Litigo: A Cost-Driven Model for Opaque Cloud Services

Shahid Al Noor, Rasib Khan, Mahmud Hossain, and Ragib Hasan
SECRETLab, Department of Computer and Information Sciences
University of Alabama at Birmingham, AL, USA
{shaahid, rasib, mahmud, ragib}@cis.uab.edu

***Abstract*—Cloud computing provides software, platform, and infrastructure as a service that helps organizations to perform several resource intensive tasks. The services offered by a cloud service provider are limited by provider-specific options in terms of the pre-specified configurations. Moreover, it is sometimes expensive to pay a fixed amount of money without any format of negotiation or price-matching deals for the cloud-based services and resources. Conversely, the negotiator-based model for opaque services has gained popularity in various markets, such as, for flights, hotels, and rentals. We posit that a similar opaque inventory for cloud-based services and resources is the next generation niche for consumer acquisition and service delivery in the cloud computing market. Such a model will facilitate the clients with flexible resource and service provisioning at reasonable prices, and will also allow a higher revenue and increase resource utilization for cloud service providers. In this paper, we propose Litigo, a cost-driven model for opaque service platforms for cloud computing. The Litigo component acts as a middle-man to deliver cloud-based services from a set of cloud service providers to the end users. We present a detailed cost model and comparison between establishing a cloud service vs. an opaque cloud service. Our empirical framework allows a Litigo service provider to analyze the profit model and creates the market niche accordingly. We performed extensive analysis using simulated model verification for Litigo. The proposed model delivers an opaque cloud as a service to clients at a reasonable price by maximizing the resource utilization and revenue of cloud service providers.**

***Index Terms*—Cost Model, Cloud as a Service, Service Negotiation, Opaque Services**

I. Introduction

As the popularity of cloud computing increases, new cloud service providers are joining the market to provide cloud-based services along with different policies and sophisticated pricing and sequestration strategies. On the other hand, there are other niche market areas, such as airline ticketing, hotel reservation, and real-estate rental, which have adopted a form of opaque service delivery. The negotiating agent between a client and a provider is introduced in an opaque service model to provide greater benefits for the clients in terms of prices and services, as well as for an increased revenue and utilization of resources for the service providers [1, 2]. In opaque inventory model, only certain attributes of any product are disclosed to the client during negotiation phase, which facilitates to attract both existing and new clients to their differentiated products with different pricing schemes [3]. Extensive market feasibility research has been done in the field of collaborative partners in developing business models [4]. Unfortunately, the concept of opaque service model has not been highlighted in the domain of cloud computing.

A recent research shows that only 20% of the total resources of the major cloud providers are used by clients [5]. The problem is worse for smaller cloud providers, given that they cannot afford the cost for advertising and promoting their services. The smaller cloud providers may not be able to maintain the required service level agreement and quality of service when there is surge in demand for their cloud services. Due to a segregated cloud market, the clients are forced to choose only the larger providers for their service requirements. This creates a significant barrier to entry for newer cloud computing providers into the niche market.

A major challenge for cloud providers is to efficiently distribute the virtual resources into the actual physical locations without degrading the quality of service and maximizing the resource utilization [6, 7]. The variation of requirements of clients' requests in run-time may also result in migrating the services following efficient resource migration strategies [8]. This results in fragmented resources for the cloud providers which may become unusable for a certain period of time [9, 10]. Resource migration strategies may be adopted to reduce the fragmentation and increase resource utilization [11]. Unfortunately, such approaches are expensive, not highly scalable, and unable to completely remove the fragmentation.

At the same time, the clients are forced to pay the full price for their required resources, irrespective of any demand-supply equilibrium in the cloud market [12]. Given that clients are tied to only a few major cloud providers, the offered services are strictly limited by the pre-configured specifications, regardless of the current needs of a client at a given time. To compensate for the low resource utilization of cloud providers' resources [5], the cloud providers impose full priced services without any option for price-matching deals for the clients. This creates a monopoly-centric market for cloud computing for the generally price sensitive mentality of consumers, similar to other online market sectors [13].

This paper presents a cost-driven model for opaque cloud service delivery. Our proposed model, Litigo[1], is a service negotiating component to deliver cloud computing as a service to end users. Similar to any opaque service model, Litigo provides a service front-end for cloud-requesting clients. A client presents Litigo with the service and price requirements for cloud-based resources. Litigo reviews the client's requirements and contacts several cloud providers for the given set of requirements. Additionally, Litigo may also communicate with independent agents that hold a large number of mobile and

[1] **Litigo** (/'li:.ti.go:/, ['li:.tI.go:]) is a Latin synonym for '*negotiation*'.

stationary computing devices and are willing to rent out the underutilized hardware as virtual resources via Litigo. After a detailed cost benefit analysis, Litigo accumulates the best possible deals that are offered by the individual cloud providers for the given set of requirements. Such an approach may also accommodate novel cloud service models, such as, resource aggregation from multiple providers, price-matching negotiations, and flash deals for very cheap prices from the fragmented cloud providers' resources.

Contributions: The contributions of this paper are as follows:

1) We have introduced Litigo, a cost and profit driven model for opaque cloud service delivery to negotiate cloud-based resource request and allocation.
2) We have presented extensive cost analysis and benefit models for Litigo with respect to cloud providers, illustrating the margin of profit for opaque cloud services.
3) We have implemented a detailed simulation model for Litigo and an open market for cloud providers.
4) We have performed extensive simulated experimental analysis to illustrate the feasibility of Litigo in the real world.

The rest of the paper is organized as follows. Section II presents the motivation for Litigo, followed by the related work in Section III. A reference architecture for Litigo is explained in Section IV. Cost models for cloud service providers and Litigo are presented in Section V. We present our simulation model and implementation in Section VI, followed by the experimental results in Section VII. Finally, we present a discussion on Litigo in Section VIII and conclude in Section IX.

II. Motivation

Opaque marketing companies, such as Priceline[2] and Hotwire[3], use several strategies so that the sellers can acquire more of the market beneath their retail prices and fill more of their capacity [14]. The profit of some companies can be enhanced up to 33% of their regular profit by adopting an opaque player [15]. Market research also shows that the existence of an opaque player actually helps the companies to find out and reach the price sensitive customers more conveniently than the self marketing approach. For example, Priceline increased their reservations by 38.8% in the last quarter of 2014, which resulted in 36.3% increase in revenue since 2012 [16]. We posit that a similar opaque model for cloud as a service will be beneficial for the cloud service providers as well as for the price-sensitive clients in the cloud computing market. Let us consider the following example scenario.

Example Scenario: *Charlie is planning to start a new business to enter the cloud computing market. However, he does not have unlimited source of funds to set up a whole cloud. He therefore uses the Litigo model to evaluate the financial feasibility and cost analysis of a new opaque cloud service model for his business. Charlie then sets up a service level agreement with the cloud service providers in the market and creates a new opaque cloud as a service company, Chatigo. On the other hand, Alice is an entrepreneur and is designing a cloud-based product for her new startup company. With very limited startup funds,*

[2] www.priceline.com/ [3] www.hotwire.com

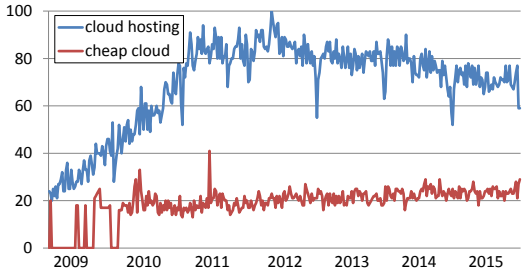

Fig. 1: Google Trend Report for Worldwide Web Search (2009 - 2015)

Alice uses Chatigo to find cheap deals for her cloud-based hosting to get her initial alpha-prototype up and running.

In the above scenario, Alice is able to find the price-matching cheap deals from Chatigo. On the other hand, Charlie does not have to bother about running his own cloud data centers, and merely acts as a mediator between Alice and the contracted cloud providers. Based on Alice's requirements, Chatigo is able to find the best deals, increases the revenue for the cloud providers, and provides a profit margin for Charlie based on the service agreements between Chatigo and the providers.

We also investigated the Google Trends Report[4] for the search keywords '*cloud hosting*' and '*cheap cloud*' from January 2009 till December 2015. As shown in Figure 1, there was a sharp increase in interest for people in cloud computing from 2009 till 2011, which then stayed almost constant till the end of 2015. At the same time, we found people are trying to find cheap cloud based services, which gradually increased from 2010 till the end of 2015. The average interest factor ratio of 'cheap cloud' and 'cloud hosting' was 0.28, implying that over one-fourth of the people interested in clouds were also trying to find a cheap cloud hosting deal on the Internet.

III. Related Work

Organizational private clouds may allow better performance for resource intensive operations. On an average, resource utilization during peak hours may exceed 80% of the total resources, resulting in degraded performance [17]. Conversely, off-peak resource utilization rarely exceeds 5% of the total resources. Therefore, having more resources than required increases the overall operational cost, as well as the costs associated with energy, maintenance, software licenses, and hiring skilled employees. Using some of the existing energy saving models, it is possible to reduce the cost associated with the maintenance [18]. However, unless the resources are properly utilized for most of the time, the company will not be able to make substantial profit.

A branch of research work has been done to integrate a group of cloud platforms into a single domain [19, 20]. For example, using an infrastructure-as-a-Service aggregator (IaaSA) [21], several IaaS providers will be able to endorse their available resources, which will facilitate the user to view their service catalogs at several levels of abstraction through a single management interface. For enhancing the performance, Costa et al. proposed a generic model for high level cloud application program interface (CAPI) with lesser vendor lock-in and fast and easier way of portability [22]. RESERVOIR is an well known research project that intends primarily the

[4] www.google.com/trends/

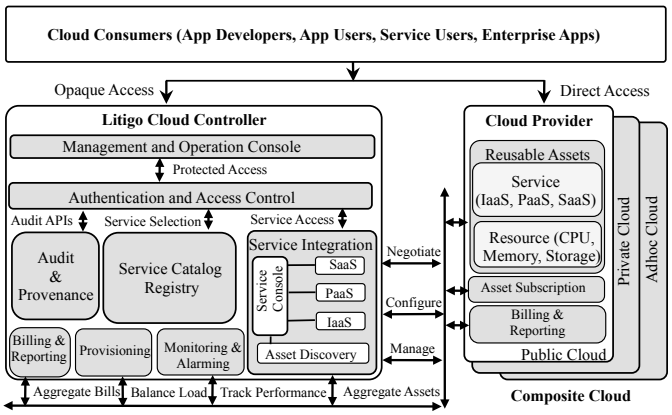

Fig. 2: Jugo Defined Architecture of Litigo [27]

deployment of a service-oriented framework in order to support dynamic interactivity among the cloud service providers [23]. A similar work is proposed by Tsai et al. [24] where the level of interactivity is improved by placing a standard abstraction layer among multiple platforms. Unlike the above research works, Litigo coalesces the services, such as provisioning, negotiation, audit and management etc., offered by various cloud service providers into a common platform for better adoptability.

Some of the researchers proposed broker based strategies where a third party broker is used to select a cloud based on user requirements. Houidi et. al. proposed a broker based approach that used an efficient splitting algorithm to split the user request and find out the best cloud platform for serving each part of the request [25]. Maximilien et al. uses a cloud agnostic middleware that serves as a broker between the client and the cloud system [26].

IV. Opaque Service Model for Cloud Computing

There are various forms of cloud service providers in the cloud market. Commonly, a cloud provider delivers Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) with abstracted cloud-based resources. We propose the concept of composite cloud as an aggregated cloud as a service via an abstracted service platform. The Litigo composite cloud is an opaque service model to deliver cloud-based resources to the end users, as shown in Figure 2. A composite cloud service may combine multiple cloud providers providing different formats of services. Moreover, a composite cloud is also expected to be able to include private cloud owners who are willing to sell cloud services from the privately owned unused and underutilized resources. Novel and evolving cloud computing frameworks may also be formed in an ad-hoc fashion using mobile and cellular devices, and the Internet-of-Things [28–30]. As shown in Figure 2, a composite cloud concept can bring all formats of cloud computing platforms under a single unified opaque service gateway.

Here, we consider a Jugo defined architecture for Litigo enabled opaque cloud service [27]. As shown in Figure 2, a Litigo-enabled cloud allows users to request for cloud resources via the Litigo controller. A Jugo-compliant architecture provides interfaces for service specifications and pricing expectations for the cloud users [27]. Additionally, the Litigo controller also provides additional internal services, such as management interfaces, audit and provenance preservation, provisioning, run-time monitoring of quality of contracted service, as well as aggregated billing and reporting. Litigo negotiates the user's service requirements with the service catalog of cloud service provider (CSP) and determines the best price and resource matching contract. Hence, a user is oblivious of the tedious task of determining the most suitable cloud from the (currently) diverse and segregated market of cloud providers. Moreover, such an opaque service model allows an opportunity for marginal cloud companies to enter the market niche, but with the control of the opaque Litigo provider in terms of quality of service.

V. Economic analysis of Litigo

The traditional approaches [31] where an internal financial unit monitors the expenditures and costs, frequently provide inaccurate result specifically during the estimation of the storage cost as they always overlook several indirect, hidden, or environmental costs associated with the product. Therefore, we provide a full cost accounting approach analysis based on the work done by Dutta et. al [32] and Schempf et al. [33], and derive a mathematical model from that analysis. Any individual, who wants to start a Litigo based business, can use our model to estimate the financial outcomes before starting the business.

A. Startup Cost

Initially, the provider needs to buy some hard drives, networking devices (e.g. connectors, wires, switches, routers etc.), and some auxiliary equipments, such as racks, cooling fans, cabinet power supply etc. Additionally, the provider needs to own or rent a floor in order to accommodate those devices. Moreover, some softwares need to be installed for handling the client's requests, organizing client's contents in the cloud, and serving those requested contents or tasks. The cloud provider often hires skilled people or third party organization on a one-time basis for installing and configuring those applications. The number of staffs required is proportional to the number of devices used in the cloud. Let us assume that for setting up a cloud for n_{client} number of clients, we estimate n_{floor} number of floors where each floor contains n_h number of hard drives, n_c number of cooling fans, n_r number of racks, n_{net} number of networking devices, and n_d number of accessories for the decoration. Suppose c_f is the cost of renting a floor for a month, c_h, c_c, c_r, c_{net}, and c_d are the average cost of the hard drives, cooling devices, racks, the networking devices, and decoration respectively. Then the initial investment cost is, $c_{device} = n_h * c_h + n_c * c_c + n_r * c_r + n_{net} * c_{net} + n_d * c_d$.

Suppose n_{sf} is the total software requirements with an average price of c_{sf} for each software, the cost of hiring one person is c_{em} and on an average, a hired person can configure n_{dev} devices. Then the one-time device installation cost is, $c_{install} = n_{sf} * c_{sf} + \frac{n_h+n_c+n_r+n_{net}}{n_{dev}} * c_{em}$.

B. Maintenance Cost

The primary costs associated with the maintenance cost are power costs for operating the servers, computers, cooling devices, networking devices, and the people working in the company. Suppose, c_{e1}, c_{e2}, c_{e3}, and c_{e4} are the daily power

cost for turning on the server, networking devices, cooling devices, and employe usage equipment respectively. Then the total power cost in each month is, $c_{power} = (c_{e1} + c_{e2} + c_{e3} + c_{e4}) * 30$.

For computing the repairing cost, suppose, p_{fail} is the probability of a devices failing in a day and c_{rep} is the average cost of repairing the device. Then the average repairing cost in a month is, $c_{repair} = p_{fail} * c_{rep} * (n_h + n_c + n_r + n_{net} + n_d) * 30$.

The installed software frequently needs to be upgraded or renewed. Suppose, on an average, $n_{upgrade}$ number of softwares needs to be upgraded and n_{renew} number of software's license needs to be renewed in a month. If $c_{upgrade}$ and c_{renew} are the average costs for upgrading and renewal for each software, then the cost of updates in each month is, $c_{update} = n_{upgrade} * c_{upgrade} + n_{renew} * c_{renew}$.

For maintaining the cloud system, the provider hires several permanent employees, such as engineers, business analysts and representatives, accountants etc., and those employees receive salary on a monthly basis. If the company hires n_{sal} employees with an average salary of c_{sal}, then the cost for employee is, $c_{employee} = n_{sal} * c_{sal}$.

If the average rental cost of each floor in a month is c_{floor} then the total maintenance cost for each floor in a month is, $c_{maintenance} = \frac{c_{power} + c_{repair} + c_{update} + c_{employee}}{n_{floor}} + c_{floor}$

C. Operational Cost

Depending upon the types, sizes, and complexity of a tasks, the system consumes different amount of power [34]. Let α be the average power consumption of a task for each unit of time and γ be the average complexity of a task. If the cost of each unit of energy is c_{energy} and the average size of the input for a task is t_{task}, then the cost of the the computation is, $c_{comp} = \alpha * {t_{task}}^{\gamma} * c_{energy}$

The user request for resources goes through the cloud management layer via a network. The cost varies with the distance of the data center, the type of network (LAN, WAN), and the total data transferred on the link. Suppose, for transferring each byte of data, the cost of a network link is c_{link} and the power consumption cost is c_{ls} and c_{lr} for sending and receiving respectively. If b_1 bytes of task is sent that generates b_2 bytes of result then the cost of network is, $c_{net} = c_{link} * (b_1 + b_2) + c_{ls} * b_1 + c_{lr} * b_2$

D. Analysis from Provider's Viewpoint

For analyzing the effect of Litigo in terms of providers' financial benefit, first we compute the revenue of a provider before Litigo is introduced. Let us assume that, $n1_{soft}$, $n1_{plat}$, and $n1_{infra}$ are the total number of clients taking SaaS, PaaS, and IaaS respectively from the cloud within a time period and $c1_{soft}$,$c1_{plat}$, and $c1_{infra}$ are the average price charged by the cloud provider for those three services respectively. Then the revenue of the cloud provider in that time period is, $rev1_{prov} = t1_{soft} * c1_{soft} + t1_{plat} * c1_{plat} + t1_{infra} * c1_{infra}$.

Now we consider that after Litigo is introduced, the clients using SaaS, PaaS, and IaaS from the cloud within the same time period is $t2_{soft}$, $t2_{plat}$, and $t2_{infra}$ respectively and pays $c2_{soft}$,$c2_{plat}$, and $c2_{infra}$ respectively on an average for those services. Then the revenue of the cloud provider in that time period is, $rev2_{prov} = t2_{soft} * c2_{soft} + t2_{plat} * c2_{plat} + t2_{infra} * c2_{infra}$.

A cloud provider will sign an agreement with any third party to deploy Litigo only if $rev2_{prov} > rev1_{prov}$. It is expected that in order to attract clients, Litigo will try to provide services at a cheaper rate than the actual price of the services. Hence, the Litigo has to get those services from the cloud provider with much cheaper rate, i.e. $c2_k < c1_k$,where k $\in \{SaaS, PaaS, IaaS\}$. Hence, according to the above equations, for a service k, where $k \in \{SaaS, PaaS, IaaS\}, t2_k > t1_k$ if the cloud provider wants to make profit using Litigo. Suppose $\delta_1, \delta_2, \delta_3$ are the percentage of discount offered by the provider for Saas, Paas, and Iaas to Litigo. If we assume that the provider wants more profit after proving the discounts, i.e. $rev2_{prov} > rev1_{prov}$, then the following equivalence relationship has to be satisfied:

$t2_{soft} * (c1_{soft} - \delta_1 * c1_{soft}/100) + t2_{plat} * (c1_{plat} - \delta_2 * c1_{plat}/100) + t2_{infra} * (c1_{infra} - \delta_3 * c1_{infra}/100) > t1_{soft} * c1_{soft} + t1_{plat} * c1_{plat} + t1_{infra} * c1_{infra}$

E. Analysis from Litigo's Viewpoint

Suppose, we want to target n_{client} number of clients and provide them the cloud service. First, we compute our financial benefit if we build a private cloud and provide cloud service from there. Let us assume that our private cloud requires $n1_{ds}$ number of data centers and a cloud central system (CCS). Suppose, for setting up the CCS, we need $nf1_{ccs}$ number of floors and each data center requires $nf1_{ds}$ number of floors on average. If we assume that the startup cost for each floor is c_{start}, then the startup cost of setting up a cloud system is, $c_{Pstrart} = nf1_{ccs} * c_{start} + nf1_{ds} * c_{start} * n1_{ds}$.

Similarly, the maintenance cost of the private cloud considering $c_{maintenance}$ cost for each floor is, $c_{Pmaintenance} = nf1_{ccs} * c_{maintenance} + nf1_{ds} * c_{maintenance} * n1_{ds}$.

Assume that a client initiates n_{task} number of tasks on an average in each month. If the operational cost for each task is c_{op}, then the cost of the operation is $c_{Pop} = n_{client} * n_{task} * c_{op}$.

Now, we assume that instead of setting up a private cloud for providing service to the client, we are building Litigo. The provider just needs to invest for building and maintaining the CCS where it will manage all the operations. If $nf2_{Litigo}$ is the total number of floors required for Litigo setup then the initial set up cost of Litigo is, $c_{Cstart} = nf2_{Litigo} * c_{start}$.

Similarly, the maintenance cost of the Litigo considering $c_{maintenance}$ cost for each floor is, $c_{Cmaintenance} = nf2_{Litigo} * c_{maintenance}$.

In Litigo, the CCS contacts the other cloud system for services. Let us assume that the contacted cloud provider charges c_{hire} for using a resource for each unit of time. Then the operational cost in Litigo is $c_{Cop} = (\alpha * {t_{task}}^{\gamma} * c_{hire} + c_{net}) * n_{client} * n_{task}$

In order to use Litigo,the total cost in Litigo should be less than the total cost of installing a private cloud. If we assume that the startup and maintenance cost of the central unit for both private cloud and Litigo is the same then we need to ensure the following equivalence relationship,

$$(\alpha * {t_{task}}^{\gamma} * c_{hire} + c_{net}) * n_{client} * n_{task} < nf1_{ds} * c_{start} * n1_{ds} + nf1_{ds} * c_{maintenance} * n1_{ds} + c_{Pop} \quad (1)$$

Since the network cost for sending task or receiving result is same in both cases, we can simplify (1) as,

$$\alpha * {t_{task}}^{\gamma} * c_{hire} * n_{client} * n_{task} < nf1_{ds} * c_{start} * n1_{ds} + nf1_{ds} * c_{maintenance} * n1_{ds} + \alpha * {t_{task}}^{\gamma} * c_{energy} * n_{client} * n_{task} \quad (2)$$

Equivalence relation (2) will be applicable only if we consider a cloud service for one month. However, the startup cost will be the same even if we continue our cloud business for multiple years. Therefore, for n number of months, we need to satisfy the following equivalence relationship in order to choose Litigo as a business model instead of building our own cloud,

$$\alpha * {t_{task}}^{\gamma} * c_{hire} * n_{client} * n_{task} * n < nf1_{ds} * c_{start} * n1_{ds} + nf1_{ds} * c_{maintenance} * n1_{ds} * n + \alpha * {t_{task}}^{\gamma} * c_{energy} * n_{client} * n_{task} * n$$

VI. Simulation Model

We designed a detailed model for our simulation setup to evaluate the proposed Litigo framework. Figure 3 illustrates the overview of the simulation model, which was set up using CloudSim[5]. CloudSim is a popular open-source project, and allows a generalized and extensible simulation modeling for experimentation of cloud computing infrastructures. The simulation model for Litigo is described in details in the following subsections.

A. Cloud Service Providers

We have created a pool of Cloud Service Providers (CSPs) for an open market operation. Each of the CSPs is described using a random number of data-centers. Resources for each of the data-centers, were allocated in a random fashion. As a result, the simulation setup allows us to create random sizes of the CSPs in terms of their available resources and prices for their offered services. The CSPs in the CloudSim simulation were created using the following functional components.

- **Cloud Service Provider (CSP):** The CloudSim simulation created c-number of CSPs for each instance of the execution.
- **Data Center (DC):** Each CSP owns d-number of DCs to host the resources for CSP_x. The number of DCs, d_x, at CSP_x, is independent of the number of DCs, $d_{\bar{x}}$, at $\text{CSP}_{\bar{x}}$.
- **Host:** Each DC owns h-number of hosts to allocate the resources for CSP_x. The number of hosts, h_y, at DC_y, is independent of the number of hosts, $d_{\bar{y}}$, at $\text{DC}_{\bar{y}}$.
- **Cloud Resource Generator (CRG):** The CRG is the randomized resource generation module within each CSP_x, where $x \in [1...c]$, and is responsible for creating the pricing and specifications of the data center resources at CSP_x. The CRG_x and $\text{CRG}_{\bar{x}}$ are independent of each other. The resource pools for the CRG are defined for the set [CPU, RAM, Storage, Bandwidth, Price].
- **Cloud Deployment Engine (CDE):** The CDE is responsible for deploying the simulated components for the CSP. The CDE_x for CSP_x obtains the number of DCs, d_x, the number of hosts, h_y in each DC_y, and the allocated resources for each of the hosts. The CDE then creates and deploys the CloudSim instances for the corresponding CSPs.

[5] http://www.cloudbus.org/cloudsim/

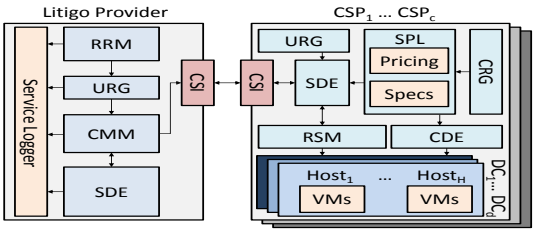

Fig. 3: The CloudSim Simulation Implementation Architecture for Litigo

- **Service Policy Layer (SPL):** The SPL holds the pricing information and the specifications of the VM configurations for CSP_x generated by the CRG_x.
- **Virtual Machine Pool (VM):** The VM pool is the CloudSim VM virtual hardware resources which are created within each of the hosts at the DCs owned by CSP_x. The specifications of the VMs are obtained from the SPL.
- **Resource Manager (RSM):** The RSM is responsible for interacting with the DCs and hosts at a CSP to create, launch, and destroy the VM instances.
- **User Request Generator (URG):** The URG is an independent looping process within a CSP. URG_x generates the user requests for VM allocations within CSP_x for a random duration of time, and represents the direct clients owned by CSP_x. The model for URG_x implies the variable amount of resources at CSP_x due to incoming resource allocation requests from CSP_x's direct clients. Therefore, the amount of available resources at the various DCs for all CSPs varies with respect to time, and cannot ensure guaranteed availability for the resource allocation requests made via Litigo.
- **Cloud-as-a-Service Interface (CSI):** The CSI defines a set of software-defined interfaces for the CSP to receive service requests from the Litigo component. We designed two services for the CSI: (a) request for the currently available resources and the corresponding prices for all DCs owned by CSP_x, and (b) request the allocation of a specified VM configuration and price at a particular DC_y owned by CSP_x.
- **Service Decision Engine (SDE):** The SDE receives any incoming VM allocation requests from both the URG (CSP's own clients) and the CSI (Litigo's clients). The SDE obtains the specification and pricing information and executes the decision process for the particular VM request. If the request is invalid, the SDE sends a failure response to the requesting component (either URG or CSI). In case of a valid request, the SDE forwards the request to the RSM to allocate the VM. If the RSM is unsuccessful in launching the VM with the currently available resources, a failure message is sent back to the SDE.

B. Litigo Component

The Litigo component is defined as an aggregated service provider for requesting cloud-based resources for Litigo-clients from a set of available CSPs. The model generates batches of user requests which includes specific amounts of resources and an estimated payable price. The Litigo component in the

CloudSim simulation was created using the following functional components.

- **Resource Requirement Modeling (RRM):** The RRM creates various combinations of VM resource requests when requested. The model for the requested resources are based on the resource pools similar to the CRG in the CSP from the set [CPU, RAM, Storage, Bandwidth, Price].
- **User Request Generator (URG):** The URG is an independent looping process within the Litigo component. The URG generates the user requests for VM allocation requests and represents the clients owned by Litigo for a random duration of time. The model for URG implies the incoming resource allocation requests from CSP_x's indirect clients via the Litigo. Therefore, given that the amount of available resources at the CSPs varies with respect to time, CSP cannot ensure guaranteed availability for the resource allocation requests made via Litigo on behalf of their clients.
- **Cloud-as-a-Service Management Module (CMM):** The CMM within the Litigo component is responsible for managing the CSP clients for Litigo. The CMM maintains the list of available resources for all DCs for CSPs and their corresponding unit prices. The CMM is also responsible for issuing VM allocation requests to a particular DC owned by a specific CSP.
- **Cloud-as-a-Service Interface (CSI):** The CSI defines a set of software-defined interfaces for the Litigo component to request services from the CSP.
- **Service Decision Engine (SDE):** The SDE for the Litigo component is responsible for finding a suitable CSP to host the requested VM for a given RRM-generated model. The SDE obtains the list of available resources from the CMM. The price and resource matching for the RRM request is performed with the currently available resources and their unit prices from the CSPs. If a match is found, a VM allocation request is dispatched to CSP_x via the CSI by the CMM. However, the available resources at CSP_x may change during the time that the Litigo is trying to decide an RMM-match. In case a failure is received from CSP_x for the given VM request, the SDE performs the process again till there are no prospective matches.
- **Service Logger:** The service logger maintains the list of user requests generated by the Litigo component and maintains their statuses of successes and failures.

VII. Experimental Evaluation

Next, we utilized the CloudSim simulation model implementation for experimental evaluation of Litigo.

A. Experimental Setup

We identified the minimum and maximum amount of processing cores, processing power, ram, hard drives, and bandwidth used in the hosts of major cloud service providers and configured the hosts for simulation by randomly selecting a size for each of those resource categories. We chose the size of the datacenter in a cloud and number of hosts in a datacenter randomly from the maximum datacenter size and

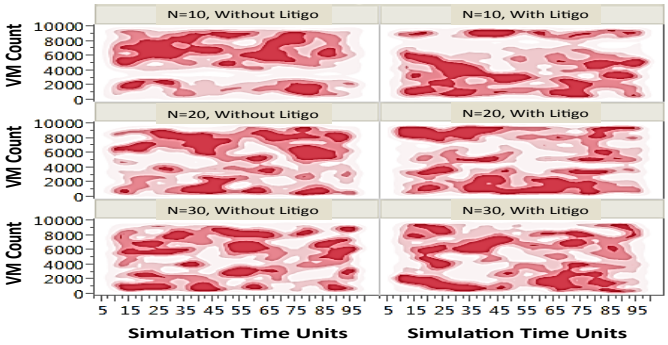

Fig. 4: Density Map for VM Request Generation in CloudSim

host size among all the clouds in the current environment. For each CSP_x, the number of DCs, d_x, is selected randomly between the range 1 and 10. Within each DC_y, the number of hosts, h_y, is randomly selected between the range 1 and 30000. The CRG creates the resource specifications as follows: CPU core $\in [10 * 2^i | 0 <= i <= 5]$, RAM $\in [10 * 512 * i | 0 < i <= 400]$, Storage $\in [10 * 4 * i | 0 < i <= 12000]$, Bandwidth $\in [10 * 2 * i | 0 <= i <= 250]$. We did a comprehensive analysis of the offered services and the corresponding prices for some of the major cloud service providers and selected the unit price for each of the resources randomly within their lowest and highest offered values. We executed the CloudSim simulation for Litigo for 100 simulation time units. After successful creation, the VM is terminated randomly after $1 < T_D < 5$ simulated time units (i.e. between 1 to 5 hours). We performed the measurements on multiple iterations, with 10 to 30 CSPs, with an increment of 10 additional CSPs each time. The URG_{CSP} creates v-number of VM requests within the CSP, where $5000 < v < 10000$. On the other hand, the URG_{CaaS} creates wv-number of VM requests on the CaaS to be sent to the CSPs, where w $\in [2, 4]$.

B. Simulation Results

Request Distribution: Initially, we examined the request generation pattern and distribution of the requests over time. Figure 4 illustrates the contour heat maps for VM requests generated in our simulated environment. We ran the simulation in 6 different settings: (a) $N = 10$, $V = 10000$, without Litigo, (b) $N = 10$, $V = 10000$, with Litigo, (c) $N = 20$, $V = 10000$, without Litigo, (d) $N = 20$, $V = 10000$, with Litigo, (e) $N = 30$, $V = 10000$, without Litigo, and (f) $N = 30$, $V = 10000$, with Litigo. As seen from Figure 4, the service requests to the CSP was randomly and evenly distributed over the 100 simulated time units. Therefore, at a given time, the amount of available assets was randomly distributed over the CSPs for all cases.

Mean Launch Time: Next, we evaluated the mean times to launch a VM for both cases of with and without Litigo for all the three sizes of CSPs. The summary of the results are presented in Table I. As seen from the results, we observed that the mean time to launch VMs for CSPs reduced with the presence of Litigo. Moreover, we also measured the time taken for Litigo to negotiate and find a suitable CSP to request a cloud resource. We observed that the mean time taken for Litigo requested VMs were similar to the CSP requested VMs, for 10 and 20 CSPs. However, with 30 CSPs, the launch

	Number of Clouds = 10		**Number of Clouds** = 20		**Number of Clouds** = 30	
	Without Litigo	With Litigo	Without Litigo	With Litigo	Without Litigo	With Litigo
Mean Time to Launch - Simulated Time Units (Std. Dev.)						
CSP Requested VMs	0.110 (0.029)	0.107 (0.026)	0.106 (0.028)	0.100 (10^{-10})	0.099 (10^{-9})	0.103 (0.019)
LitigoRequested VMs	n/a	0.100 (10^{-9})	n/a	0.100 (10^{-10})	n/a	0.191 (0.021)

TABLE I: Litigo Simulation Summary

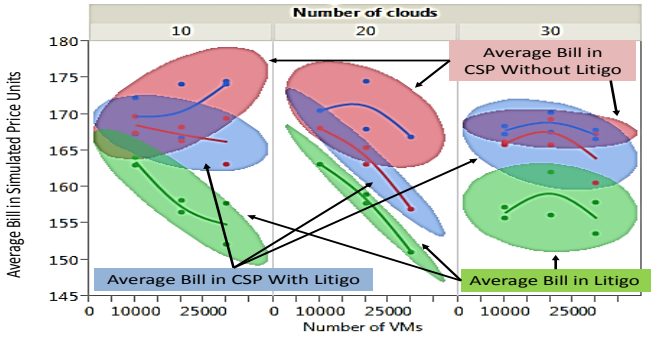

Fig. 5: Average Bill Vs. Number of VMs and CSPs

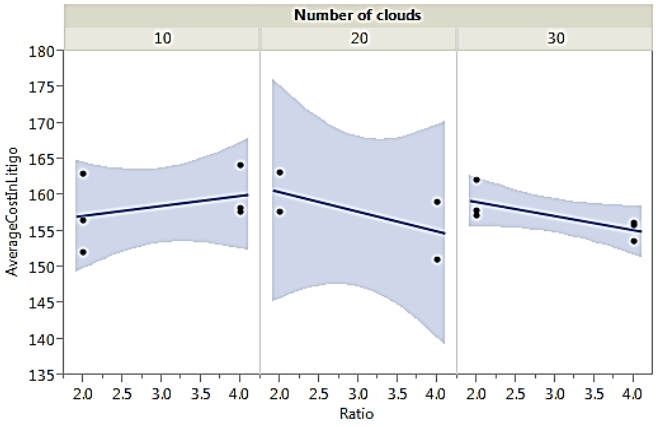

Fig. 6: Average Bill Vs. Litigo-Request-Ratio and Number of CSPs

time increased slightly, implying a more intense negotiation mechanism between Litigo and the 30 CSPs.

Average Bill: We evaluated the effects of the number of VMs and the number of CSPs on the average accumulated bill for requested resources. From Figure 5, we see that the average bill for running a VM in Litigo is less than that of the CSP. Moreover, we can see that the average bill of CSP with Litigo also decreases while the bill increases in CSP without Litigo. We also observed that the bill decreases rapidly in Litigo for larger numbers of VMs when the number of CSPs are 10 and 20. However, the average negotiated bill stabilizes as the number of CSPs grows to 30. As a result, this confirms a demand-supply equilibrium for cloud resources for a negotiation-based cloud service delivery.

We also analyzed the effects of the average accumulated bill for Litigo with respect to the ratio of VM requests between Litigo and each cloud. Figure 6 illustrates the billing trend for increasing ratio of requests for Litigo with respect to the number of CSPs. We found that increasing the ratio for Litigo for 10 CSPs increases the usage bill. However, for increasing number of CSPs, we observed significant decrease in the usage bills for resources allocated via Litigo. Therefore, we can posit that increasing popularity of opaque services, such as Litigo,

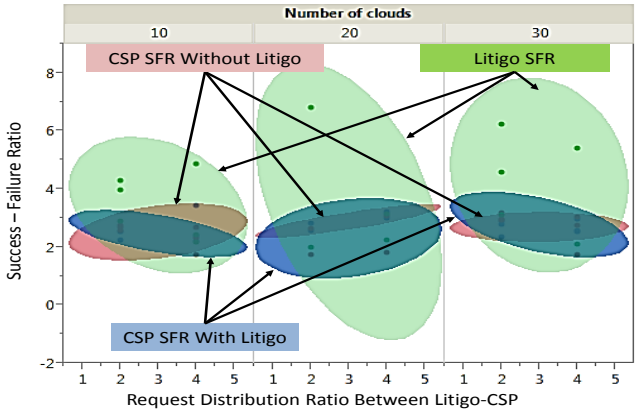

Fig. 7: Success-Failure Ratio Vs. Litigo-Request-Ratio and Number of CSPs

will result in reduced costs for cloud users, given that more CSPs are contracted via Litigo.

Success Failure Ratio: Finally, we examined the success-failure ratio (SFR) of VM allocation requests for CSPs (with and without Litigo) and for the Litigo provider. Figure 7 illustrates the range map for SFRs with respect to the number of CSPs and the request distribution ratio between Litigo and each CSP. We found that increasing the number of CSPs results in increased SFRs for Litigo and the CSPs, both with and without Litigo. However, we found the SFR for Litigo is significantly high than CSPs for all cases. Hence, we can say that the increasing popularity of Litigo will directly motivate cloud users to find better price-matching deals for cheaper prices.

VIII. Discussion

From our experimental results, we see that along with providing higher success rates for VM creation, Litigo also takes smaller amount of time for serving client requests for VM creation. Moreover, clients can also get their requested VM for a much cheaper rate. However, we only considered VM creation as a sample service in our experiment. The VM access time might take a little bit longer time as additional latency will be added for sending each access request through the Litigo controller.

Fragmented resources is one of the major concerns for the current cloud service providers. In our current work, we did not show the effect of Litigo on the fragmented resources for an individual cloud provider. As our future work, we are planning to provide a comparison of total fragmented resources between environments with Litigo and without Litigo.

Additionally, there are several technical challenges associated with implementing Litigo. A Litigo-enabled provider should allow dynamic mapping of resources for flexible migration plans between cloud providers. Moreover, the Litigo controller must ensure the quality of service and service level agreements for all the contracted cloud providers, especially for newer

cloud providers. As a result, the Litigo provider must provide mechanisms for service elasticity to evade performance deterioration for a given cloud provider. Our current work involves the development of Jugo, a detailed Litigo-compliant operational architecture for opaque cloud services [27]. The future directions for Litigo includes development of intelligent negotiation strategies with cloud providers for fragmented resource pricing.

IX. Conclusion

In this paper, we proposed a concept of opaque service delivery for composite cloud frameworks. Our proposed model for Litigo-enabled providers illustrates the practical aspects of a novel service platform for delivering cloud-based services to the end users. We have presented market research evidence from various market niche domains, along with the interests for cloud users, to emphasize the necessity of such an opaque service model for cloud computing. Moreover, our detailed cost-model provides an insightful motivation for opaque players in the cloud market. We posit that an opaque service provider will allow a healthier cloud market competition, provide price-matching for cloud users, allow flexible service specifications, and deliver cheap deals for cloud resource rentals. Moreover, such a model will also allow an easier market entry process for new and diverse cloud service providers, without a monopoly-oriented and fixed-pricing model for cloud users.

We evaluated the proposed Litigo approach for opaque cloud services using an extensive and fine-grained simulated model checker. The CloudSim simulation was executed using numerous control parameters. The Litigo and cloud service provider models for CloudSim were defined using service specifications and hardware configurations which are currently offered by the major cloud providers. Our results depicted promising results for the proposed Litigo-provider in a cloud market, resulting in cheaper average prices paid for cloud-based resources. Moreover, we also found a greater success rate for both Litigo and cloud providers for their requested resources. This implies that an opaque middle-man in cloud computing can be beneficial for both the end users and the current cloud service providers.

X. Acknowledgements

This research was supported by the National Science Foundation CAREER Award CNS-1351038.

References

[1] D. Shapiro and X. Shi, "Market segmentation: The role of opaque travel agencies," *Journal of Economics & Management Strategy*, vol. 17, no. 4, 2008.

[2] L. Illge and L. Preuss, "Strategies for sustainable cotton: comparing niche with mainstream markets," *Corporate Social Responsibility and Environmental Management*, vol. 19, no. 2, pp. 102–113, 2012.

[3] C. K. Anderson, "Pricing and market segmentation using opaque selling mechanisms," http://scholarship.sha.cornell.edu/cgi/viewcontent.cgi?article=1373&context=articles, 2012.

[4] K. Popp and R. Meyer, *Profit from Software Ecosystems: Business Models, Ecosystems and Partnerships in the Software Industry*. BoD–Books on Demand, 2010.

[5] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, 2010.

[6] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Proc. of IEEE/ACM CCGRID*, 2010.

[7] G. Foster, G. Keller, M. Tighe, H. Lutfiyya, and M. Bauer, "The right tool for the job: Switching data centre management strategies at runtime," in *Proc. of IFIP/IEEE IM*, 2013.

[8] M. Tighe, G. Keller, M. Bauer, and H. Lutfiyya, "A distributed approach to dynamic vm management," in *Proc. of IEEE CNSM*, 2013.

[9] L. Tomás, B. Caminero, and C. Carrion, "Improving grid resource usage: Metrics for measuring fragmentation," in *IEEE/ACM CCGRID*, 2012.

[10] D. Pandit, S. Chattopadhyay, M. Chattopadhyay, and N. Chaki, "Resource allocation in cloud using simulated annealing," in *Proc. of AIMoC*, 2014.

[11] S. Chen, J. Wu, and Z. Lu, "A cloud computing resource scheduling policy based on genetic algorithm with multiple fitness," in *Proc. of IEEE CIT*, 2012.

[12] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," in *Proc. of IEEE HPCC*, 2008.

[13] C. Péloquin, http://veilletourisme.ca/2005/02/08/les-modeles-%C2%ABopaques%C2%BB-revolutionneront-ils-la-distribution/, 2005.

[14] J. Wilner, "Exclusive interview: Priceline.com," http://www.ecommercetimes.com/story/2267.html, 2000.

[15] Y. Jiang, "Price discrimination with opaque products," *J Revenue Pricing Manag*, vol. 6, no. 2, 2007.

[16] M. McGrath, "Priceline beats on profit and revenue, boosted by 38.8% increase in bookings," http://www.forbes.com/sites/maggiemcgrath/2014/02/20/priceline-beats-on-profit-and-revenue-boosted-by-38-8-increase-in-bookings, 2014.

[17] C. Longbottom, "Cutting through the promises and problems of private cloud services," Quocirca, http://www.computerweekly.com/tip/Cutting-through-the-promises-and-problems-of-private-cloud-services, Tech. Rep., 2012.

[18] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future generation computer systems*, vol. 28, no. 5, pp. 755–768, 2012.

[19] L. Osmani, S. Toor, M. Komu, M. Kortelainen, T. Linden, J. White, R. Khan, P. Eerola, and S. Tarkoma, "Secure cloud connectivity for scientific applications," *IEEE TSC*, vol. PP, no. 99, pp. 1–1, 2015.

[20] M. Komu, M. Sethi, R. Mallavarapu, H. Oirola, R. Khan, and S. Tarkoma, "Secure networking for virtual machines in the cloud," in *Proc. of IEEE CLUSTER WORKSHOPS*, Sept 2012, pp. 88–96.

[21] B. S. Lee, S. Yan, D. Ma, and G. Zhao, "Aggregating iaas service," in *Annual SRII Global Conference*, 2011.

[22] B. Costa, M. Matos, and A. Sousa, "Capi: cloud computing api," *Inforum, Simpósio de Informática*, 2009.

[23] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres *et al.*, "The reservoir model and architecture for open federated cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, 2009.

[24] W.-T. Tsai, X. Sun, and J. Balasooriya, "Service-oriented cloud computing architecture," in *Proc. of IEEE ITNG*, 2010.

[25] I. Houidi, M. Mechtri, W. Louati, and D. Zeghlache, "Cloud service delivery across multiple cloud platforms," in *IEEE SCC*, 2011.

[26] E. M. Maximilien, A. Ranabahu, R. Engehausen, and L. C. Anderson, "Toward cloud-agnostic middlewares," in *Proc. of ACM SIGPLAN OOPSLA*, 2009.

[27] M. M. Hossain, R. Khan, S. A. Noor, and R. Hasan, "Jugo: A generic architecture for composite cloud as a service," in *Proc. of IEEE CLOUD*, 2016.

[28] R. Hasan, M. M. Hossain, and R. Khan, "Aura: an iot based cloud infrastructure for localized mobile computation outsourcing," in *IEEE MobileCloud*, 2015.

[29] S. Al Noor, R. Hasan, and M. M. Haque, "Cellcloud: A novel cost effective formation of mobile cloud based on bidding incentives," in *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*. IEEE, 2014, pp. 200–207.

[30] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. of ACM MCC*, 2012.

[31] C. T. Horngren, G. Foster, S. M. Datar, M. Rajan, C. Ittner, and A. A. Baldwin, "Cost accounting: A managerial emphasis,," *Issues in Accounting Education*, vol. 25, no. 4, 2010.

[32] A. K. Dutta and R. Hasan, "How much does storage really cost? towards a full cost accounting model for data storage," in *Economics of Grids, Clouds, Systems, and Services*, 2013.

[33] N. Conway-Schempf, "Full cost accounting: A course module on incorporating environmental and social costs into traditional business accounting systems," *Pittsburgh: Carnegie Mellon University*, 1998.

[34] M. Rashid, L. Ardito, and M. Torchiano, "Energy consumption analysis of algorithms implementations," in *ACM/IEEE ESEM*, 2015.