Optimal Dataset Allocation in Distributed Heterogeneous Clouds

Min Sang Yoon, Ahmed E. Kamal
Department of Electrical and Computer Engineering
Iowa State University, IA, 50010, USA
Email : {my222, kamal}@ iastate.edu

Abstract— **We consider optimal datasets allocation in distributed Cloud Computing systems. Our objective is to minimize processing time and cost. Processing time includes virtual machine processing time, communication time, and data transfer time. In distributed Cloud systems, communication time and data transfer time are important component of processing time because data centers are distributed geographically. If we place datasets far from each other, this increases the communication and data transfer time. The cost objective includes virtual machine cost, communication cost, and data transfer cost. Cloud service providers charge for virtual machine usage according to usage time of virtual machine. Communication cost and transfer cost are charged based on transmission speed of data and dataset size. The problem of allocating datasets to VMs in distributed heterogeneous Clouds is formulated as a linear programming model with two objectives: the cost and processing time. After finding optimal solutions of each objective function, we use a heuristic approach to find the Pareto front of multi objective linear programming problem. In the simulation experiment, we consider a heterogeneous Cloud infrastructure with five different types of Cloud service provider resource information, and we optimize dataset placement by guaranteeing Pareto optimality of the solutions.**

I. INTRODUCTION

Cloud Computing is a promising solution to handle a large size datasets. Since Cloud systems have large size computing resource pools, large size datasets can be processed in parallel after being divided to smaller partitions. For example, MapReduce and Hadoop are representative frameworks for distributed processing used in Cloud systems. MapReduce model is composed of Map and Reduce part. Map divides large size datasets to smaller sub problems by filtering and sorting them. Then, Reduce part collects all the results from sub-problems and merges them.

Distributed Cloud Computing systems are designed with geographically separated Cloud data centers. When a user wants to send a request to the Cloud, the Cloud manager can choose the closest data center for processing. Thus, it is beneficial to reduce latency between users and data centers. However, user requests have many different requirements. Some applications require high performance computing, some other application need low latency, and for some others low cost is the most important requirement. It cannot be guaranteed that the closest data center can satisfy all user demands. Therefore, efficient resource allocation is required so that Cloud systems satisfy various user demands. Processing time is an important objective for both of service providers and users. Service providers can offer service to more users, and hence increase revenue, by reducing the processing time, which will also offer improved performance to users. Cost is also an important objective for both of users and service providers. Needless to say, users want to spend less money to use Cloud service. The main cost for users is the virtual machine cost. Users pay virtual machine cost depending on usage time of virtual machines, which increases with increased VMs' computational power. The service provider can also help reduce network cost by allocating datasets to virtual machines having low communication cost between each other.

Processing time includes data transfer time, virtual machine processing time, and communication time. Data transfer time occurs when a data center, which receives the dataset initially, needs to send subsets of these datasets to other data centers for processing. If the dataset size is too large to handle in the initial data center, the Cloud manager will divide the dataset and send the subsets to other data centers. So, data transfer time is the time before the start of processing. Virtual machines processing time is the processing time in virtual machines to process jobs. Virtual machines have different computational power, memory, storage size and operating systems. Therefore, processing time is very different depending on which type of virtual machine we use. The Cloud manager will select high performance virtual machines to reduce processing time within given constraints. Communication time is time for communication after complete processing of allocated dataset. The final results of computing over divided datasets need to be merged. Communication time occurs in this step to share status information of those datasets.

Cost consists of data transfer cost, virtual machine cost, communication cost, and bandwidth cost. Data transfer cost is related to data transfer time. When datasets have to be sent to different data centers from the first data center, this cost is incurred. Most Cloud providers charge data transfer cost only for outgoing data from data centers according to dataset size. Therefore, data transfer cost will be only charged to move sub-datasets from initially allocated data center to other data centers. Virtual machine cost increases depending on how long user uses the virtual machine. So, reducing virtual machine cost is also related to minimizing virtual machine processing time. However, high performance virtual machines have high cost for using that. Thus, efficient selection of virtual machines is important to reduce both of the virtual machine cost and

processing time. Communication cost is also the cost for sending overhead of sub-dataset after finishing processing in order to share their processing status. Communication cost is also only charged for outgoing data from the data center. So we only consider outgoing overhead from each allocated virtual machine to Cloud manager.

Contribution of this paper is to obtain the optimal allocation of datasets to VMs, using two objectives: processing time and cost. Processing time and cost is changed a lot according to which virtual machine or which types of service platform we use. Therefore, our purpose is finding optimal dataset placement in heterogeneous computing resource environment in order to reduce processing time and cost simultaneously.

Our model is multi-objective linear programming model. We can solve each objective functions by using many known solutions. After solving each objective independently, we can start to find Pareto front by using Benson's algorithm. We will find Pareto front of multi objective linear problem.

Previous research of resource allocation issues in Cloud systems are presented in Section Ⅱ. Section Ⅲ describes the dataset allocation problem by using multi objective linear Programming model. We will introduce an algorithm that can solve multi linear objective problem in Section Ⅳ. We developed a simulation model based on practical Cloud service providers' resource in Section Ⅴ. In the Section Ⅵ, we will compare results of each objective function separately and show Pareto front of our simulated Cloud systems. The conclusions are given in Section Ⅶ.

II. Related work

The importance of virtual machine placement is recognized by various objectives. Since the Clouds have heterogeneous resources, it is possible to achieve various objectives by allocating virtual machines according to different strategies.
In distributed Clouds, latency could be minimized by using standard assignment algorithms as in [1]. Alicherry et al. used an integer linear programming model for reducing latency between virtual machines. Latency is mainly affected by the geographical distance between data centers. The model is optimized using one of two objective functions. The first is minimizing the total access time and the other is minimizing the maximum access time. Minimizing total access time can reduce overall access time of running the job. However, minimizing maximum access time is used when performance of the job is the most important. In other words, minimizing maximum of objective function could guarantee performance of the application but it increases total access time compared to minimizing total access time.

Many optimization formulations have been introduced to reduce the cost of power consumption in Clouds. Optimizing power allocation and load distribution has been achieved in [7] by using a queuing network model. Also, a load balancing model was introduced to reduce Carbon Emission in [5]. However, minimizing the cost of virtual machine usage has not received enough attention even though it is a very important factor from a user's perspective.

Also, only a few attempts to optimize multiple objective in Clouds exist in the literature. In [8], Ruiz-Alvarez et al. minimized cost, latency, and bandwidth. However, they have scalarized multi objectives to a single objective function. Consequently, the problem could not guarantee Pareto optimality of each object.

An important contribution of this paper is finding Pareto Optimal solutions [10]. We use a special case of Benson's algorithm [16], outer approximation algorithm for multi objective linear programming problem, which uses a vector summation of objective values, and then finds solutions that guarantee Pareto optimality.

III. Problem description

Our goal is to optimize the allocation of datasets to Cloud Computing systems that have heterogeneous resources. We have two objective functions: 1) processing time, and 2) cost. Processing time and cost are in inverse proportion to each other. In order to decrease processing time, it is helpful to allocate datasets to high performance virtual machines. However, dataset placement to high performance virtual machines increases the cost. Therefore, we attempt to find Pareto optimal points that guarantee optimality of processing time and cost simultaneously.

Large size datasets will be divided into small dataset, and the datasets will be assigned to different virtual machines that have different resources. We assume parallel processing of datasets and do not consider serial processing of partitioned datasets.
We develop a linear programming model for this problem. There are two objective functions that have the dataset size as a decision variable. Both objective functions share the same constraints. There are four types of constraints in the problem: Storage capacity, summation of total dataset size, individual dataset size, and communication latency. Table 1 shows the glossary of all terms used in our model.

TABLE1. PROBLEM MODEL VARIABLES

Variable	Description
D*mnl*	Decision variable. The dataset size allocated to the *l* th machine in the nth data center, which is derived from *m*th dataset
i	Constant. The number of datasets
j	Constant. The number of data centers
K*n*	Constant. The number of virtual machines in *j*th data center
DS*mn*'	Constant. The *m*th dataset initially submitted to *n*'th data center
B*nl*	Constant. Network bandwidth of nth data center *l*th virtual machine(Gb/sec)
B*n'n*	Constant. Bandwidth for data transfer from *n*'th data center(initially arrived) to nth data center (Gb/sec)
P*nl*	Constant. Processing time of nth data center *l*th virtual machine (Gb/sec)
CC*nl*	Constant. Communication Cost about outgoing data from nth datacenter *l*th virtual machine ($/GB)
TC*j'n*	Constant. Transfer Cost about outgoing data from initial arrived datacenter *j*'th to nth data center ($/GB)
VC*nl*	Constant. Virtual machine cost of nth data center *l*th virtual machine ($/GB)
C*nl*	Constant. Available Capacity of *l*th virtual machine in nth data center
LT*nl*	Constant. Maximum available latency of communication time from *n*th data center (sec)
BC*nl*	Constant. Cost to upgrade the bandwidth of nth data center *l*th virtual machine ($)

α	Constant. Communication overhead rate from original dataset size ($0 \leq \alpha \leq 1$)

A. Processing Time

Minimize

$$\sum_{m=1}^{i}\{\sum_{n=1}^{j}\sum_{l=1}^{kn}\frac{\alpha}{\mathrm{B}nl}\times \mathrm{D}mnl+\sum_{\substack{n=1\\ \mathrm{n}\neq j'}}^{j}\sum_{l=1}^{kn}\frac{(1+\alpha)}{\mathrm{B}j'n}\times \mathrm{D}mnl+\sum_{n=1}^{j}\sum_{l=1}^{kn}\frac{\mathrm{D}mnl}{Pnl}\} \quad (1)$$

The processing time objective function includes time to allocate sub-datasets to other data centers, virtual machine processing time, and time to send dataset overhead to Cloud manager for share their processing status. Each datacenter has Cloud manager. The data center that receives dataset first becomes Cloud manager for the dataset. Therefore, we consider data transfer time from initial arrived data center and communication time to send processing status to initial datacenter, and virtual machine processing time of sub-datasets in each virtual machine. The first term is communication time between all virtual machines. Since a large size dataset is divided into several smaller datasets, each virtual machine needs to check processing status of sub-datasets. Therefore, after finishing the processing in each virtual machine, virtual machines send their status to the Cloud manager. We consider only the outgoing time from allocated virtual machines for communication time. Some Cloud service providers provide high bandwidth for outgoing data transfer from the virtual machine with additional cost. Thus, only considering outgoing time from the virtual machine will be the dominant factor in communication time. We don't need to transfer all data for communication. Communication overhead is configured with some rate α from dataset size D*mnl,* allocated sub-dataset in the *l*th virtual machine in *n*th data center. By dividing the dataset overhead size with bandwidth of *l*th virtual machine in *n*th data center, communication time of the virtual machine can be obtained.

The second term refers to the data transfer time. The initial dataset is such that their required resource exceeds available resources in a certain data center. If it is able to reduce processing time by dividing initial dataset and allocating sub-datasets to other virtual machines in a certain data center, we have to transfer the datasets to other data center's virtual machines. This process would consume some transfer time. So transfer time can be calculated by dividing allocated dataset and communication overhead size by communication bandwidth of the initially allocated data center. We assume datasets have no serial relationship. All datasets are possible to be processed in parallel model in this problem.

The third term refers to the virtual machine processing time. Virtual machine processing time is one of the most important factors in deciding the processing time objective value. Cloud service providers offer different types of service depending on user's demands. The most representative difference is the number of CPUs. High performance virtual machines provide more number of cores. So it is able to reduce processing time by the allocating more data to the virtual machine that has high performance processor or high processing ability.

B. Cost

Minimize

$$\sum_{m=1}^{i}\{\sum_{n=1}^{j}\sum_{l=1}^{kn}\alpha\times \mathrm{D}mnl\times CCnl+\sum_{\substack{n=1\\ n\neq j'}}^{j}\sum_{l=1}^{kn}(1+\alpha)\times \mathrm{D}mnl\times TCj'n+\sum_{n=1}^{j}\sum_{l=1}^{kn}\frac{\mathrm{D}mnl}{Pnl}\times \mathrm{VC}nl\}+\sum_{n=1}^{j}\sum_{l=1}^{kn}BCnl \quad (2)$$

We consider similar factors in the cost objective function: communication cost, data transfer cost, virtual machine cost, and bandwidth cost.

The first term stands for communication cost. Most Cloud Computing service providers charge for only outgoing data transfer depending on data size and communication bandwidth. In the first term, we only consider cost for dataset size. Similar to communication time, only outgoing communication overhead is charged for communication cost.

The second term is about data transfer cost. As mentioned in section A, initial dataset has to be moved to other data center as necessary. Since data transfer cost is also only charged for outgoing data, we multiply cost by dataset and overhead size.

The third term is virtual machine cost. Virtual machine cost charged depends on hours of use and its performance. High performance virtual machines are more expensive than low performance virtual machine. It may seem cheaper if we just use low performance virtual machine to reduce virtual machine cost, but virtual machine cost also increases with usage time of the virtual machine. If the virtual machine processing time increases, the virtual machine cost is increased simultaneously. So the virtual machine cost is multiplied by virtual machine processing time in the third term in order to make the best decision between performance and usage time of virtual machine.

Bandwidth cost is not proportional to dataset size. When user wants to upgrade communication bandwidth of certain virtual machines, most Cloud service providers charge additional cost per month. We do not consider that long term processing time over a month. Therefore, bandwidth cost is fixed and is not affected by dataset allocation.

C. Constraints

$$\mathrm{D}mnl \leq \mathrm{C}nl,\ \forall m,n,l \quad (3)$$

$$\sum_{n=1}^{j}\sum_{l=1}^{kn}\mathrm{D}mnl \geq \mathrm{DS}mn',\ \forall m \quad (4)$$

$$\mathrm{DS}mj' >= \mathrm{D}mnl \geq 0,\ \forall m,n,l \quad (5)$$

$$\sum_{l=1}^{kn}\frac{\alpha * \mathrm{D}mnl}{\mathrm{B}nl} \leq \mathrm{Lt}n,\ \forall \mathrm{m,n} \quad (6)$$

(3) is virtual machine capacity constraint. Virtual machines have their own given amount of storage. New allocated dataset size cannot exceed the current available storage size of virtual machine.

(4) is summation of allocated dataset size constraint. Total size of assigned datasets originated from same dataset has to be greater than or equal to the original dataset size.

(5) is each dataset size constraint. Allocated dataset size cannot be greater than original dataset size.

(6) is latency constraint. This provides performance guarantees on the communication time from each data center. Latency will be different according to the types of datasets. Some datasets are sensitive to latency but some are not.

Therefore, we can set latency constraint depending on dataset's property.

The problem is multi-objective linear programming model. There are many known algorithms for solving single objective linear programming model like Simplex algorithm, Criss-cross algorithm, and Conic sampling algorithm. After solving each objective function, we will use a special case Benson's algorithm to find Pareto front of optimal solutions.

IV. ALGORITHM FOR MULTI OBJECTIVE OPTIMIZATION

In this section, we introduce a heuristic algorithm for finding the Pareto front of the multi objective linear programming model. The algorithm, which is shown in Algorithm 1, is designed for the maximization problem. We will reformulate the problem to an equivalent vector form of the maximization problem.

Maximize

$$Zetap = -P^T \times D \quad (7)$$

$$Zetac = -C^T \times D + b \quad (8)$$

P^T denotes the transpose of processing time coefficient vector about each D. D denotes the vector of decision variables D_{mnl}. Since P^T and D have the same dimension, we can obtain the objective value of processing time. Let us assume we allocate one dataset to two data centers which have 2 virtual machines each. Then D vector becomes [D111 D112 D121 D122]. The elements of vector P are summation of each term in (1). The each element in P will be summations of the data transfer time, virtual machine processing time, and communication time about all D_{mnl} elements in the. C^T is also transpose of cost coefficient vector having same dimension with D. Elements of the C vector are the summations of each term in (2) for all elements of D, just as like the P vector. b is the summation of all bandwidth cost, the last term in (2), which is not relevant to the decision variable D_{mnl}. So we can also obtain objective function of cost.

The algorithm starts with two Pareto optimal solutions of each objective functions: Dp and Dc. If Dp and Dc are on the same line, we can say all points between Dp and Dc are Pareto solutions. In order to verify if the Dp and Dc are on the same line or not, a new objective function is generated that has potential optimal points between Dp and Dc.

$$Zeta_k = [P^T(D_c - D_p)C + C^T(D_p - D_c)P]^T D \quad (9)$$

(9) is a new objective function that has optimal values between the optimal values of Dp and Dc. The first term in (9) is processing time vector having size between processing time of Dp and Dc in processing time. Also, the second term in (9) is cost vector having size between cost of Dc and Dp in cost axis. By summing two vectors, we can get new vector having new Pareto optimal point not dominated by both of Dp and Dc. Since we solve new objective function by same constraints, we can obtain new feasible optimal solution Dk. The next step is observing whether three Pareto points are in the same line or not. If $Zeta_k(D_k)=Zeta_k(D_p)=Zeta_k(D_c)$, the three points are in the same line.

If they are on the same line, which guarantees that the line between Dp and Dc is the Pareto front. If they are not on same line, we have to find new optimal points between Dp and Dk, and then between Dk and Dc. If we can find all pairs of adjacent lines connecting all Pareto points, we stop the iteration of the algorithm.

Algorithm 1 Exploring Pareto Front

1. **Input** : D_p, D_c, $Zeta_p(D_p)$, $Zeta_c(D_p)$, $Zeta_p(D_c)$, $Zeta_c(D_c)$
2. **Output** : All pairs of Pareto points and Pareto front
3. $D_i = D_p$, $D_j = D_c$
4. Hash (K_1, $D_i = D_p$, $D_j = D_c$, Optimal Pair)
5. $l = 1$
6. **While** ()
7. {
8. $K_l = \frac{Zeta_p(K_l D_i)+Zeta_c(K_l D_i)+Zeta_p(K_l D_j)+Zeta_c(K_l D_j)}{2}$
9. $K_{list} \leftarrow K_l$
10. $Zeta_{K_l} = [P^T(K_l.D_i - K_l.D_j)C + C^T(K_l.D_j - K_l.D_j)P]^T D$
11. D_{kl} = Find Optimal Solution of $Zeta_{K_l}$
12. **if** {
13. $Zeta_{K_l}(D_i) = Zeta_{K_l}(D_{kl}) = Zeta_{K_l}(D_j)$
14. line between D_j and D_j is Pareto front
15. $K_l \leftarrow$ Optimal Pair(D_i,D_j)
16. **if** {
17. l is the biggest index among K_l in K_{list}
18. **break**;
19. }
20. **else**
21. $l=l+1$
22. }
23. **else if** {
24. $Zeta_{K_l}(D_i) = Zeta_{K_l}(D_{kl}) \neq Zeta_{K_l}(D_j)$
25. line between D_j and D_{kl} is Pareto front
26. $K_l \leftarrow$ Optimal Pair(D_i, D_{kl})
27. Make hash
28. Hash (K_{l+1}=(D_i, D_{kl}), $D_i = D_i$, $D_j = D_{kl}$, Optimal Pair)
29. $l=l+1$
30. }
31. **else if** {
32. $Zeta_{K_l}(D_i) \neq Zeta_{K_l}(D_{kl}) = Zeta_{K_l}(D_j)$
33. line between D_{kl} and D_j is Pareto front
34. $K_l \leftarrow$ Optimal Pair(D_{kl}, D_j)
35. Make hash
36. Hash (K_{l+1}=(D_{kl}, D_i), $D_i = D_i$, $D_j = D_{kl}$, Optimal Pair)
37. $l=l+1$
38. }
39. **else if**{
40. $Zeta_{K_l}(D_i) \neq Zeta_{K_l}(D_{kl}) \neq Zeta_{K_l}(D_j)$
41. Make hash
42. Hash (K_{l+1}=(D_i, D_{kl}), $D_i = D_i$, $D_j = D_{kl}$, Optimal Pair)
43. Hash (K_{l+2}=(D_{kl}, D_i), $D_i = D_{kl}$, $D_j = D_j$, Optimal Pair)
44. $l=l+1$

45. }
46. }
47. Sort K_l in K_{list} by K_l value
48. K'_{list} : Sorted K_l list
49. **for**(n=1;n=l;n++)
50. {
51. Obtain optimal value pairs of all Pareto points in optimal value domain
52. $On = (Zeta_{p(K'n.Di)}, Zeta_{c(K'n.Di)}),(Zeta_{p(K'n.Dj)}, Zeta_{c(K'n.Dj)})$
53. }
54. If we connect all paris of On, we can get Pareto front

The algorithm starts from two optimal points of each of the objective functions and objective values (Line 1). In order to manage each point efficiently, hashmap function is employed including key value, two nearest optimal points not on the same line, and optimal pairs when optimal pairs exists (Line 4). Key value, which shows position of points, is determined depending on optimal values of nearest two points. By averaging optimal values of nearest two points, we can get key value sequentially to arrange optimal pairs (Line 8). After obtaining key value, Kl is saved in K$list$ (Line 9). New objective function is generated with two optimal points in Kl hashmap as explained in previous paragraph, and then solves the problem with the same constraints (Line 10 – Line 11). If optimal value of Zetakl(Di)=Zetakl(Dkl)=Zetakl(Dj), three points are in the same line. So we add optimal points Di and Dj to Kl hashmap as optimal pairs. If we cannot find anymore Kl in the Klist, then the Pareto front has been found. So we stop the iteration (Line 12 – Line 21).

If Zetakl(Di)=Zetakl(Dkl)≠Zetakl(Dj), Di and Dkl are in the same line but Dkl and Dj are not in the same line. Therefore, only Di and Dkl are added to Kl hashmap as optimal pairs. Since Dkl and Dj are not in the same line, new objective function having optimal point between Dkl and Dj has to be generated again. So we make new hashmap K$l+1$ including two points Dkl and Dj and empty optimal pairs (Line 23 – Line 30).

If Zetakl(Di) ≠Zetakl(Dkl) =Zetakl(Dj), Dkl and Dj are in the same line but Di and Dkl are not in the same line. As like previous paragraph, we save Dkl and Dj to optimal pairs of Kl hashmap and make new K$l+1$ hashmap including Di and Dkl (Line 31- Line 38).

If all three points are not in the same line, two hashmaps are generated including Di and Dkl, and then Dkl and Dj (Line 39 – Line 45).

The iteration is repeated until there is no more Kl in the K$list$. If we cannot find any more Kl, this guarantees that every hashmap k will include optimal pairs or empty pairs. By sorting Kl according to the Kl key value, Kl will be ordered sequentially. Then the Pareto front can be drawn by connecting all optimal points in Kl (Line 49 – Line 54).

V. SIMULATION MODEL

In this section, we evaluate the performance of the model and find Pareto optimal points of the multi objective problem by using the proposed algorithm.

We consider a Cloud consisting of 10 data centers having heterogeneous virtual machines. Data center resource information is configured from actual data of Cloud Computing service providers. Five types of Cloud data center infrastructures are employed: Microsoft Azure, Rackspace, Google Cloud, Amazon EC2, and IBM Cloud. Each Cloud provider offers different infrastructure services. For an example, Microsoft Azure has four options for CPU and has an option for upgrading network bandwidth. Rackspace provides all different network bandwidth depending on performance of virtual machines. A single core virtual machine provides 60Mbps of bandwidth, but a dual core virtual machine has 120Mbps network bandwidth. Google Cloud provides the highest performance virtual machine. Google services provide 16 core virtual machine model. All data centers have heterogeneous resources similar to this. Microsoft Azure (data centers number 1 and 6) includes 8 types of virtual machines, Rackspace(data centers number 2 and 7) has 10 types of virtual machines, Google Cloud (data centers number 3 and 8) has 5 types of virtual machines, Amazon EC2 (data center number 4 and 9) has 12 types of virtual machines, and IBM cloud (data center number 5 and 10) includes 8 types of virtual machines, and two data centers exist from each data center platform. So we have a total of 86 types of virtual machines to allocate dataset. Each virtual machine in a different data center has a different cost, processing time, and bandwidth. We generate the storage capacity of virtual machine randomly between 0 to data the storage size of virtual machines for constraint (3).

Three datasets are allocated to the Cloud. The first dataset has1000GB size and assigned to datacenter 3. The second dataset is assigned to data center 1 with 500GB size. The third dataset is placed to data center 4 with 700GB size.

VI. SIMULATION RESULTS

We used GUSEK, open source Linear/Mixed Integer Linear Programming solver to solve the two objective functions separately. The problem includes 258 decision variables and 548 constraints. In order to find the Pareto front, we implemented Algorithm 1 in MATLAB.

Figure 1 is the result of minimizing processing time. Dnl stands for the lth virtual machine in nth data center. High performance virtual machine has large index number l. In Figure 1, we can see most of the dataset is assigned to high performance virtual machines. Also, the dataset is mainly assigned to the initial datacenter to which it is submitted in order to reduce the data transfer time. The third dataset shows that 83% of the dataset is assigned to data center 4 in order to reduce data transfer time. Also, 90% of the first dataset is allocated to high performance virtual machine of data center 9 and 10 in the result.

Figure 2 is result of minimizing the Cost objective function. Compared to processing time minimization problem, the datasets are evenly distributed. Cost minimization also shows a similar allocation trend. In order to reduce data transfer cost, the dataset is mainly allocated to the initial data center. However, most datasets are not only assigned to low performance virtual machines, but many datasets are also allocated to high performance virtual machines.

This means that virtual machine cost is not a dominant factor in deciding the Cost. It seems like the effect of the data transfer

cost and communication cost is relatively more important than virtual machine cost according to the result.

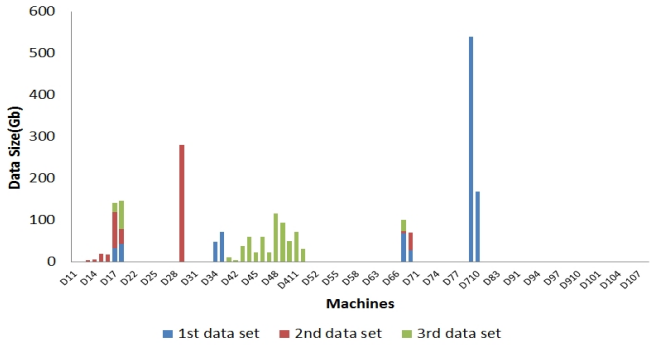

Fig 1. Minimum Processing Time Optimization Result

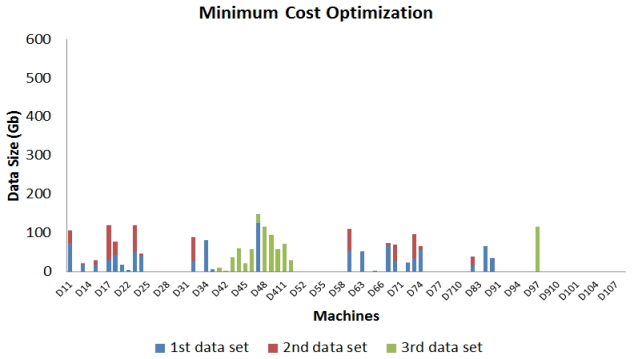

Fig 2. Minimum Cost Optimization Result

According to the optimization result of two separate objective functions, we could find two Pareto optimal points. The first point has cost value of 2088 and processing time of 8215. The second point has cost value of 2047 and processing time value of 8651 in Figure 3. Both points are not dominated by each other. Thus, both points are Pareto optimal points.

According to the introduced algorithm, we could find more Pareto optimal points between initial optimal points.

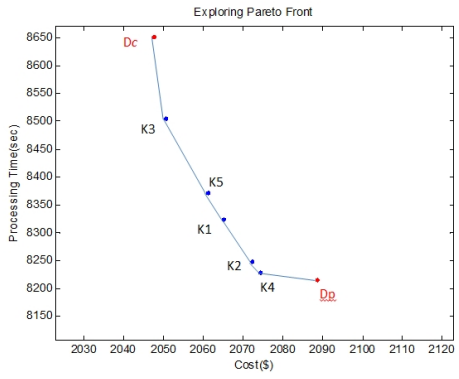

Fig 3. Pareto optimal points with iteration 5

At iteration 2, we could find two Pareto optimal pairs, K1 and K*2*. According to the algorithm, K*1* and K*2* are on the same line but K*2* and D*p* are not in the same line. So another Pareto optimal points will exists between K*2* and D*p*. Since K*1* and K*2* are on the same line, we can get partial Pareto front by connecting these two points. In iteration 3, we could find K*3* point on the same line with D*c*. In the same way, the line between D*c* and K*3* will be partial a Pareto front and there will be one more Pareto points between K*3* and K*1*.

After five iterations of the algorithm we could search Pareto points between D*c* and D*p*. If we connect these points sequentially, we can obtain Pareto front of this simulation model.

VII. CONCLUSIONS

In this paper, we have developed a strategy for allocating data sets to VMs in a heterogeneous cloud consisting of different types of data centers with each datacenter offering different types of VMs. The strategy is based on jointly considering the cost and processing time. We therefore formulated the assignment problem as a dual objective optimization problem, and introduced an algorithm for finding the Pareto front of optimal solutions. The result of separate objective function optimization shows proper characteristics of each objective. When we minimize processing time, the dataset is usually assigned to high performance virtual machines. However, dataset is allocated relatively to lower performance virtual machine when cost is optimized.

With the joint optimization, many of the Pareto points enable us to allocate datasets in many ways depending on properties of datasets.

References

[1] M. Alicherry and T.V. Lakshman. Optimizing Data Access Latencies in Cloud Systems by Intelligent Virtual Machine Placement. IEEE INFOCOM, 2013.

[2] M. Alicherry and T.V. Lakshman. Network Aware Resource Allocation in Distributed Clouds. IEEE INFOCOM, 2012.

[3] Xiaoqiao Meng, Vasileios Pappas, and Li Zhang. Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. IEEE INFOCOM, 2012.

[4] Mohamed Abu Sharkh, Manar Jammal, Abdallah Shami, and Abdelkader Ouda. Resource Allocation in a Network-Based Cloud Computing Environment: Design Challgnes. IEEE Communization Magazine, November 2013.

[5] Joseph Doyle, Robert Shorten, and Donal O'Mahony. Stratus: Load Balancing the Cloud for Carbon Emissions Control. IEEE Transactions on Cloud Computing, vol. 1, no. 1, pp.116–128, 2013.

[6] Tsahee Zidenberg, Isaac Keslassy, and Uri Weiser. Optimal Resource Allocation with MultiAmdahl. IEEE Computer Architecture Letters, vol. 11, no. 2, pp.65-68, 2012.

[7] Junwei Cao, Keqin Li, and Ivan Stojmenovic. Optimal Power allocation and Load Distribution for Multiple Heterogeneous Multicore Server Processors across Clouds and Data Centers. IEEE Transaction on Computers, vol. 63, no. 1, 2014.

[8] Arkaitz Ruiz-Alvarez, Marty Humphrey. A Model and Decision Procedure for Data Storage in Cloud Computing. IEEE/ACM 12th International Symposium on Cluster, Cloud, and Grid Computing, 2012.

[9] Yin Li, Min Yao, and Chuang Lin. Joint Study on Optimizations of Data Center Deployment, VM Assignment and Migration. IEEE/ACM 21st International Symposium on Quality of Service. 2013.

[10] R.T.Marler and J.S. Arora. Survey of Multi-Objective Optimizations Methods for Engineering. Struct Multidisc Optim 26, pp.369-395,2014.

[11] Google Cloud price: https://developers.google.com/storage/pricing

[12] Amazon Cloud price: http://aws.amazon.com/ec2/pricing/

[13] IBM Cloud: http://www.ibm.com/cloud-computing/us/en/iaas.html

[14] Rackspace Cloud price: http://www.rackspace.com/cloud/servers/pricing/

[15] Microsoft Cloud price: http://azure.microsoft.com/en-us/pricing

[16] Harold P. Benson. An Outer Approximation Algorithm for Generating All Efficient Extreme Points in the Outcome Set of Multi Objective Linear Programming Problem. Journal of Global Optimization13 (1), pp 1-24, 1998.