs

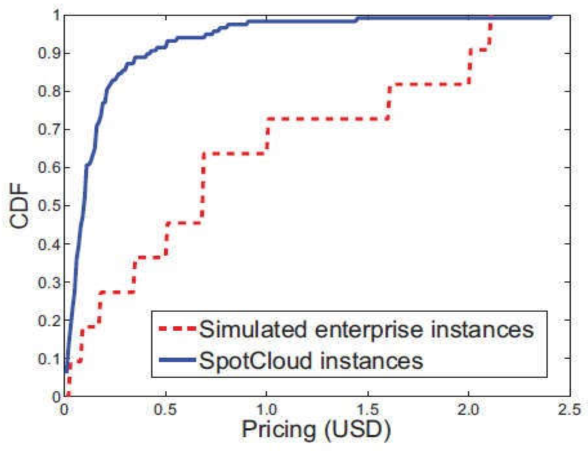


Fig.7: Memory Size

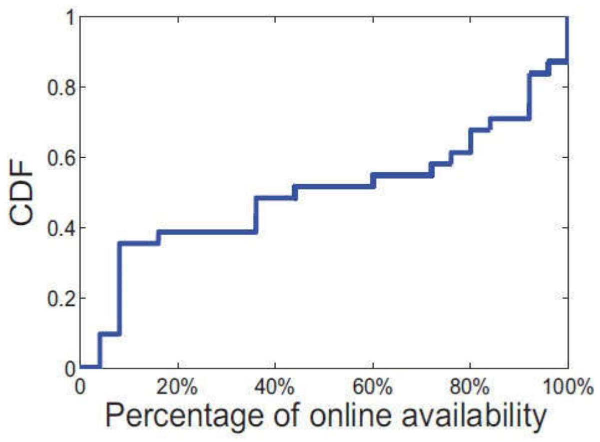


Fig.8: Online Availability

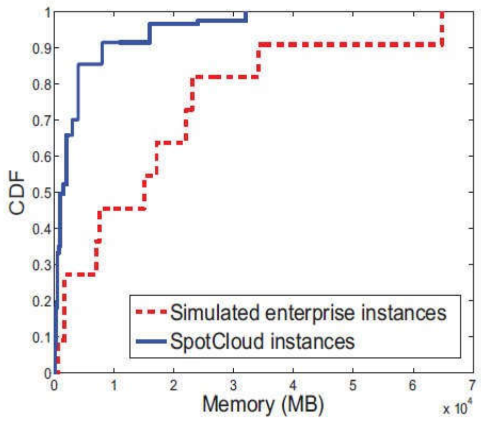


Fig.9: Server-client throughput

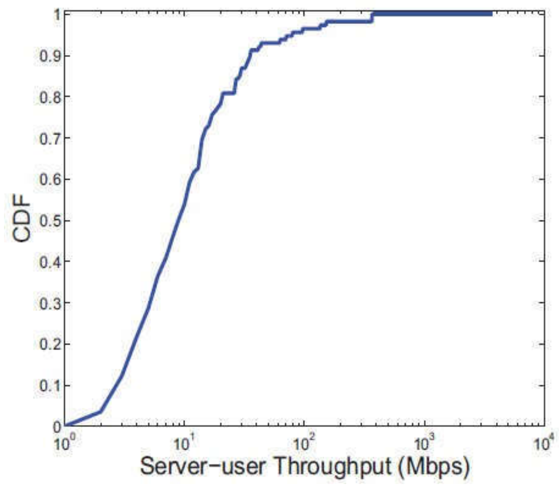


Fig.10: Pricing distribution

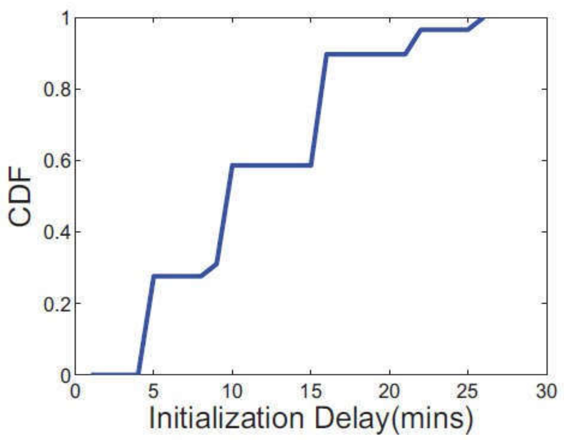


Fig.11: Pricing distribution

4.1 Modeling of Resource Pricing

In the greater part of the current undertaking cloud stages, settled valuing remains the most mainstream methodology. Amazon EC2, as a normal sample, publicizes \$0.02−2.62 every hour for each of its On Demand Virtual Machine occurrences, contingent upon their sorts. As of late, element valuing additionally been presented, e.g., the "spot estimating" in EC2 [30] that goes for better using the empty limits in the datacenters. It is realized that the spot cost will be powerfully acclimated to coordinating the supply and request, however the full detailhave not been uncovered. Subsequent to the potential asset suppliers in SpotCloud are heterogeneous and are not compelled to contribute their assets, a working plan of action is important to offer them enough motivator. Along these lines, rather than setting an institutionalized valuing principle for unit asset, we recommend a circulated market that permits the potential suppliers (i.e., merchants) to choose the quality, amount, and evaluating of their nearby assets to be contributed, in which: (1) A dealer will publicize the arrangement (sum and accessibility) of its neighborhood assets and also the asking cost; such data will be seen by different venders and purchasers; (2) Both asset venders and purchasers are normal: given the promoted costs, a purchaser will attempt to minimize the expense for asset provisioning, and a vender will attempt to augment the benefit; (3) After seeing others' promoted data, a dealer will modify her/his own setup and cost to amplify her/his potential benefit. The instinct behind this configuration is that the venders have better learning of their own assets as far as both running expenses and expected qualities. In the event that they can't locate a decent approach to pick up benefits, any altered or element valuing guideline will neglect to give them the motivating force to join cloud markets. This plan of action can be figured as a variety of a Repeated Seller Competition amusement [31] as takes after. To encourage our discourse, we list the key documentations that will be utilized as a part of this paper in

Table. 1. We begin with a general model with N asset dealers. Every merchant can be considered as a cloud administration supplier working s/he claim neighborhood assets. We indicate the vector of dealers' asset by c

= (c1, ..., cN). Putting resources into asset is exorbitant. Specifically, the expense of sharing asset ci for dealer i is γici where γi > 0 for i ∈ {1, ...,N}. We signify the cost charged by dealer i (per unit request) by ei and mean the vector of costs by e = (e1, ..., eN). Then again, the purchaser interest is given by a set W =

{w1, w2, ...}, whichmay result from a continuum of buyers who request precisely one unit up to their reservation prices(the unit value that normal by the purchasers). Accordingly, |W| demonstrates the total interest in the framework. For the merchants in the framework, their conceivable methodologies are: (1)Provides ci cloud assets with unit cost ei; (2) Inactive and not willing to contribute any nearby assets (n.a). This procedure space can be composed as: Si := {ei, ci} {n.a} (1) where {ei, ci} implies that merchant i is dynamic and furnishes ci cloud assets with unit cost ei, and {n.a} implies that dealer i is not willing to contribute any nearby assets. The diversion is along these lines played as takes after: every dealer i reports a limit ci and a unit cost ei which it is willing to offer. For this situation, its benefit is given by:

$$\pi(Si) = ei \cdot ci - \gamma i \cdot ci - F \text{ for } ei \in [0, v] \; 0 \text{ else when } Si = n.a \quad (2)$$

where F is the altered expense (e.g. the arrangement overhead) when merchants design their neighborhood assets for cloud service7. v is the interfere with cost showing the maximin cost of the merchants; no interest will face to dealer i if ei > v. The target of every vender is to amplify the benefit π. Note that, all purchasers will submit their requests with the least expensive dealers. These requests are satisfied until the least expensive dealers' abilities are all depleted. If not all requests can be satisfied by the least expensive merchants, we accept that the purchasers will be apportioned in light of the Beckmann-proportioning principle [32]. Specifically, an arbitrary specimen of customers will be served by the least expensive venders. The remaining (unserved) purchasers will put in their requests with the following least expensive venders. This technique is rehashed until either every one of the clients are served or all dealers are depleted. The interest confronted by a vender i is given by:

$$Hi = \max\_ |W| \cdot (3)$$

where |W|[w=ej] is the quantity of requests with reservation costs equivalent to ej and I[ej=ei] is the number venders with the costs equivalent to ei. In this way we have qi = min{Hi, ci}. Since it is a rehashed amusement, when we utilize Z(t) to indicate the methodology vector of period t , the normal result of vender i over all unendingly numerous periods is given by: ∞_ t=1 δπi[Z(t)] (4) where 0 < δ < 1 is a rebate component, meaning the dealers' understanding (near 0: not persistent; near 1: exceedingly tolerant). Practically speaking it means the likelihood of whether the merchants will have the opportunity to play the amusement later on [33]. By Theorems [34], the ideal technique for playing this rehashed amusement is not to more than once play a Nash methodology of the stage recreations, however to participate and play a socially ideal system.

We will accordingly examine whether such a stationary result balance exist in this rehashed diversion. Accept that ¯ S is a stationary result harmony set crosswise over N merchants in the rehashed amusement with unit value e (where the dealers charge the same cost per unit asset in every period). Since all merchants are dynamic and the amusement is symmetric, we overlook the file i where no disarray results. 1 1−δ . Along these lines the normal benefit (over vastly numerous periods) at this stationary balance is 1 1−δ π(¯ S). It is anything but difficult to see that the stationary result harmony will exist when 1 1−δ π(¯ S) > 0 Otherwise if 1 1−δ π(¯ S) ≤ 0, the venders will like to be inert (n.a) with zero benefit. At the end of the day, if there is a methodology set a cross N dealers such that π(¯ S) is sure, the stationary result harmony will dependably exist for δ adequately near 1. Subsequent to the lower part if Eq.6 is the upper bound of π(¯ S), it is not hard to see that π(¯ S) is diminishing in N; it will advance gets to be negative for N adequately expansive because of the presence of F.

Hence, we can give the upper bound of N in the framework (the presence condition for the stationary balance) as takes after: N∗ = max{N|π(¯ S) > 0} (6) Therefore, for every one of the 2 ≤ N ≤ N∗ there is dependably a stationary result harmony ¯ S for the framework where the venders charge the same cost per unit asset in every period. An instinctive clarification for N∗ is the greatest number of merchants in the cloud market. Additionally, if ¯ S is a stationary result balance, it is anything but difficult to see that for any vender i Si _= n.a (since the benefit is bigger than zero, the merchants will have no motivating force to be inert). Note that the unit value e for this stationary result balance is not one of a kind, and it is relying upon the purchasers' interest at value e: L(e) = |W|[w≤e]. The bound of this cost will be further talked about in our future works.

4.2 Discussions

We can discover that on the off chance that we send a cloud business sector where the merchants can progressively conform their offering amount and costs, the dealers will be capable find suitable systems to offer their assets for a higher advantage (rather than leaving the business sector which results to zero benefit). In this plan of action, the merchants' motivation and their aggregate populace are both relying upon the purchaser request. Expansive requests can give sharing motivating force to more dealers (bigger N∗) and littler requests will lessen the quantity of N∗. Yet, unless |W| = 0, this plan of action can simply offer motivating forces to a given number of dealers to join the cloud market. In addition, giving the boundless round of this amusement, the cost per unit asset has the pattern toconverge to a stationary balance [35].

Distinctive with the altered valuing models, this stationary cost is progressively chosen and balanced by the venders in light of purchasers' interest. It is likewise significant that the valuing imposing business model and agreeable duping can barely happen in such a framework. The fundamental reason is that there are likewise some other option cloud suppliers over the Internet, for example, EC2 and Google. On the off chance that the dealers' costs surpass the unit costs in different stages, the venders' benefits will likewise be diminished to zero. This component was additionally caught by the interfere with value v in Eq.4. It merits underscoring that we see customerprovided assets a supplement to the datacenter assets, not a substitution. Given that the configuration and streamlining of datacenter-based cloud have been broadly contemplated, in this paper, we will concentrate on the viable usage of clients' nearby assets and their consistent combination into the general cloud stage. While investigating appropriated nearby assets have been nearly analyzed in such different settings as lattice processing [36] and distributed [37], the stateof-the-workmanship cloud environment represents a progression of new challenges for our proposed arrangement. Specifically, to empower endeavor level administrations, we need to guarantee high administration accessibility when coordinating clients' assets. Not the same as datacenters, there is no surety that a specific client's nearby assets will be constantly online for distributed computing 8. Luckily, through follow examination and a versatile calculation plan, we exhibit that the steady administ

V. PROVISIONING ACROSS DYNAMIC CUSTOMER PROVIDED RESOURCES

We begin from analyzing the issue of asset provisioning crosswise over element client gave assets.

5.1 The Resource Provisioning Problem

We consider a nonexclusive model of N asset suppliers (rather than one monster supplier as in the routine cloud, i.e., the datacenter). Without loss of all inclusive statement, we expect every supplier just offers one Virtual Machine (VM) for the business sector, meant by S = {s1, s2, ・ , sN}. The asset limit of VM si incorporates calculation power psi, memory size msi , plate size dsi , and system data transfer capacity bsi. Given that such a VM is accessible on the cloud stage just when the supplier does not plan to utilize it locally, we utilize Asi (t) to mean the accessibility of VM si at time t; that is, Asi (t) = 1 if VM si is accessible, and generally Asi (t) = 0. Practically speaking, such data can be acquired by requesting that the supplier show when it offers the VM to the cloud stage. For a client that hopes to rent pl.

```
Algorithm OptimalProvisioningSchedule()
1:  Sort $S$ with ascendant order based on $rules$ 1-2;
2:  for $t \in [t_{start}, t_{end}]$, $Req[t] \leftarrow (P, M, D, B)$; end for
3:  Set $Stack$ empty; $k \leftarrow 0$; $time \leftarrow t_{start}$;
4:  $Cost \leftarrow 0$; $Cost_{min} \leftarrow \infty$; Set $Stack^*$ empty;
5:  while true,
6:     if $time > t_{end}$ or $Cost \geq Cost_{min}$,
7:        if $Cost < Cost_{min}$,
8:           $Cost_{min} \leftarrow Cost$; $Stack^* \leftarrow Stack$;
9:        end if
10:       goto 26;
11:    end if
12:    $k \leftarrow k + 1$;
13:    if $k > |S|$,
14:       goto 26;
15:    else if $A_{s_k}(time) = 0$,
16:       continue;
17:    end if
18:    Push $\{k, time, S\}$ in $Stack$;
19:    $Req[time] \leftarrow Req[time] - (p_{s_k}, m_{s_k}, d_{s_k}, b_{s_k})$;
20:    Update $Cost$ according to $f(W)$;
21:    if $Req[time] \leq (0, 0, 0, 0)$,
22:       $time \leftarrow time + 1$; $k \leftarrow 0$;
23:       Sort $S$ with ascendant order based on $rules$ 1-3;
24:    end if
25:    continue;
26:    if $Stack$ is empty, break;
27:    else
28:       Pop $Stack$; Update $Req$ and $Cost$ accordingly;
29:    end if
30: end while;
31: Generate optimal schedule $W$ by $Stack^*$;
32: return $W$;
```

Fig. 12: Algorithm to compute the optimal provisioning

resources from the cloud platform, her/his demands include the aggregate computation power, the aggregate memory size, the aggregate disk size, and the aggregate bandwidth, denoted by P, M, D and B, respectively. Such demands are also accompanied by a request period [tstart, tend] indicated by the customer. Define a provisioning schedule as W = {(x1, t1, l1), (x2, t2, l2), · · · , (xk, tk, lk)} (tstart ≤ t1 ≤ t2 ≤ · · · ≤ tk ≤ tend), where each tuple (xi, ti, li) (xi ∈ S and li > 0 for i = 1, 2, · · · , k) means that, starting at time ti, VM xi is provisioned for period li. The problem is thus to find a proper provisioning schedule given the demands, subjecting to the following constraints: (1) VM Availability Constraint: ∀(xi, ti, li) ∈ W, ∀t ∈ [ti, ti + li],Axi(t) = 1;
(2) VM Utilization Constraint: ∀(xi, ti, li) ∈ W, if ∃(xj, tj, lj) ∈ W and xi = xj , 8. The enterprise cloud service, such as Amazon EC2, is enabled by highly available dedicated datacneters. This is why the service availability was seldom considered in the existing cloud models. 1045-9219 (c) 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission

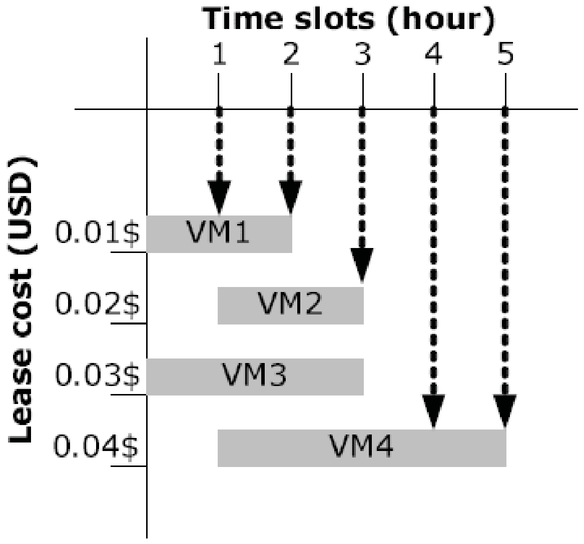
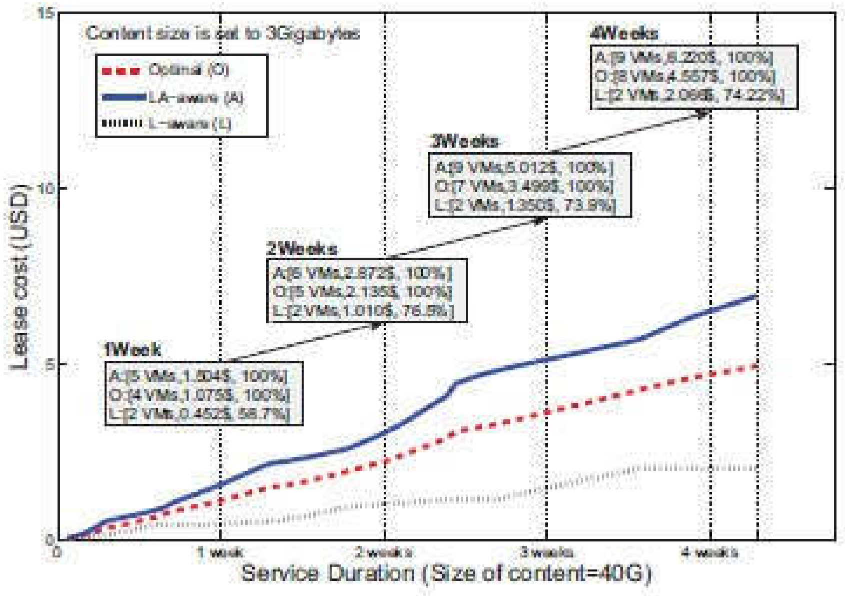


Fig.13: An example of resource provisioning

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. then [ti, ti + li] ∩ [tj, tj + lj] = ∅; (3) Resource Requirement Constraint: ∀t ∈ [tstart, tend], _k i=1 pxi · I_ t∈[ti,ti+li] ≥ P, _k i=1 mxi · I_ t∈[ti,ti+li] ≥ M, _k i=1 dxi · I_ t∈[ti,ti+li] ≥ D, _k i=1, bxi · I_ t∈[ti,ti+li] ≥ B; where I[·] is the indicator function. The above constraints ensure the resource availability during the lease period, a VM can be leased only in one schedule at any time, and the resource demands are fulfilled at any time instance within the lease period, respectively. Let csi be the lease cost of VM si per unit time. Our objective is thus to minimize a cost function f(W), which involves two parts in our scenario:
Lease Cost:

Which is the cost to migrate the service/data to a new VM should the old VM become unavailable. Given the dynamic resource availability, migration across different providers in the lease period is necessary in our system to ensure the demands are fulfilled within the whole lease period. Accordingly, it is calculated as the number of VMs that becomes unavailable before time tend. It is worth noting that, for a fully cooperative non-profit system, the costs here can simply be the provider's net costs for offering the services. Yet if the providers are profit-motivated, which is natural for a commercial system, the costs depend on the provider's expected resource prices, which we will examine in the next section.



(a)

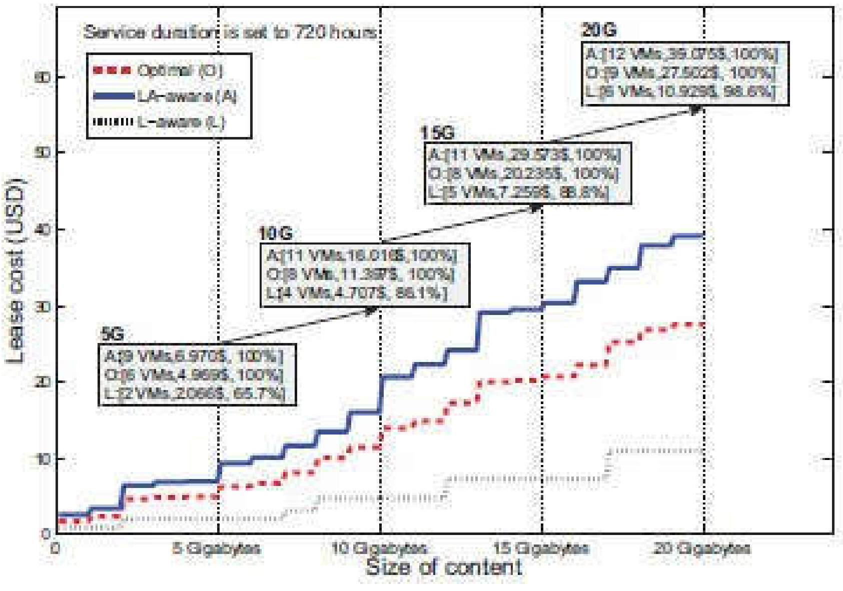
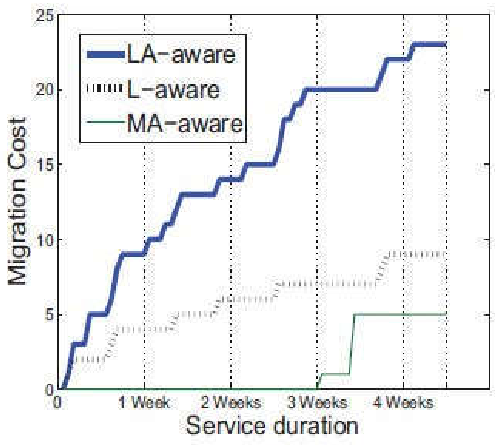


Fig.14: Lease cost analysis with different: (a) service durations and (b) content sizes.



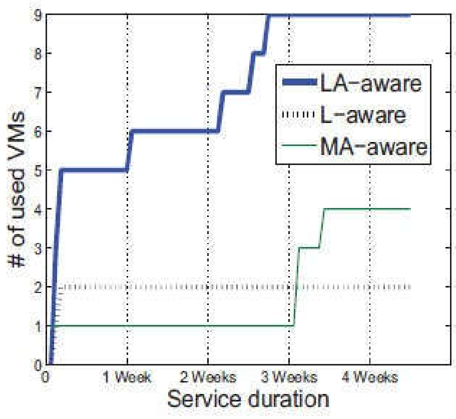
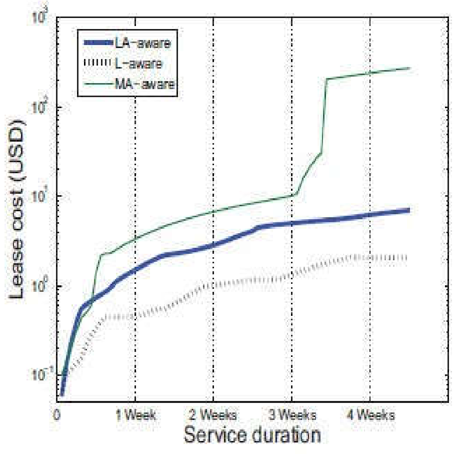


Fig.15: Migration cost with different service durations. Fig.16: # of used VMs with different service durations

5.1 Optimal Provisioning with Dynamic Resources

The lease cost is a major concern in any cloud platform. Yet given the dynamically available resources across providers, the migration cost 9 is not negligible 9. In this paper the migration cost refers to the frequency of migration during the cloud deployment



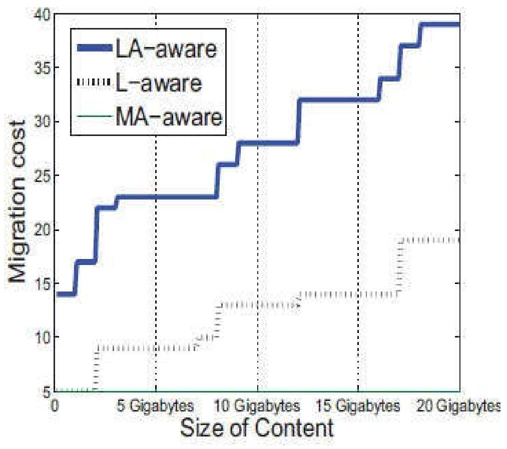
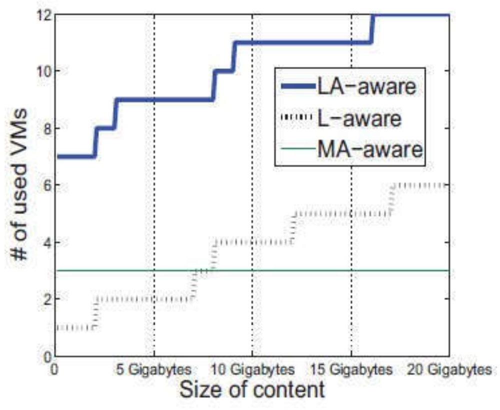


Fig.17: Lease cost with different service durations.

Fig.18: Migration cost with different content sizes



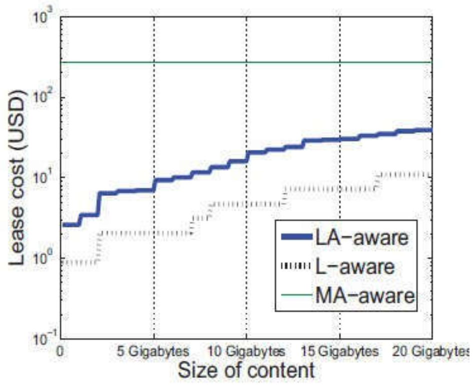


Fig.19: # of used VMs with differentcontent sizes ***Fig.20: Lease cost with different content sizes***

```
Algorithm OptimalProvisioningSchedule()
1: Sort S with ascendant order based on rules 1-2;
2: for t ∈ [tstart, tend], Req[t] ← (P,M, D,B); end for
3: Set Stack empty; k ← 0; time ← tstart;
4: Cost ← 0; Costmin ←∞; Set Stack∗ empty;
5: while true,
6: if time > tend or Cost ≥ Costmin,
7: if Cost < Costmin,
8: Costmin ← Cost; Stack∗ ← Stack;
9: end if
10: goto 26;
11: end if
12: k ← k +1;
13: if k > |S|,
14: goto 26;
15: else if As
k (time) = 0,
16: continue;
17: end if
18: Push {k, time, S} in Stack;
19: Req[time] ← Req[time] − (ps
k,ms
k, ds
k, bs
k );
20: Update Cost according to f(W);
21: if Req[time] ≤ (0, 0, 0, 0),
22: time ← time +1; k ← 0;
23: Sort S with ascendant order based on rules 1-3;
24: end if
25: continue;
26: if Stack is empty, break;
27: else
28: Pop Stack; Update Req and Cost accordingly;
29: end if
30: end while;
31: Generate optimal schedule W by Stack∗;
32: return W;
```

Fig. 12: Algorithm to compute the optimal provisioning schedule. in our system, either. The availability requirement also introduces another dimension. As such, the solution space for the provisioning problem becomes much larger. We now present a branch-and-cut algorithm for optimizing a general cost function f(W). We then present a series of heuristics to consider such overhead as migration costs and further minimizing the search space for online dynamic provisioning. Our algorithm is summarized in Fig. 12. We first sort the VMs in S in ascending order of the lease cost per unit resource10 (referred to as *rule 1*).

For the VMs with the same lease cost per unit asset, we advance request them in plunging request on their first distracted times after tstart (alluded to as principle 2). This permits the VMs that are for the most part accessible and with less expensive assets being investigated first and close ideal arrangements can then be rapidly found. With such arrangements, we can promote cut other hunt branches with equivalent or higher costs (line 6-11) and incredibly lessen the quest space for the ideal arrangement. We additionally check whether current VM sk is accessible at time (line 13-17). If not, we will avoid the present one and go ahead to the following. Each time a VM sk is chosen (line 18-20), it will be viewed as rented at time and Req[time] will be lessened by the VM's asset ca-pacity (psk,msk, dsk, bsk), in like manner. After that, the calculation checks if whatever other VM should be rented (line 21-24). If not, the reality of the situation will become obvious eventually expanded by one unit and k will be reset to

0. What's more, like line 1, we sort the VMs in S in view of principles 1-2, yet then move the VMs utilized at past time unit ahead to the starting (alluded to as standard 3). At the point when the hunt completes, the ideal provisioning plan W will be produced and returned (line 31-32). This is very like the exemplary Interval Scheduling Problem [38]. The distinction is that our client interest is fragmentary and can be easily relocated between various VMs. Because of the sorting operation, this calculation requires significant investment O(nlogn) on inputs with n virtual machines. A sample of lease cost minimization is appeared in Figure

13. For this situation, the clients need to settle on choices crosswise over four VMs to bolster a five-hour assignment. We can see that for every time opening we look the accessible VM with least rent cost. In Figure 13, we will utilize VM1 for the two hours, VM2 for one hour and VM3 for two hours. The optimality of this calculation can be demonstrated as takes after:

Confirmation: We demonstrate the optimality of this calculation by inconsistency. For a given time space t, we accept that the undertaking (right now opening) is alloted to VMx in timetable W. Assume VMy is a superior task of this assignment with littler lease cost right now space. Taking into account guideline 1 (all VMs in S are sorted in climbing request of the lease cost per unit asset), we can understand that our calculation officially checked VMy before doling out the errand to VMx. This is on the grounds that the lease expense of VMy is littler than the lease expense of VMx. In any case, the errand is not doled out to VMy at time t. In view of Step5 to Step9 of Figure 13, this implies VMy is not accessible a this time space (t). This prompts an inconsistency on the grounds that VMy is a legitimate task at time t. _ Depending on the application, lease expense and relocation expense may have astoundingly diverse commitments. We can then further accelerate the online calculation by expecting one of them is ruling. Specifically, given an expense of interest, we can first sort the VMs in S by the principle (guideline 1 or tenet 2) relating o this expense. For the VMs tied with the standard for the expense of interest, we assist sort them by the principle for the other expense. At that point we pick the principal k accessible VMs that can satisfy the requests at tstart. At the point when a picked VM gets to be occupied, we pick various accessible yet not yet utilized VMs from the earliest starting point of S to top off the interest hole until tend. The viability of the speedup calculation and in addition the tradeoffs in the middle of lease and movement expenses will be assessed in Section VII.

VI. PERFORMANCE EVALUATION: LEASE- VSMIGRATION-COST

Given that the lease cost and migration cost play key roles in the resource provisioning and in pricing, we now take a closer look at their impact and tradeoffs with the SpotCloud trace data. We will also examine the effectiveness of heuristics respectively focusing on lease and migration costs, as discussed in Section 4. We call the algorithm that treats lease cost with more interest as *LA-aware*, and that treats migration cost with more interest as *MA-aware*. Note that both of them strike to ensure service availability out of dynamic customer-provided resources. For comparison, we also implement a state-of-the-art cost-aware-only algorithm (*L-aware*) [17], which however is difficult in ensuring service availability in SpotCloud as we will show. In the following discussions, *LA-aware* and *MA-aware* refer to our designs, and *L-aware* refers to the existing resource provisioning approach [17]. TABLE 2: Smapled

SpotCloud Instances.

Price	**vCPU**	**Memory**	**Storage**	**Country**
$0.002	1	256MB	10GB	Isle of Man
$0.005	1	256MB	15GB	United States
$0.010	1	512MB	10GB	United States
$0.010	2	8GB	10GB	Isle of Man
$0.019	1	256MB	10GBB	Netherland
$0.020	1	4GB	20GB	United States
$0.041	1	256MB	15G	Iceland
$0.06	1	512MB	30GB	Poland
$0.238	4	8GB	50GB	Iceland

We apply the genuine information follows (as talked about in Section 3) from SpotCloud to run the calculations. Table. 2 demonstrates a tested arrangement of the Spotcloud VMs. In this assessment, the purchasers will rent diverse VMs at various time to get administration accessibility. This is unique in relation to the VM relocation [15] in light of the fact that the relationship in the middle of VMs and physical machines are not changed. The chose Spotcloud VMs in this manner comprise of both KVM and XEN VMs. To better comprehend the execution of our heuristics, the ideal results are additionally identified as baselines for examination. From the follows, we find that the online examples of client gave assets can be all around fitted by a self-comparable procedure with Hurst parameters [39] around 0.7 (more subtle elements can be found in our specialized report [40]). We will be concentrating on an online stockpiling application, which, when contrasted with CPUintensive applications, makes more difficulties when assets are dispersed at assorted areas. Yet, with specific changes (generally rearrangements), our analyses and conclusions can be stretched out to CPUintensive applications.

This capacity administration empowers a purchaser to rent an arrangement of SpotCloud assets to store her/his information substance; both the substance size and the administration term will be powerfully balanced in our investigation. Note that we don't furnish the cost correlation with big business cloud administrations, for example, Amazon S3. This is first in light of the fact that SpotCloud framework is intended to supplement the venture cloud benefits yet not to supplant them. Second, the estimating model of Amazon S3 is likewise to a great extent unique in relation to SpotCloud; for instance, other than the lease cost, S3 will likewise charge the clients in view of the quantity of solicitation and the measure of their information exchange. All things considered, an immediate correlation can be very troublesome. Yet it is workable for a client to exploit both frameworks by part the capacity, which can be an intriguing heading deserving of further examination.

L-mindful versus LA-mindful: We consider C-mindful and LA-mindful as the strategies for asset provisioning. Figure 14(a) demonstrates the outcomes when purchasers use Spot-Cloud assets to have a capacity for a 3 Gigabytes content with various administration lengths of time. The cases in the figure demonstrate the quantity of utilized VMs, the lease cost, and in addition the rate of administration accessibility. It is anything but difficult to see that on the off chance that we apply the current Laware calculations for client gave assets, the purchasers will experience the ill effects of low administration accessibility, which will be around half to 70% just, contingent upon the administration length of time. This administration accessibility is clearly not able to bolster most Internet stockpiling administrations. Luckily, the proposed LA-mindful calculation gives extremely stable administration accessibility, notwithstanding when the purchasers need to convey this administration for quite a while, around 720 hours. As an exchange off, the purchasers need to pay more to empower their administration on more VMs.

The lease expense of the LA-mindful calculation is additionally very close with the ideal results. Yet the intricacy of LA-mindful heuristic is much lower than the ideal calculation, as examined prior. Figure 14(b) gives a further examination when clients need to send bigger substance for one month (720 hours). This outcome demonstrates that the lease expense is additionally entirely touchy with the expanding of substance size. This is on the grounds that more VMs will be utilized to hold bigger substance for such a long administration length of time. It is important that the administration accessibility appears to be actually expanding when more VMs are utilized as a part of L-mindful calculation. Be that as it may, this accessibility is relying upon the haphazardly joined accessibility over all chose VMs, and there is no certification for the purchasers.

LA-mindful versus MA-mindful: We now present the relocation cost and additionally the quantity of utilized VMs as a part of our correlation. Figure 15 demonstrates the movement cost when purchasers send their administration from 1 week to over 4 weeks. We can see that one downside of the LA-mindful calculation is the expanding of movement expense. This is not astounding in light of the fact that the LA-mindful calculation is intended to plan the

substance crosswise over various VMs for lower lease cost. The MA-mindful calculation, then again, can better decrease the relocation cost for the purchasers (with the movement cost under 5 in the figure). As appeared in Figure 16, we can see that the MA-mindful calculation just utilized one VM to serve purchasers' substance for the initial 3 weeks while the LA-mindful calculation utilized more than 5 VMs for lower lease cost. As a tradeoff, minimizing the movement expense will normally build the lease cost. As appeared in Figure 17, we can see that the lease expense of MA-mindful calculation is the most astounding among all calculations. Specifically, its lease expense could be 10 times higher than that of the LA-mindful calculation. The L-mindful calculation, then again, gives low rent cost and sensible movement cost. Be that as it may, it again experiences the low administration accessibility as appeared in Figure 21. Much more dreadful, its administration accessibility diminishes quick when the purchasers need to convey their administration for more lengths of time. Figure 18 looks at the relocation cost when the purchasers need to convey bigger substance for an altered time term (720 hours). We can see that the LA-mindful calculation will again give low rent cost however higher relocation cost. The MA-mindful calculation, then again, gives a consistent relocation expense of 0 utilizing 3 VMs (Figure 19). When we assist check these 3 VMs, we find that they are all extremely stable VMs with high stockpiling limits. The selecting of these VMs can in this way give low relocation cost. In any case, as appeared in Figure 20, the lease expense of MA-mindful calculation is additionally very high (around $200). This is very nearly 20 times higher than that of the LA-mindful calculation. It is important that for a settled administration length of time, selecting more VMs will possibly build the administration accessibility. As appeared in Figure 22,the administration accessibility of the L-mindful calculation is expanding with bigger substance. Yet, as we have as of now talked about some time recently, such an administration accessibility can't be ensured. It is important that our calculations are adaptable for purchasers to set up a redid administration accessibility, i.e., lower than 100%, for their applications. Figure 23 demonstrates the situation when a purchaser needs to send a 5 Gigabytes content for 720 hours with various administration availabilities. Note this figure alludes just to the LA-mindful calculation. We can see that the purchasers' administration accessibility is directly related with the lease cost. This figure elucidates the exchange off between administration accessibility and lease cost. For instance, the purchasers will spend around $0.6 to expand their administration accessibility by 10 percent.

VII. CONCLUSIONS AND FURTHER DISCUSSIONS

This paper researched the achievability and the framework outline of empowering client gave assets to distributed computing. We intently analyzed the asset provisioning and estimating issues with powerfully accessible assets, and created proficient arrangements. We then introduced the beginning outline of SpotCloud, a working framework going for coordinating the cloud assets from both venture and individual dealers. Follow examination approved SpotCloud as a supplement of awesome possibilities to datacenter-based cloud. Our work speaks to a beginning endeavor toward this heading, and there are numerous conceivable future roads. Other than advancing SpotCloud and investigating the best assignment of administrations in the middle of SpotCloud and datacenter-based cloud, we are especially inspired by the accompanying four basic issues: Privacy and Security Issues: It is significant that the usage of client gave assets can acquaint protection and security issues with the cloud frameworks. Like the associate helped content stockpiling frameworks, we trust that such an issue likewise should be precisely moderated in our framework. Also the current encryption capacities in Spotcloud, one of our progressing works is to plan a security mindful burden task for such a client gave cloud stage.

Arbitrary Failure of clients: When a client gives assets to SpotCloud, s/he needs to assert the periods that the neighborhood assets are accessible. The clients will likewise be inspired to well act given the benefit from giving assets. This is not the same as shared systems where the companions can leave the framework openly, and hence the online/disconnected from the net practices are significantly more unsurprising in our framework.

In the uncommon instance of wild arbitrary disappointments, some shrewd reinforcement calculations can be investigated for issue recuperation. Specifically, we expect to evaluate the likeness of VMs' online accessibility, advance the periodical status reporting and sort out them into a paired tree structure to reinforcement one another. Applying Amazon Elastic Block Store (EBS) administration with essential modifications is additionally a conceivable choice.

Movement cost with better granularity: It is significant that the relocation cost characterized in this concentrate just considers the recurrence of movements and serves as a guess of the genuine movement cost. Practically speaking, genuine relocation expense of various applications can be distinctive notwithstanding when their movement frequencies are indistinguishable; for instance, the genuine movement expense can rely on upon the application convention, the substance size or the transmission capacity between VMs. Consequently, this relocation expense could be better characterized given the point by point attributes of various applications. Such data may likewise encourage keen allotments in the middle of SpotCloud and datacenters.

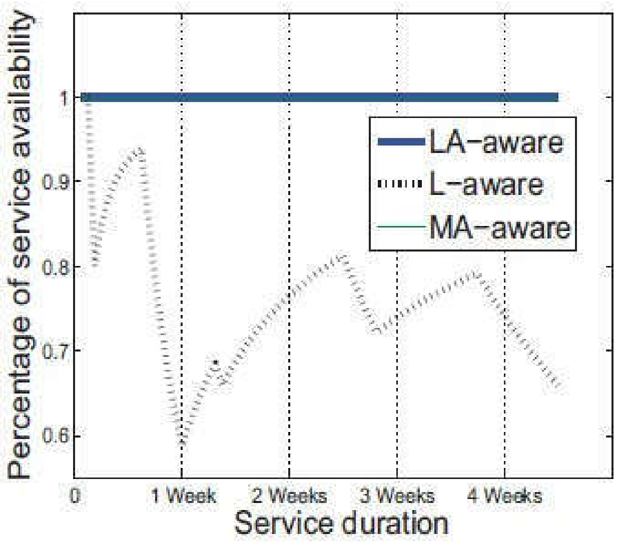


Fig.21: Service availability with different service durations

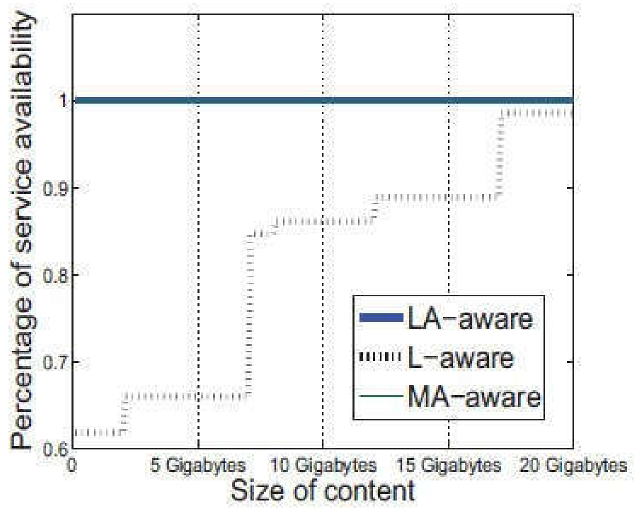


Fig. 22: Service availability with different content sizes

Modularized System Enhancement: It is known that high cost and high availability do not necessarily be better than low cost and low availability in real-world system deployments. Instead of considering everything for the users, it is also important to modularize different performance metrics and let the users decide their own priorities/tradeoffs. In our SpotCloud system, such an enhancement will be able to fulfill different user demands with lower costs.

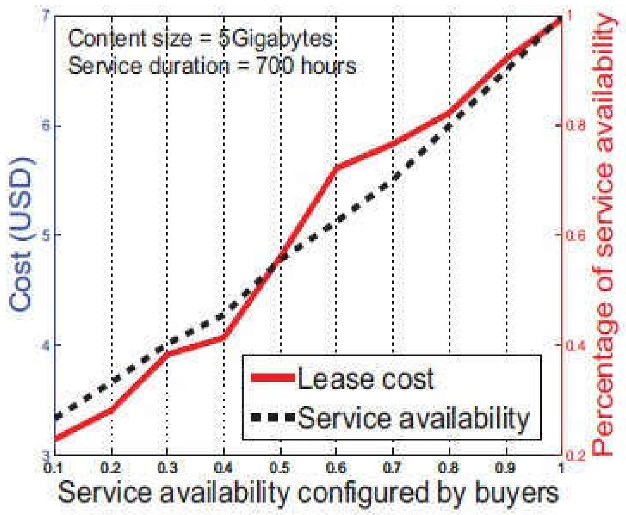


Fig.23: Trade-off between cost and service availability

REFERENCES

[1] M. Hajjat, X. Sun, Y. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai,
and M. Tawarmalani, “Cloudward Bound: Planning for Beneficial Migration of Enterprise Applications to the Cloud,” in *Proc. ACM SIGCOMM, 2010.*
[2] Amazon Web Service. [Online]. Available: http://aws.amazon.com/
[3] GoGrid Cloud Hostin. [Online]. Available: http://gogrid.com/
[4] Google AppEngine. [Online]. Available: http://code.google.com/appengine/
[5] Microsoft Windows Azure. [Online]. Available: http://www.microsoft.com/
[6] Rackspace Cloud. [Online]. Available:http://www.rackspacecloud.com/
[7] M. Armbrust, R. G. A. Fox, A. D. Joseph, R. H. Katz, A. Konwinski,

F. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the Clouds: A Berkeley View of Cloud Computing,” University of California, Berkeley, Tech. Rep., 2009.
[8] K. Sripanidkulchai, S. Sahu, Y. Ruan, A. Shaikh, and C. Dorai, “Are Clouds Ready for Large Distributed Applications?” in *Proc. SOSP LADIS Workshop, 2009.*
[9] SETI@HOME. [Online]. Available:http://setiathome.berkeley.edu/
[10] S. Garfinkel, “An Evaluation of Amazon s Grid Computing Services : EC2 , S3 and SQS,” *Harvard University Tech, Rep., 2008.*
[11] E. Walker, “Benchmarking amazon EC2 for high-performance scientific computing,” *Proc. USENIX Login, 2008.*
[12] A. Li and X. Yang, “CloudCmp: Comparing Public Cloud Providers,” *Proc. ACM/USENIX IMC, 2010.*
[13] J. S. Ward, “A Performance Comparison of Clouds: Amazon EC2 and Ubuntu Enterprise Cloud,” *Proc. SICSA DemoFEST, 2009.*
[14] T. Wood, P. Shenoy, and Arun, “Black-box and Gray-box Strategies for Virtual Machine Migration,” in *Proc. USENIX NSDI, 2007.*
[15] V. Shrivastava, P. Zerfos, K. Lee, H. Jamjoom, Y. Liu, and S. Banerjee, “Application-aware Virtual Machine Migration in Data Centers,” in *Proc. IEEE INFOCOM, 2011.*
[16] Y. Seung, T. Lam, L. E. Li, and T. Woo, “Seamless Scaling of Enterprise Applications into The Cloud,” in *Proc. IEEE INFOCOM, 2011.*
[17] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh, “Kingfisher: Cost-aware Elasticity in the Cloud,” in *Proc. IEEE ICDCS, 2011.*
[18] Y. Wu, C. Wu, B. Li, X. Qiu, and F. Lau, “CloudMedia: When Cloud On Demand Meets Video On Demand,” in *Proc. IEEE ICDCS, 2011.*
[19] S. Kannan, A. Gavrilovska, and K. Schwan, “Cloud4Home – Enhancing Data Services with Home Clouds,” in *Proc. IEEE* [20] A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, and P. Gauthier, “Cluster-based Scalable Network Services,” in *Proc. SOSP, 1997.*
[21] J. Chase, D. Anderson, P. Thakar, A. Vahdat, and R. Doyle, “Managing Energy and Server Resources in Hosting Centers,” *ACM SIGOPS Operating Systems Review, 35(5):103-116, 2001.*
[22] L. Grit, D. Irwin, A. Yumerefendi, and J. Chase, “Virtual Machine Hosting for Networked Clusters: Building the Foundations for Autonomic Orchestration,” in *Proc. VTDC, 2006.*
[23] T. F. Abdelzaher, K. G. Shin, and N. Bhatti, “Guarantees for Web Server End-Systems: A Control-Theoretical Approach,” *IEEE Trans on PDS, 13(3):80?6, 2002.*
[24] D. Villela, P. Pradhan, and D. Rubenstein, “Provisioning Servers in the Application Tier for E-commerce Systems,” in *Proc. IEEE IWQoS, 2004.*
[25] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, “A Survey and Comparison of Peer-to-Peer Overlay Network Schemes,” *IEEE Communications Surveys and Tutorials, 7(2):72-93 , 2005.*
[26] Planetlab. [Online]. Available: http://www.planet-lab.org/ [27] Seller API and Third Party Provider Integration Guide. [Online]. Available: http://spotcloud.com/fileadmin/docs/SpotCloudProviderGuide.pdf
[28] AWS Management Console. [Online]. Available: http://aws.amazon.com/console/
[29] J. L. H. Wang, F. Wang and J. Groen, “Measurement and Utilization of Customer-Provided Resources for Cloud Computing,” *Proc. IEEE INFOCOM, 2012.*
[30] Amazon EC2 Spot Instances. [Online]. Available: http://aws.amazon.com/ec2/spot-instances/
[31] K. Zhu, “Information Transparency of Business-to-Business Electronic Markets: A game Theoretic Analysis,” *Management Science, 50(5):670-685 , 2004.*
[32] M. Beckmann, “Bertrand-Edgeworth Duopoly Revisited,” *In: Henn R (ed) Operations Research Verfahren III, Meisenheim: Sonderdruck, Verlag Anton Hain, 55-68, 1965.*
[33] J. Farrell and E. Maskin, “Renegotiation in Repeated Games,” *Journal of Economic Theory, 1(4):327-360 , 1989.*
[34] M. Kandori, “Introduction to Repeated Games with Private Monitoring,” *Journal of Economic Theory, 102(1):1?5, 2002.*
[35] D. Abreu and A. Rubinstein, “The Structure of Nash Equilibrium in Repeated Games with Finite Automata,” *Journal of the Econometric Society, 56(6):1259-1281 , 1988.*
[36] K. Krauter, R. Buyya, and M. Maheswaran, “A Taxonomy and Survey of Grid Resource Management Systems for Distributed Computing,” *Software: Practice and Experience, 32(2):135-164 , 2002.*

[37] J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast," *Special Issue on Recent Advances in Distributed Multimedia Communications, 96(1):11-24 , 2008*.
[38] A. W. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. Spieksma4, "Interval Scheduling: A Survey," *Naval Research Logistics (NRL), 54(5):530543, 2007*.
[39] V. Paxson and S. Floyd, "Wide-area traffic: the failure of Poisson modeling," *Proc. ACM SIGCOMM, 1994*.
[40] H. Wang, F. Wang, and J. Liu, "Measurement and Gaming Analysis of SpotCloud," *Simon Fraser University, Tech, Rep.,2011, [online]: http://netsg.cs.sfu.ca/spdata/sc.pdf.*