Deadline Oriented Resource Broker for Cloud Computing

LI Tao，CHEN Weiwei，LI Zhigang, LIU Zhao
College of Command Information System, PLA Univ. of Sci. & Tech., Nanjing 210007, China
Email:lee_xiamu@163.com

Abstract

Cloud broker can provide more flexible price options by integrating and scheduling all the requirements of their clients. The existing work does not consider the deadline of the requirement, which may fail to satisfy the clients before the deadline time. Therefore, we propose a novel strategy for the cloud broker to minimize overall cost and optimize the resource scheduling when considering the deadline constraints. The simulation shows the cost can reduce one third for Aliyun by our algorithms.

Keywords—SLA; Reserved Instance; Resource Allocation

I. Introduction

Cloud computing [1],[2],[3],[4] has emerged as a technology becoming quite popular among academic and industry. Most IaaS CSPs will offer their users two options of computing service: reservation and on-demand option. Reservation option allows the user to prepay a one-time reservation fee and then to reserve a computing instance for a long period (usually months or years). On-demand instances are economically inefficient to users, owing to the higher rates.

Most users who choose the reserved instances can’t fully use the instance, so these instances have a number of spare time. Therefore we hope to integrate as much as possible users’ on-demand requests and exploit time-division multiplexing (TDM) [5] to run these on-demand requests in reservation instances’ spare time. In this way, a cloud user can enjoy cost savings due to reservation option, while avoiding its inefficiency usage.

II. The Instance Reservation for deadline constrained jobs

1) Smoothing user requests to reduce number of reserved instances

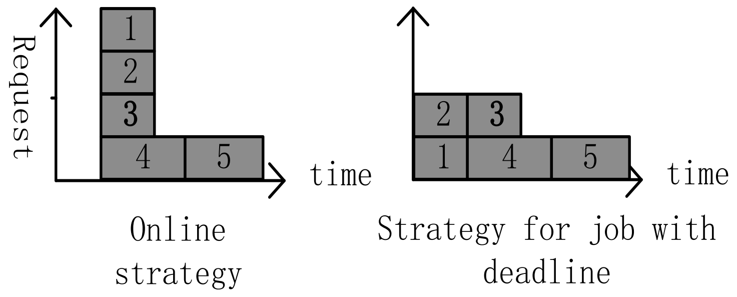


Figure 2 Smooth the request and reduce the number of reserved instances

Online reserved [6] strategy purchase on-demand instance to satisfy the burst demand which beyond their expectation. However, as shown in figure 2, if the demand number of the requests is vast and burst is very transient. At this moment, CSP need large number of server and high quality bandwidth to deal with the requests, bear huge computing and networking pressure, and pay large fund on the datacenter's building for reliability and availability. However, it's a vast waste for CSP. Though our strategy, the burst demand will be smoothed and dispersed, CSP's datacenter will keep in steady state with no pressure.

2) Exploiting reservation instances sufficiently

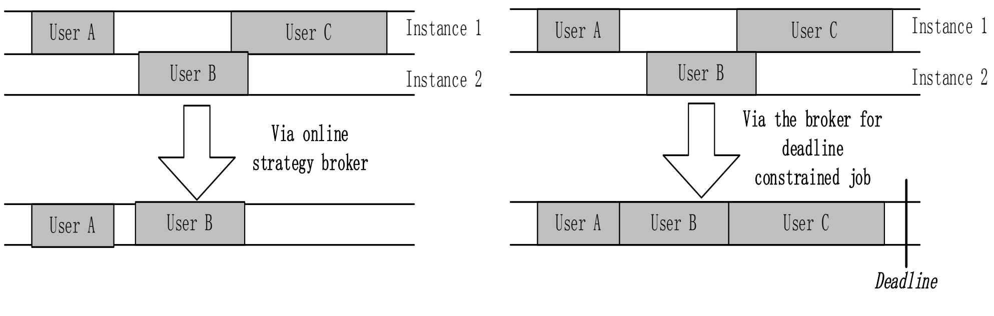


Figure 3 Oriented deadline broker let a reserved instance "full" of use

As shown in figure 3, user B request and user C request has overlap working time, if cloud broker adopting the working model of the online strategy, they can’t be conducted time-division multiplexing on one reserved instance. However, via deadline oriented broker, user B request could be translated before deadline. The deadline oriented broker put the request B ahead of time. Therefore, user B and user C request have no overlap working time. In this way, we can realize the "seamless connection" for on-demand request in the reserved instances, this will let a reserved instance "full" of use, at the same time also can reduce the cost of the user

III. Strategy for the problem

There is a close relationship between reservation instance cycle and the number of reserved instances. In order to facilitate research, we proposed to divide a long term reserved strategy into short period reserved strategy for research. The relationship between reservation strategy for a short-term period and long-term period. In reality, user put forward their demand tuple (r_i, dt_i, D_i), where r_i is the resource requirement of job i, dt_i is the duration time and D_i is job’s finish deadline. Our strategy can save cost efficiently, exploit the reservation option better, and smooth the request burst in our strategy.

The following we focus on Deadline oriented short-term reservation strategy. Assuming that all service requests arrive in zero moment, demand of instance using time is stochastic. According to the users’ deadline, this paper analyze two situation:

A. **All users’ deadline are set in the end of the reservation cycle**

All users’ deadline are in the end of the reservation cycle, namely, $D_i = \tau, i \in I$. Owing to the characteristic that demand requests for instance is vast, and average duration of instance relative to reservation cycle is of short duration. We assume every reservation will be filled by these requests. Our method is fulfill the reservation one by one. At first, we will let total demand $\sum dt_i$ of all users divide the reservation cycle.

1) If $\sum dt_i$ can be divided exactly by reservation cycle, users directly reserve $N = \sum dt_i / \tau$ instances. The total cost for the user:

$$\text{cost} = \frac{\sum dt_i}{\tau} \cdot \gamma$$

2) If $\sum dt_i$ cannot be divided exactly by reservation cycle, set $\mu = \gamma / \rho$. For users' requests, set the actual use time is δ, we set $\delta = (\sum U_i \ mod \ \tau)$. If $\delta \geq \mu$, the reservation option for user is more appropriate, if $\delta < \mu$, the on-demand option is more appropriate. In the situation that deadline is τ. If $\delta \geq \mu$, the broker only apply for reserved instances. Otherwise, the broker apply for reserved instances and a demand instance for surplus requests δ . The total cost for user:

$$cost = \left\lfloor \frac{\sum dt_i}{\tau} \right\rfloor \cdot \gamma + \delta \cdot p$$

B. **All users' deadline are different in the reservation cycle**

First we study a simple example, assuming $\tau = 2$. In figure 6, in scene 1,2,3, all request arrive at zero. Every situation has one instance that their deadline is one, and three instances that their deadline is two.

It is easy to find in figure 5, regardless of the applying number of reserved instances, the cost of scene 1 are not higher than scene 2 and scene 3. We already found smoothing the users' request, can maximum utilization of reserved resources to reduce costs. However, scene 4 shows when the Deadline is due, we can't reduce the demand peak. The only way we can smoothing the demand peak is to translate instance before its deadline. Though the simple example, we set two laws:

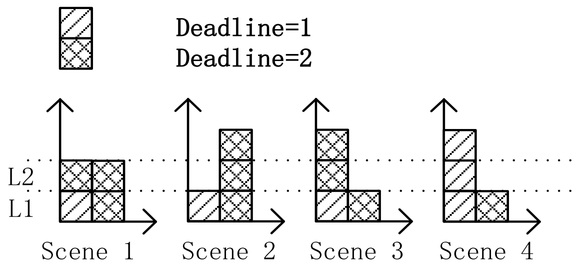


Figure 5 A simple example

1) the smoothing requests will cost lower

2) every job must be done before its deadline

In order to meet every job get over before its deadline in a real-time system, the earliest-deadline-first (EDF) [7] scheduling policy is used. On this basis, we proposed EDF translating algorithm:

EDF translating algorithm

```
Input: $(r_i, dt_i, D_i)$, $i \in N$
Output: $(n_1, n_2, n_3, \cdots, n_\tau)$
```

Procedure EDF translating

```
Sort all the requests by deadline from early to late
For t=1:$\tau$
    $d_t$=0;      //initialization
For i=1:N
    For j=1:$D_i - dt_i$
        Find the time t when has the lowest $d_i$;
    Boot the request i in time t;
    For k=1:$dt_i$
        $d_{t+k-1}$++;
For ($i = 1; i \leq \tau; i = i + 1$)
    $n_i$=$d_i$;
```

n_i can be get through EDF translating algorithm, then we can calculate the number of reserved instance and the total cost where we denote by $X^+ := \max\{0, X\}$.The Deadline of different cases, we analysis the minimum cost formula is:

$$\text{cost} = \min(k \cdot \gamma + \sum_{i=0}^{\tau} (n_i - k)^+ \cdot \rho)$$

IV. Experiment evaluation

Through simulation using the cloud broker we proposed can greatly reduce the cost of the user.

Taking the instance price of Aliyun cloud as example, the common configuration is in table 2. In the simulation we choose the instance price in Beijing area which is configured for single core, 512 m memory, 1 MBPS bandwidth as a reference, as shown in table 2,3.

Table 2 Configuration Of the Instance（part）

	core	Memory(M)	Bandwidth(Mbps)
Configuration 1	1	512	1
Configuration 2	2	1	1
Configuration 3	4	1	1

Table 3 Price Of The Instance In Beijing

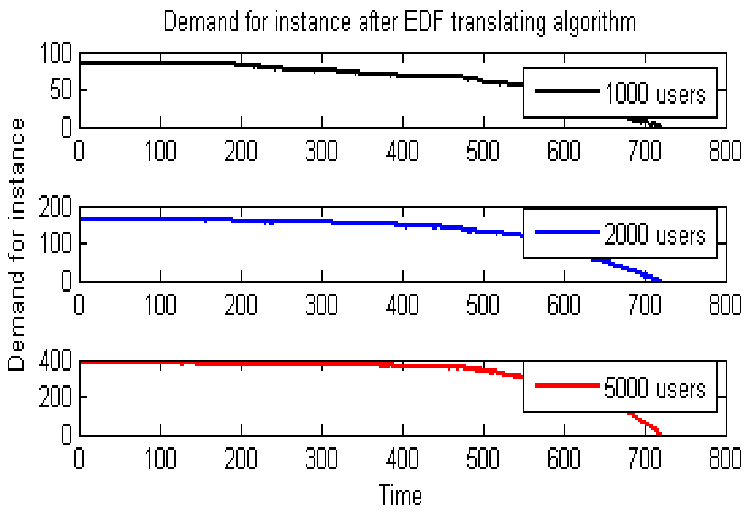
	On-demand instance (￥/hour)	Reserved instance (￥/month)
Configuration 1	0.2	55
Configuration 2	0.88	271
Configuration 3	1.76	519

In [8], the data indicates, a quarter of customers use only a few minutes to several hours, 45% of the customers use long time about half a day, 15% of customers use time about a day to a few weeks, the rest of the users to use time more than a few weeks. , $\mu = \gamma/\rho = 275$, if the use time more than 275,customers can buy reserved instances individually, without the need to reduce costs through the cloud broker, so, this paper focuses on the customers that using time is less than 275 hours. As shown in table 4.

Table 4 The Duration Of Users' Demand

Request demand (hours)	(1,2)	(2,24)	(24,120)	(120,275)
proportion	25%	45%	15%	15%

The cloud service cycle of Aliyun reserved instances is $24 \cdot 30 = 720$ hours, the user requests randomly distributed in the reserved cycle, assuming each request has a random Deadline. In figure 6, it's easy to find that level of the instance in every time declined obviously with the EDF translating algorithm.



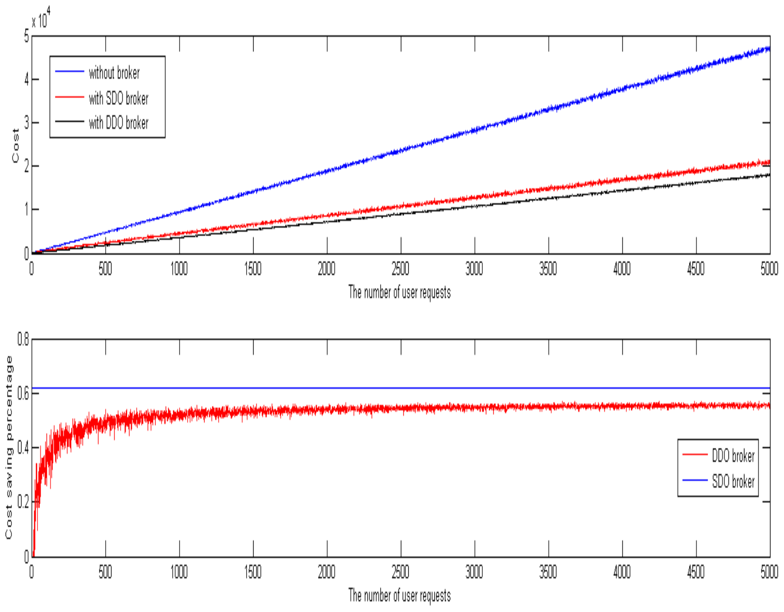


Figure 6 Demand for instance after EDF translating algorithm（a）
and Cost and cost saving percentage(b)

We simulated respectively 1000, 2000, 5000 requests. Data according to the figure 7 shows that using this cloud broker service can significantly lower total cost than the users without using cloud broker service. The reserved strategy proposed in this paper work better than online reserved strategy. The disparity between the service with same deadline and different deadline is not big when the requests are more than 1000. But the users generally have obviously demand for own Deadline, so, using different Deadline cloud broker service is more close to the practice and have better application prospect.

we can easily see that the proportion of the total cost with same deadline broker (SDB) reduction do not change with different number of users, but basically stable at around 62%, so it can be applied to different scale of users with stable cost saving. On the other hand, the proportion of different deadline broker (DDB) reduce cost is increasing until the cost saving percentage close to 57%. As we can see, both SDB and DDB can greatly reduce the users' cost.

Reference

[1] Akilandeswari, J. and Sushanth, C., 2014. A Review of Literature on Cloud Brokerage Services. International Journal of Computer Science and Business Informatics, Vol. 10, No. 1, pp. 25-40.

[2] Armbrust, Michael, et al. "A view of cloud computing." Communications of the ACM 53.4 (2010): 50-58.

[3] Ostermann, Simon, et al. "A performance analysis of EC2 cloud computing services for scientific computing." Cloud Computing. Springer Berlin Heidelberg, 2010. 115-131. http://azure.microsoft.com/zh-cn/

[4] Tsai W T, Sun X, Balasooriya J. Service-oriented cloud computing architecture[C]//Information Technology: New Generations (ITNG), 2010 Seventh International Conference on. IEEE, 2010: 684-689.

[5] Nestle, Elliot, and Robert F. Schunneman. "TIME DIVISION MULTIPLEXING." U.S. Patent No. 3,599,160. 10 Aug. 1971.

[6] Wang W, Niu D, Li B, et al. Dynamic cloud resource reservation via cloud brokerage[C]//Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on. IEEE, 2013: 400-409.

[7] Stankovic, John A., ed. Deadline scheduling for real-time systems: EDF and related algorithms. Springer, 1998.

[8] Bazarbayev S, Hiltunen M, Joshi K, et al. Content-Based Scheduling of Virtual Machines (VMs) in the Cloud[C]//Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on. IEEE, 2013: 93-101.