A Decomposition-based Architecture for Distributed Cloud Computing

Chuan Pham, Cuong T.Do, Kyithar, and Choong Seon Hong
Department of Computer Engineering, Kyung Hee University
E-mail: {pchuan, dtcuong, kyithar, cshong}@khu.ac.kr

Abstract: Almost all of cloud services nowadays are built atop geographically distributed infrastructure. In this paper, we consider the problem of placing virtual machines (VMs) to host several independent applications on a shared resource pool based on cloud platforms. To improve the quality of services and optimize the social welfare, we propose a distributed algorithm based on the dual decomposition method. The solution is coordinated from multi data centers to find an optimal solution. The efficiency of our proposed model is demonstrated on the simulation environment.

1. Introduction

Cloud computing is becoming a rising paradigm in the information and communications technology industry today. Many cloud services are deployed on multiple data centers that are located in different region for improving performance and reliability. The term distributed systems (such as distributed computing, distributed databases, etc.) generally refer to that scenario. And, Fig.1 presents the architecture of distributed cloud computing.

There are many advantages in distributed cloud system. Firstly, today the number of requests from users in cloud services increases significantly. The distributed clouds not only mitigate the suffering from massive connection but also improve the load balancing and fault tolerant in cloud services. Secondly, the client requests come from the wide area. They must be directed to an appropriate datacenter, which is closest to them to reduce the latency and bandwidth on the network. In distributed system, a big problem can be divided into many tasks, each of which is solved by one or a pool of sharing resource, which is communicated autonomously and easily by the virtualization technique in cloud computing. Finally, each datacenter has different revenue from others at instant of time. Since, with different regions, each datacenter has difference amount of available resource, the power consumption, etc.

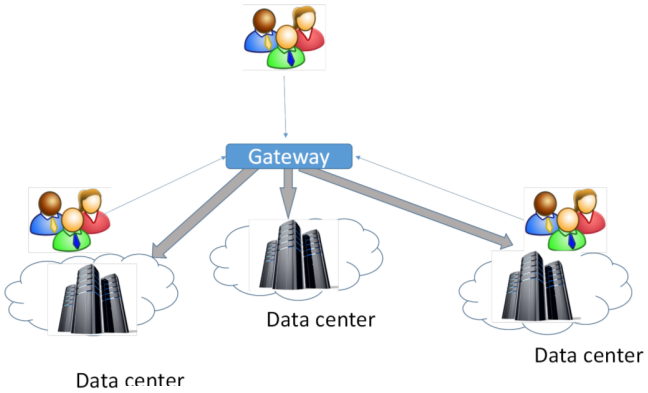

Fig.1. Distributed cloud architecture

Distributed cloud solutions are very useful to cloud providers to enable virtual network services that span a wide geographical area. It supports more securely. Because all data are not in the same place. It will not be damageable too much in any disaster and will easily recover in small amount of time. This technique also does not need a large resource pool such as network capacity, CPU, memory, and storage at datacenters. The cloud service is spread out everywhere and will be more likely to be close from where users are accessing it. Furthermore, with small capacity, distributed cloud computing requires less electricity almost cuts off power consumption from air conditioning.

However, today, geographical load balancing and energy cost are managed independently. It causes poor performance and high costs in many cases [1]. In addition, many papers in VM placement problem do not focus on distributed cloud computing [2, 3]. Minghong Lin et al [3] proposed a method that dynamically scale the number of active servers in datacenter depending on workloads. This method is a good way to reduce the power consumption in datacenter. But they did not propose for the distributed cloud.

In this paper, we propose a framework that joints load balancing and minimizing the power consumption. By using the theory of dual decomposition method, we divide the master problem into multiple sub-problems, which can easily solve at each datacenter.

The rest of the paper is organized as follows. The system model and problem formulation is represented in Section II. In Section III, we solve the optimal problem by decomposition method. In Section IV, we describe the experiment environment and simulation results. Finally, we conclude our work in Section V.

2. System model and problem formulation

In this section, we show an architecture of our method that illustrates how to partial execution into each datacenter and how to control provisioning resource in distributed datacenters.

Each datacenter consists of a set of active physical machines (PMs). Each PM can host multiple VMs through a hypervisor. To reduce the power consumption, the number of active servers at each datacenter can be changed depending on workloads. We define $X = (x_1, x_2, \ldots, x_j, \ldots, x_N)$ is a vector active servers, which x_j is the number of active servers at datacenter j.

We consider a discrete time model, where in each slot t the cloud service provider receives amount of tasks $A = (a_1, a_2, \ldots, a_i, \ldots, a_M)$. Each task a_i is associated with specific performance goals specified in a service-level agreement (SLA) contract. A task a_i can embed an arbitrary application topology, which requires a list of VMs, which is called a cluster. To reduce network transfer, a cluster is often hosted at one datacenter.

To model the provision resource in cloud, we consider each VM belong to class k. For example, Amazon EC2 defines a list of instants that clients can choose properly to their application. We define that N_i is the VM allocation vector of task $N_i = (n_{i1}, n_{i2}, \ldots, n_{ik}, \ldots, n_{ic})$ where n_{ik} is the number of VMs of class k attributed to task a_i, and c is maximum number of class. Each task a_i requires amount of resource (such as: memory, CPU unit, storage, etc.) that can be calculated based on vector N_i as follows:

$$R_i^{mem} = \sum_k^c n_{ik} * Ins_k^{mem}$$
$$R_i^{CPU} = \sum_k^c n_{ik} * Ins_k^{CPU} \quad (1)$$

where Ins_k^{mem} and Ins_k^{CPU} are attributes of instance type k.

To host application a_i, the controller or gateway must distribute this task to one datacenter that has available resource greater than the total requirement resource of a_i.

While improving the quality of services (QoS), we consider that the allocation of datacenters and demand assignment must satisfy a set of constraints, such as: demand constraint and SLA performance constraint. We define σ_j is the demand arrival rate from the gateway assigned to datacenter j, the demand constraint ensures that all demands are satisfied:

$$\sum_j^N \sigma_j = D \quad (2)$$

where D is the average demand arrival rate originated from the gateway. In addition, we define d_j as the network latency between the gateway and datacenter j. At datacenter j, we assume that the arriving demand σ_j is equally split among x_j active servers. From the queueing model M/M/1 [5], the queueing delay between the gateway to a server can be computed as:

$$L_j = \frac{1}{\mu - \frac{\sigma_j}{x_j}} \quad (3)$$

where μ is the service rate of each server. The constraint of delay is defined as:

$$\frac{1}{\mu - \frac{\sigma_j}{x_j}} + d_j \le d_j^{\max} \quad (4)$$

Where d_j^{max} is the maximum delay that the service provider tries to achieve.

By defining the constant $\alpha_j = \mu - \frac{1}{d_j^{\max} - d_j}$ (5)

we can rewrite the constrain (4) as: $x \ge \frac{\sigma_j}{\alpha_j}$ (6)

We aim at finding the VM allocation scheme for each task a_j while maximizing a global utility value U, which is expressed as a weighted sum of the application-provided resource-level utility functions and operating cost function [2]. The resource-level utility function ui for task ai is defined as $u_i = f_i(a_i)$. To capture the reconfiguration cost, we use the function $g(x_i)$, which depends on the number of active servers on each data centers. The distributed cloud model can be formulated as:

$$\max U = \sum_j^N \sum_i^M H_{ij} \omega_i u_i - \sum_j^N \varepsilon * g(x_j))$$
$$s.t \sum_i^M H_{ij} R_i^s \le C_j^s(x_j)$$
$$\sum_j^N H_{ij} = 1$$
$$x_j \ge \frac{\sigma_j}{\alpha_j}$$
$$\sum_i^M \sum_j^N H_{ij} c_{ij} \le B$$
$$H_{ij} = [0,1]$$

The matrix mapping H_{ij} shows the result that task i is directed to datacenter j. To simplify the constrain (1), we use R_i^s to define the total resource type s in task i and $C_j^s(x_j)$ is the function capacity of datacenter j. In this case, we consider that the capacity of each datacenter depends on the number of active servers that is formulate by the function $C_j^s(x_j)$.

Constraint (9) is to require each task should be hosted in one datacenter. Constraint (11) captures the cost constraint that the total cost of serving all the requests should not exceed the given budget B. The constraints from (8-10) could be solved separately from each datacenter, and the constraint (11) is called a coupling constraint, as it couples many datacenters together by the budget. Based on the dual decomposition method, we relax the constraint (12) and decompose this problem into many sub-problem, which can solve at each datacenter.

3. Distributed cloud computing embedding decomposition architecture

The idea of decomposition problem above is to separate the master problem into many sub-problem, in which each datacenter tries to optimal the resource allocation independently. The gateway will collect the information from all datacenters and control resource price for all the sub-problems.

At each datacenter, the sub-problem is as follows:

$$\underset{H_{ij},x_j}{Max}\ u_j = \sum_{i}^{M} H_{ij}\omega_i u_i - \varepsilon * \cos t(x_j) \quad (13)$$

$$s.t \quad \sum_{i}^{M} H_{ij} R_i^s \le C_j^s(x_j) \quad (14)$$

$$x_j \ge \frac{\sigma_j}{\alpha_j} \quad (15)$$

By solving this problem at each datacenter, we obtain the mapping matrix and the number of active servers. The gateway will collect these values to solve the primal problem [6].

In general, a gateway can instantiate a set of policies at the master problem, after receiving requests, dictating the order in which the variables need to be optimized and obtain the mapping matrix *H* and vector *X*. The gateway receives all information from datacenters and updates prices (based on the formula in [6]) to calculate for the next step. This process runs until convergence.

4. Simulation results

In this section, our goal is in two-folds: First, we implement our model through a simulator and evaluate efficient of our system in random data. Second, we seek to illustrate the saving of cost and power consumption.

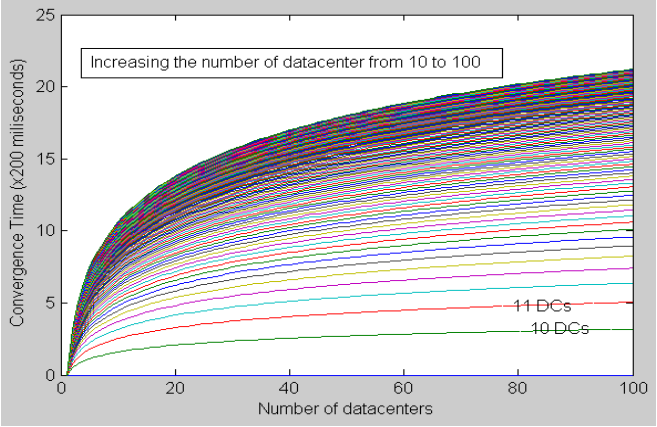

Fig.2. Evaluation of convergence

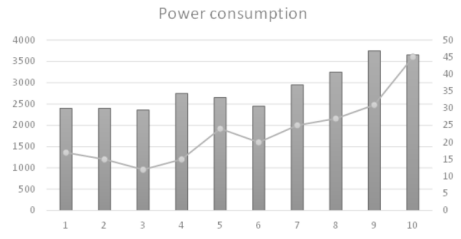

Fig.3. Evaluating power consumption

We consider a scenario that the system has maximum 100 active servers and create randomly 100 to 500 VMs from three instances types, which are referred in [7]. With an average request arrives at 10 minutes, we consider that our system can easily get convergence point that shows in the Fig. 2.

The Fig.3 demonstrates the efficiency of reduce power consumption. By optimal the number of active servers at each datacenters, we can reduce amount of carbon footprint when having low workload.

5. Conclusion

In this paper, we propose a decomposition framework that can apply in large scale geographically distributed cloud computing. Our method can maximize the total revenue and reduce the power consumption for servers in data centers. Especially, our work separates computations into many datacenters to obtain the optimal solution. This considering helps us not only to minimize the power consumption, but also to maximum the utilization on each datacenter. The analysis and simulation show that our proposed system is adaptable and can be applied distributed data centers.

Acknowledgment

This work was supported by the ICT R&D program of MSIP/IITP, Republic of Korea. (2014-044-011-003, Open control based on distributed mobile core network) *Dr. CS Hong is the corresponding author

References

[1] M. Hajibaba and S. Gorgin, “A review on modern distributed computing paradigms: Cloud computing, jungle computing and fog computing,” *CIT. Journal of Computing and Information Technology*, vol. 22, no. 2, pp. 69–84, 2014.

[2] H. Nguyen Van, F. Dang Tran, and J.-M. Menaud, “Autonomic virtual resource management for service hosting platforms,” in *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*. IEEE Computer Society, 2009, pp. 1–8.

[3] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, “Dynamic rightsizing for power-proportional data centers,” *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 5, pp. 1378–1391, 2013.

[3] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, “Dynamic rightsizing for power-proportional data centers,” *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 5, pp. 1378–1391, 2013.

[4] P. Jakovits, S. N. Srirama, and I. Kromonov, “Stratus: A distributed computing framework for scientific simulations on the cloud,” in *High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS), 2012 IEEE 14th International Conference on*. IEEE, 2012, pp. 1053–1059.

[5] S. M. Ross, *Introduction to probability models*. Access Online via Elsevier, 2006.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2004.

[7] [Online]. Available: http://aws.amazon.com/ec2/pricing/