International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-2, December, 2019

Choosing the best Bin Packing Algorithm for Replica Placement in Multi-Tenant Cloud System

Debabrata Sarddar, Sougata Chakraborty

Abstract*: *Of late, Cloud Computing is visibly seen to reduce infrastructure costs with high data availability and performance conforming to service level agreement for both the service providers and the users. With the rapid and explosive growth of the number of cloud users, Cloud Data Management System must serve an array of different analytical and transactional workloads. Hence, to ensure the scalability in a multi-tenant system, replica placement algorithms always come into the picture appropriately. In our work, we have vividly analyzed various replica placement algorithms in terms of their performance and tried to find the beneficial aspects to be the fittest one to tackle the situation when the actual observed workloads are immensely deviated from the estimated workloads.

Keywords*: *DBaaS, Replica Placement, Bin Packing, Multi-tenant System

I. INTRODUCTION

With the predominant implementation of cloud computing, nowadays, many service providers have started offering Database-as-a-Service (DBaaS) [1], [2] to their customers, such as Amazon Relational Database Service (RDS), Microsoft SQL Azure, salesforce.com, database.com, Xeround cloud database, Oracle cloud etc. [3]. Primarily, DBaaS is a paradigm where the end users can request database services, consume it for one's own purpose as per the contractual agreement and then the service is automatically de-provisioned and returned to the resource pool. DBaaS provides flexible resource offering and pricing model. Moreover, multi-tenancy is one of the prime features of DBaaS. Multi-tenant data management is a key application of Software as a Service (SaaS) in advanced form, where a third-party service provider hosts database as a service and provides its customers as on-demand services. SaaS applications are deployed on a shared environment that can be accessed by the users from client-end software via Internet. Multi-tenancy is a software architectural principle where a single instance of the software runs on a server, serving multiple tenants as and when required. Multiple tenants can interact with the main software with a common user interface (UI) and they can save data in a single database to reduce the total infrastructural cost of ownership.

Revised Manuscript Received on December 30, 2019.

* Correspondence Author

Debabrata Sarddar*, Assistant Professor, Department of Computer Science and Engineering, University of Kalyani, West Bengal, India.

Sougata Chakraborty, Senior System Engineer, IBM India Private Limited, Kolkata, West Bengal India.

Nowadays, within an organization, a data center can have thousands of individual database management systems. Since, thousands of Database Management instances are deployed in its internal infrastructural backbone of a large-sized organization and each database is deployed on a dedicated server. Very naturally, it is always unexpected that all the Database Management instances within a data center will be hit by peak loads at the same time. Occasionally, the number of tenants may be highly increased at any point of time. However, it is the biggest challenge to the DBaaS providers to deal with the phenomenon when the actual workload is very high compared to the estimated workload [4], [5]. Because of this challenge, to get the most feasible solution, we are to find the best strategy among all the replica placement algorithms which can ensure the optimal scalability in a multi-tenant cloud system and the Quality of Service (QoS) of end-users with the minimized cost. Categorically, the online replica placement algorithm is chosen to find out the placement for the replicas of a given tenant whenever, a tenant arrives in the system, without having a priori knowledge of the workloads of the entire system. An offline replica placement algorithm is completely different in its working approach [5]. Hence, the online replica placement algorithm is more likely and realistic approach to overcome the hurdles pertaining to the performance issue in multi-tenant cloud system. While studying all the Replica Placement algorithms we have vividly analyzed the performance of bin packing algorithms so that how efficiently replicas can be placed ensuring the optimal scalability in a multi-tenant cloud system and quality of service of end users with the minimized cost.

II. LITERATURE SURVEY AND REVIEW

There are several research activities have been published on the various aspects of Database Management System in Cloud. In [1], we have studied on Oracle Database Cloud Service. In [2], we have studied on Database as a Service (DBaaS) using Oracle. In [3], we have seen the intelligent database placement in cloud environment. In [4], the research work has been carried out on the database workload management. The research work pertaining to the high-performance cloud data management has been published in the paper [5]. In [6], we have seen the necessity of the cloud DBMS which is designed explicitly for cloud environments. Authors explained here the architecture of the cloud DBMS. Authors in [7] described the data management system to improve security and availability in Cloud Storage.

We have gone through the research work on the several aspects of Bin Packing Algorithm in [8], [9], [10] and [11]. In [12], a survey on Replica Server Placement Algorithms for Content Delivery Networks has been carried out by the authors thoroughly. A comparative study has been done on heuristic procedures in [13] to solve Bin Packing Problems.

III. OVERVIEW OF BIN PACKING ALGORITHM

The Bin Packing Problem is NP Complete. More illustratively, we can consider the classical bin packing problem as following,
We are given a set $I = \{1, \dots, n\}$ of items, where item $i \in I$ has size $si \in (0,1)$ and a set $B = \{1 \dots n\}$ of bins with capacity one. Find an assignment $a : I \rightarrow B$ such that the number of non-empty bins is minimal. However, we can write, $S(J) = \sum_{j \in J} S_j$ for any $J \subseteq I$.
First Fit Bin Packing algorithm says to scan the bins according to the order and place the new item in the first bin that is large enough to hold it.
A new bin is created only when an item does not fit in the previous bins. Let N be the total number of bins required to fit all items and N_0 be optimal number of bins required so mathematical derivation of bin packing algorithm,

$$a_1 + a_2 + \dots + a_i > 1$$
$$a_i + a_{i+1} + \dots + a_j > 1$$
$$\dots$$
$$a_l + a_{l+1} + \dots + a_m > 1$$

$$If, a_1 + a_2 + \dots = \sum$$

$$then,\ 2\sum > N - 1$$
$$thus,\ 2\sum \geq N$$
$$implies, N_0 \geq \sum \geq \frac{N}{2}$$
$$\frac{N}{N_0} \leq 2$$

Next Fit Bin Packing algorithm says if the current item fits in the current bin. If so, then place it there, otherwise start a new bin. Again, according to Best Fit Bin Packing algorithm a new item is placed in a bin where it fits the tightest. If it does not fit in any bin, then start a new bin.

The Next Fit algorithm works as follows:
1. Initially all bins are empty, and we start with bin $j = 1$ and item $i = 1$.
2. If bin j has remaining capacity for item$i$, assign item i to bin j, i.e., $a(i) = j$, and consider item $i + 1$.
3. Otherwise consider bin $j + 1$ and item i.
4. Repeat step 1 to 3 until item n is assigned.

Now, let N be the number of non-empty bins in the assignment a found by Next Fit. Let N_0 be the optimal number of bins. We show the slightly stronger statement that $N \leq 2N_0 - 1$. New item is placed in a bin where it fits the tightest. If it does not fit in any bin, then start a new bin. If optimal algorithm uses M bins, then Best Fit uses at most ~ *1.7M*.
It can be shown that

$$FF(N)/N_0 \geq 17N/(10N + 17) > 1.72/N_0$$
$$BF(N)/N_0 \geq 17N/(10N + 17) > 1.72/N_0$$

IV. PERFORMANCE ANALYSIS OF ALGORITHM

The asymptotic approximation ratio for next fit is less than or equal to 2 which is 1.7 in case first fit and best fit algorithm. The next fit algorithm runs in linear time i.e. $O(n)$ whereas first fit and best fit algorithm runs in logarithmic time i.e. $O(n \log n)$.

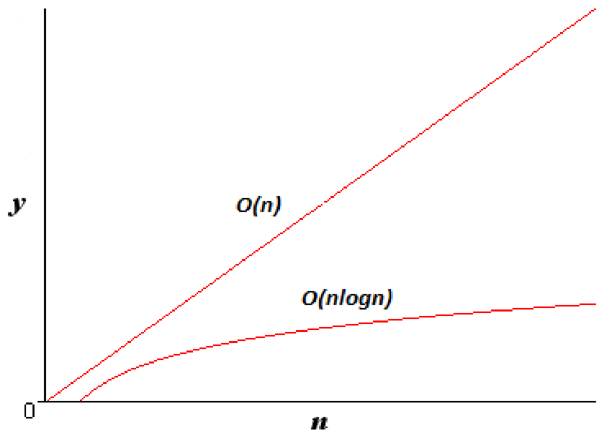

Fig. 1.Performance comparison between *O(n)* and *O(log n)*

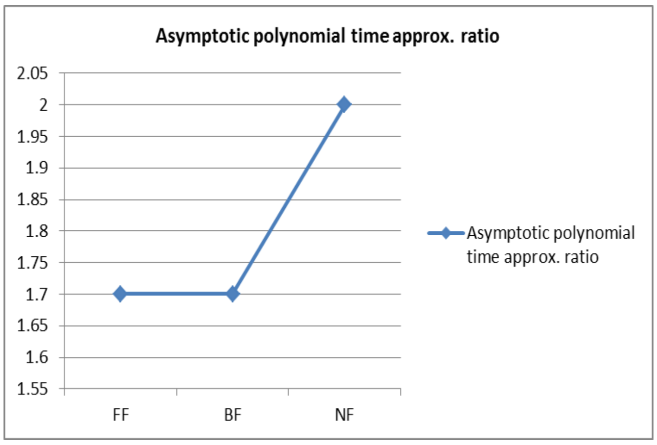

Fig. 2.Asymptotic polynomial time approximation ratio

V. RESULT

We have conducted our experiment with a suitable program which can identify whichever Bins are allocated or not allocated based on the size of the Bin on completion of the algorithms case wise.
Here, we have taken the following Bin size and Item size as the input.

Case 1: First Fit

Input: Bin Size [] = {210, 450, 300, 420, 675};
Item Size [] = {235, 578, 105, 426};

We have found some of the items are allocated in the bin after applying 'First Fit Algorithm' and at the same time we have identified which item is not allocated on completion of the algorithm.
Output:

Table- I: Demonstration of the First Fit algorithm

Number of Items	Size of the Item	Allocated Bin or Not Allocated
1	235	2
2	578	5
3	105	2
4	426	Not Allocated

Here, we have taken the following Bin size and Item size as the input.

Case 2: Best Fit

Input: Bin Size [] = {210, 450, 300, 420, 675};
Item Size [] = {235, 578, 105, 426};
We have found all the items are allocated in the bin after applying 'Best Fit Algorithm'.

Output:

Table- II: Demonstration of the Best Fit algorithm

Number of Items	Size of the Item	Allocated Bin or Not Allocated
1	235	3
2	578	5
3	105	1
4	426	2

Here, we have taken the following Bin size and Item size as the input.

Case 3: Worst Fit

Input: Bin Size [] = {210, 450, 300, 420, 675};
Item Size [] = {235, 578, 105, 426};

We have found some of the items are allocated in the bin after applying 'Worst Fit Algorithm' and at the same time we have identified some of the items are not allocated in the bin on completion of the algorithm.
Output:

Table- III: Demonstration of the Worst Fit algorithm

Number of Items	Size of the Item	Allocated Bin or Not Allocated
1	235	5
2	578	Not Allocated
3	105	2
4	426	Not Allocated

Here, we have taken the following Bin size and Item size as the input.

Case 4: Next Fit

Input: Bin Size [] = {210, 450, 300, 420, 675};
Item Size [] = {235, 578, 105, 378};
We have found all the items are allocated in the bin after applying 'Next Fit Algorithm'.
Output:

Table- IV: Demonstration of the Next Fit algorithm

Number of Items	Size of the Item	Allocated Bin or Not Allocated
1	235	2
2	578	5
3	105	1
4	378	4

Here, we have taken the following Bin size and Item size as the input.

Case 5: First Fit Decreasing

Input: Bin Size [] = {210, 450, 300, 420, 675};
Item Size [] = {235, 578, 105, 378};
We have found the all the items are allocated in the bin after applying 'First Fit Decreasing Algorithm'.
Output:

Table- V: Demonstration of the First Fit Decreasing algorithm

Number of Items	Size of the Item	Allocated Bin or Not Allocated
1	235	5
2	578	4
3	105	3
4	378	1

VI. CONCLUSION

We have analyzed various Bin Packing Algorithms here in our research work. Although, Bin Packing Algorithm is truly a heuristic technique, it is guaranteed that this problem-solving methodology cannot produce the full-proof optimal result to find the optimal scalability as a best replica placement process in a multi-tenant cloud system, still the Best Fit Algorithm is said to be chosen as the best one in this context from our derived and proved experimental result.

Further research work can be carried out for the speed up factor of the Bin Packing Algorithm by proposing a modified algorithm which could make the algorithm faster. In that case, we can analyze and compare the overall Throughput of the modified algorithm versus the existing one.

ACKNOWLEDGMENT

I would be thankful to my research advisor Asst. Prof. Dr. Debabrata Sarddar for his extensive support at every phase of my work.

REFERENCES

1. An Oracle White Paper, Oracle Database Cloud Service, May (2012).
2. An Oracle White Paper, Delivering Database as a Service (DBAAS) using Oracle Enterprise Manager 12 C, October (2013).
3. Yu1 T.; Qiu J.; Reinwald B.; Zhi L.; Wang Q.; Wang N.; Intelligent Database Placement in Cloud Environment, IEEE 19th International Conference on Web Services, IEEE, 545-551 (2012).
4. Babu S.; Graefe G.; Kuno A.H.; Database Workload Management, ACM, 2(7), 73-91(2012).

5. Floratou A.; High-Performance Cloud Data Management, University of Wisconsin–Madison (2013).
6. Januzaja Y.; Ajdaria J.; Selimia B.; DBMS as a Cloud service: Advantages and Disadvantages. World Conference on Technology, Innovation and Entrepreneurship, Procedia - Social and Behavioral Sciences, 195, 1851 – 1859(2015).
7. Banyal K. R.; Jain K. V.; Jain, P.; Data Management System to Improve Security and Availability in Cloud Storage, International Conference on Computational Intelligence & Networks, 125-129 (2015).
8. Johnson S. D.; Fast Algorithms for Bin Packing. Journal of Computer and System Sciences, 8, 272-314 (1974).
9. Zehmakan N. A.; Bin Packing Problem: Two Approximation Algorithms. International Journal in Foundations of Computer Science & Technology, 5(4) (2015).
10. Dosa G.; Sgall J.; Optimal analysis of Best Fit bin packing. International Colloquium on Automata, Languages, and Programming, 429-441 (2014).
11. Korte B.; Bin Packing, Combinatorial Optimization, Springer-Verlag Berlin Heidelberg, 407-408 (2000).
12. Sahoo J.; Salahuddin M. A. ; Glitho R.; Elbiaze H.; Ajib W.; A Survey on Replica Server Placement Algorithms for Content Delivery Networks, IEEE Communications Surveys and Tutorials.
13. Yesodh R.; Amudha T; A Comparative Study on Heuristic Procedures to solve Bin Packing Problems, International Journal in Foundations of Computer Science & Technology (IJFCST), Vol. 2, No.6, November 2012.

AUTHORS PROFILE

Debabrata Sarddar, is an Assistant Professor in the Department of Computer Science and Engineering, University of Kalyani, West Bengal, India. He obtained Ph.D. from Jadavpur University. He received M.Tech. in Computer Science & Engineering from DAVV, Indore in 2006, and B.E. in Computer Science & Engineering from NIT, Durgapur in 2001. He published more than 200 research papers in various journals and conferences. His research interest includes wireless and mobile system and Cloud computing.

Sougata Chakraborty, is a Senior System Engineer at IBM India Private Limited in Kolkata, He received M.Tech. in Computer Science & Engineering from Jadavpur University in 2011. He also received B.Tech. in Information Technology from Murshidabad College of Engineering & Technology under West Bengal University of Technology in 2008. His research interest includes Cloud Computing and Mobile Computing.