S-CDA: A Smart Cloud Disk Allocation Approach in Cloud Block Storage System

Hua Wang†, Yang Yang†, Ping Huang‡, Yu Zhang†, Ke Zhou†*, Mengling Tao†, Bin Cheng§
†Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, China
‡Temple University, America §Tencent Technology (Shenzhen) Co., Ltd., China
*Corresponding Author: k.zhou@hust.edu.cn

***Abstract*—Cloud disk provided to users by cloud providers has been a prevalent form of cloud storage. Assigning disk storage space for cloud disks among available data warehouses remains a challenging task, as the load characteristics of cloud disks are not known at the time of disk creation. Methods considering only subscribed capacity could be prone to result in resource under-utilization and load imbalance among warehouses. We propose a Smart Cloud Disk Allocation (S-CDA) approach, which uses clustering and classifying to predict the load information for new cloud disks, and then realizes multi-dimensional allocation based on Manhattan distance. Experimental results with realistic cloud workloads show that, compared with the existing one-dimensional allocation scheme, S-CDA increases the overall space/IOPS/disk bandwidth utilization, while decreasing the load imbalance.**

***Index Terms*—Cloud Disk Allocation, Clustering, Classifying, Manhattan Distance**

I. Introduction

With the rapid development of cloud computing and other Internet technologies, the amount of data increases rapidly. Cloud storage technology can effectively store and manage massive data, hence has been widely attended and supported, and has become the main development trend in the field of information storage. Cloud Disk (CD) is a typical kind of product in Cloud Block Storage (CBS) system [1] [2], which provides efficient and reliable storage service for tenants. Currently, a large number of cloud storage providers, such as AWS, Alibaba and Tencent, provide such block-level storage service for cloud tenants.

A classic CBS system consists of four parts, namely, clients, Virtual Machines (VMs), Cloud Disks (CDs) and cloud storage (as shown in Fig.1). Tenants run applications on their own VMs and can apply for many CDs for their VMs. CD is configured with a generic block-level access interface, through which client-end application programs can access remote data stored in the cloud just like referencing local disks. Each CD is mounted to VMs as a virtual volume. For the consideration of data security and management convenience, storage layer is divided into multiple physically isolated storage rooms. A storage room is named as a warehouse. Each warehouse is subject to various resource constraints, such as storage capacity, disk bandwidth, IOPS and etc. Each CD is allocated into only one warehouse according to a certain allocation policy, which has a direct impact on the resource utilization and load balance of warehouses. Since upon the time of allocation, only the storage capacity is known through tenants' subscriptions, current used allocation policy is one-dimensional allocation(ODA) which considers only subscribed capacity, therefore, ODA could only guarantee high efficiency of storage capacity.

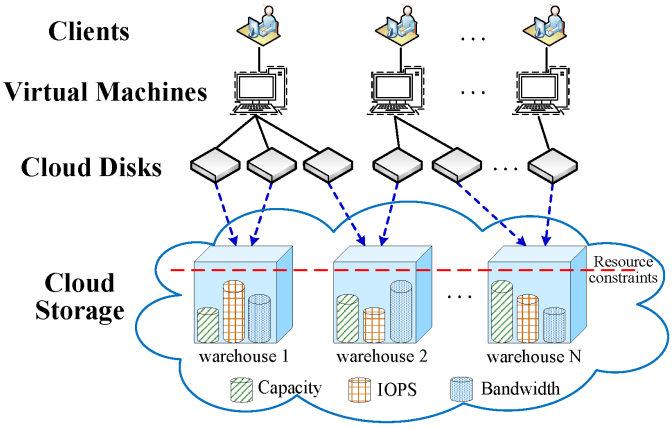

Fig. 1. Architecture of Tencent cloud block storage systems

A. Motivation

As Jitpukdeebodintra [3] et al., pointed that in solving an optimization problem, considering only a single dimension, while ignoring other dimensions, could lead to sub-optimal resource utilization. This is not unexpected and we have further verified this by analyzing the realistic trace of Tencent (the largest social network company in China) CBS system, whose current policy is based on ODA that only considers capacity. The main observations from the trace provided by Tencent include:

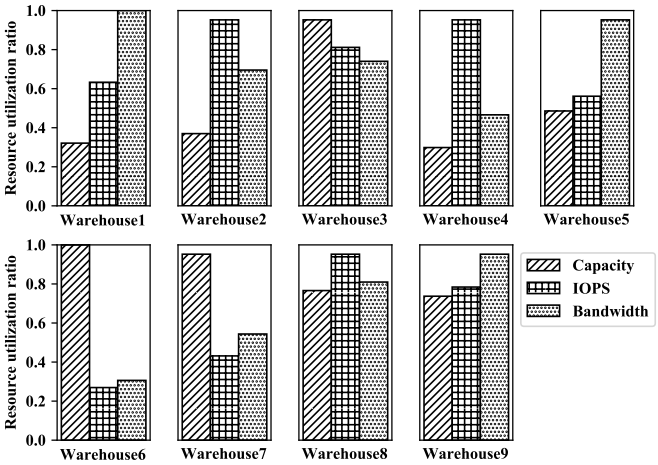

Fig. 2. The warehouse resource utilization under the ODA policy

978-1-7281-1085-1/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.

1) Low Utilization of Multi-Dimensional Resources: With Tencent real historical trace, we have analyzed one-dimensional resource utilization by simulating the process of disk allocation. The simulator allocates new CDs to one of the 9 warehouses using ODA policy. The resource utilization results are shown in Fig. 2. Apparently, the utilizations of the multi-dimensional resources (Capacity, IOPS, and disk bandwidth) are all below 75%. There is much room for improvement.

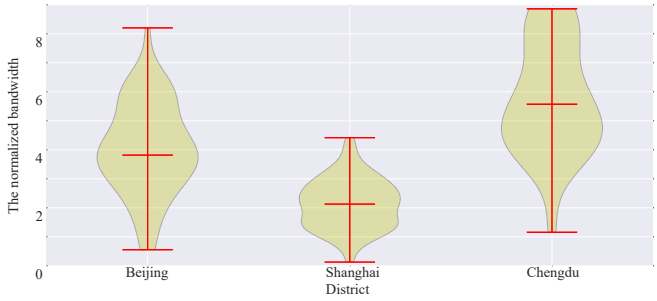

Fig. 3. Disk bandwidth distribution among warehouses in different districts

2) Load Imbalance: The ODA policy also leads to unbalanced load across warehouses. We made a statistical analysis of the disk bandwidth distribution of thousands of warehouses in three regions of Tencent CBS system. As illustrated in Fig. 3, it shows current load-balance (i.e., disk bandwidth) scenarios of three data centers (in three different districts), with each comprised of thousands of warehouses. The y-axis of Fig. 3 is the normalized average disk bandwidth of warehouse. From this figure, we can observe that the average disk bandwidth exhibits a large variation among the warehouses in different districts.

B. Our Idea

In light of the above observed inefficiency of ODA policy, it is urgent to design a more appropriate CD allocation policy to alleviate the problems mentioned above. From the theoretical viewpoint, this problem is indeed an optimization problem under constraints of multiple resources. But the challenge with CD allocation lies in that the load features (e.g., IOPS, disk bandwidth, and so on) of a newly CD are unknown beforehand. If we would be able to gain the information about the features of the new CDs, the problem boils down to an optimization under multiple constraints which are known.

For cases like the above where load features are missing, machine learning techniques can be applied to obtain the missing information based on existing related data. Compared with traditional heuristic scheme, machine learning-based approach can improve the accuracy and effectiveness of decision-making [4]. Inspired by some previous works [5], [6], [7], [8], we cluster the historical CDs' trace according to their load features to obtain labels, then use the subscription features and labels of historical trace to do classifier training. Then when a new CD is created, it can be classified according to its subscription features before allocation. The load features of the most similar category can be used to approximately represent future load features of the new CD. Having the information of predicted load features, we can realize a multi-dimensional resource allocation policy.

The contributions of this paper are summarized as follows:

1) Based on the trace of historical CDs provided by Tencent, we propose a Smart CD Allocation (S-CDA) approach, which uses k-means clustering algorithm to cluster the load features of historical CDs to obtain labels. Then it uses the subscription features of the trace and labels to train the decision tree classification model. S-CDA then identifies the most similar label for each new CD by the decision tree model through subscription features, so as to obtain the label of load features for the new CD.
2) After obtaining the load features through clustering-classifying method, S-CDA employs a multi-dimensional resource allocation policy based on Manhattan distance to select the appropriate warehouse for CD allocation.
3) We conduct comprehensive experiments to evaluate the efficiency of S-CDA. Experimental results show that S-CDA improves multi-dimensional resource utilization and decreases load imbalance.

II. Load Prediction by Machine Learning

It is beneficial to develop a multidimensional CD allocation strategy to optimize the problems of resource waste and load imbalance in a CBS system caused by the ODA policy that considers only capacity of CDs. In a CD allocation strategy, load dimension is considered to be indispensable in improving the resource utilization of the warehouse and ensuring a more balanced load. However, the main challenge lies in that the new CD's future load features are unknown, which disallows us to adopt the multi-dimensional allocation. Therefore, it is necessary to predict the load of the new CD before allocation.

Because the load behavior of different CDs varies greatly, it is impossible to establish a uniform and effective prediction model. Modeling for each CD separately is even more impractical. Considering the above factors, it is feasible to predict CD with similar load behavior for each class. And it is very suitable to use clustering method to distinguish the type of historical trace. CDs with similar load features in the historical trace are grouped into one class by clustering. In this case, the CD trace can be firstly clustered according to their load features, and each category can be defined as a label according to the clustering results, and then the classification model can be trained based on these labels and subscription features to judge the label of new CDs, and the load features of this type can be used to represent load features of the new CD. The application of clustering algorithm and classifying algorithm in CBS system to predict the load of CDs will be introduced as follows.

A. Clustering Algorithm

We need to classify the historical CDs trace into classes through their load features. Clustering is the most widely used method in un-supervised machine learning, whose goal

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.

is to exploit the inherent nature and laws of the data by learning the samples. In recent resource scheduling researches, many works use clustering to analyze load features to realize resource scheduling and load balancing in cloud systems [7], [8]. Besides, the tag information of clustering training samples is also unknown, Clustering technology is very suitable for unlabeled data analysis. Therefore, we cluster historic CDs through their load features.

1) Selection of Clustering Algorithm: Clustering Algorithm has a great influence on the prediction accuracy of load features for new CDs. In order to search for the most appropriate algorithm, we make comparisons among the three representative and commonly used clustering algorithms: k-means, DBSCAN and AGNES. Tencent provides trace of about 37k CDs from 61 warehouses in Tencent's data centers for 15 days. We select randomly 20k records to analyze the load features of CDs. Each record includes average IOPS, average disk bandwidth, peak IOPS and peak disk bandwidth of the CD. We evaluated clustering performance of the above three algorithms using two common evaluation metrics: silhouette score and running time. Silhouette score considers the cohesion and coupling to evaluate the clustering effect, and its value ranges from 0 to 1. The higher the degree of intra-class cohesion and the lower the degree of inter-class coupling, the larger the silhouette score will be.

TABLE I
COMPARISON OF DIFFERENT CLUSTERING ALGORITHMS

Algorithm	Silhouette score	Run time (s)
k-means	0.936	0.173
AGNES	0.724	2.26
DBSCAN	0.937	6.12

From Table I, we find that the Silhouette scores of k-means and DBSCAN are almost the same, whereas the running time of k-means is much smaller than that of DBSCAN. As to AGNES, it behaves far poorly than k-means both in Silhouette score and running time. Therefore, we choose k-means as our clustering algorithm.

2) Configuration of k-means: The k-means algorithm needs to determine value k in advance, and we use the elbow method to determine it. With the increase of cluster number k, the sample division becomes more refined, and the degree of aggregation of each cluster will gradually increase, so the Sum of Square Error (SSE) will gradually decrease. After comprehensive sensitive tests, we set the value of k to be 3.

3) Load Features: The features of each CD include two types: subscription features and load features. In clustering, only load features are required. The CDs with similar load features can be clustered into a category. Load feature refers to average IOPS, average disk bandwidth, peak IOPS and peak disk bandwidth within 15 days of each CD.

B. Classifying Algorithm

After clustering similar CD in the historical trace into classes, the trace is labeled. A new CD only has some subscription features before allocation. The label of new CD can be determined by subscription features and classifier. The classifier model is trained according to subscription features and label of the historical trace, so that the label of new CD can be distinguished, and the load features of this label can be used to represent load features of the new CD.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT CLASSIFIERS

Algorithm	Macro-P	Macro-R	Accuracy	F1-score	run time(s)
Decision Tree	0.837627	0.822917	0.823146	0.830207	0.251355
Random Forest	0.839855	0.824910	0.825140	0.832315	2.345045
AdaBoost	0.724274	0.696530	0.696696	0.710131	5.519235
Naive Bayes	0.639049	0.590739	0.590359	0.613945	0.220436
Logic Regression	0.568024	0.512458	0.513234	0.538812	1.382274

TABLE III
DEFINITIONS OF METRICS IN CLASSIFICATION

Metric	Definition
Precision (P)	P = True Positive / (True Positive + False Positive)
Recall (R)	R = True Positive / (True Positive + False Negative)
Accuracy	The proportion of the number of correctly classified samples to the total number of samples.
Macro-P	Take one kind as positive samples and the rest as negative samples to calculate the mean of P.
Macro-R	Take one kind as positive samplse and the rest as negative samples to calculate the mean of R.
F1-score	The harmonic mean of Macro-P and Macro-R.

1) Selection of Classifying Algorithm: Classifying Algorithm has a great influence on the prediction accuracy for labeling for new CDs. We makes a comprehensive comparison of the current mainstream classifying algorithms. We use the trace labeled after clustering processing. Table II shows the performance evaluation of each classifier. The decision tree algorithm and the ensemble learning algorithm (random forest and AdaBoost) are obviously superior to other algorithms in terms of performance. It shows the performance of the random forest algorithm was only improved by 0.15%, but the training time cost was nearly 17 times that of the single decision tree. We decide to use the decision tree [9] as the final classifier in this paper. And the detail description of performance metrics in Table II could be found in Table III.

TABLE IV
THE EXPLANATION OF SUBSCRIPTION FEATURES

Subscription features	intention
DiskType	The type of CD, such as data disk, system disk.
IsVip	Is the user a Vip?
CPU	CPU cores of the user's VMs.
Memory	The memory size of the user's VMs.
PayMode	Payment method of users.
IsLocal	Is the CD local?
Is1c1g	Is the user's VMs 1 CPU and 1GB?
SnapUse	The snap size of the CD.
DiskSize	The capacity of CD.

2) Subscription Features: We use the trace labeled after clustering. The subscription information of the statistical history CDs is taken as the input features, and the label as the output result. There are 9 subscription features in the trace, as shown in Table IV. Then the trace was divided into training set and testing set. The training set was used to complete classifier training, and the testing set was used to cross-verify and evaluate the performance of different classifiers.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.

3) Feature Selection: The feature selection of the model aims to find appropriate features from the original feature set of user subscription information, avoiding the interference of invalid features on the classifier. In this paper, the gini index in the process of decision tree training is employed to determine the appropriate features. The larger the gini index, the more helpful the feature is to the classification. Based on this method, we finally adopts $\{DiskSize, CPU, Memory, DiskType, IsLocal\}$ as the final feature set to train the classifier.

III. Multi-Dimensional Allocation Algorithm for Cloud Disks

The existing ODA policy for CDs only considers capacity. As a result, the utilization of multi-dimensional resources among warehouses is suboptimal, and the load imbalance among warehouses is very high. After predicting the load features for the new CD in Section II, we could solve this problem through multi-dimensional optimization.

In fact, CD's multi-dimensional allocation is similar to VM allocation in cloud, with the difference being that the multi-dimensional info of the former are unknown, whereas those of the latter are given. The VM allocation problem is usually translated into a bin-packing problem [10], [11], [12], which assigns objects to the appropriate boxes by using some policy, so that the least boxes are used or a certain attribute could achieve the optimal value. In our scenario, the CDs and warehouses correspond to objects and boxes respectively. The aim is to maximize resource utilization and reduce load imbalance. We consider three dimensions: capacity, IOPS, and disk bandwidth, which are three important and constraint resources in a warehouse.

A. Problem Modeling

We represent n warehouses as $\mathrm{W} = \{W_1, ..., W_n\}$, and the upper bound resource of capacity, IOPS and disk bandwidth for warehouse j is expressed as $\mathrm{L_j} = \{C_j, I_j, B_j\}$. The actual used capacity, IOPS and disk bandwidth of warehouse j is expressed as $\mathrm{l_j} = \{c_j, i_j, b_j\}$. We represent the requested capacity (subscribed by the tenant) average IOPS and average disk bandwidth (obtained from the most similar clustered category) of a new CD as $r = \{x, y, z\}$. The utilization of capacity, IOPS and disk bandwidth of W_j could be expressed as the following equation:

$$\begin{cases} W_{\mathrm{j}}^c = \dfrac{c_j}{C_j} & W_j^i = \dfrac{i_j}{I_j} \quad W_j^b = \dfrac{b_j}{B_j} \end{cases} \tag{1}$$

A new CD may be placed in W_j if the following conditions are met:

$$\{x + c_j <= C_j \quad y + i_j <= I_j \quad z + b_j <= B_j \tag{2}$$

After the CD is assigned to W_j, the utilization of W_j for all the three dimensions are expressed as the following equations:

$$\begin{cases} W_j^c = \dfrac{x + c_j}{C_j} & W_j^i = \dfrac{y + i_j}{I_j} \quad W_j^b = \dfrac{z + b_j}{B_j} \end{cases} \tag{3}$$

B. Multi-Dimensional Allocation Policy Based on Manhattan Distance

In a CBS system, CD allocation must be performed online due to the real-time demands generated by users. Online scenarios are very common in cloud data centers. Filiposka el. [13] point out that due to the uncontrollable arrival of cloud service requests, it is unacceptable to determine the best location of all cloud services using the batch offline method. There are researches studying online multi-dimensional allocation in cloud. The most common methods are classical packing methods, such as First Fit (FF) [14], Next Fit (NF) [15] or Best Fit (BF) [16].

To deal with multi-dimensional resource constraints, the common practice is to reduce dimensions, that is, multi-dimensions are transformed into one normalized dimension. Inspired by this, we compare commonly used distances in mathematical modeling. Based on the experimental results, we propose a heuristic online and multi-dimensional allocation policy based on Manhattan distance. To our knowledge, we are the first to use this method in CBS systems.

In order to reduce the load imbalance, we calculated the Manhattan distance between the resource utilization for the three dimensions of the warehouse and the average resource utilization. The average resource utilization of the three dimensions are calculated using the following equation:

$$\mathrm{U}_j = \frac{W_j^c + W_j^i + W_j^b}{3} \tag{4}$$

After the new CD is put into the warehouse, based on (3) and (4), the Manhattan distance is defined as the following equation:

$$M_j = \left|W_j^c - U_j\right| + \left|W_j^i - U_j\right| + \left|W_j^b - U_j\right| \tag{5}$$

Algorithm 1 Manhattan heuristic allocation algorithm

Input: The resource requirement vector r for new CD; Warehouse set W; Warehouse's resource upper bounds L; The warehouse's used resource set l.

Output: $FLAG$//The warehouse number of the final placement.

```
FLAG = −1;
MinDis = INF;
for each W_i in W do
    if r + I_i < L_i then
        Mi = ManhattanDis(r, W_i);//Based on (4), (5).
        if MinDis > M_i then
            FLAG = i; MinDis = M_i;
        end if
    end if
end for; return FLAG;
```

Algorithm 1 shows the pseudo-code of our Manhattan heuristic algorithm about a new CD allocation. The Manhattan distance between the new CD r and each warehouse W_i was calculated based on (4) and (5), and the warehouse number denoted as $FLAG$ indicating the minimum distance, was finally chosen as the destination of disk allocation.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.

IV. Design and Implementation

A. Architecture Overview

Fig. 4 illustrates the architecture of CD allocation system, which consists of four main parts: Client, backend storage, clustering component and classifier, in which the latter two parts are the new components over traditional CD allocation architecture. The main function of clustering component is to cluster historical trace of CDs in storage according to their load features. The function of classifier is to classify the new CD into the most similar category based on its subscription features, then the average load features of that category could be applied to the new CD as its predicted load features.

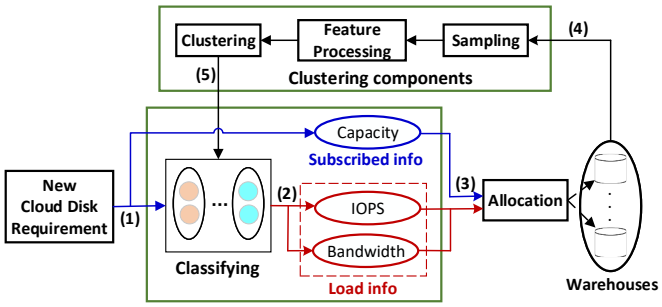

Fig. 4. The architecture of Cloud Disk allocation

With the clustering component and the classifier added, the allocation workflow proceeds as follows:

1) CD Allocation System receives a request from the client;
2) The classifier classifies the new CD according to its subscription information and assigns the average load features of the corresponding category to the new CD;
3) After obtaining load features for the new CD, the system selects the appropriate target warehouse according to the multi-dimensional heuristic allocation policy based on Manhattan distance;

   The training process comprises of the following:
4) Train clustering model by using historical trace;
5) The trace labeled after clustering is used to train the classifier.

Through clustering and classifying, the load features of new CD could be predicted. Consequently, S-CDA uses a multi-dimensional heuristic allocation policy to allocate a new CD, so as to improve resource utilization and alleviate load imbalance among data warehouses in a CBS system.

V. Experimental Results

A. Evaluation Setup

We use the trace from Tencent on the cloud simulation platform to conduct comprehensive tests to evaluate S-CDA. The trace comprises of about 37k CDs from 61 warehouses in Tencent's data centers for 15 days(detailed in Section II-A3 and Section II-B2). We use about 20k CDs from 50 warehouses to train clustering model and classifying model (detailed in Section II). Then we use the remaining 17k CDs from 11 warehouses to perform CD allocation simulation in cloud simulation platform to test S-CDA. In order to accomondate each warehouse with the available trace as much as possible, the number of warehouse used for disk allocation in the simulation test should be no more than 9. We vary the number of warehouses gradually from 6 to 9. In order to evaluate the impact of different allocation policies on allocation, we compared ODA, Manhattan distance, Euclidean distance, Cosine distance, and Pearson. Configuration of the experimental environment is as follows: Intel Xeon CPU E5-2609 @ 2.5GHz, 32GB RAM.

B. Evaluation Metric

1) Resource Utilization: The total resource utilization rate is expressed by the following equation:

$$\begin{cases} C_{uli} = \dfrac{\sum_{j=0}^{n} c_j}{\sum_{j=0}^{n} C_j} & I_{uli} = \dfrac{\sum_{j=0}^{n} i_j}{\sum_{j=0}^{n} I_j} \quad B_{uli} = \dfrac{\sum_{j=0}^{n} b_j}{\sum_{j=0}^{n} B_j} \end{cases} \tag{6}$$

The symbols are defined in Section III-A.

2) Load Imbalance: Load imbalance is defined by the following equation:

$$imbal(r) = \frac{\sigma^r}{\bar{w^r}} = \frac{\sqrt{\frac{1}{n}\sum_{j=0}^{n}\left(w_j^r - \frac{1}{n}\sum_{j=0}^{n} w_j^r\right)^2}}{\frac{1}{n}\sum_{j=0}^{n} w_j^r} \tag{7}$$

σ^r is the standard deviation of resource utilization in respect of r for all warehouses and w^{r} is the average resource utilization in respect of r for all warehouses. Other parameters are defined in Section III-A.

C. Experimental Results

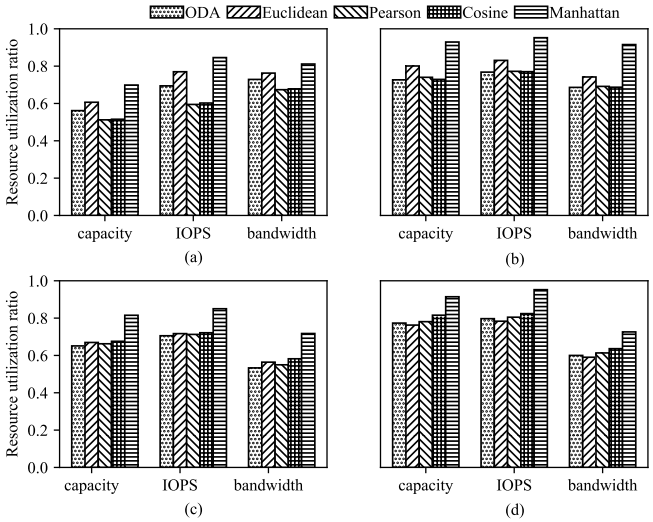

Fig. 5. Figure 5(a)-(d) show the comparison of total resource utilization when the number of allocated warehouses increases from 6 to 9.

1) Comparison of Resource Utilization: Fig. 5(a)-(d) shows the comparison of total resource utilization. When the number of allocated warehouses increases from 6 to 9, Manhattan distance performs stable and is superior to other methods. Compared to ODA, Manhattan distance improves about 13% to 27%, 14% to 26%, and 9% to 20% in capacity utilization, average IOPS utilization and average disk bandwidth utilization respectively.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.

2) Comparison of Load Imbalance: Fig. 6(a)-(c) shows the comparison of load imbalance for allocation policies in terms of sales capacity utilization, average IOPS utilization and average disk bandwidth utilization. When the number of allocated warehouses increases from 6 to 9, the perfomance of Manhattan distance is the most prominent. Compared to ODA, Manhattan distance reduces about 68% to 80%, 78% to 87%, and 45% to 87% in terms of the imbalance on sales capacity, average IOPS and average disk bandwidth respectively. The experimental results show that S-CDA is superior to other methods in all the respects.

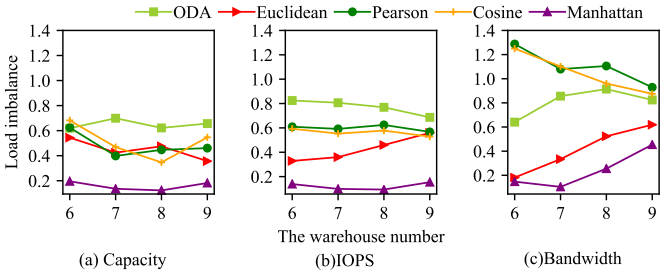

Fig. 6. Figure 6(a)-(c) show load imbalance comparison when the number of allocated warehouses increases from 6 to 9.

VI. Related Work

A. Using Machine Learning on Resource Allocation in Cloud

In recent resource scheduling researches employing machine learning and artificial intelligence technology to optimize resource scheduling in cloud is a hot topic in both academia and industry [4], [6]. And many works used clustering to analyze load features to realize resource scheduling and load balancing in cloud computing [7], [8]. Alam el. [7] analyzed the Google cluster data based on the job classification using k-means clustering. Rahman el. [8] present a detailed analysis of Google clustering data by the hierarchical clustering method.

B. Resource Allocation for VMs in Cloud

Resource utilization and load balancing of physical machines in cloud directly influence operational costs and benefits. Both academia and industry have focused on the research of the resource allocation to cut cost in cloud. In most studies, VMs resource allocation is often abstracted as a bin packing problem [17], [18]. Bin packing problem could be divided into offline packing and online packing according to whether all object information is known in advance or not. Main solutions to offline bin packing problems include various heuristic algorithms [19], [20].

Resource allocation for VMs in cloud are also divided into online and offline scenarios. In the offline scenario, there are many researches [11], [12]. But there are few researches on online multi-dimensional allocation for VMs. The most commonly used methods are classical bin packing, such as greedy FF, or Heuristic BF. Guo el. [17] designed weighted algorithms based on BF, so as to realize online multi-dimensional allocation of efficient resource utilization in the cloud environment. Yao el. [21] proposed the MVBFD algorithm by improving the BF algorithm, which was applied to the cloud storage resource scheduling.

VII. Conclusion

In order to meet the challenge that load features of new CD are unknown and the multi-dimensional allocation of new CD to warehouse is unattainable, we propose a Smart Cloud Disk allocation (S-CDA) policy. S-CDA uses the k-means method to cluster historic CDs and the decision tree model to classify new CD, in this way, the load info of new CD can be predicted and a multi-dimensional resource allocation policy based on Manhattan distance be realized. Experimental results show that S-CDA has improved all the aspects of Capacity/IOPS/Bandwidth utilization and reduced warehouse load-imbalance by varying degrees.

Acknowledgment

This work is supported by the Innovation Group Project of the National Natural Science Foundation of China No.61821003.

References

[1] K. Z. et al., "LEA: A lazy eviction algorithm for SSD cache in cloud block storage," in *ICCD*. IEEE, 2018, pp. 569–572.
[2] Z. J. et al., "Reasoning about block-based cloud storage systems," *CoRR*, 2019.
[3] J. et al., "Efficient cloud gaming resource provision via multi-dimensional bin-packing," *GSTF Journal on Computing (JoC)*, 2018.
[4] R. Y. et al., "Intelligent resource scheduling at scale: A machine learning perspective," in *SOSE*. IEEE, 2018, pp. 132–141.
[5] R. N. C. et al., "Workload prediction using ARIMA model and its impact on cloud applications' qos," *IEEE Trans. Cloud Computing*.
[6] C. D. et al., "Quasar: resource-efficient and qos-aware cluster management," in *ASPLOS*, 2014, pp. 127–144.
[7] M. A. et al., "Analysis and clustering of workload in google cluster trace based on resource usage," in *15th Intl Symposium on Distributed Computing and Applications for Business Engineering*, 2016.
[8] A.-R. et al., "Google users as sequences: A robust hierarchical cluster analysis study," *IEEE Transactions on Cloud Computing*, 2017.
[9] E. Alpaydin, "Introduction to machine learning," *MIT press*, 2014.
[10] M. T. et al., "A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers," *Neural Processing Letters*, pp. 211–221, 2015.
[11] M. A. K. et al., "Solving bin packing problem with a hybrid genetic algorithm for VM placement in cloud," in *19th KES*, 2015, pp. 1061–1069.
[12] S. R. M. A. et al., "Using the multiple knapsack problem to model the problem of virtual machine allocation in cloud computing," in *16th CSE*. IEEE, 2013, pp. 476–483.
[13] S. F. et al., "Balancing performances in online VM placement," in *ICT*, 2015, pp. 153–162.
[14] J. D. Ullman, "The performance of a memory allocation algorithm," 01 1971.
[15] D. S. Johnson, "Fast algorithms for bin packing," *J. Comput. Syst. Sci.*
[16] L. T. K. et al., "Multidimensional bin packing algorithms," *IBM Journal of Research and Development*.
[17] L. G. et al., "Energy saving and maximize utilization cloud resources allocation via online multi-dimensional vector bin packing," in *SDS*, 2018, pp. 160–165.
[18] A. et al., "Incorporation of weighted linear prediction technique and queuing theory for improving energy efficiency of cloud computing datacenters," in *LISAT*. IEEE, 2016, pp. 1–5.
[19] F. C. R. Spieksma, "A branch-and-bound algorithm for the two-dimensional vector packing problem," *Computers & OR*, 1994.
[20] A. C. et al., "Lower bounds and algorithms for the 2-dimensional vector packing problem," *Discrete Applied Mathematics*, pp. 231–262, 2001.
[21] Z. Y. et al., "Multi-dimensional scheduling in cloud storage systems," in *IEEE ICC*, 2015, pp. 395–400.

Authorized licensed use limited to: Huazhong University of Science and Technology. Downloaded on May 13,2023 at 16:36:37 UTC from IEEE Xplore. Restrictions apply.