2015 IEEE 7th International Conference on Cloud Computing Technology and Science

A Privacy and Price-Aware Inter-cloud System

Yuanfang Chi[1], Wei Cai[1], Zhen Hong[1], Henry C.B. Chan[2] and Victor C.M. Leung[1]
[1]Department of Electrical and Computer Engineering, The University of British Columbia, Canada
[2] Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[1] {yuanchi, weicai, vleung}@ece.ubc.ca, [1] hongz.ubc@gmail.com, [2] cshchan@comp.polyu.edu.hk

***Abstract*—Cloud service selection and financial expense are two main concerns of users considering adoption of cloud computing services. In this paper, we propose a novel cloud federation that is cognitive to the dynamic prices. The cloud federation system first determines on which cloud services should the user applications be deployed. Then, when a cloud provider is charging too high for a VM, the proposed system automatically migrates user tasks to a cloud system that is charging at a lower rate. We discuss the architectural framework and platform design, provide a mathematic formulation and investigate a total service fee minimization approach with privacy constraints. Preliminary simulation results demonstrate the proposed system can lower the cost of cloud services by exploiting the advantages of different price policies provided by multiple cloud providers.**

***Index Terms*—Inter-Cloud; Privacy; Pricing; Cognitive; Cloud Computing**

I. INTRODUCTION

The computing industry has started to realize that cloud computing is becoming the next modality of computing infrastructures. With the ease of summon and expand computing power, the cloud computing has also become a decent solution for parallel computing. However, in contrary to local computing model, there are arising concerns on hosting applications on the clouds. One of the obstacles that prevents the adoption of cloud services [1] is the service availability and business continuity. Since the cloud is the best effort service, many organizations are affraid to rely on a third-party service provider to ensure the organizations' service availability and business continuity. Another concern relates to data security, confidentiality and auditability. Crackers and hackers have never stopped their efforts on attacking information systems to steal users' virtual properties. Since the access is public, cloud exposes their systems to more public attacks than conventional data centers. On the other hand, users cannot easily extract their data and programs from one cloud service to another. When a cloud data center crushes, users can hardly move their data to another data center on time. Plenty of research works are devoted to investigate secured and robust cloud service infrastructures. All of these obstacles can be removed by using services from a cloud federation or inter-cloud [2] system that supports distributed cloud services where one cloud could use the computational, storage or network resources of other clouds [3] to complete a task together [4]. Similar to cloud-RAID [5], better privacy protection can be achieved if users were able to control on which clouds the highly sensitive software classes or objects are hosted and distribute the rest of system function codes on other clouds. This way, no public cloud provider could acquire all the program pieces. Also, to select the best mix of service offering from an abundance of possibilities, users must consider complex dependencies and heterogeneous sets of criteria. Even professional technicians will be confused by the variety of service providers, service utility levels, pricing models, geographical distance and so on. Some researchers have investigated decision support techniques for automating cloud service selection [6]. However, this static supporting tool cannot adapt the users' need in varying pricing scenarios and it cannot exploit all advantages among multiple clouds simultaneously. With programs and data scattered among and backed-up by different data centers maintained by different cloud providers, also with the automated service choosing and program/data sets migration ability, the cloud service are more reliable and convincible for users to adopt.

In this paper, we propose an inter-cloud based price-aware system intended to provide a tool for users to automatically choose and migrate their applications to a cloud provider that is charging at a lower rate. Further, we take the advantage of program decomposition techniques, where a large parallel computing program can be breaked down into system functions as smaller classes or objects and data entities[1]. Our inter-cloud system allows users to configure the privacy constraints of their program by defining the number of program or data sets that could be hold in one cloud service concurrently. The system looks up current prices a cloud provider charges for their VM at a pre-defined time interval. Then, each of the program and data sets can be assigned to a cloud service charging at the lowest rate according to the privacy constraints. The cost of running the program or data set at the destination serive, along with the cost of migrating the program or data sets to the destimation service are considered before the migration decision is made. As a price-aware cognitive system, the proposed platform collects pricing information from multiple clouds and dynamically adapts its execution modality on purpose of minimizing total cost. This dynamic adaption seeks optimal assignment of codes and data sets of user decomposed application, which is a procedure that uses the update-to-date prices to predict future trend of cost and eventually lead to a lower priced solution. Hence, it is also a user-oriented system that benefits its customers financially. The rest of this paper is organized as follows. First, we study some related studies in Section II. Then in Section III, we

---

[1] https://en.wikipedia.org/wiki/Decomposition

978-1-4673-9560-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CloudCom.2015.50

provide an overview of our proposed system. A problem formulation of the system is presented in Section IV. We evaluate our proposal by computer simulations in Section V. Section VI concludes this paper.

II. RELATED WORK

A. Inter-cloud Systems

The term of cloud federation, or inter-cloud, the cloud of clouds, was first introduced by Kevin Kelly in 2007. The inter-cloud is analogous in the way the Internet works. An Internet service provider that has an endpoint attached to it will access or deliver traffic from/to source/destination addresses outside of its service area by using Internet routing protocols with other Internet service providers with pre-arranged exchange or peering relationships. The work [7] first proposed the inter-cloud blueprint to describe the high level architecture of the interoperating of multiple clouds. With the concept of inter-cloud, distributed software systems [8] explored a new application scenario. Federated cloud system helps to achieve better QoS, reliability and flexibility [9]. Authors in [10] proposed a cloud federation system that provides profit-aware solutions in order to receive benefits from pricing policies provided by multiple clouds. Some of the other cloud federation systems are proposed for load balancing and distribution of elastic applications among different cloud data centers to achieve reasonable QoS levels [11].

B. Privacy Regulation Through Across-Clouds Distribution

Keeping computing tasks that involve sensitive data in a private cloud and outsourcing the rest of computing tasks that involve insensitive data to public clouds is a preliminary solution for data security. However, as more data-intensive computing tasks are required, this kind of hybird cloud computing has became inappropriate. [12] proposed a Privacy-Aware Data Intensive Computing on Hybird Clouds that automatically splites data-intensive tasks according to security levels of the data. They modified MapReduce's distributed file system to replicate data and send sanitized data to the public cloud. Using multiple public clouds to ensure data security is also a popular solution. Faults in software or hardware in cloud computing are known as Byzantine faults. Many research work has been done on Byzantine fault tolerance [13]. DepSky virtual storage cloud system [14] leverages the Byzantine quorum system protocols to ensure data security. The DepSky system is consisted of n clouds, while the DepSky system reads and write to each cloud seperatelly. With intensive tests, experiences of working on a specific cloud and extrodinary positive user reviews, a cloud provider might be trustworthy. Besides, a series of cloud-RAID work [5] tackle the problem by encrypting and encoding the original data and later by distributing the fragments transparently across multiple providers. This way, none of the storage vendors can see the full picture of the client's data.

C. Pricing Models

Appropriate pricing or charging models are the key to success for businesses in any industry. Cloud providers are devoted to optimize their pricing strategies to increase resource utilization, in order to achieve higher revenues. Infrastructure as a Service (IaaS) provides computing infrastructures such as processing unit, storage and network to customers. Some examples of IaaS are Amazon EC2 and Google Compute Engine. **Computing Pricing:** there has been various pricing policies in the market, regarding the combination of CPU and memory usage: *i) Reserved Instances:* provides the option for customers to subscript to a cloud service for a period with a non-refundable one-time payment at a lower rate [15]. *ii) Freemium and Usage-based:* provides free but limited amount of resources for a limited time period to encourage potential cloud tenants to try-out their cloud services. *iii) Spot Instances:* sets its spot prices through a market-driven auction and publishes its spot price for the next time period online so customers can run those instances as long as their bid exceeds the current spot price. Spot Instance pricing allows cloud providers to sell more of their unused resources at the highest possible rate while preserving its control over the spot price [16]. **Networking Pricing:** Despite various types of pricing policies, almost all networking pricing models follow a principle that charge users' internet usage bidirectional with different standards[2]. In general, ingress is cheaper than egress. **Storage Pricing:** Storage pricings provided by various cloud services are relatively lower compared to computing and networking. Microsoft Azure [3] provides four types of storage, including block blobs, page blobs and disks, tables and queues and files (in preview stage). It also has four redundancy levels, including locally redundant, zone redundant, geographically redundant and read-access geographically redundant. In addition to basic storage price, Amazon S3 and Google Cloud Storag also charge users for data operations (request, delete, etc.) and data transfer. In this work, we simply use network transmission to represent these additional cost applied to the storages.

D. Cognitive Parallel Computing Software System

Parallel computing has already been widely adopted as a solution in big data processing and complex computing [17]. The key of parallel computing is the decomposed and distributed storage and processing. For example, Hadoop decomposes data into chunks and distributes them among computer clusters. Then, based on the data each node contains, the Hadoop MapReduce sends appropriate program module to each node to process data in parallel [4]. With decomposed and distributed programs, cognitive optimization of resource allocation is also an open issue. Intrinsically as a group of dynamic partitioning problem, research on the dynamic partitioning between cloud and users' mobile terminal has

---

[2] https://cloud.google.com/products/compute-engine/

[3] https://azure.microsoft.com

[4] https://en.wikipedia.org/wiki/Apache_Hadoop

been conducted from the perspectives of offline K-step approach [18] and flexible partitioning [19]. A similar idea has been used in [20], which has designed and developed a cognitive platform that enables task migration and dynamic task allocation between the cloud server and the devices. [21] proposed a decomposition algorithm that decompose a sequential program into speculatively parallel threads that can run on multi-processor chip, with considerations of data dependency and load imbalance. In [22], authors proposed a program decomposition framework that provides near-optimal mappings of program segments to machines with minimum-cost. Data elements needed for the program segments can be transfered among machines. The cost of executing a program segment depends on machine selections and associated data transfer costs.

III. SYSTEM OVERVIEW

In this section, we provide a system overview for the proposed inter-cloud based price-aware parallel computing system.

A. Architecture

The architectural framework we consider for the inter-cloud parallel computing system is shown in Fig. 1. The platform concatenates multiple cloud services to provide a unified inter-cloud environment for the parallel computing applications. Essentially, it is a three-layer software system: application layer (consists of the code layer and the data set layer), platform layer and cloud infrastructure layer, from top to bottom. As the middle-ware between the cloud infrastructure and application layer, the inter-cloud platform monitors the real-time pricing from different providers and cognitively adjust the decomposed module of the parallel computing application to minimize the overall price.

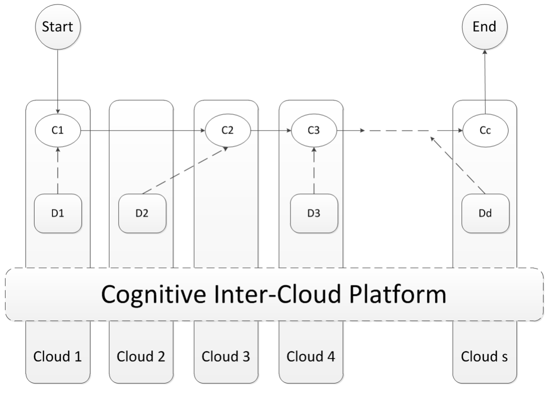

Fig. 1. Architectural Framework for Cognitive Inter-Cloud System

1) Code Definition: According to the decomposable nature of parallel computing programs, we define c program codes in set C and d data chunks in set D as illustrated as ovals and rectangles in Fig. 1. We also define s cloud service providers S. Note that some cloud server hosts both code and data set, while some of the others only supports either code or data set. Considering the code-data relationship between program codes and data chunks, we define that a program code can be associated with 0 to d data chunks, while a data chunk can be accessed by 0 to c program codes. Furthermore, we consider the code-data relationship as a directed graph G, where the directed edges are the data flow from data chunks to the program codes, the weights of the edges are the data size to be transmitted. In general, a program code only require a small proportion of data in the data chunk. In this paper, we denote ω as the data access proportion. Note that if a program code and its associated data chunk are located in different cloud server, an inter-cloud data transmission is required.

2) Code Migration: One of the key features of our inter-cloud system is the capability of program and data set migrations among multiple cloud services on demand. With the development of inter-cloud networking technologies [23] and flexible dispatch of mobile agents [24], the inter-cloud code migration is feasible in existing software systems. A mobile agent is a composition of computer software and data, which is able to migrate from one cloud to another autonomously and continue its execution at the destination. In this particular system, once a better assignment solution is adopted, designated program codes or data chunks can be encapsulated into mobile agents and dispatched to the destination cloud.

3) Inter-Cloud Message Exchange: In order to facilitate the work flow in a parallel computing system, the program codes need to communicate with each other through messages, including native context states, processed data, control signals, etc. Since theprogram codes are executed in a distributed manner among multiple clouds and a single code can be hosted in different clouds under different circumstances, a message exchanges between two codes can be either local invocations (e.g., when the two codes are hosted in the same cloud) or remote calls (e.g., when the two codes are executed in distinct clouds). Hence, a dynamic message forwarding mechanism that determines the destination of a message is needed in the cognitive inter-cloud platform.

B. Platform Design

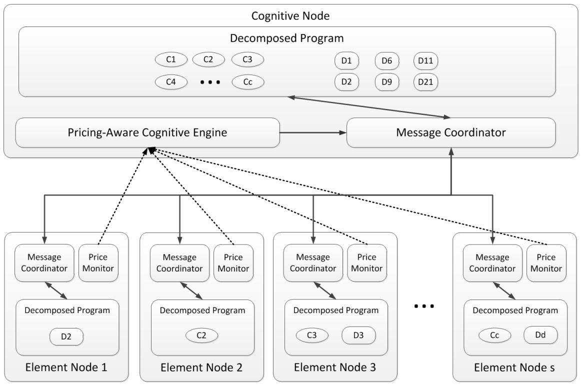

Fig. 2. Design of Decomposed Inter-Cloud Software Platform

The inter-cloud system introduces challenges including decomposition granularity, response latency, synchronization frequency and application programming interface design. In order to facilitate the proposed architectural framework, we design the cognitive inter-cloud platform with both security and pricing awareness as shown in Fig. 2.

Similar to the concept of *Master Node* and *Slave Node* implemented in the Hadoop[5] system, the proposed platform incorporates *Cognitive Node* and *Element Node* to facilitate the cognitive feature. A *Cognitive Node* is the core of the whole network, which is supposed to be hosted in a secured cloud infrastructure, e.g., a private cloud. The *Pricing-Aware Cognitive Engine* collects the pricing data from *Element Node*s in real-time and makes cognitive decisions to optimize the whole inter-cloud system. Note that both *Cognitive Node* and *Element Node* contain a *Message Coordinator*, which serves as the router for inter-code message exchange. The *Pricing-Aware Cognitive Engine*'s decision provides the reference for message redirection, which is broadcasted by the *Cognitive Node* to all *Element Node*s, together with the routing information of all nodes spreading in multiple clouds.

IV. PROBLEM FORMULATION

In this section, we mathematically formulate the inter-cloud system. For a particular user application, we denote the number of available cloud service providers as s, the number of program codes as c, and the number of data chunks as d.

A. Program Feature

In this work, we model the code-data relationship as a $1-n, \{n = 0, 1, 2, 3.., d\}$ pair, which means a program code can access 0 to d data chunks distributed in various cloud servers. We formulate the code-data relationship as $c \times d$ logical matrix L, in which the numeric value of elements are defined to be either 0 or 1. Hence, $L_{ij} = 1$ indicates that the ith program code requires data chunk j for its procedure. In addition, we denote the message exchange between code as a $c \times c$ matrix M, where M_{ij} indicates the message data size between the ith and jth program codes.

B. Dynamic Pricing

The motivation of cognitive selection in the proposed system is to pursue overall lower costs. With the assumption that each cloud service is charging with dynamic prices, hereby we formulate instance prices for the s cloud service providers as vectors α and β with length of s, where α represents the unit computational resource prices and β represents the unit data storage prices, respectively. In addition, we formulate the network prices for the s cloud service providers as ρ and θ, representing the unit input and output bandwidth prices.

C. Assignment of Codes and Data Sets

A key of system design is to determine the assignment of program codes and data chunks. We formulate the assignment of cloud codes over cloud service providers as a $c \times s$ matrix A. It is defined as a logical matrix, where $A_{ij} = 1$ represents that the ith code is executed at the jth cloud service provider. Similar to code assignment, we define a $d \times s$ logical matrix B to formulate the assignment of data chunks over cloud service providers. $B_{ij} = 1$ indicates that the ith data chunk is stored at the jth cloud service provider.

D. Assignment Constraints

1) Software Integrity: In order to guarantee the completeness of the software system, every program code and data set chunk is required to be assigned to either one or more cloud service providers. In order to simplify our model, we only consider the case that no duplicate exists in this work. Therefore, the constraints on software integrity can be described by the following equations:

$$\sum_{j=1}^{n} A_{ij} = 1, \forall i \in A_{ij} \tag{1}$$

$$\sum_{j=1}^{n} B_{ij} = 1, \forall i \in B_{ij} \tag{2}$$

2) Privacy Assurance: Privacy assurance involves a series of techniques and SLAs, which constrain the access of sensitive data. With some strict security requirements, some sensitive data even need to be stored in cloud servers within certain geographical areas. For instance, some healthcare data cannot be transferred outside the US territory, according to specific laws and legislations. In this paper, we demonstrate the security of a software system by a set of *security levels* Θ and Φ, which represent the security restriction on the assignment of program codes and data chunks, respectively. The value of *security level* $Theta$ is defined as the maximum quantity of program codes that are allowed to be hosted in the same cloud. Similarly, Φ constrains the coexistence of multiple data sets for a specific cloud service. Accordingly, $\Theta \in [1, c]$ and $\Phi \in [1, d]$, in which value of 1 represents that all program codes or data chunks shall be distributed among different clouds, while the values of c and d provide complete freedom for assigning the program codes and data chunks. Thus, the smaller Θ and Φ are, the higher security level is assured. With these definitions, we derive the constraints of security assurance as follows:

$$\sum_{i=1}^{n} A_{ij} \leq \Theta, \forall j \in A_{ij} \tag{3}$$

$$\sum_{i=1}^{n} B_{ij} \leq \Phi, \forall j \in B_{ij} \tag{4}$$

Apparently, security and pricing represent a pair of trade-off that the software users shall be aware of.

E. Price Calculation

1) Computing Price: According to above formulations, the total computing price P_c is derived as follows:

$$P_c = ||(C^T \cdot A) \odot \alpha^T||_1 \tag{5}$$

---

[5] http://hadoop.apache.org/

2) Storage Price: According to above formulations, the total storage price P_d is derived as follows:

$$P_d = ||(D^T \cdot B) \odot \beta^T||_1 \tag{6}$$

3) Networking Price: We first derive the data networking volume matrix N as following algorithm:

Algorithm 1 Data Networking Matrix Algorithm

```
Initiate a $s \times s$ all 0 matrix $N$
for each $(i, j)$ in $L_{ij} == 1$ do
    for each $(x, y)$ satisfies $A_{ix} == 1$ and $B_{jy} == 1$ do
        if x!=y then
            $N_{yx} \leftarrow \omega D_y$
        end if
    end for
end for
```

According to the above formulations, the total inbound bandwidth price P_i is derived as follows:

$$P_i = ||N \times \rho||_1 \tag{7}$$

while the total outbound bandwidth price P_o is derived as follows:

$$P_o = ||N^T \times \theta||_1 \tag{8}$$

Also, we derive the message networking volume matrix E as follows:

Algorithm 2 Message Networking Matrix Algorithm

```
Initiate a $s \times s$ all 0 matrix $E$
for each $(i, j)$ in $M_{ij}! = 0$ do
    for each $(x, y)$ satisfies $A_{ix} == 1$ and $A_{jy} == 1$ do
        if x!=y then
            $E_{xy} \leftarrow M_{ij}$
        end if
    end for
end for
```

According to the above formulations, the total message inbound bandwidth price Q_i is derived as follows:

$$Q_i = ||E \times \rho||_1 \tag{9}$$

while the total message outbound bandwidth price Q_o is derived as follows:

$$Q_o = ||E^T \times \theta||_1 \tag{10}$$

Therefore, the overall network price P_n is derived as

$$P_n = P_i + P_o + Q_i + Q_o \tag{11}$$

4) Total Price: Hence, we derive the total price P of the inter-cloud parallel computing system:

$$P = P_c + P_d + P_n \tag{12}$$

F. State-Transition Pricing

When the price changes, the cognitive system needs to adapt to the new optimal solution. However, the transition between two assignment states involves network transmissions. Since the package size of a program code is relatively insignificant comparing to the size of a data chunk, we only consider the networking volume produced by the migration of data chunks. Given the current data chunk assignment matrix B at the total price of P, we assume that with up-to-date pricing status, the proposed system optimizes the system and derive a set of new data chunk assignment matrix B' with new total price P'. The state-transition network matrix T can be derived by the following equations:

$$T = B' - B \tag{13}$$

Here we derive the inbound matrix I, a $d \times s$ logical matrix, by

$$I_{ij} = pos(T_{ij}) = \begin{cases} T_{ij}, & T_{ij} > 0 \\ 0, & otherwise \end{cases} \tag{14}$$

According to the calculation procedure, a non-zero element I_{ij} indicates a network transmission of the ith data chunk to the jth cloud in the state-transition from B to B'.

Similarly, we derive the outbound matrix O, a $d \times s$ logical matrix, by

$$O_{ij} = pos(-T_{ij}) = \begin{cases} -T_{ij}, & -T_{ij} > 0 \\ 0, & otherwise \end{cases} \tag{15}$$

Hence, the total inbound bandwidth price R_i is derived as follows:

$$R_i = ||D^T \cdot I \cdot \rho^T||_1 \tag{16}$$

while the total outbound bandwidth price P_o is derived as follows:

$$R_o = ||D^T \cdot O \cdot \theta^T||_1 \tag{17}$$

Therefore, the state-transition pricing R can be derived by:

$$R = R_i + R_o \tag{18}$$

G. Optimization Target

In the starting stage of the system, the price-aware feature requires the system to seek an optimal combination of program code assignments A and data chunk assignments B, which minimizes the overall cost according to a given prices. Hence, the optimization target is formulated as follows:

$$\begin{array}{ll} \text{Minimize:} & P(A, B) \\ \text{Subject to:} & (1)(2)(3)(4) \end{array} \tag{19}$$

Once the system is launched, to maintain the price-aware performance, the cognitive engine needs to make decisions to adjust the assignments of program codes and data chunks, according to the real-time pricing fluctuation. It is a continuous process that keeps the assignment up-to-date with changing pricing environment. Note that, this procedure is not simply seeking for new optimal solution, but need to take the state-transition cost into consideration. We formulate this optimization target as follows: Assume the pricing policy for all clouds will last for a pricing variety time interval t, thus the system is able to make decisions by predicting the overall cost for current price P and new minimal price P' with the state-transition price R. Hence, the objective function is constructed as:

$$\begin{aligned} \text{Minimize:} \quad & R + P'(A,B)t \\ \text{Subject to:} \quad & R + P'(A,B)t < P(A,B)t \\ & (1)(2)(3)(4) \end{aligned} \tag{20}$$

V. SIMULATION

To validate the performance of our proposed system and the efficiency of optimization approach, we conduct simulations from the perspectives of pricing optimization and cognitive pricing-aware transition.

A. Simulation Setup

TABLE I
DEFAULT SIMULATION PARAMETERS

cloud service provider s	10
code quantity c	8
data set quantity d	8
code-data relationship probability p	0.1
computational requirement C (GB)	$1 \sim 30$
size of data sets D (GB)	$1024 \sim 10240$
size of message exchange M (GB)	$0 \sim 0.1$
data access proportion ω	1×10^{-5}
minimum computing pricing α	$0.0073 \sim 0.0089$
maximum computing pricing α	$0.036 \sim 0.044$
minimum storage pricing β	$0.000032 \sim 0.000040$
maximum storage pricing β	$0.000050 \sim 0.000062$
minimum inbound network pricing ρ	$0.0011 \sim 0.0013$
maximum inbound network pricing ρ	$0.0019 \sim 0.0023$
minimum outbound network pricing θ	$0.11 \sim 0.13$
maximum outbound network pricing θ	$0.19 \sim 0.23$
code and data set security level (Θ, Φ)	$(4, 4)$

This section describes the default settings for our simulations. For the parallel computing software to be deployed over an cognitive inter-cloud system, we specify the quantities of codes and data sets, while initiate their code-data relationship by randomly generating non-zero values for all elements in the matrix L with a probability of p. For the pricing variances from different cloud service providers, we set up random values for minimum and maximum prices for a specific service and simulate the variety of spot instance prices within the intervals. Note that the computing pricing is given by dollars per unit per hour, the storage pricing is given by dollars per gigabyte (GB) per hour, and the network pricing is given by dollars per GB. From existing commercial cloud service pricing policies, the default values for parameters of the simulation are set in Table I. Note that, all random parameters follow uniform distributions.

B. Pricing Optimization

We first simulate the starting stage of the system to demonstrate the efficiency of seeking an optimal solution with the lowest cloud service fee. In addition to our proposed optimization approach, we also derive the cloud service fee for the conventional *Single-Cloud* solution, which selects the cloud with lowest total service fee to host all codes and data sets.

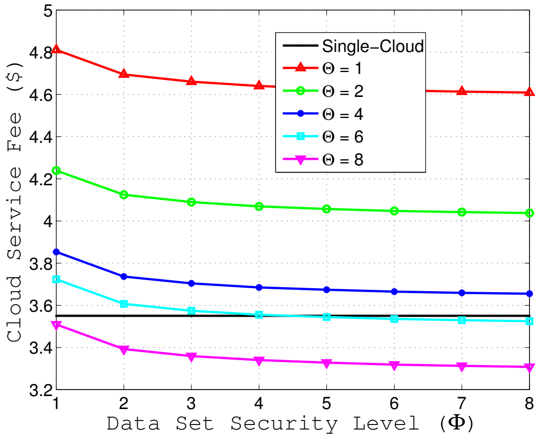

Fig. 3. Tradeoff between Security Level and Cloud Service Fee

Fig. 3 reveals the trade-off between security assurance and pricing optimization. We derive the cloud service fee for various combinations of security levels in codes and data sets. It is obvious that as the security level increases, either in codes Θ or in data sets Φ, the total cloud service fee grows. This requires the user to choose from different security levels based on various requirements. According to our experimental settings, the proposed cognitive optimization can only outperform the *Single-Cloud* system with some combinations of Θ and Φ in terms of cloud service fee, e.g., $\Theta = 6$ and $\Phi > 6$.

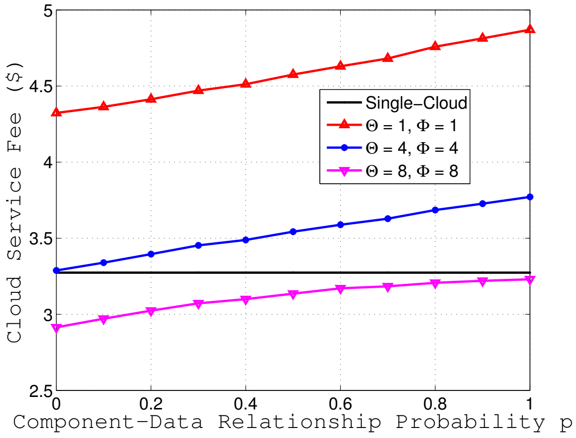

Fig. 4. Effect of Data-Code Relationship on the Cloud Service Fee

One of the most important features of a parallel computing software system is the data-code relationships. Fig. 4 shows its impact on the cloud service fee. We increase the value of p, indicating the probability of code-data relationship, to illustrates its impact on cloud service fee. Note that, along with the growth of p from 0 to 1, the cloud service fees for distinct combinations of Θ and Φ all linearly rise to a higher level. Apparently, these increases are caused by additional inter-cloud data transmissions from data chunks to program codes. Similar comparison on Θ and Φ combinations are also conducted. As depicted, if we select the highest security level with $\Theta = 1$ and $\Phi = 1$, the cloud service fee is higher than that of *Single-Cloud* within the range of $30\% \sim 46\%$. In contrast, the setting of lowest security level with $\Theta = 8$ and $\Phi = 8$ will save the users' cost by $3\% \sim 20\%$.

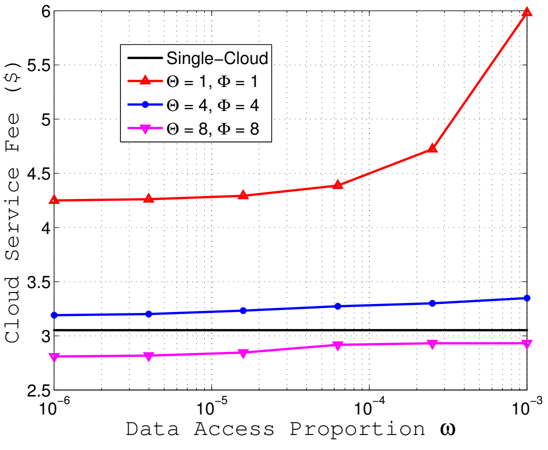

Fig. 5. Effect of Data Access Proportion on the Cloud Service Fee

Another critical feature of the parallel computing software system that impacts the cloud service fee is the data access proportion ω. As shown in Fig. 5, we evaluate the system performance with various values of ω. Apparently, a larger ω yields a higher volume of data transmissions between multiple clouds, which results in a higher cloud service fee. Note that the cost for highest security level with $\Theta = 1$ and $\Phi = 1$ dramatically climbs to 6 dollars, with relatively higher growth rate than other schemes. This is because the highest security level restricts the program codes and data chunks to be completely distributed, and thus the solution space for pricing-aware optimization is much smaller than others.

C. Cognitive Pricing-Aware Transition

After the establishment of the optimal assignment solution, we perform a simulation over time to evaluate the platform's cognitive capacity to the dynamic spot instance prices. With the default security level setting at $\Theta = 4$ and $\Phi = 4$, we compare the performance of three methodologies: *Single-Cloud:* to host all program codes and data chunks in the lowest-cost cloud and never change deployment over time, *Optimal-Static:* to optimize the program codes and data chunks assignment and never change deployment over time, and *Optimal-Cognitive:* to be cognitive to the pricing variety of multiple clouds and keep optimizing deployment strategy over time.

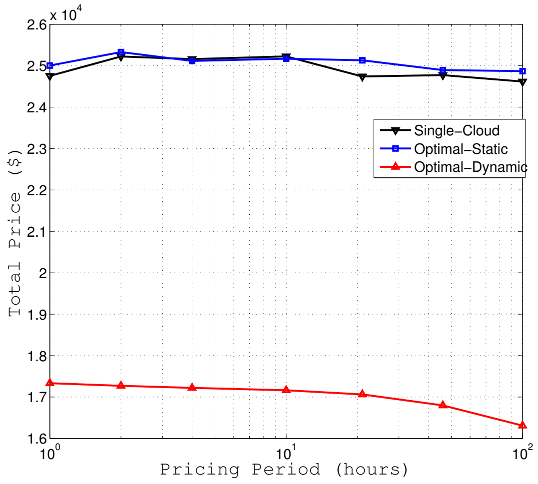

Fig. 6. Efficiency of Proposed Platform with Pricing Variety Time Interval

According to our discussion before, the length of pricing variety time interval will impact the state transition decisions. In our formulation, the optimization target in state transition involves the pricing variety time interval t, which implies that larger value for time interval will make the cost in state transition worthy. Therefore, we expect to see a decrease in total cost, given a longer interval in our simulations. Fig. 6 illustrates the results from our experimental settings with value of t ranging from 1 hour to 100 hours. In fact, in contrary to relatively flat values of total fee from Single-Cloud and Optimal-Static methodologies, we do observe the decline of the Optimal-Cognitive scheme's total cloud service fee, although the degree of elevation decrease is insignificant. In a nutshell, the performance of proposed dynamic optimization is not severely impacted by the frequency of price fluctuation. This phenomenon indicates that, the occurrence of state-transition in our simulations rarely involves inter-cloud transfer of data chunks, which will significantly increase the cost of state-transition.

Given price variety time interval $t = 1$, Fig. 7 illustrates the average instant cloud service fee over 50 hours for 100 random iterations. In fact, in the first hour, *Optimal-Static* and *Optimal-Cognitive* share the same assignment for program codes and data chunks; thus their costs are identical. Afterwards, since the *Single-Cloud* and *Optimal-Static* schemes are not adaptive to the price fluctuations of clouds, their performances become worse as time progresses. In contrast, the *Optimal-Cognitive* scheme dynamically adapts new strategies for different pricing portfolios, thus yields an around 30% reduction on total cloud service fee.

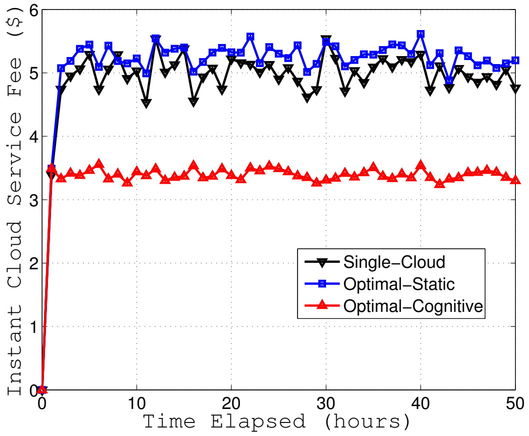

Fig. 7. Efficiency of Proposed Platform with Time Elapsed

VI. CONCLUSION AND FUTURE WORK

In this paper, we have investigated a novel privacy and price-aware inter-cloud system that automatically chooses and migrates user tasks. We have mathematically formulated and derived optimal solutions in both first-stage pricing minimization and cognitive strategy. Preliminary numeric results have revealed the trade-off between privacy and cost. Simulation results also demonstrated the efficiency of the proposed system in reducing total cloud service fee while considering privacy constraints.

ACKNOWLEDGEMENT

This research work is supported by the Canadian Natural Sciences and Engineering Research Council through grant STPGP 447230, a University of British Columbia (UBC) Four Year Doctoral Fellowship and a UBC Work Learn International Undergraduate Research Award.

REFERENCES

[1] 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, pp. 50–58, Apr. 2010.

[2] D. Bernstein, D. Vij, and S. Diamond, "An intercloud cloud computing economy - technology, governance, and market blueprints," in *SRII Global Conference (SRII), 2011 Annual*, pp. 293–299, March 2011.

[3] M. Gall, A. Schneider, and N. Fallenbeck, "An architecture for community clouds using concepts of the intercloud," in *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, pp. 74–81, March 2013.

[4] M. Schnjakin and C. Meinel, "Evaluation of cloud-raid: A secure and reliable storage above the clouds," in *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*, pp. 1–9, July 2013.

[5] M. Schnjakin, D. Korsch, M. Schoenberg, and C. Meinel, "Implementation of a secure and reliable storage above the untrusted clouds," in *Computer Science Education (ICCSE), 2013 8th International Conference on*, pp. 347–353, April 2013.

[6] M. Zhang, R. Ranjan, A. Haller, D. Georgakopoulos, and P. Strazdins, "Investigating decision support techniques for automating cloud service selection," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pp. 759–764, Dec 2012.

[7] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, "Blueprint for the intercloud - protocols and formats for cloud computing interoperability," in *Internet and Web Applications and Services, 2009. ICIW '09. Fourth International Conference on*, pp. 328–336, May 2009.

[8] R. Ranjan and R. Buyya, "Decentralized overlay for federation of enterprise clouds," *CoRR*, vol. abs/0811.2563, 2008.

[9] N. Grozev, R. Buyya, and K. Words, "Inter-cloud architectures and application brokering: Taxonomy and survey."

[10] A. Toosi, R. Calheiros, R. Thulasiram, and R. Buyya, "Resource provisioning policies to increase iaas provider's profit in a federated cloud environment," in *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*, pp. 279–287, Sept 2011.

[11] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," in *Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2010*, pp. 21–23, Springer.

[12] K. Zhang, X. Zhou, Y. Chen, X. Wang, and Y. Ruan, "Sedic: privacy-aware data intensive computing on hybrid clouds," in *Proceedings of the 18th ACM conference on Computer and communications security*, pp. 515–526, ACM, 2011.

[13] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[14] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "Depsky: dependable and secure storage in a cloud-of-clouds," *ACM Transactions on Storage (TOS)*, vol. 9, no. 4, p. 12, 2013.

[15] A. Gohad, N. Narendra, and P. Ramachandran, "Cloud pricing models: A survey and position paper.," in *Cloud Computing in Emerging Markets (CCEM), 2013 IEEE International Conference on*, pp. 1–8, Oct 2013.

[16] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon ec2 spot instance pricing," *ACM Trans. Econ. Comput.*, vol. 1, pp. 16:1–16:20, Sept. 2013.

[17] J. Ekanayake and G. Fox, "High performance parallel computing with clouds and cloud technologies," vol. 34 of *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, pp. 20–38, Springer Berlin Heidelberg, 2010.

[18] I. Giurgiu, O. Riva, D. Juric, I. Krivulev, and G. Alonso, "Calling the cloud: enabling mobile phones as interfaces to cloud applications," in *Proceedings of the ACM/IFIP/USENIX 10th international conference on Middleware*, Middleware'09, (Berlin, Heidelberg), pp. 83–102, 2009.

[19] B. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*, EuroSys '11, (New York, NY, USA), pp. 301–314, 2011.

[20] W. Cai, C. Zhou, V. Leung, and M. Chen, "A cognitive platform for mobile cloud gaming," in *2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom 2013)*, 2013.

[21] T. A. Johnson, R. Eigenmann, and T. N. Vijaykumar, "Min-cut program decomposition for thread-level speculation," in *Proceedings of the ACM SIGPLAN 2004 Conference on Programming Language Design and Implementation*, PLDI '04, (New York, NY, USA), pp. 59–70, ACM, 2004.

[22] D. Watson, J. K. Antonio, H. Siegel, and M. J. Atallah, "Static program decomposition among machines in an simd/spmd heterogeneous environment with non-constant mode switching costs," in *Heterogeneous Computing Workshop, 1994., Proceedings*, pp. 58–65, Apr 1994.

[23] M. Mechtri, D. Zeghlache, E. Zekri, and I. Marshall, "Inter-cloud networking gateway architecture," in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 2, pp. 188–194, Dec 2013.

[24] D. Lange and O. Mitsuru, *Programming and Deploying Java Mobile Agents Aglets*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st ed., 1998.