World Journal of Engineering Research and Technology
WJERT
www.wjert.org
SJIF Impact Factor: 5.924

A BENCHMARK STUDY FOR MCMC AND OPTIMISATION (MCBO) MODEL ON CLOUD APPLICATIONS

***Aws Ismail Abu Eid and Mohammad Ibraigheeth**

Faculty of Science, Computer Science Department, Northern Boarder University,1321 Arar, Saudia Arabia.

Article Received on 24/01/2022 **Article Revised on 14/02/2022** **Article Accepted on 06/03/2022**

***Corresponding Author**
Aws Ismail Abu Eid
Faculty of Science, Computer Science Department, Northern Boarder University, 1321 Arar, Saudia Arabia.

ABSTRACT

The Markov Chain Monte Carlo and Optimization MCBO model is a strategy designed to reduce the Time taken for high data availability to access replicas in the cloud environment. This study provides a new model MCBO as a new approach to defining an optimal path for solving one of the most common replication-related problems in the cloud: time-consumption in the cloud environment. Besides, MCBO output is further aided by the process of providing optimal results. This work developed a dynamic replication strategy capable of improving efficiency by reducing time in cloud replicas' distributed environment. The Markov Chain Monte Carlo and Optimization MCBO model is a strategy designed to reduce the Time taken for high data availability to access replicas in the cloud environment. In general, replicas' development is activated in two situations: a certain number of replicas is not achieved, or a goal of unsatisfactory response time is not achieved.

KEYWORDS: Data replication, Cloud environment, MCBO model, Bat algorithm.

1. INTRODUCTION

In the current environment, the wide-ranging and large-scale convergence of Internet resources and big data has made the cloud the perfect solution to the growing demands for extended storage to its provision of limitless space, fast availability, and rapid access time. In today's technological and academic sectors, the cloud computing model itself is highly popularized, drawing scholarly interest due to the prospect of comprehensive gains for both

the business and the community.[1] For cloud providers, specific users' use of resources can be carried out after their availability, leading to their improved usage.[2] That enables several large-scale data replicas to be set up in geographically dispersed areas. Following this, data replication is established as an efficient solution to fault tolerance provisions, optimized end-user bandwidth, and reduced time consumption and resource sharing. Replica management has now developed as a challenging area for these providers.

Cloud computing is typically an excellent up-and-coming infrastructure that provides infrastructure, communication, and storage over a network. Because of lacking connectivity resources, various cloud systems and their service providers are caught in gridlock. This allows a beneficial approach to data replication as it puts data (e.g., databases) closer to users (e.g., cloud applications)[3] As well as alleviating difficulties with time delays and use of bandwidth. As a result, there is a significant requirement for distributed storage due to the massive quantity of data handled and disseminated by web application providers in the provisioning of multiple tenants.[4] Contextually, data replication is considered a common strategy that, by distributing various data replicated around different locations, ensures availability and performance.[5] It increases the possibility that at least one copy will be available in the event of losses and in the context of different goals, such as reducing storage costs and increased error tolerance and access delays.[6] Replication is, thus, unsurprisingly, a key component in cloud computing applications in which a multitude of customers use the services to access their data at a low bandwidth rate from different locations. Data replication in cloud systems has been identified in numerous works in the literature.[7,8] In order to achieve an improved level of data availability and load balancing[4], improved performance[9,10], and reduced bandwidth consumption[11], such methods have been implemented. However, the outlined objectives are seen to be discordant: data replication, for example, ensures their availability, but this can occur at the cost of inter-site connectivity. This results in a network that is overwhelmed, thereby affecting efficiency. Also, a majority of the strategies dismiss both replications cost and provider profit. As such, this work design applies and positions the technique of the Markov Chain Monte Carlo MCMC in the production of random samples (paths) in a cloud environment is needed to execute the outcome assessment to use a one-test-at-a-time meta-heuristic algorithm (BAT-algorithm). This study provides an optimal path for solving one of the most common replication-related problems in the cloud, using MCBO as a new approach. As such, this work design applies and positions the Markov Chain Monte Carlo MCMC in the production of random samples

(paths) in a cloud environment is needed to execute the outcome assessment to use a one-test-at-a-time with meta-heuristic algorithm (BAT-algorithm). This study provides a new model MCBO as a new approach to defining an optimal path for solving one of the most common replication-related problems in the cloud: time-consumption in the cloud environment. Besides, MCBO output is further aided by the process of providing optimal results.

The following structure structures this study: section two reviews the correlated studies carried out concerning replica location and cloud platforms. In contrast, the resulting section outlines the proposed answer to the problem. Section four then presents the results of the simulation obtained when evaluating the approach and its performance. Finally, section five offers a thorough conclusion.

1LITERATURE REVIEW AND RELATED WORK

Research authors have previously located approaches in cloud systems for data replication. For example, to achieve reduced data movement across resource-limited links, Kloudas et al.[11] have detailed a scheduler referred to as Pixida. This is accomplished by introducing Silo, which is a new abstraction method crucial as a graph partitioning issue for modeling the scheduling goals of the Pixida. Also, the authors disclose that the pre-existing problem formulations for graph partitioning are not mapped to the way significant data jobs operate, thereby making the solutions incongruent with data movement prevention opportunities. Consequently, a new problem of graph partitioning has been developed and a new algorithm is suggested as a solution. In comparison with current schedulers, Pixida has been further incorporated into the Spark in which preliminary results show that it achieves traffic reduction by up to nine times using the links. Tos, U., Mokadem, R., Hameurlain, A., Ayav, T., & Bora, S., meanwhile.[9] developed a data replication strategy aimed at providing tenants with adequate performance assurance while ensuring the viability of cloud providers at the same time. This approach provides the approximate response time for any problems and the costs influencing the profitability mentioned above. Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., & Zomaya, A. Y. In another case,[3] Models have been developed as a solution to the need for energy consumption and bandwidth posed by database access in cloud data centers. In addition, according to the models, the work has detailed an efficient energy replication strategy, resulting in an improved Quality of Service (QoS) and reduced contact delays. The resulting evaluation created by detailed simulations revealed tradeoffs in

performance and energy efficiency, which subsequently served as guidance in the design of upcoming solutions for data replication.

Next, Bonvin et al.[12] detailed a dispersed key-value store called Skute in which autonomous agents operate in virtual nodes, making decisions without external control to the benefit of data owners. The economic model put in place here revolves around a virtual economy. With regard to storage usage and query load, these nodes will make rental payments to other nodes for replica hosting. Moreover, according to the number of queries answered, they are revenue-generating in nature. The placed study took into account the availability initially followed by the net profit according to their financial fitness through replica placement to nodes. In Skute, performance guarantees are not considered to be a key component of the SLA implemented in it, but over Time, a reduction in the average query load per node is recorded and obtained.

A SLA-focused provisioning approach for cloud databases, which is consumer-centric in nature, has been outlined in Sakr and Liu's work.[13] Here, according to the SLA specifications, the database servers are subject to scaling in and out; the total execution time of transactions is therefore chosen as the primary SLA target for decision-making processes. This approach includes near cloud system control, while cloud providers define application-specific rules comprehensively to ensure adaptive resource scaling. While the advantages of SLA-aware provisioning for scaling are known, cloud provider replication's economic effects have not been detailed.

Janpet and Wen[14] have suggested a data replication strategy to reduce the Time to access data, which is accomplished by finding the shortest route to access data objects. In order to yield the node that is the nearest and most suitable for replica placement, the work has modeled the access frequency, delay, and replication budget. In particular, in order to ensure a regulated number of replicas, the replication budget is predefined in nature and merely enforced as a limiting factor for users. The work did not, however, discuss the economic connection between users and cloud providers in detail. However, the work has shown that a closer location between data objects and high access frequency nodes increases the response time.

Next, by Kumar et al.[15], which is a workload-aware data placement and replica selection scheme, SWORD was detailed. A novel metric called query span, which is the average

number of nodes implemented for query execution, has been placed in the work. This approach aims to reduce the span of the query to achieve decreased overhead contact, resource usage, energy footprint, and transaction costs. Thus, the authors have reported that SWORD manages performance degradation by data repartitioning increments. In addition, supplier benefit is not emphasized in this work, but the study's efficacy has been demonstrated through an experimental review performed to assess query duration and transaction times.

In the meantime, in implementing a replica placement strategy, Zhang et al.[16] have depicted an auction model aimed at meeting the availability aspect in a large-scale cloud storage environment. If the required degree of availability is not maintained, the location of a new replica is decided through the retention of the bid. The bidding price depends on various node properties, such as the probability of failure, network bandwidth, and availability of space. The objective feature does not incorporate the response time, but the authors documented improved performance and satisfactory availability during the experiments.

In addition, replica, which is a method for elastic multitenant database replication, has been offered by Sousa and Machado.[17] It considers SLA output and makes elastic changes to the number of replicas by studying the method's implementation. Any adjustments and differences in the workload can be handled by targeting the transactions with appropriate resources against replicas. A replica has been juxtaposed by experimental work with a rule-based scaling system, which results in the technique reaching the QoS satisfactorily with minimal SLA breaches.

A data replication approach with an emphasis on improving the energy efficiency of cloud datacenters was described by Boru et al.[18] Here, at the inter-data centre and intra-datacenter levels, the strategy advances energy consumption, bandwidth implementation, and network delays. In this work, the data center's power usage and bandwidth consumption of database operations are modeled accordingly, whereby recurring tests are carried out when deciding the replication decision. In future years, they can also estimate the power and bandwidth utilization of the replicas. Therefore, a simulation study carried out showed that a closer placement between replicas results in better power consumption and reaction time. Regardless, there is no focus on economic incentives.

2. PROPOSED APPROACH

The method proposed in this work is referred to as the MCBO model, which is subjected to four stages during the design phase to achieve the goals of reduced time consumption when cloud replica updates are required. This is shown clearly in Figure 1, however, it displays all four phases and their processes as follows:

1. To produce more random walks (i.e., more iterations), implement the MCMC Algorithm (see Algorithm 1). It is necessary to categorize a random walk as a mathematical object, otherwise known as a probabilistic or random process that defines a route consist of sequential, random moves on specific mathematical spaces.

2. The results of random walks (i.e., paths), start location and Time. for each iteration generated from the MCMC algorithm obtained to cover all locations in the blind environment (Phase 1). Software tools for managing such data, such as Excel, can be used.

3. Generate the MCBO test suite, which implements the BAT Algorithm BA to enhance the solution as the enhanced algorithm and use the result of phase 1 as input for BAT echo to define the optimal path. This depends on the starting location and the Time taken by the MCMC algorithm.

4. In a distributed cloud environment, the MCBO strategy will define the optimal path.

4. Experiments

The scale and the limits used in this work could be established by determining the starting position for the MCBO model. The scale of 1:100 was then introduced, symbolic of every 1 being equal to 100 kilometers on the ground. Furthermore, by operating on the first quarter (x, y) of the Cartesian coordinate system or Universal Transverse Mercator (UTM), the avoidance of a coordinate X or y negative signal was achieved. Our scaling boundary from 0 to 100 means we function in a positive field: distribution method x Cartesian level axis and y axis (0,100). Next, using the BAT algorithm to determine the optimal path or variables (i.e. start position, Time (Steps)) could proceed to the model's corresponding phase. This allowed the model efficiency to be calculated to reduce time and resource usage while using the cloud replicas' update operation.

Start Position: In the first iteration, the first location value is randomly defined.

Time (Steps): The number of steps observed from the initial position to the end of the domain boundary during the model movement.

Two main goals are the experimental assessment of MCBO

1. Characterization of the MCMC's results concerning the BA applied.
2. BA against other competitive strategies to measure.

- In the first section, MCMC produces the Friedman test for candidate sets with Time.
- The second part, the optimal solution for every candidate set against the outcome of MCMC.

4.1 Run of the MCBO model

The first phase in the MCBO model detailed the output generating outputs for 9000 iterations created by the MCMC algorithm in this paper and the extraction of the (Start Position, Times (step))

The second phase in MCBO analysis and Grouping data results from phase one; five group locations will be studied in this paper.

5. RESULT AND ANALYSIS

In this sub-section, the MCM findings are described in three parts. In the first segment, the MCMC variants and The Fridman Strategies Test of MCMC Development were mentioned in the second section. In the third section, the optimal path finds by enhancing BA algorithm.

5.1.1 result for 9000 iterations for five groups generated by the MCMC Algorithm

The MCMC Random walk algorithm (i.e., start point, current location, next location to reach the final position (100), and how long it takes (i.e., steps) to reach the final position (100)) was used to calculate the outputs.

Iteration Number 1:

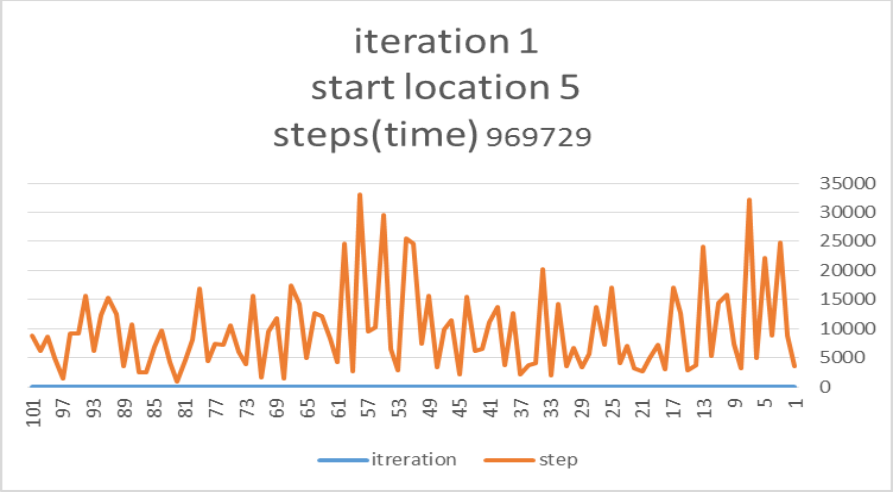

Figure 2: Iteration Number 1.

Figure 2 shows the route obtained by the random walk algorithm. It included the position at start point 5 of the two necessary values and the steps needed to cover boundary 969729.

I 1 = {L1, T1}, numerically expressed as {[5], [969729]}.

- Iteration Number 2

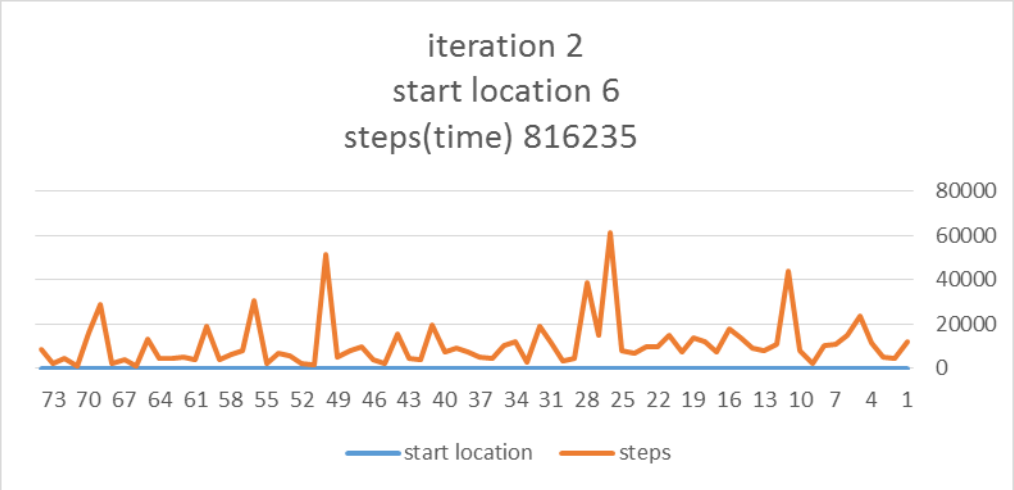

Figure 2: Iteration Number 1.

Figure 2 shows the route obtained by the random walk algorithm. It included the position at start point 6 of the two necessary values and the steps needed to cover boundary 969729.

I 2 = {L2, T2}, numerically expressed as {[6], [816235]}.

- Iteration Number 3

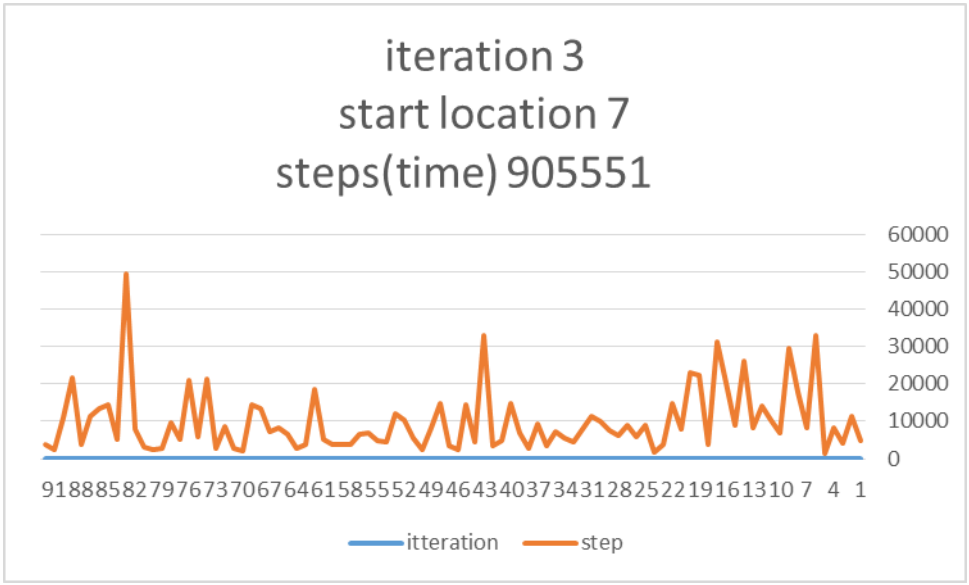

Figure 4: Iteration *Number* 3.

The route established by the random walk algorithm is depicted in Figure 4. It included the location of two required values at the start point 7 and the steps necessary to cover boundary 905551.

- I3={L3,T3}, represented numerically as {[7], [905551]}.

- Iteration Number 4

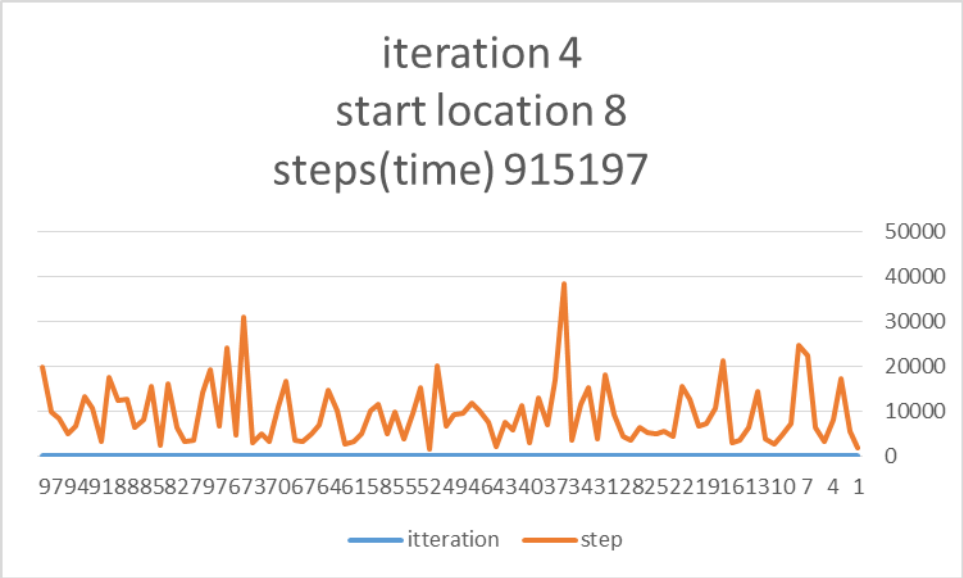

Figure 5: Iteration Number 4.

The route obtained by the random walk algorithm is illustrated in Figure 5. It included the location of two required values at the start point 8 and the steps necessary to cover boundary 915197.

- I4= {L4, T4}, represented numerically as {[8], [915197]}.

- Iteration Number 5

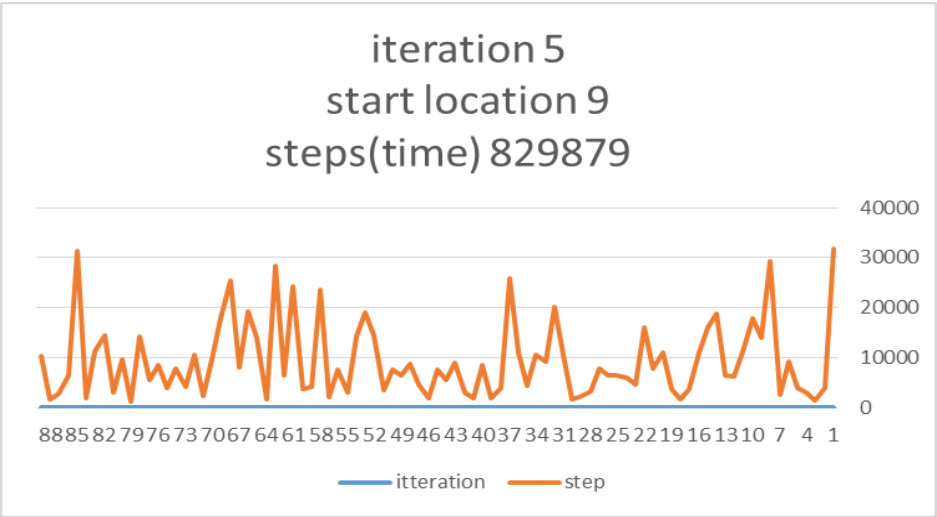

Figure 6: Iteration Number 5.

The route obtained by the random walk algorithm is illustrated in Figure 6. It included the location of two required values at the start point 9 and the steps necessary to cover boundary 829879.

- I5= {L5, T5}, represented numerically as {[9], [829879]}.

5.2 Analysis and Collect results MCMC

Figure 7 shows the route created by a random walking algorithm for 9000 iterations and extracts two necessary starting location values and the steps to cover all starting locations at the boundary.

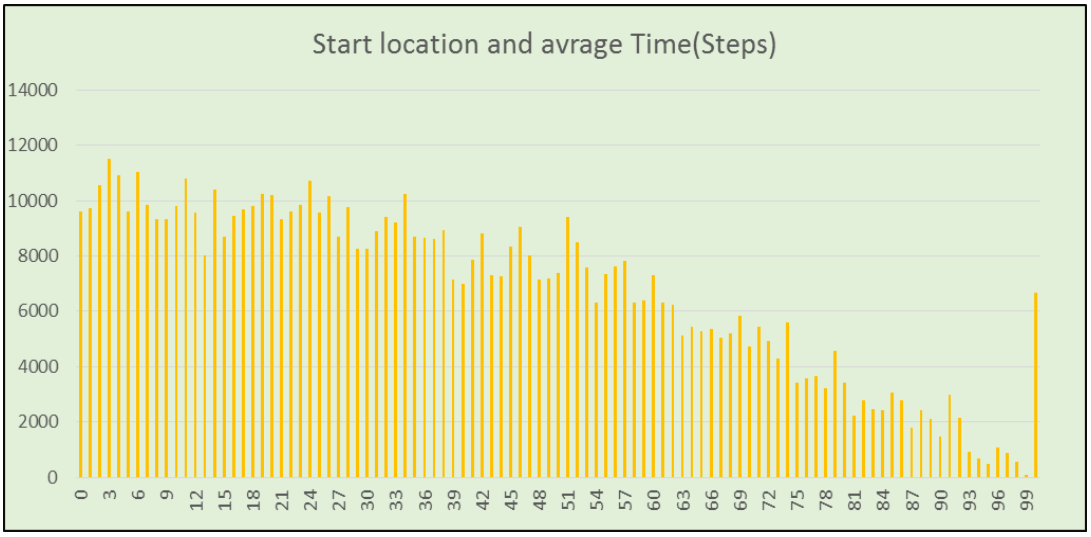

Figure 7 Graph the start location and time (step) to reach the domain boundaries.

In this paper, we split our output from 9000 iterations to 5 sets based on Start position (0-99) {5,6,7,8,9} extract, and use it as the candidate sets, table (1) shows the candidate set, each set of iterations that generated Time (steps) to reach the upper bound.

To determine the optimal path (Time or step), we have entered these steps into the BAT algorithm.

Table (1): The output for candidate sets run 9000 iteration.

Start location	Repeat	Total (Time)
5	101	969729
6	74	816235
7	92	905551
8	98	915197
9	89	829879

The representative results for Five candidate sets were used during this particular step to delineate the input for the BAT algorithm clearly.

- **Candidate Set Number 1:** (see table 2).
- **Candidate Set Number 2:** (see table 3).
- **Candidate Set Number 3**: (see table 4).
- **Candidate Set Number 4**: (see table 5).
- **Candidate Set Number 5**: (see table 6).

5.2.1 The Use of BAT Algorithm

The representative components are defined in the current step as Numerical Legal Values.

I = {Li, Ti}, represented numerically using Table 1{[start location], [(Time)]}. Typically, BAT algorithm uses echo sets and solutions to in order to assess the optimal Solution for each candidate set. This is depicted accordingly in Table7.

Table 7: BAT population for candidate sets.

Size (N)	Start location	Steps (Time)
Set 1 [5]	[X1,X2.X3,..X101]	[3607, 8885, 24832,..,8857]
Set 2 [6]	[X1,X2.X3,..X74]	[12029, 4303, 4898..,8608]
Set 3 [7]	[X1,X2.X3,..X92]	[4643, 11407, 4137, ..3877]
Set 4 [8]	[X1,X2.X3,..X98]	[1574, 5560, 17007…,19620]
Set 5 [9]	[X1,X2.X3,..X89]	[31736, 3973, 1441,…,10240]

The BAT algorithm variable is determined using Table8, where the size of the BAT algorithm size (N) is determined using the mean algorithm population. Meanwhile, the candidate sets for each set indicate the iteration for each set and the value of each iteration to obtain the solution = (Time)/Population.

Table 8: Bat input value.

Input Name	Values	Describe
Bat populations	5 sets	Each set has a specific number of iteration and Time
Solutions	Defined by BAT Algorithm	Rounded (Average(Time))
Lower boundary	0	Lower boundary
Upper boundary	3	Upper boundary
Tolerance	0.001	Tolerance value

NEXT The Bat population solution was determined (See figure 2). The BAT as mentioned earlier algorithm outputs were composed of population, solution, speed measurement (velocity), and frequency.

5.2.2 Outcome for BAT Algorithm Generated Candidate Sets

In order to obtain the solution, each candidate set was included in the BAT algorithm. For the candidate sets, Table 9 indicates the solution obtained.

Table 9: Solution for Candidate Set Generated by the BAT algorithm.

Set	xi	Total (Time)	Average (Time)	Rounded (average(Time))
5	101	969729	9601.277228	9601
6	74	816235	11030.2027	11030
7	92	905551	9842.945652	9843
8	98	915197	9338.744898	9339
9	89	829879	9324.483146	9324

5.2.3 Friedman Test

To determine the significant values and the best location of the candidate sets, the Friedman test was used. In Table 12, the results obtained are shown significant for start location Five.

To assess the importance or insignificance of the findings, a test search was carried out in this paragraph on all Candidate sets.

- **Candidate Set Five-location Friedman test**

Table 10: Descriptive Statistics for Five –location.

Descriptive Statistics					
	N	Mean	Std. Deviation	Minimum	Maximum
Start Location	101	5.00	.000	5	5
Time	101	9601.28	7138.622	842	33144

Table 11: Friedman rank for Five-location.

Ranks	
Start Location	1.00
Time	2.00

Table 12: Friedman test for five-location.

Test Statistics	
N	101
Chi-Square	101.000
df	1
Asymp. Sig.	.000

The result Indicates that the Chi-square value of 101 is at the level of 0.01, which means the error happened every 100 iterations one Time.

Figure 5.3 shows the results of the MCMC versus the MCBO model for the candidate set (FIVE-Location).

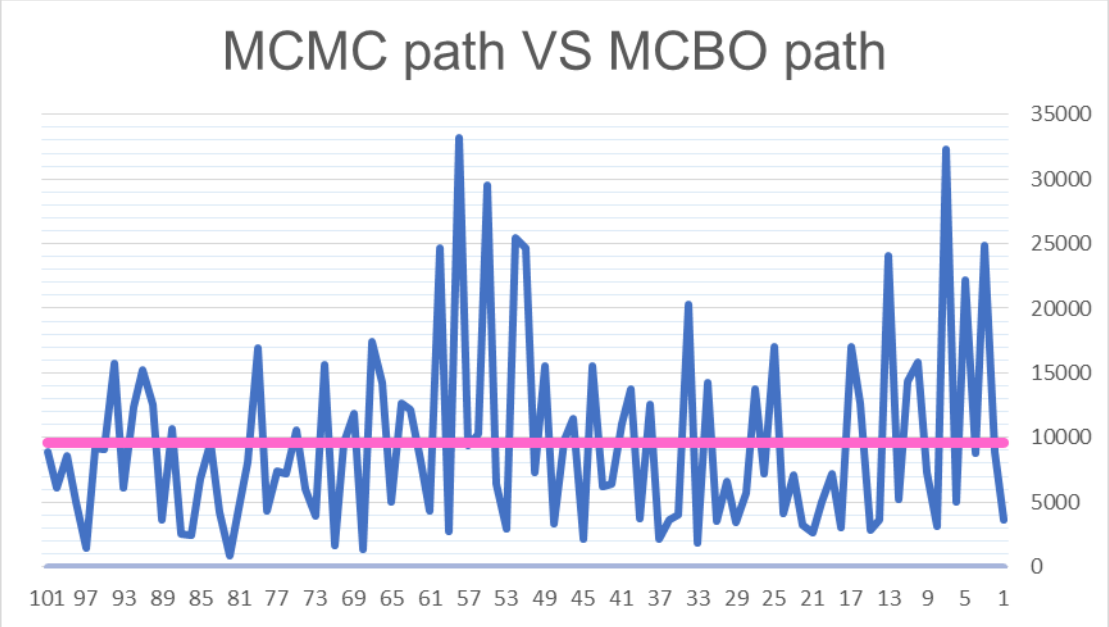

Figure 8 MCMC versus the MCBO model FIVE-Location.

The blue line displays the significant results of the Random Walk MCMC algorithm. Furthermore, the pink line indicates the optimal solution outcome of the MCBO model, so this is the best way to decrease resource and time usage when scheduler updating a distributed cloud at any time in the 9601 range set Five-Location.

5.3 Results Summary

Together with the Friedman Test by Chi-Square analysis, the results from the MCMC algorithms are obtained. MCBO calculates the average of various paths produced in the same set.

The percentage of time reduction for the first candidate set (five-location) resulted in a 3% percent decrease in time consumption in the first candidate set, where the Time or step summation gives (279227) in 101 iterations with the condition (Time or step) above the optimal Time generated from Bat Algorithm solution (9601).

6. CONCLUSION

This work established a data replication strategy to meet both the availability and performance requirements with a concomitant effect. The use of a model called MCBO could determine the optimal path in order to ensure time preservation.

MCBO benchmarking requires benchmarking of existing methods as well as the associated statistical analysis. Statistically meaningful results were obtained by MCBO efficiency. This article, building on the current content.

As the MCBO model introduced in this study is still a prototype, a simple starting point for future work would be to realize automated test replication strategies based on Markov Chain Monte Carlo and Optimization on Cloud Applications. In particular, several MCBO attributes (i.e., input-output interaction, candidate sets, and scale) should be included.

In this job, only five sets are checked by the MCBO. Therefore, as far as possible, there is a need to review more sets to be offered by MCBO to enhance its applicability.

REFERENCES

1. Ali, M., Khan, S. U., & Vasilakos, A. V. (2015). Security in cloud computing: Opportunities and challenges. *Information sciences*, *305*: 357-383.
2. Rittinghouse, J. W., & Ransome, J. F. (2017). *Cloud computing: implementation, management, and security*. CRC press.
3. Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., & Zomaya, A. Y. (2015, June). Models for efficient data replication in cloud computing data centers. In *2015 IEEE International Conference on Communications (ICC)* (pp. 6056-6061). IEEE.
4. Limam, S., Mokadem, R., & Belalem, G. (2019). Data replication strategy with satisfaction of availability, performance and tenant budget requirements. *Cluster Computing*, 1-12.
5. Selvi, M. S. A. E., & Anbuselvi, R. (2015, March). An Analysis of Data Replication Issues and Strategies on Cloud Storage System. In *International Journal of Engineering Research & Technology (IJERT), NCICN-2015 Conference Proceedings, pp18-21*.
6. Kumari, P., & Kaur, P. (2018). A survey of fault tolerance in cloud computing. *Journal of King Saud University-Computer and Information Sciences*.

7. Milani, B. A., & Navimipour, N. J. (2016). A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions. *Journal of Network and Computer Applications*, *64*: 229-238.
8. Tabet, K., Mokadem, R., Laouar, M. R., & Eom, S. (2017). Data replication in cloud systems: a survey. *International Journal of Information Systems and Social Change (IJISSC)*, *8*(3): 17-33.
9. Tos, U., Mokadem, R., Hameurlain, A., Ayav, T., & Bora, S. (2016, July). A performance and profit oriented data replication strategy for cloud systems. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)* (pp. 780-787). IEEE.
10. Xiong, R., Luo, J., Song, A., Liu, B., & Dong, F. (2011, September). QoS preference-aware replica selection strategy using MapReduce-based PGA in data grids. In *2011 International Conference on Parallel Processing* (pp. 394-403). IEEE.
11. Kloudas, K., Mamede, M., Preguiça, N., & Rodrigues, R. (2015). Pixida: optimizing data parallel jobs in wide-area data analytics. *Proceedings of the VLDB Endowment*, *9*(2): 72-83.
12. Bonvin, N., Papaioannou, T. G., & Aberer, K. (2010, June). A self-organized, fault-tolerant and scalable replication scheme for cloud storage. In *Proceedings of the 1st ACM symposium on Cloud computing* (pp. 205-216). ACM.
13. Sakr, S., & Liu, A. (2012, June). Sla-based and consumer-centric dynamic provisioning for cloud databases. In *2012 IEEE Fifth International Conference on Cloud Computing* (pp. 360-367). IEEE.
14. Janpet, J., & Wen, Y. F. (2013, March). Reliable and available data replication planning for cloud storage. In *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)* (pp. 772-779). IEEE.
15. Kumar, K. A., Quamar, A., Deshpande, A., & Khuller, S. (2014). SWORD: workload-aware data placement and replica selection for cloud data management systems. *The VLDB Journal—The International Journal on Very Large Data Bases*, *23*(6): 845-870.
16. Zhang, H., Lin, B., Liu, Z., & Guo, W. (2014, September). Data replication placement strategy based on bidding mode for cloud storage cluster. In *2014 11th Web Information System and Application Conference* (pp. 207-212). IEEE.

17. Sousa, F. R., & Machado, J. C. (2012, November). Towards elastic multi-tenant database replication with quality of service. In *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing* (pp. 168-175). IEEE Computer Society.
18. 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.