Research of Promoting the Performance of IaaS with Combined Clouds

To cite this article: Xing Ding et al 2019 J. Phys.: Conf. Ser. 1176 022054

View the article online for updates and enhancements.
Research of Promoting the Performance of IaaS with Combined Clouds

Xing Ding¹, Jing Wu¹*, Hongtao Zhu¹
¹Cet Huayun Information Technology Co., Limited, Shanghai, China

*Corresponding author e-mail: wujing_wj163@163.com

Abstract. The article studies IaaS service providers in combined clouds, including private clouds and public clouds, so as to improve the performance of IaaS services, and considers selective services provided through public clouds. Then, by providing selective services in combined clouds, it involves complicated processing, reliability, and other compatibility topics. The recommended system uses cloud ontology to ensure and rank selective cloud services to develop agent-based systems. Users can obtain the required services in case of failure or failure. This paper also uses transaction technology to ensure the performance of service adoption process. The recommended framework is evaluated by many kinds of experiments. They can prove that it improves the performance of services in combined clouds.

1. Introduction
Combined cloud environments are general frameworks that can use private clouds to provide some services internally, and obtain some services from public clouds. For example, key task services and information may be developed into private clouds; however, the public clouds can set other application. As we know, combined clouds are sometimes vulnerable to various failures. These failures can be attributed to services, intricacy of preferential cloud infrastructure, extensive and widely distributed computing resources.

The article develops a combined cloud framework to ameliorate the performance of IaaS (abbr. of Infrastructure-as-a-Service) services (including space of data storage, CPU, memory, and so on) by considering selective services provided by public clouds. It contains complicated transaction and automatic decision-making to set out what selective services could get provided to cloud users without the required services firstly. Secondly, the services which are demanded can be obtained with dependable method, even in the case of system failure or network communication failure; they are consistent and correct [1].

So as to identify and arrange the selective cloud services that users can obtain when they fail or fail to obtain the required services, we developed a proxy-based system using cloud ontology. Each agent resides in the public cloud and monitors and timely transfers usability and service state variation to proxy system. Via heuristic algorithms, proxy servers rank the available services in the public cloud in the order most in line with user request specifications. Once a service is named a proxy server, it begins to acquire the service. Sometimes we use transaction processing technology to maintain requests from users in service adoption. At the same time, it is necessary to ensure consistency and correctness with the required services. Transaction processing can be applied to a variety of cloud services. But so far they have not been used for services in a composite cloud environment. It should be believed that the use of transaction processing in the adoption of composite cloud services can
theoretically better guarantee the consistency and correctness of applications and services. Although we know that sometimes computing nodes or networks can fail.

2. Related Works

Many researchers around the world have begun to introduce search engines based on proxy cloud services into the management of cloud resources. In fact, SLA and network transit time are often not processed and organized. The author identifies the IaaS characteristics and uses Analytic Hierarchy Process method (AHP) is a multi-level of making important decisions technology to select cloud providers of IaaS. For us, we try to seek for the provider service which most suitably meets whose needs [2]. Although the article use alike aims from the past work, we also pay much attention on system concept and use proxy to discover and select services of IaaS in order to give flexible supplementary in combined cloud computing. The article is not calculated to replace present or classical cloud computing resilience technologies, while it may be found as a supplement. Based on the selective architecture, this work examines the performance and cost of running such applications on different commercial cloud services, such as Google and Amazon AWS. The survey results show that all major cloud service providers adopt different architectures to configure cloud services. Therefore, the cost and property of cloud services still vary widely based on workload. The article proposes an extensible transaction management method based on fast phase isolation. The aim is to complete advanced scalability through separating transaction control functions from the systems of storage and mingling them with applicative level disposes [3]. The article affords a special system that ensures ACID attributes for all of the data and in a cloud system. The article also divides processions from basic data for the past is managed by transaction components. While the late one is by data elements.

3. Framework of Combined Cloud

Figure 1 includes private clouds, proxy servers and many public clouds. People who use the system can access services from private clouds. When private clouds cannot provide the required services, they may be obtained from unpredictably public clouds. Hereby we pay close attention to the IaaS services, including disk space, memory, CPU, etc.

Imagine such an environment which the Freeway Control Department wants to make Cloud Applications to collect large amounts of data about current traffic conditions on the road. It includes sensors, cameras, traffic vehicles, etc. They process these data to assess the impact of traffic congestion, accidents and other events. Consider of the evaluation and analysis of the data, the Freeway Control Department may provide useful information for road operators. Then they could reply in a seasonable mode to make sure that the road is clear. For the purpose of handle so huge data in time, this progress will need many types of IaaS services which includes memory, CPU, disk space etc. The Freeway Control Department can use private clouds to manage these applications. However, when the private cloud can’t give the whole necessary services, the progress doesn’t start. It will finally come to nothing. In this case, the combined cloud can provide a selective supply of services from the public cloud.

Within the recommended framework, a proxy server (in the private cloud) is deployed to provide selective service providers. Broker servers provide various functions that we categorize in the follow-up categories:

**Election and sorting service:** Figure 1 includes different types of public-clouds. They are providing alike selective services. While the quality standard is varies. Therefore, it’s really very significant to identify, sort and choose selective services. The recommended proxy server is established with multi-proxy and cloud system to search, sort and select services.
Figure 1. Combined Cloud Architecture.

**Service Adoption:** For the sake of running Cloud Applications (for example, highway applications), to ensure that the whole demanded services are available in a uniform and dependable method is very important. For cloud applications, acquiring CPU services but not disk space or system memory is unacceptable. Proxy servers use transactions to avoid this situation.

4. **Selection and Ranking Service**
This part describes a proxy-based system for collecting and monitoring the usability and status of services which are given by chosen public cloud service providers of IaaS. That system includes proxy servers and various proxies - that is, each public cloud has a proxy arranged there. These agents send all update information in regard to the services required by users belong to the public cloud to the proxy server. That’s to say, all update services for each provider of cloud are stored in proxy unit. It contributes to handle user demands in real time, which will be described in following chapters. For instance, we give two cases as below [4]:

4.1. **One Public Cloud Provider Services**
Based on the requirements of the user application, the proxy server first searches for all members with a common cloud service and lists the minimum requirements that can meet the needs of all users. In the first list, proxy servers will use heuristic algorithms to search all cloud services individually. Then rank all providers that can provide services that meet this need.

The top providers are those with the best service performance and the largest service accumulation, and the second one will continue to rank according to this strategy. Ideally, all cloud services that meet the needs are provided by one vendor, which minimizes the communication overhead of the system.

4.2. **Other Public Clouds Services**
In terms of that some other customers or who are competing for the same public cloud service, one or more of the top providers may not be able to use the same public cloud service when making proxy servers. Its choice is decided. Then the proxy server will go to the next ranking cloud provider, and so on.

But if a set of cloud service providers cannot provide a service set, the proxy server will search for unavailable request services in the next level of providers. Sometimes the proxy server can't find the matching option at all, so the same process needs to be submitted repeatedly to the next lower program. Until that list in the system runs out of options. When no matching options are found until the end, the proxy server needs to repeat the same process from scratch and view the updated database.

4.3. **Results and Observations**
In Tables 1 and 2, two users are listed and ranked according to the search process described above. Here is a list of the first five public cloud service providers. All cloud providers are able to provide users with the services they need. This includes the SLA protocol guarantees that all public cloud
providers can provide. There are some differences between them. If multiple suppliers have the same cumulative score, we can use SLA protocol to decompose it. If the system has the same lottery score, the random selection method is adopted.

5. Adoption of Service

5.1. Definition
To run the application, users issue requests (or start transactions) to get the necessary services of the combined cloud (2GB disk space, 200GB memory, 4-Core CPU, Ubuntu OS). All of the services maybe required for users to start applications. In case part of resources cannot be obtained from a cloud provider. After that they still may be obtained through selective public cloud provider [5].

The article defines the following rules (attributes) that combined cloud transactions must follow [6].

Semantic atomicity requires combined cloud transactions to be executed as atomic work units. In other words, its processes of component should start totally. Therefore, they can get the services necessary to run user applications. If any necessary resources are not available, that’s all resources are required.

Uniformity: This needs combined cloud transactions to keep consistency of data concerned to cloud resource usability and user demand of application.

Persistence: This needs the impact of whole combined cloud transactions be bound to permanently preserved in their individual information sources to ensure recovery in case of failure.

Elasticity refers to the ability to complete combined cloud transactions despite failures or unavailability of services. Elasticity is done primarily by providing selective services to the public cloud. If we can't get disk space from only one cloud provider, we can get disk space from another one. Because there are many cloud service providers available in the system.

5.2. Transaction Protocol of Combined Cloud
Next, let's go on to show how the previous protocols are implemented in the combined cloud transaction processing. The previous protocol is implemented through cloud cooperator (that is HCC) and subassembly transaction coordinator (that is CTC). HCC means an important element of proxy server. All CTCs are arranged on separate public clouds (Figure 1). The coordination and execution of all cloud transactions related to component transactions is done by HCC. CTC is mainly responsible for coordinating the transactions of various components, such as obtaining disk space and CPU resources. CTC Maintenance Log and HCC work together to complete all records of data needed by the combined cloud firms. [7].

HCC needs to perform combined cloud transactions in order to obtain the resources required by user requests. In the process of completing composite cloud transactions, CTC and HCC communicate many kinds of information. The main steps are as follows:

Step 1. HCC receives new composite cloud transactions and records them in log files. At the same time, HCC sends "start" information to CTC. Component service transactions are officially launched. Then step 2 will be started.

Step 2. CTC also writes down the beginning of component transactions in log files. When it is completed, CTC sends information to HCC according to its own situation, including "success" and "failure". If cloud resources, such as computing resources, needed by component service firms can be obtained at the same time, it will give "success" information. If the cloud resources required by component service firms are unavailable, the information of "failure" is given.

When encountering "failure" information, CTC will record the failure of component service transaction and indicate "failure". Otherwise, CTC will make a mark of "success" and wait for HCC's response.

Step 3. When HCC receives all feedback from CTC, if all information is "successful", that is, all cloud resources are available, HCC will record "all successful" in the log and notify all CTC of "success" message. CTC terminates transactions by submitting decisions and be ready to start processing new transactions.
Table 1. Requirements of User1.

| Num | key | Provider            | service       | CPU (core) | RAM (GB) | DISC (GB) | SLA (%) | SPEND (JPY) | OS                |
|-----|-----|---------------------|---------------|------------|----------|-----------|----------|-------------|--------------------|
| 1   | 3   | Softbank Telecom    | Type_Dual     | 4          | 4        | 200       | 99       | 15650       | WindowsServer2013 Ubuntu CentOS |
|     |     |                     |               |            |          |           |          |             | WindowsServer201    |
| 2   | 3   | NIFTY               | Spec_Type6    | 4          | 4        | 60        | 99.99    | 25310       | Ubuntu RedHatEnterpriseLinux CentOS Ubuntu Debian |
|     |     |                     |               |            |          |           |          |             | RedHat Enterprise Linux WindowsServer2013 CentOS |
| 3   | 2   | GOGRID              | Large         | 4          | 40       | 400       | 100      | 13040       | WindowsServer201    |
|     |     |                     |               |            |          |           |          |             | Ubuntu CentOS       |
| 4   | 2   | NTTCom munications | Plan_v2       | 4          | 8        | 20        | 99.99    | 7550        | WindowsServer201    |
|     |     |                     |               |            |          |           |          |             | Ubuntu CentOS       |
| 5   | 2   | IDC_Frontier        | Type_M2       | 4          | 16       | 65        | 99.99    | 21100       | Ubuntu RedHatEnterpriseLinux CentOS |

Table 2. Requirements of User2.

| Num | key | provider            | service       | CPU (core) | RAM (GB) | DISC (GB) | SLA (%) | SPEND (JPY) | OS                |
|-----|-----|---------------------|---------------|------------|----------|-----------|----------|-------------|--------------------|
| 1   | 4   | GOGRID              | Large         | 4          | 24       | 400       | 100      | 13040       | Ubuntu Debian RedHatEnterpriseLinux WindowsServer2013 CentOS |
|     |     |                     |               |            |          |           |          |             | WindowsServer201    |
| 2   | 2   | Softbank Telecom    | Type_Dual     | 4          | 4        | 200       | 99       | 15650       | Ubuntu CentOS       |
|     |     |                     |               |            |          |           |          |             | WindowsServer201    |
| 3   | 2   | NTTCom munications | Plan_v4       | 4          | 4        | 40        | 99.99    | 15020       | Ubuntu CentOS       |
|     |     |                     |               |            |          |           |          |             | WindowsServer201    |
| 4   | 3   | IDC_Frontier        | Type_S        | 2          | 8        | 65        | 99.99    | 29300       | Ubuntu RedHatEnterpriseLinux CentOS WindowsServer2013 CentOS |
|     |     |                     |               |            |          |           |          |             | WindowsServer201    |
| 5   | 2   | NIFTY               | Spec_Type3    | 2          | 4        | 60        | 99.99    | 18044       | Ubuntu RedHatEnterpriseLinux CentOS |

4. When there are alternative transactions, such as selective services from the public cloud, then MTC can initiate selective component services, and CTC decides the "success" or "failure" of services.
If a transaction is irreplaceable, the HCC records a "global failure" and then notifies all CTCs in the system. If this happens, the entire composite cloud transaction means failure and the cloud assets may not be retrieved.

5. When receives the global signal, CTC will record the "global success" in the component service transaction. If it succeeds locally, CTC receives the "global failure" signal via HCC. After that, CTC need to stop its service. That condition mainly occurs when a subassembly service transaction is only partially satisfied when accessing cloud resources, while other resources are not satisfied. At this point, all the resources it acquires must be cancelled. This can generally be constrained by the aforementioned Semantic Atomicity Rules.

6. Empirical Evaluation

An assessment of the recommended framework for failure adaptive capacity and correspondence costs is presented here. The purpose of the recommended framework is the improvement of the adaptive capacity of system services through selective public. Hence, assess the side of the system's adaptive capacity. This resilience can result in delays in network communication resulting in performance delays. As a result, the public cloud selective service announcement produced an elevated assessment [8].

6.1. Failure Resiliency

According to the evaluation criteria of fault recovery capability, the model is based on possibility principle. To imitate the accomplishment /fault rate of combined cloud transactions, we designed probability sets as below:

Private Cloud Completion (PRCA): This concept is used to describe the probability of success in obtaining services from private cloud services.

Private Cloud Failure (PRCF): This indicator is used to describe the probability of failure to obtain services and resources from private clouds. Failure means that private clouds cannot satisfy users' minimum requirements. At this time, the user's needs can be sent to the proxy server.

AltSA: This indicator is used to describe the probability of getting services from the public cloud. This means the possibility of providing services to users through the public cloud. Users still have the opportunity to get the services they want from the public cloud. In this article, we still use heuristic method to sort all public cloud service providers.

Public Cloud Completion (PUCA): The concept is used to describe the probability of success in obtaining services from public cloud services.

Public Cloud Failure (PUCF): This indicator is used to describe the probability of failure to obtain services and resources from private clouds.

Total Success Ratio (TSR): This concept is used to describe the overall likelihood of users getting services and resources from both public and private clouds.

Total Failure Ratio (TFR): Based on previous concepts, the TSR (total success ratio) and the combined cloud total failure ratio (TFR) transactions calculated by multiple mathematical expressions are designed in this paper. TSR and TFR were calculated by probability PRCS, PRCF, AltSA, PUCS and PUCF.

We design a variety of selective cloud services from the public cloud. If the private cloud cannot provide the required storage, it can be obtained from the public cloud.

Table 3. Totally Accomplishment and Fail Rates.

|          | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 |
|----------|--------|--------|--------|--------|--------|
| PRCS     | 0.89   | 0.81   | 0.69   | 0.59   | 0.49   |
| PRCF     | 0.09   | 0.21   | 0.29   | 0.39   | 0.49   |
| TFR1     | 0.027  | 0.071  | 0.131  | 0.207  | 0.29   |
| TFR2     | 0.0207 | 0.0463 | 0.0815 | 0.1311 | 0.19   |
| TSR1     | 0.971  | 0.927  | 0.867  | 0.791  | 0.69   |
| TSR2     | 0.9791 | 0.9535 | 0.9183 | 0.8687 | 0.79   |
Table 3 illustrates our design to assess the elasticity of the recommended system. We imagine that the possibility of selective service availability (AltSA) is 0.8 - indicating that has a 80% possibility of obtaining selective services through the public cloud. Actually, the diverse result of PRCS and PRCF possibility are still adopted with AltSA. Consider all conditions. For instance, we may meet high possibility of failure and cases with high possibility of success.

The TSFR of combined cloud transactions have been figured up in 3rd Tab. It is described as picture in Fig. 2 and Fig. 3. The figure distinctly shows that the recommended system improves the flexibility of service provision. Which is accomplished by successfully implementation combined cloud transactions and acquiring clouds. Such as, when public clouds (TSR1 and TSR2) provide services, combined cloud transactions in Figure 2 have a better success rate. In like manner, Figure 3 means TFR1 and TFR2 have worse failure rates than PRCF if the public cloud services are adoptable.

6.2. Processing and Communication Delays
Increased fault recovery capability may lead to attenuation of system function. When some services are obtained through the public cloud, communication overhead (latency) may occur. They can do harm to system application. Hereby we evaluate such total by considering some key parameters. For example, the network transferring cost, the amount of information communicated among combined cloud coordinators and many other parts transaction cooperator, the transaction time for each parameter for parts transactions that obtain requirements from the public cloud. Total subassembly transactions number that obtain services through the public cloud. The general time to handle the mixed cloud transaction $T_{proc}$ is calculated as below:

$$T_{proc} = \sum_{i=0}^{n} CTPR_i + \sum_{j=0}^{m} CTPUB_j + NET_d$$

$T_{proc}$ represents the sum of all processing times of $CTPR_i$ resources from private clouds, $CTPUB_i$ resources from public clouds, network communication delay ($NET_d$) of the system, and so on. The processing time of $CTPUB_i$ includes the sum of all the time required to obtain cloud services such as disk storage and memory.

Figure 2. Success Rate of Combined Cloud Transactions.

Figure 3. Combined Cloud Transactions Failure rate.
NET_d presents the system delay which can be estimated as below:

\[
NET_d = \left( M_{pr} \times CD_{pr} \times N_{pr} \right) + \left( M_{pub} \times CD_{pub} \times N_{pub} \right) \times P_{pub}
\]  

(2)

M_{pr} and M_{pub} consist of information transferred among the combined cloud framework. So they can individually obtain services through both of the private cloud(s) and public one(s). CD_{pr} and CD_{pub} present the information costs for both of private clouds and public clouds. N_{pr} and N_{pub} describe the amount of component transactions individually getting services through them. P_{pub} presents the possibility. Many cloud services could be get through public cloud. When all of the applications be caught through a private cloud, P_{pub}= 0. At that time, there will not be extra communication acquired services from the public clouds. When all service is obtained through the public cloud, P_{pub}= 1.

The general time cost was simulated to deal with combined cloud transaction. T_{proc} is by bringing into four diverse experiments statistics. Some users request is considered. A combined cloud transaction needs getting the same number of services through the component transactions.

Figure 4 shows the general time of the corresponding situations. Figure 1 shows how the sum services are retrieved through the private cloud, so the general handling cost T_{proc} is lower compared to others. The 4th case shows a sharp increase in processing time as opposed to all services obtained from the public cloud. 2nd case shows that about 50% services may be obtained through private clouds and next 50% through public clouds. 3rd case presents only 3/4 services can be obtained through the public cloud and the left through the private one.

7. Conclusion
The article designs a combined cloud framework to better the performance of laaS services through the collaboration of agents, ontologies and transactions. The framework uses proxies and ontologies to collect the present condition of laaS services through each user's public cloud. Service data states are merged on proxy servers, which can use heuristic selection algorithms to provide the 1st schedule of public cloud providers that corresponding with needs from private clouds at worktime. Transaction technology is afterwards adopted to make sure the reliability and uniformity of the service adoption transactions. Cloud users and providers can use the recommended framework to enhance failover capabilities and communication delays when accessing services through private clouds and public ones by considering key factors ranking of selecting selective cloud services through public clouds and intelligent decision-making for service consumption/configuration in combined clouds.

References
[1] Ghafouri Robabeh, Movaghar, A, Mohsenzadeh, Mehran. “Time-Cost Efficient Scheduling Algorithms for Executing Workflow in Infrastructure as a Service Clouds”, WIRELESS PERSONAL COMMUNICATIONS, vol 9(4), pp. 2035-2070(2018).
[2] Muhammad-Bello, Bilkisu Larai, Aritsugi, Masayoshi. “A Robust Algorithm for Deadline Constrained Scheduling in laaS Cloud Environment”. IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS, vol E101D(12), pp. 2942-2957(2018).
[3] Mandhi, Tarek, Mezni, Haithem. “A prediction-Based VM consolidation approach in IaaS Cloud Data Centers”. JOURNAL OF SYSTEMS AND SOFTWARE, vol 146, pp. 263-285 (2018).

[4] Androcec, D., Vrcek, N., Seva, J. “Cloud Computing ontologies: a systematic overview”. Proc. of the 3rd Int. Conf. on Models and Ontology-based Design of Protocols, Architectures and Services, pp. 9–14 (2017).

[5] Levandoski, J.J., Lomet, D.B., Mokbel, M.F., Zhao, K. “Deuteronomy: transaction support for cloud data”. 5th Biennial Conference on Innovative Data Systems Research, January 9-12, Asilomar, California, USA, pp. 123–133 (2014).

[6] Xue, Yuanzheng, Jin, Shunfu, Wang, Xiushuang. “A Task Scheduling Strategy in Cloud Computing with Service Differentiation”. KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, vol 12(11), pp. 5269-5286 (2018).

[7] Mohamed, Mohamed, Belaid, Djamel. “Extending OCCI for autonomic management in the cloud”. JOURNAL OF SYSTEMS AND SOFTWARE, vol 122 pp. 416-429 (2016).