Efficient Pricing of Cloud Platform Datacenters using Improved Data Replication Scheme

Sreelatha K
Computer Science And Engineering
Intenational School Of Technology And Sciences (affiliated to Jntuk)
Rajahmundry,India

***Abstract* - Data in the Cloud need to be stored with high efficiency and cost effectiveness while meeting the requirement of reliability. Data replication is a technique used by large-scale cloud storage systems to ensure high availability, reliability and consistency. A replication strategy that supports replica management in terms of replica creation, deletion and placement, which reduces both job execution time and network traffic, can be considered as a present consideration in this paper. In this proposed work we present an efficient pricing scheme in which two issues are addressed here as first profit of cloud service provider (datacenter) and user often contradict and second is the VM-Maintenance price overhead like start up time are considerably high. In addition we propose an efficient replication strategy based on the proposed models which results in improved Quality of Service (QoS) with reduced communication delays.**

Keywords - Cloud computing, IaaS, pricing scheme

I. INTRODUCTION

Business users can profit greatly from access to computer services through central controlled backups, high computer capabilities and flexible billing techniques globally as Cloud computing [1] is an internet-based computing platform that provides on-demand access to the virtualized computing resources like servers, storage, applications, software, computer networks, services etc. The efficient use of servers, data center energy planning, virtualization and optimized software stakes is leveraged by cloud computing. The virtual resources, both hardware and software are provided as on demand services to the end-users over Internet namely Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a service (IaaS) [2]. Modern distributed storage systems currently generally use data replication in order to support data reliability. In order to ensure data reliability, for example, data storage systems such as Amazon S3 [3], Google File System [5] and the Hadoop Distributed File System [4] all have a three-fold replication strategy, i.e., to store 3 data copies at once. A 3-replicate strategic approach implies that a gigabyte of data would require three gigabytes of data space and cost twice, which has a substantial impact on the cost-efficacy and safety of the cloud system.

There are generally two serious problems with the deployment and provision of the instances of VM over IaaS, refined resource allocation [6] and precise resource rental pricing [7]. There are generally two serious problems with the deployment and provision of the instances of VM over IaaS, refined resource allocation [6] and precise resource rental pricing [7]. Highly refined resource allocation is typically implemented by deploying VM instances and adjusting their resources on demand, affecting VMs' performance to complete the workload of customers. Accurate pricing is also known as Pay-as-you-go and includes several kinds of resources such as the CPU, memory and I/O [8]. Pricing is a key component of cloud computing because it directly impacts the profits of providers and the expenditure of customers [9]. In the IaaS environment, the design of a suitable pricing system to satisfy both providers and customers becomes an important concern. The smallest price unit in an on-demand instance in Amazon EC2, for example, is one hour. For short-term users such rough-grained hourly prices are likely to be economically inefficient. For example, users have to pay full-time, even if their jobs consume a small part (such as 15 minutes) of resources during their one-hour period. This phenomenon is called partial usage waste, which often appears as cloud jobs are generally quite short [10].

The storage space consumed by a replication strategy is important feature to be considered. Some of the strategies ensured reduced storage space consumption by maintaining an optimal number of replicas. It is found that there is no single strategy that addresses all issues involved in data replication.

II. RELATED WORK

In [11], the authors proposed an elastic replica management system known as ERMS, which would be based on HDFS for an active / standby data storage model. Live traffic data are categorized as hot or cold by means of the sophisticated event processing engine and on the basis of that classification replicas are generated dynamically. Replica of the hot data is increased and additional replicas of cold data are cleaned and deleted. ERMS categorizes storage nodes to active nodes. New hot data replicas are placed in the standby nodes if the standby nodes are used extensively and standby nodes can become better than active nodes. When cold data replicas are reduced, the ERMS need not rebalance replicas in standby nodes. The node is shut down to save energy after all data are removed in a standby node. ERMS dynamically adapts to changes in data access patterns and data popularity. In [12], the authors proposed a dynamic data replication strategy for D2RS which defines the connection between system availability and number of replica considering factors including size, access time and the likelihood of failure of every data. By analysing the access history, different accessed data can be weighted. The popular data are very important. If a dynamic threshold value passes the data popularity, the replication operation is triggered. In order to meet an effective system byte rate demand, the appropriate number of replicas is calculated. The new replicas are placed in data nodes and are performed in a balanced way by examining directly linked data center access information. The results of the assessment show that the proposed strategy increases the data available and reduces the bandwidth consumption of the cloud system. In [13], the authors proposed a proactive data management framework for the Hadoop cluster, based on predictive patterns of data activities. This process will forecast future data requests in the HDFS cluster using the Fourier Series Analysis [12] and calculate the replication factor dynamically in order to meet the availability requirements, improve efficiency and reduce stock usage. This framework maintains statistics on data access and transmits them as a prediction component for service-oriented time series. The prediction component creates prediction models based on previous access values and defines future user data. The replica of anticipated popular future data is increased and popularity is declining. Only limited replication scenarios are classified in this method file.In [14] the authors addressed an energy-efficient data center data reproduction system for cloud computing that refers to the bandwidth and energy consumption needed for data access. This scheme contains a topology of a three-tier tree data center that consists of a central database, local datacenter databases, each hosting datacenter and the hosting of racks. The Central DB comprises all the necessary data for cloud applications. A module regarded as the Replica Manager (RM), situated in the Central DB, regularly analyses data-access statistics to locate the most appropriate data items for replication and by which replication sites. DB is often used to replicate the most widely used Central DB data objects and Rack DB is used for later replication from DB DC. Access statistics and updates are also used to reduce system power and bandwidth consumption. In [15] the authors proposed HDFS adaptive replication management (ARM) on the grounds of controlled learning to increase the data location metric to ensure that data is highly available in HDFS. In this way the data files are replicated based on predictive analysis. For both the hyper parameter learning and training stages to increase efficiency of the prediction, a complexity reduction method for prediction technique was introduced. The popularity of every data file is predicted and significant potential files are replicated as erasure code for low-positive files is applied. The new replicas are placed on low-use nodes with low blocking probability to redirect tasks to the idle nodes and equilibrium the calculation. The results of the assessment show that this approach improves availability while maintaining reliability.In [16] the authors proposed a Dynamic Replication Strategy (DARS) of the Markov HDFS model which comprises a HRPS Dynamic Replica Address Strategy (DARS). This method first builds a transitional probability matrix on file access in a time period. DRS then evaluates the number of replicas to be allocated for each file based on the current distribution and transition probability matrix. The results data are categorized as hot or cold. Extra hot data replicas are created and cold data replicas are removed. The homogeneous replica placement strategy displays replicas across an HDFS cluster that considers the relationship between different types of data that places the relevant data on the same node or rack, thus reducing the data transmission time and bandwidth consumption among nodes or rack systems. This technique does not take into account the efficacious cold data management which leads to a probability of loss of data. The authors have suggested in [17] a weighted dynamic cloud replication policy in which data is dynamically replicated by classifying it into heat, warm or cold data. Each data item is assigned a weight based on its popularity in the system. A popularity index is calculated for each data item based on its number of accesses, weight and current factor of replication. Then, the data is classified as hot, warm, or cold in view of the weight of its popularity index. Replica factor for hot and warm data is dynamically calculated. A minimum factor for replicating cold data is set and erasure coding is applied in order to prevent loss of data. The evaluation results show that this strategy maintains an optimal replication factor that is sufficient to guarantee the data's reliability and availability. It also reduces storage costs by reducing the consumption of storage space.

III.PRELIMINARIES

A. *Resource pricing scheme*:

The existing classic IaaS cloud pricing schemes, and then analyze the partial usage waste issue, and finally formulate our optimized fine-grained pricing model by taking VM maintenance overhead into consideration. Our main objective is to satisfy both customers and providers, and reach maximum social welfare meanwhile. In recent times, the pricing schemes broadly adopted in IaaS cloud market can be categorized into three types: pay-as-you-go offer, subscription option and spot market. Under the pay asyou-go scheme, users pay a fixed rate for cloud resource consumption per billing cycle (e.g., an hour) with no commitment. On-Demand Instances are often used to run short-jobs or handle periodic traffic spikes. In the subscription scheme, users need to pay an upfront fee to reserve resources for a certain period of time (e.g., a month) and in turn receive a significant price discount. The billing cycles in the subscription scheme are relatively long compared to the pay-as you- go scheme, and can be

B. Fine-Grained pricing Scheme:

The aim of the work is coarse-grain pricing strategy and the unavoidable overhead VM maintenance. On the one hand, we strive to provide both users and CSPs with a cost range which is acceptable and to obtain the best price to satisfy both sides by maximizing total utility. Yet on the other hand, we hope that we will find a better billing cycle to maximize social welfare on both sides like win-win strategy.

That is, *PK* can be represented as Formula (1), where *I* indicates an increment cost (≥ 0).

$$P_k = \frac{P\left(k - T_0\right)}{60 - T_0} + I \qquad (1)$$

This increment price *I* can be used to not only cover providers' loss due to VM-maintenance overhead but also gain more profit for providers.

C. Statistics

As indicated in Fig. 1, in the hourly pricing, majority of users in both traces get low (< 20%) instance time utilizations, which implies a serious phenomenon of partial usage waste
The users are spilt into three groups (i.e., Low Utilization group, Medium Utilization group and High Utilization group) based on different levels of instance time utilization. Users' instance time utilization in Low group, Medium group, High group is [0, 20%), [20%, 80%), [80%, 100%] respectively. For example, 79.8% (42.4%) of users in DAS-2 (Google) get low (i.e., < 20%) instance time utilization.

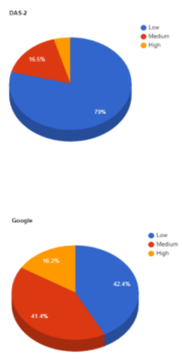


Fig. 1: Statistics of users' instance time utilization in the hourly pricing

IV. PROPOSED WORK

It is important to adjust the trade-off between adopting the refined pricing scheme and controlling the impact of extra overheads with decreasing length of billing cycles. Figure 2 presents the timeline of a workload execution in data center. It begins with the user request arrival at the datacentre gateway. After being scheduled it is forwarded through the data center network to the selected computing resource for execution. At the server, the workload can request data item if it is needed for its execution. For this, it queries a database and waits for the database reply to arrive. The database querying delay corresponds to the round-trip time and depends on the database location. As soon as the database reply is received, the workload execution is started. At the end of the execution, some workloads will send a modified data item back to the database for the update. As a result, the total delay associated with the workload execution in datacentres can be computed.

A. Optimized fine-grained pricing algorithm:

The total cost for user u with billing cycle = K-minutes can be written as Formula (2),

$$Cost_{u,K} = C_K \sum_{i=1}^{N} \left\lceil \frac{L_u\left(i\right)}{K - T_0} \right\rceil \qquad (2)$$

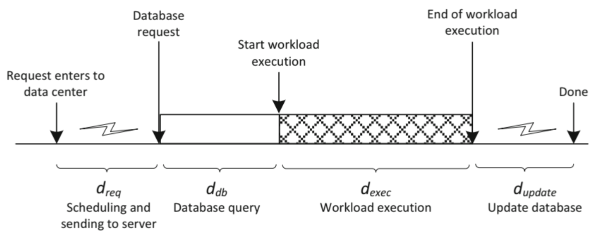


Fig 2

Apparently, indicates the cost of user u running all of his/her jobs in the classic coarse grained hourly pricing scheme. With the computation of , we can compute the maximum price that can be accepted by users (denoted as) per billing cycle based on Formula (3) (when billing cycle = K-minutes).

$$MaxP_{u,k} = \frac{P\sum_{i=1}^{N}\left[\frac{L_u(i)}{60-T_0}\right]}{\sum_{i=1}^{N}\left[\frac{L_u(i)}{K-T_0}\right]} \qquad (3)$$

The pseudo-code in computing the maximum prices for user u with the fine-grained billing cycle K-minutes is described in Alg. 1.

Algorithm 1: Computing maximum cost that can be there accepted by users

Input: The length Lu(i) of User us all jobs.

Output: The maximum social welfare and the best-t billing cycles BF.

```
Step 1: Calculate  for all of the user's jobs with hourly pricing.
Step 2: for all billing cycles K do
Step 3: For all billing cycles for all of u's jobs created on the fine-grained pricing calculate the social welfare WK for all users and the provider.
Step 4: end for
Step 5: Select the maximum social welfare when the billing cycle is K-minutes
```

B. Calculate Maximum User-accepted Price:

A cloud service provider like (Amazon, Google) has ***U*** different users. For a particular user ***u*** who has ***J*** jobs, we use $L_u(i)$ to denote the length of the user's ***i***th job running on some cloud instances. Specially, if two conjoint jobs (i.e., user's *ith* and *i*+1th job) submit in the same billing cycle, they are deemed as one job. When running a user's jobs in cloud with the fine-grained billing cycle = *K*-minutes, the unit price C_K is supposed to be higher than the original cost value calculated based on hourly pricing, as Formula (1) shows.

C. Calculate Minimum Provider-accepted Price:

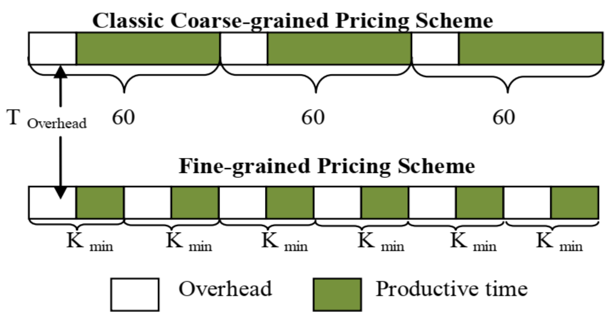


Fig. 3: Service time analysis between existing pricing schemes

In Coarse-grained pricing scheme [18] the maintenance of VMs become an erroneous, for example, initiating cost of the VMs. It is the burden for short-running job users. As depicted in Fig. 3 if the user running job time is lesser than productive time then at that time also user has to bare overhead cost. Fine-grained pricing scheme [19], majorly concentrate on short employment users. This is scheme mainly consists of three terms: time granularity, unit price, and resource bundle. Time granularity states that when pricing the resources consider the minimum length. The unit price is the price paid by the customer based on the time granularity for the consumption of resources. The resource package serves a customer with demanded task workloads.

For the same service period with length = Tservice, we use m and n to denote the number of billing cycles in the classic pricing scheme and that in our optimized fine-grained pricing scheme respectively. In the example shown in Fig. 3, m = 3 and n = 6. Then, it is easy to derive Equation (4) based on Fig. 3.

$$T_{service} = m \cdot 60 = n \cdot K \qquad (4)$$

On the other hand, the minimum price accepted by the provider is reached when the provider's total gains in the new pricing scheme is equal to the payment earned in the classic pricing scheme, i.e., Equation (5), where P refers to the price in the classic hourly pricing scheme.

$$m.p = n .MinPK \qquad (5)$$

Based on Equation (4) and Equation (5), we can get Formula (6).

$$MinP_K = \frac{K}{60}.P \qquad (6)$$

Together with Formula (1), we can get Formula (7).

$$I_{overhaed} = MinP_k - \frac{P(K-T_0)}{60-T_0} \qquad (7)$$

Obviously, I overhead is the increment price to overcome the overhead in our optimized fine-grained pricing scheme.

In this paper, we declare a job scheduling plan for the fine-grained pricing scheme to reduce the wasteful payment. Scheduling refers to the group of plans to manage the order of jobs executed by a computer system. The best scheduler adapts its scheduling scheme according to the environment and kind of task variation. Wasteful payment can be mitigated by ordering the user's jobs effectively. It can be alleviated by combining user's scheduling knowledge.

In this work partitioning of a data center in fig.4 takes place. Each partition allotted with some no of VMs, based the capacity of VMs and partition of the data center the jobs will execute. Controller will be there internally for giving the current status of the VMs and partitions.

Algorithm 2: Optimized User Job Scheduling (Improved Data replication scheme)

Inputs: Requests with user jobs

```
Outputs: Response time/state of servers, fine-grained cost
Step1: Read requests from the queue.
Step2:
1. begin
2. While request do
3. Search Best Server (request);
4. IF Server State==IDLE || server State==NORMAL
a. THEN send request to Server;
b. Processing request.
5. else
a. Update/Compute refresh period based on load.
 b. If all servers are OVERLOADED, wait for the server to NORMAL.
6. end if
7. end while
8. end
```

Setting an order of jobs is a specific piece of work which requires being allotting the system resources to the different tasks that are held back for the CPU time and issued in a queue. The scheme has to determine that for giving CPU time which job should take first for processing, in order to execute all the jobs in efficient and fair manner. Besides fairness in scheduling is the significant standard that furnishes the resource allotment in an optimal manner and improves efficiency.

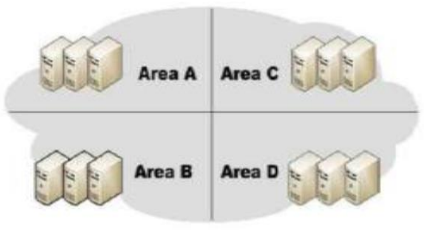


Fig. 4: partitioning data center

D. Extension work:

Geo-partitioning grants developers row-level replication control. By default CockroachDB lets us control which tables are replicated to which nodes. But with geo-partitioning, we can control which nodes house data with row-level granularity. This allows you to keep customer data close to the user, which reduces the distance it needs to travel, thereby reducing latency and improving user experience. To geo-partition a table:

- Define location-based partitions while creating a table.
- Create location-specific zone configurations.
- Apply the zone configurations to the corresponding partitions

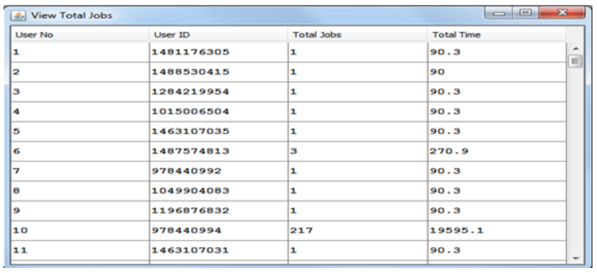
E. Results and discussions

Amazon S3 [18] offers two levels of services: “Standard Storage” which has 11 nines of storage availability for 0.03$ per Gigabyte per month, while “Amazon S3 Reduced Redundancy Storage (RRS)” has 4 nines of data availability for 0.024$ per GB per month. The simulated market is summarized in Table 1.

Providers	Price ($/GB/month)	Data availability
Prov 1	0.1	99.99999999 %
Prov 2	0.095	99.99999995 %
Prov 3	0.09	99,9999999 %
Prov 4	0.085	99,9999995 %
Prov 5	0.08	99,999999 %
Prov 6	0.075	99,999995 %
Prov 7	0.07	99.99999 %
Prov 8	0.065	99,99995 %
Prov 9	0.06	99,9999 %
Prov 10	0.055	99,9995 %
Prov 11	0.05	99,999 %
Prov 12	0.04	99,995 %
Prov 13	0.03	99.99 %
Prov 14	0.02	99.95 %
Prov 15	0.01	99.9 %

TABLE 1

FIG.5 SHOWS THE SIMILAR JOBS FROM GIVEN USER AS THEY ARE HAVING SAME TASK FOR EXECUTION SO THEY BELONGS TO SAME GROUP USERS. IN THE GIVEN TRACES WITH THE GIVEN USER ID 978440994 HAVE TOTAL OF 217 SIMILAR JOBS

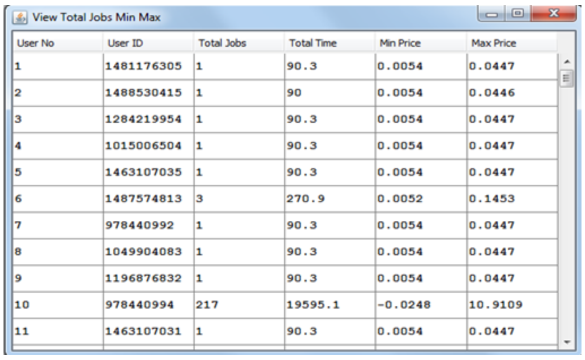

View Total Jobs

User No	User ID	Total Jobs	Total Time
1	1481176305	1	90.3
2	1488530415	1	90
3	1284219954	1	90.3
4	1015006504	1	90.3
5	1463107035	1	90.3
6	1487574813	3	270.9
7	978440992	1	90.3
8	1049904083	1	90.3
9	1196876832	1	90.3
10	978440994	217	19595.1
11	1463107031	1	90.3

FIG.5

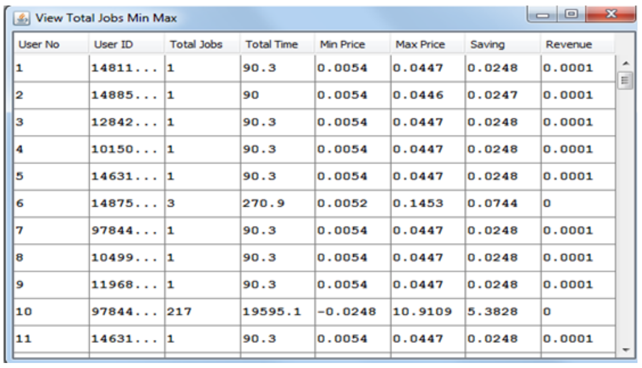
In fig.6 as it is mentioned the minimum cloud provider accepted cost in column 5th and maximum user accepted cost in column 6th. In the above calculation we can get the min price of provider and max price of user both are displayed.

FIG.6


View Total Jobs Min Max

User No	User ID	Total Jobs	Total Time	Min Price	Max Price
1	1481176305	1	90.3	0.0054	0.0447
2	1488530415	1	90	0.0054	0.0446
3	1284219954	1	90.3	0.0054	0.0447
4	1015006504	1	90.3	0.0054	0.0447
5	1463107035	1	90.3	0.0054	0.0447
6	1487574813	3	270.9	0.0052	0.1453
7	978440992	1	90.3	0.0054	0.0447
8	1049904083	1	90.3	0.0054	0.0447
9	1196876832	1	90.3	0.0054	0.0447
10	978440994	217	19595.1	-0.0248	10.9109
11	1463107031	1	90.3	0.0054	0.0447

IN FIG.7 AS IT IS SHOWN THE SAVING AND REVENUES OF THE TOTAL JOBS AND ITS CORRESPONDING PROVIDER MINIMUM AND USER MAXIMUM COSTS ACCORDINGLY.


View Total Jobs Min Max

User No	User ID	Total Jobs	Total Time	Min Price	Max Price	Saving	Revenue
1	14811...	1	90.3	0.0054	0.0447	0.0248	0.0001
2	14885...	1	90	0.0054	0.0446	0.0247	0.0001
3	12842...	1	90.3	0.0054	0.0447	0.0248	0.0001
4	10150...	1	90.3	0.0054	0.0447	0.0248	0.0001
5	14631...	1	90.3	0.0054	0.0447	0.0248	0.0001
6	14875...	3	270.9	0.0052	0.1453	0.0744	0
7	97844...	1	90.3	0.0054	0.0447	0.0248	0.0001
8	10499...	1	90.3	0.0054	0.0447	0.0248	0.0001
9	11968...	1	90.3	0.0054	0.0447	0.0248	0.0001
10	97844...	217	19595.1	-0.0248	10.9109	5.3828	0
11	14631...	1	90.3	0.0054	0.0447	0.0248	0.0001

FIG.7

IN TABLE 2. CONTENTS THE PERFORMANCE OF THE REDUCED TIME DURING THE EXECUTION OF USER JOBS.

IT ANALYSED THAT THE PARTITIONING DATA CENTER AND SCHEDULING USER JOBS TECHNIQUE OUT PERFORMANCE THE EXISTING METHOD BY ALLEVIATING THE WASTEFUL PAYMENT. IT EXPLICITLY SHOWS THE TIME REDUCTION DURING JOBS EXECUTION. IN THIS WAY, JOBS CAN EXECUTE IN LESS TIME, AND PAYMENT CAN REDUCE.

TABLE 2

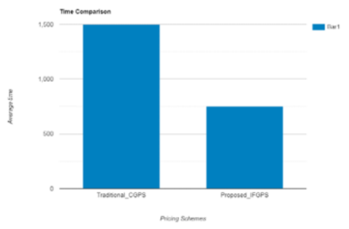


fig:time comparision for course grained pricing scheme and fine grained pricing scheme

V CONCLUSION

This whole paper takes the first step in identifying and examining the partial application of waste problem in cloud computing by analysing its importance with real world traces. In order to solve the partial usage waste issue, we propose efficient pricing with regard to the unavoidable VM overheads and find a best fit billing cycle to maximize social welfare. Our study has provided a clear option for the cloud storage scheme that prioritizes cost. Also it is shown that the different pricing schemes and work done in the perspective of short-running jobs; it includes some favour to the providers also. We evaluated an optimized fine-grained pricing scheme with scheduling jobs effectively. This work reduces the wasteful payment in the cloud.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their insights and comments that help us improve the quality of this paper.

REFERENCES

[1] [1] Buyya, R., et al., Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst., 2009. 25(6): p. 599-616.
[2] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.
[3] Amazon. (2008). Amazon simple storage service (Amazon S3). Available: http://aws.amazon.com/s3
[4] D. Borthakur. (2007). The Hadoop distributed file system: Architecture and design. Available: http://hadoop.apache.org/common/docs/r0.18.3/hdfs_design.html
[5] S. Ghemawat, H. Gobioff, and S. Leung, "The Google file system," in ACM Symposium on Operating Systems Principles pp. 29 - 43, 2003.
[6] S. Di and C.-L. Wang, "Error-tolerant resource allocation and payment minimization for cloud system," IEEE Transactions on Parallel and Distributed Systems (TPDS), vol. 24, no. 6, pp. 1097–1106, 2013.
[7] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "The resource-as-a-service (RAAS) cloud," in Proceedings of the 4th USENIX conference on Hot Topics in Cloud Computing (HotCloud'12), 2012, pp. 12–12.
[8] CloudSigma.[Online].Available:http://www.cloudsigma.com/

User Id	Total no.of jobs	Total Time	Max.User-accepted Cost	Min.Provider-accepted Cost	Optimal Cost
1	1	90.3	0.0447	0.0054	0.0248
2	1	90.3	0.0447	0.0054	0.0248
3	1	90.3	0.0447	0.0054	0.0248
4	1	90.3	0.0447	0.0054	0.0248
5	1	90.3	0.0447	0.0054	0.0248
6	3	270.9	0.1453	0.0052	0
7	1	90.3	0.0447	0.0054	0.0248
8	1	90.3	0.0447	0.0054	0.0248
9	1	90.3	0.0447	0.0054	0.0248
10	217	19595.1	10.9109	0.0248	0

[9] H. Xu and B. Li, "Dynamic cloud pricing for revenue maximization," IEEE Transactions on Cloud Computing (TCC), vol. 1, no. 2, pp. 158–171, 2013.
[10] S. Di, D. Kondo, and W. Cirne, "Characterization and comparison of cloud versus grid workloads," in International Conference on Cluster Computing (CLUSTER), 2012, pp. 230–238.
[11] Cheng, Zhendong, Zhongzhi Luan, You Meng, Yijing Xu, Depei Qian, Alain Roy, Ning Zhang, and Gang Guan. (2012) "Erms: An elastic replication management system for HDFS." in Cluster Computing WorkshopS (CLUSTER WORKSHOPS), 2012 IEEE International Conference on, IEEE: 32-40.
[12] Sun, Da-Wei, Gui-Ran Chang, Shang Gao, Li-Zhong Jin, and Xing-Wei Wang. (2012) "Modeling a dynamic data replication strategy to increase system availability in cloud computing environments." Journal of computer science and technology 27 (2): 256-272.
[13] Kousiouris, G., Vafiadis, G., & Varvarigou, T. (2013) "Enabling proactive data management in virtualized hadoop clusters based on predicted data activity patterns." In P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Eighth International Conference on IEEE: 1-8

[14] Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., & Zomaya, A. Y. (2015). “Energy-efficient data replication in cloud computing datacenters.” Cluster computing, 18(1), 385-402.
[15] Bui, Dinh-Mao, Shujaat Hussain, Eui-Nam Huh, and Sungyoung Lee. (2016) “Adaptive replication management in HDFS based on supervised learning.” IEEE Transactions on Knowledge and Data Engineering28 (6): 1369-1382.
[16] Qu, Kaiyang, Luoming Meng, and Yang Yang. (2016) “A dynamic replica strategy based on Markov model for hadoop distributed file system (HDFS).” in Cloud Computing and Intelligence Systems (CCIS), 2016 4th International Conference on, IEEE: 337-342.
[17] S. Gopinath and E. Sherly, "A Dynamic Replica Factor Calculator for Weighted Dynamic Replication Management in Cloud Storage Systems", Procedia Computer Science, vol. 132, pp. 1771-1780, 2018.
[18] Y.-J. Hong, J. Xue, and M. Thottethodi, “Dynamic server provisioning to minimize cost in an IAAS cloud,” in Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems (SIGMETRICS), 2011, pp. 147–148.
[19] “Optimal resource rental planning for elastic applications in cloud market”, in IPDPS, 2012, pp. 808–819 by H. Zhao, M.Pan, X. Liu, X. Li, and Y. Fang.