Open Access | **Baghdad Science Journal** | **P-ISSN: 2078-8665**
2021, 18(4) Supplement: 1356-1364 | | **E-ISSN: 2411-7986**

DOI: https://dx.doi.org/10.21123/bsj.2021.18.4(Suppl.).1356

Crucial File Selection Strategy (CFSS) for Enhanced Download Response Time in Cloud Replication Environments

M.A. Fazlina[1,2] ***Rohaya Latip[1]**** ***Azizol Abdullah[1]*** ***Hamidah Ibrahim[1]***
Mohamed A. Alrshah[1]

[1]Faculty of Computer Science and Information Technology, University Putra Malaysia, Serdang, 43400 Selangor, Malaysia.
[2]Malaysian Administrative Modernisation and Management Planning Unit (MAMPU), Setia Perdana 2, Setia Perdana Complex, Federal Government Administrative Centre, 62502 Putrajaya, Malaysia.
[*]Corresponding author: rohayalt@upm.edu.my
E-mails: sygfaz5@gmail.com , azizol@upm.edu.my , hamidah.ibrahim@upm.edu.my , mohamed.asnd@gmail.com

Received 11/10/2021, Accepted 14/11/2021, Published 20/12/2021

Abstract:

Cloud Computing is a mass platform to serve high volume data from multi-devices and numerous technologies. Cloud tenants have a high demand to access their data faster without any disruptions. Therefore, cloud providers are struggling to ensure every individual data is secured and always accessible. Hence, an appropriate replication strategy capable of selecting essential data is required in cloud replication environments as the solution. This paper proposed a Crucial File Selection Strategy (CFSS) to address poor response time in a cloud replication environment. A cloud simulator called CloudSim is used to conduct the necessary experiments, and results are presented to evidence the enhancement on replication performance. The obtained analytical graphs are discussed thoroughly, and apparently, the proposed CFSS algorithm outperformed another existing algorithm with a 10.47% improvement in average response time for multiple jobs per round.

Keywords: Cloud computing, Crucial File Selection, Data Replication, Replication Algorithm.

Introduction:

Cloud computing is a resilient and well-known technology to serve enormous data from various platforms [1–3]. Cloud computing offers many critical services, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), to support a variety of business scales. Cloud computing architectures, protocols, and tools give opportunities for service innovation, which helps cloud clients in a multitude of ways [4-7]. One of the prominent services to the client is providing data replication which enables faster data retrieval.

Data replication is a multi-dimensional approach for storing one or more copies of data across several cloud storage services [8,9]. Although different studies defined data replication in different ways, the gist and aims of data replications are the same as promising methods for ensuring data accessible in a heterogeneous system environment. Researchers are eagerly and proactively placed their efforts to establish an appropriate data replication strategy according to business goals to achieve optimal performance in cloud replication systems [10,1]. Explicitly, replication strategies are a broad umbrella with extensive technology in the cloud storage system, accelerating data access through efficient processing mechanisms and significantly improving various cloud replication performance metrics [12]. Therefore, cloud providers and researchers developed numerous approaches like; sub-strategies, techniques, methods, and algorithms to facilitate a comprehensive replication mechanism for cloud replication environments.

Similarly, data selection strategies are a prevalent mechanism introduced in numerous researches works ensuring that only essential data are chosen for the replication process [13-15]. However, the unforeseen drawbacks in existing approaches usually remain significant challenges, which often degrades replication performance. The drawbacks are not limited to high response time during file retrieval, high storage usage, and massive network usage, leading to expensive costs in cloud replication management [16,17]. Thus, this study proposes a comprehensive data selection strategy called Crucial File Selection Strategy (CFSS) to address one of the main limitations of current solutions. Subsequently, CFSS guarantees faster file retrieval and enhance performance with a low vulnerability that would satisfy the cloud users.

The remainder of this paper is structured as follows: Related Works discusses data replication and data selections strategies in the cloud environments. Subsequently, a detailed explanation of the proposed model, process diagram, algorithm and system implementation are presented. Further, results and discussions of the experiments are delivered. Finally, research conclusions summarized, and the future directions for this research work are recommended.

Related Works:

There are two (2) typical approaches for replication strategies in data replication systems [10,18,19]. First, static replication is a predetermined technique for specific replica situations that is simple to apply but does not adapt to all environments [5,9]. The second approach is dynamic replication, also known as agile replication strategies, in which the algorithm efficiently generates and deletes any replicas based on system users' access patterns [13,14].

Static and Dynamic Replication

Regardless of the limitations in static replication approaches, some research work still carried adapting static replication mechanism. Researchers [20] proposed a combined solution inclusive of data placement and replica placement in one single process called Combining Data and Replica (CDR). Subsequently, the unified CDR recognized as a generic framework called UnifyDR with a static system workload approach. The research work attained minimizes communication cost, traffic, and storage cost by analyzing execution time performance between three (3) existing techniques, Hyper, Spectral, and OverlapH, which influence replication system performance. However, the disadvantages of the work are high network usage because the study environment uses join-intensive online analytical processing (OLAP) queries and a location-based online social network (OSN) service that requires massive data movement in bandwidth.

Numerous researchers and practitioners have adopted the dynamic replication strategy in various cloud environments, including grid, cloud, edge, and fog computing. This method is widely utilized because of its ability to effectively control data replication based on the access patterns of system users. [21,22]. A dynamic replication approach was proposed [23]. The researcher devised a consensus-based replication technique for acknowledging replica updates ready for cloud customers to read at storage. Before placing updated copies, the researchers addressed significant security elements in data transfer with encryption and decryption capabilities, as well as secure hashing operations, in their suggested approach. The study's objectives are met because of a short data update time and significant data consistency. On the other hand, the study has a high network usage because of the continual acknowledgement procedure among services, contributing to heavy traffic within the bandwidth.

Data Selection Strategies

Copying the entire dataset into a replication environment consumes mass space and results in insignificant performance improvements [18,24]. Replication strategies employ various approaches to ensure that only critical data is identified and replicated to replication storage. The approaches are also known as data selection strategies.

The first approach is a traditional replication approach that calculates the frequency of file access by users is a direct and straightforward concept to identify replication candidates in cloud services. Researchers [25] used a similar approach to determine the most popular files called Dynamic replica Creation for Availability Enhanced Storage (DRCAES). The DRCAES algorithm is integrated with an existing algorithm named File Accessing Frequency accessing ranking (FAFR) to identify frequently requested files in the replication environment. The proposed algorithm list most accessed file and arrange the list in descending order. The most top-ranked files are chosen to have a replica copy and replicated to another server based on server memory. Researchers achieved the goal through accelerates the response time, but they disregarded the bandwidth consumption caused by the mass size of transferred files during replica creation and placement.

A mathematical formula is another data selection strategy adapted vastly in current research works. Researcher [26] proposed a new Dynamic Popularity aware Replication Strategy (DPRS) to select the most popular. DPRS ingested mathematical formulation to identify crucial data in the cloud replication environment. There are few replication factors considered in the formulation: the total number of file access, file sizes, and finally, the popularity value for each accessed file is obtained. As usual, the uppermost value has a higher priority to be selected for the replication process. However, DPRS adopted the Pareto principle in the following phase, whereby only the top 20% from the listed popular files are selected as replication candidates. The researcher claims the proposed strategy is efficient because of the popularity factors presented in DPRS capable enough in choosing necessary files to fulfil user demands. However, the researcher overlooked the high replication time, which causes by the complex computation involved in the popular file selections.

In 2019, an algorithm called Data Mining-based Data Replication (DMDR) was proposed, which determines the relationship of the accessed data using user access history [27]. The data mining task is mapped into three (3) main categories, assigning a logical value to most required files, determines each task as a group of files accessed for a particular job and extraction context. The three (3) fields are used to identify accessible files and then to find the most often accessed pattern, which is then included in file access histories with a predefined value. Finally, the list of popular files is selected for the replication process. According to the researcher, the proposed DMDR contributed to a faster response time. But at the other side, the processing time overhead issue was disregarded, which was accentuated by the long techniques process used during the data mining process until the determination on popular file listing.

A researcher proposed the dynamic Replication Factor Model for Linux Container-based in cloud systems [28]. Data mining, AI, and probabilistic models are used to predict correct replication factors. This method determines crucial data and the number of replicas required to be stored in each storage container. During any failure on nodes, snapshots/container images are used to recover or restart containers. The study successfully improves data availability and replication performance. The disadvantage of the solution in this research work is high replication time. This drawback is due to the regression analysis being a rather lengthy process to determine the best replication factor to decide an adequate number of replicas to be stored.

Hierarchical Data Replication Strategy (HDRS) can identify popular files based on the prediction of subsequent access data for data files in the cloud and replicate the replicas into the best site using network-level locality [29]. The HDRS algorithm adopted the prediction approach and uses labelling concepts at the sites with specific naming such as a parent, siblings, ancestor, load, and hops. HDRS place replica based on needs and placement to a site with the most recent required replica. The researcher claimed that the HDRS successfully reduced the response time, bandwidth, and latency. However, high replication time remains one of the drawbacks of this study. The researcher ignored the long process time due to the placement strategy with various checking procedures at the master, parent, siblings, and ancestor nodes influenced the replication process overheads.

Holistically, data selection strategy has a significant role in determining only important files replicated in cloud replication environments [30-33]. Additionally, as mentioned by [26], the popularity factors in the data selection strategies greatly influence response time for file downloads. Therefore, a comprehensive data selection strategy with significant factors plays crucial roles in decreasing the response time and enhancing performance for cloud replication strategy entirely [28,34,35].

Proposed Model: Crucial File Selection Strategy (CFSS)

The detailed algorithm for CFSS is shared in the next sub-section. The overall process of the proposed CFSS algorithm is illustrated in Figure 1 and summarized in detailed steps as follows:

1. In respective Local Replica Manager (LRM) for each Cluster C_j which denotes as LRM_j, individual file access is identified based on file id (F_i) , in an array table.
2. Accumulated file access ($\hat{F}$), are calculated and sorted in descending order. Popularity factors for files (P_i) are computed using **Eq. (1)** as in algorithm, and the values for each file are stored in a list as Set_x.
3. The frequency of each popular files (Δ), is further counted using **Eq. (2)** as in algorithm and Set_x re-arranged in descending order according to (Δ) values.
4. All information from individual LRM_j are send to Global Replica Manager (GRM) inclusive the latest list of Set_x.
5. GRM continue the process to determine replication candidates by selecting the top 20% from Set_x from every LRM_j and store the top 20% list as Most Popular File (MPF_i), in Set_y.
6. Eventually, Set_y as the final replication candidates list is ready for replica process and placement activity.

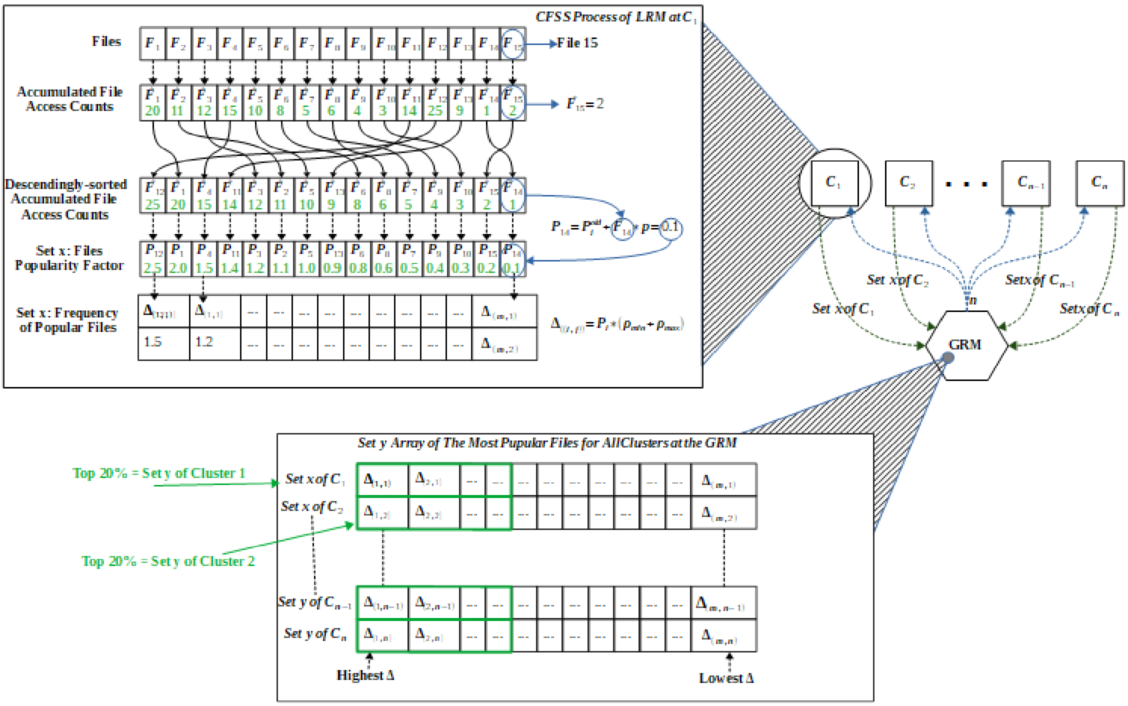

Figure 1. CFSS Process Diagram

The CFSS Algorithm is shared as follows.

Algorithm: Critical File Selection Strategy (CFSS)

// $\rho min = \frac{1}{n}$; $\rho max = \frac{Wmax}{n}$; $p = 0.1$; $q = 0.15$; $P^{Old}_{(i,j)} = 0.005$; $W_{max} = \lfloor \frac{n}{2} \rfloor$

// $Setx_j = \{\hat{F}_{(i,j)}, \ldots, \hat{F}_{(m,n)}\}$ //Set contains the total file access

//$\Delta list_i = \Delta_{(1,1)}, \Delta_{(2,j)}, \ldots, \Delta_{(i,j)}$ } //list consist the frequency of each file

//At the LRM

For all files, F_i in Clusters, C_j ; $j = \{1,2, \ldots n\}$.

User Request file, F_i ;

LRM notifies file access;

When $User_k$>1 {

$User_k$ = earliest timestamp then;

$User_l$ locked; //$User_l$ = other user;

//File Access Calculation

for ($j = 0$; $j < n$; $j = j + 1$) {

for ($i = 0$; $i < m$; $i = i + 1$) {

if $F_i \nexists LRM_j$ then

Notifies GRM (F_i);

end

$\hat{F}_{(i,j)} += 1$;

Descending Sort ($Setx_j$);

if $\hat{F}_{(i,j)} > 0$ then

$P_{(i,j)} = P^{Old}_{(i,j)} + \hat{F}_{(i,j)} * p$;

Else   **//Eq. (1)**

$P^{Old}_{(i,j)} - q$;

end

//Calculate Popular Frequency for Files

$\Delta_{(i,j)} = P_{(i,j)} * (\rho_{min} + \rho_{max})$; **//Eq. (2)**

Re-sort Descending ($Setx_j$) based on Popular Frequency; *//File is ready for download*

}

} end

//At the GRM

30. GRM get updates on popular file list ($Setx_j$) from LRM_j;{
31. $MPF_i = \lfloor \sum FR * (1-x) \rfloor$; *//Select the Top 20% from $Setx_j$ using 20:80 principal*
32. Results are identified as Most Popular File, MPF_i and store as ($Sety_j$);
33. $MPF_i = Sety_j$;
34. Replication process for all MPF_i;
35. } End.

System Implementation:

This research environment was developed identically as another work by [26] using CloudSim. We selected [26] to compare with our proposed CFSS because it reached various goals and improved many performance metrics in cloud replication, including reducing response time for file downloads. There are clusters, data centers, Global Replica Manager (GRM), and a Local Replica Manager (LRM) are interconnected as part of the system architecture. The GRM set as a broker is the core of this architecture and connected to other nodes by various routers and connections. Multiple clusters consist of data centers are associated with individual storage in this experiment architecture. The specification of every node is summarized in Table 1 adopted from [26].

Table 1. Parameters

PARAMETERS	VALUES
Total Number of Clusters	10
Total Number of Nodes	100
Numbers of Nodes within the same clusters	10
Number of Different Files	200
Size of each file	From 1 to 20 (GB)
Storage Size for every Cluster Nodes	60 (GB)
Number of Files Accessed by a Job	3-10
Round Length	100
Number of Intermediate Nodes between two nodes in the same cluster	1
Number of Intermediate Nodes between two successive cluster	3
Inter-Router Bandwidth	10 (Gbps)
Router-to-Site Bandwidth	2.5 (Gbps)
User-to-Router Bandwidth	100 (Gbps)
GRM-to-Router Bandwidth	2.5 (Gbps)
LRM-to-Router Bandwidth	1 (Gbps)
The Duration of Round (Td)	1000 (sec)
W1, W2, W3	1/3

Results and Discussions:

As evidenced to prove the capability of the CFSS algorithm, a rigorous experiment was conducted through measuring Average Response Time (ART). Average Response Time (ART) is defined as the duration between sending until receiving particular jobs. The ART formula is described as Eq. (3), adapted from [26].

$$ART = \frac{\sum_{j=1}^{m} \sum_{k=1}^{m_j} \left(tS_{jk}(rt) - tS_{jk}(st)\right)}{\sum_{j=1}^{m} m_j} \quad (3)$$

In Eq. (3), $tS_{jk}(rt)$ and $tS_{jk}(st)$ denotes (st) sending and (rt) receiving time for job k and user j. The number of jobs for a user j is referred to m_j.

The studies were carried out with 200 master files produced using Zipf' distribution, with file sizes ranging from 1000Mb to 20,000 Mb. Each task interval was simulated for a total of 100 rounds. All of the parameters utilized in individual experiments are similar to those found in previous studies. [26]. CFSS greatly decreased response time for file downloads in the replication cloud environment, as seen in Figure 2. In simulations, an experiment was done employing Eq. (3) with numerous jobs per round to produce the findings shown in Figure 2.

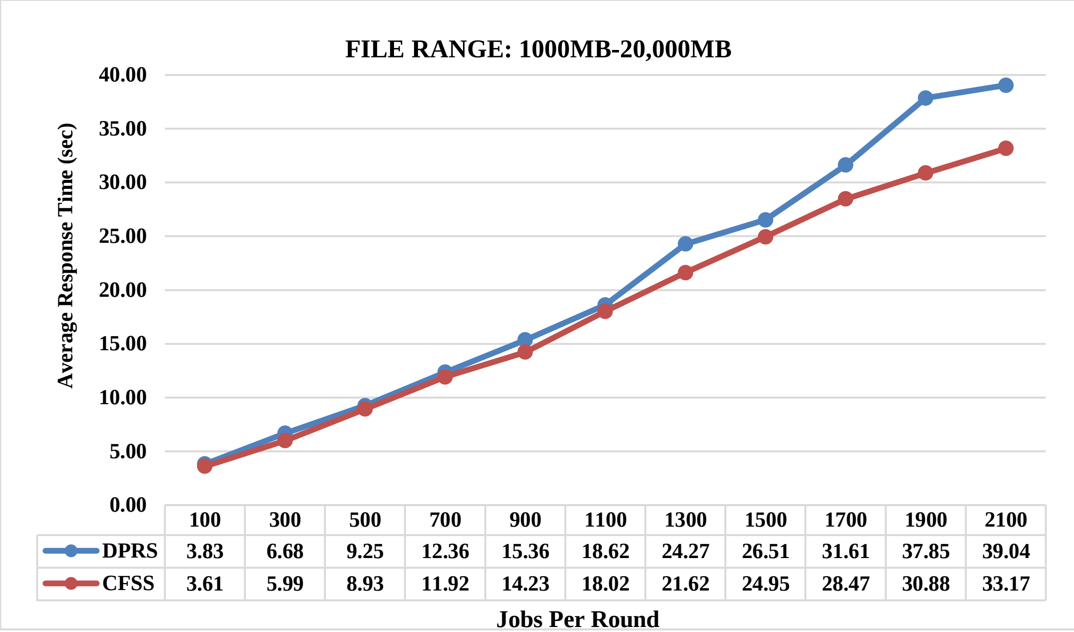

Figure 2. Average Response Time (File Download)

Figure 2, the CFSS algorithm shows 6% enhancement on response time for 100 number tasks, continued with 10.34% acceleration for 300 tasks compared to DPRS. At the peak of 500 and 700 jobs, the graph lines reflecting 3.44% and 4% improvement, respectively. Meanwhile, for 900 jobs, the results gradually improved by 7.33%, and for 1100 jobs, the improvement percentage is 3.23% for average response time. However, the average response time gradually improves when the number of jobs increases after 1100 jobs. At the point of 1300 jobs, the CFSS algorithm depicted a 10.93% enhancement in response time compared to DPRS. While, at 1500 and 1700 jobs per round, results obtained by CFSS are 5.90% and 9.94% faster response time than DPRS. Finally, drastic betterment on response time was perceived, with 18.42 % for 1900 jobs per round and 15.04% for 2100 jobs. The results trends influenced by the number of jobs and factors were varied over the number of successful replications for cumulative rounds.

The potential cause of DPRS performance degrading at every peak is due to irrelevant criteria in computation, particularly when discovering popular file values, which are extensive and complicated. This causes some additional processes to occur, as well as a delayed response time. Instead, CFSS accelerates response time, using a compact strategy to compute and obtain values for accessed files more proficiently. Further, with the essential proposed factors in the crucial file selection strategy, CFSS can ensure necessary files replicated in local storage that expedite the file retrieval during user downloads. Overall, the CFSS outreach DPRS algorithm improved by 10.47%, with a significant number of better response times.

The following experiment was carried out with constant file sizes because file size is a factor that influences job completion time. In several simulations, constant file size is tested with 100, 300, 500, 700, 900, and 1100 tasks. This experiment ensures that we got fair and accurate findings when testing the CFSS algorithm's competence. Subsequently, Figure 3 depicted the total average response time for file downloads with different constant file sizes scaled as; 1000Mb, 5000Mb, and 10,000Mb.

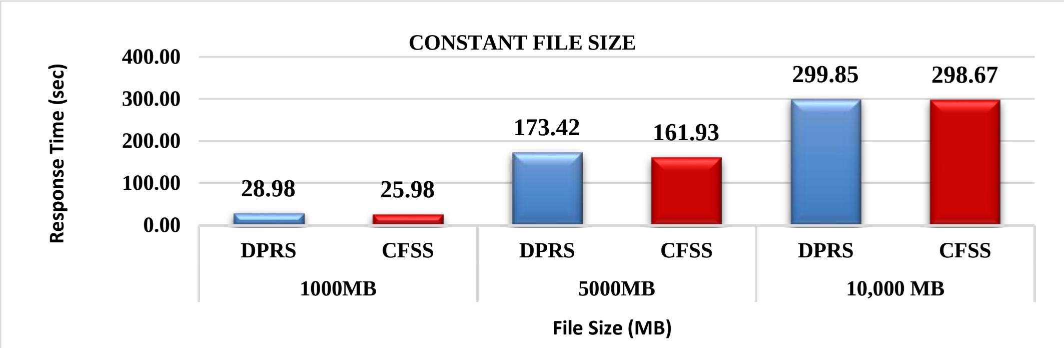

Figure 3. Total Average Response Time (File Download)

Figure 3 evidenced a total of 10.37% enhancement reached by CFSS compared to DPRS for file size 1000Mb. Whereas, for file size 5000Mb, the overall betterment achieved was 7%. Finally, for a greater file size of 10,000Mb, the response time tremendously falls to 0.4% improvement compared to DPRS. The graphs show that the response time draws an exponential pattern as the file size scales increases. Thus, the response time degrades for the CFSS algorithm when file sizes are more extensive because the replication fails due to full storage. Therefore, essential files may not be available locally for user access and require file retrieval from remote sites, which causes a longer response time for file download. Holistically, the CFSS algorithm produces better response time than DPRS, which evidenced CFSS has comprehensive popular file selections factors, which determines crucial files for the replication process and guarantees faster file retrieval.

Conclusion and Future Recommendations:

In conclusion, this research work attained the research goal with the proposed Crucial Data Selection Strategy (CFSS) for cloud replication environment. The CFSS successfully accelerates response time without any substantial drawback on popular file selections compared to DPRS algorithm. The CFSS competence, proven as required crucial files, is always available, making the file retrieval or downloads significantly faster than the DPRS algorithm. The analytical results evidenced CFSS outperformed the DPRS with a 10.47% improvement in average response time for multiple jobs per round. Thus, intensely CFSS derive enhancement on cloud replication performance.

For future researchers, it would be better to integrate data selection with features like fuzzy inferences in their research area. Additionally, measurement on popularity file accuracy is suggested to compare the proposed data selection strategies' efficiency as proof for significant contributions and performance improvement in cloud replication environments.

Acknowledgement:

The University of Putra Malaysia supports this research work. Utmost appreciation and thanks for providing sufficient facilities throughout this research.

Authors' declaration:

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are mine ours. Besides, the Figures and images, which are not mine ours, have been given the permission for re-publication attached with the manuscript.
- The author has signed an animal welfare statement.
- Ethical Clearance: The project was approved by the local ethical committee in University of Putra Malaysia.

Authors' contribution:

All authors contributed to the design and implementation of the research, to the analysis of the results and to the writing of this manuscript.

References:

1. Cirani S, Ferrari G, Mancin M, Picone M. Virtual replication of IoT hubs in the cloud: a flexible approach to smart oject management. J Sens Actuator Networks [Internet]. 2018;7(2):16.
2. Lu S, Wu J, Duan Y, Wang N, Fang J. Towards cost-efficient resource provisioning with multiple mobile users in fog computing. J Parallel Distrib Comput [Internet]. 2020; 146:96–106.
3. Rahimi M, Songhorabadi M, Kashani MH. Fog-based smart homes: a systematic review. J Netw Comput Appl [Internet]. 2020;153(November 2019):102531.
4. Xiao Z, Xiao Y. Security and privacy in cloud computing. Commun Surv Tutorials, IEEE. 2013;15(2):843–59.
5. Mansouri N, Ghafari R, Zade BMH. Cloud computing simulators: a comprehensive review. Simul Model Pract Theory. 2020;104(July).

6. Rajabion L, Shaltooki AA, Taghikhah M, Ghasemi A, Badfar A. Healthcare big data processing mechanisms: the role of cloud computing. Int J Inf Manage [Internet]. 2019;49(June 2017):271–89.
7. C.Venish Raja DLJ. A cost-effective scalable scheme for dynamic data service in heterogeneous a cost-effective scalable scheme for dynamic data service in heterogeneous cloud environment. Int J Adv Sci Technol. 2020; Vol. 28, N(January).
8. Chaturvedi N. Analysis of replication and replication algorithms in distributed system. 2012;2(5):261–6.
9. George S, Edwin EB. A review on data replication strategy in cloud computing. 2017 IEEE Int Conf Comput Intell Comput Res ICCIC 2017. 2018;1–4.
10. Seguela M, Mokadem R, Pierson J-M. Comparing energy-aware vs. cost-aware data replication strategy. Int Green Sustain Comput Conf. 2020;1–8.
11. Shorfuzzaman M, Masud M. Leveraging a multi-objective approach to data replication in cloud computing environment to support big data applications. Int J Adv Comput Sci Appl. 2019;10(3):418–29.
12. Li C, Tang J, Luo Y. Scalable replica selection based on node service capability for improving data access performance in edge computing environment [Internet]. The Journal of Supercomputing. Springer US; 2019.
13. Gill NK, Singh S. A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers. Futur Gener Comput Syst [Internet]. 2016; 65:10–32.
14. Alami Milani B, Jafari Navimipour N. A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions. J Netw Comput Appl [Internet]. 2016; 64:229–38.
15. Nannai John S, Mirnalinee TT. A novel dynamic data replication strategy to improve access efficiency of cloud storage. Inf Syst E-bus Manag [Internet]. 2020;18(3):405–26.
16. Li C, Bai J, Chen Y, Luo Y. Resource and replica management strategy for optimizing financial cost and user experience in edge cloud computing system. Inf Sci (Ny) [Internet]. 2020; 516:33–55.
17. Mousavi Nik SS, Naghibzadeh M, Sedaghat Y. Task replication to improve the reliability of running workflows on the cloud. Cluster Comput [Internet]. 2021;24(1):343–59.
18. Fazlina MA, Latip R, Ibrahim H, Abdullah A. A review: replication strategies for big data in cloud environment. Int J Eng Technol. 2018; 7:357–62.
19. Mokadem R, Hameurlain A. A data replication strategy with tenant performance and provider economic profit guarantees in Cloud data centers. J Syst Softw [Internet]. 2020; 159:110447.
20. Atrey A, Van Seghbroeck G, Mora H, Volckaert B, De Turck F. UnifyDR: A generic framework for Unifying data and replica placement. IEEE Access. 2020;8(1):216894–910.
21. Ciritoglu HE, Saber T, Buda TS, Murphy J, Thorpe C. Towards a better replica management for hadoop distributed file system. In: Proceedings - 2018 IEEE International Congress on Big Data, BigData Congress 2018 - Part of the 2018 IEEE World Congress on Services. 2018. p. 104–11.
22. Latip R, Othman M, Abdullah A, Ibrahim H, Sulaiman MN. Quorum-based data replication in grid environment. Int J Comput Intell Syst. 2009;2(4):386–97.
23. Maheshwari R, Kumar N, Shadi M, Tiwari S. Consensus-based data replication protocol for distributed. J Supercomput [Internet]. 2021.
24. Souravlas S, Sifaleras A. On minimizing memory and computation overheads for binary-tree based data replication. Proc - IEEE Symp Comput Commun. 2017;(1):1296–9.
25. Selvi SAE, Anbuselvi R. Popularity (hit rate) based replica creation for enhancing the availability in cloud storage. Int J Intell Eng Syst. 2018;11(2):161–72.
26. Mansouri N, Rafsanjani MK, Javidi MM. DPRS: a dynamic popularity aware replication strategy with parallel download scheme in cloud environments. Simul Model Pract Theory. 2017; 77:177–96.
27. Mansouri N, Javidi MM, Mohammad Hasani Zade B. Using data mining techniques to improve replica management in cloud environment. Soft Comput [Internet]. 2019;0123456789.
28. Abbes H, Louati T, Cérin C. Dynamic replication factor model for linux containers-based cloud systems. J Supercomput [Internet]. 2020;76(9):7219–41.
29. Mansouri N, Javidi MM, Zade BMH. Hierarchical data replication strategy to improve performance in cloud computing. Front Comput Sci. 2021;15(2).
30. Xiong R, Du Y, Jin J, Luo J, Yang W, Ciritoglu HE, et al. A survey of dispatching rules for the dynamic unrelated machines environment. Concurr Comput Pract Exp [Internet]. 2018; XI(Xi):555–69.
31. Singh A. Comparative study of concurrency and replica control protocols in distributed environment. 2019;14(2):329–34.
32. Yang W, Hu Y. A replica management strategy based on MOEA/D. In: Proceedings of the 13th IEEE Conference on Industrial Electronics and Applications, ICIEA 2018. IEEE; 2018. p. 2154–9.
33. He L, Qian Z, Shang F. A novel predicted replication strategy in cloud storage. J Supercomput. 2018;76(7):4838–56.
34. Mansouri N, Javidi MM, Mohammad Hasani Zade BA. A CSO-based approach for secure data replication in cloud. J Supercomput [Internet]. 2020;77(6):5882–5933.
35. Shakarami A, Ali MG, Mohammad S, Hamid M. Data replication schemes in cloud computing: a survey. Cluster Comput [Internet]. 2021;7.

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

إستراتيجية اختيار الملفات الحاسمة (CFSS) لتحسين وقت استجابة التنزيل في بيئات النسخ المتماثل السحابي

فضلينا محمد علي [1,2] روحيه لطيب [1,*] عزيزول عبدالله[1] حميده إبراهيم[1] محمد الرشاح[1]

[1]كلية علوم الحاسب وتكنولوجيا المعلومات ، جامعة بوترا ماليزيا ، سيردانج ، 43400 سيلانجور ، ماليزيا
[2] وحدة التخطيط الإداري والتحديث الإداري الماليزي (MAMPU) ، سيتيا بيردانا 2 ، مجمع سيتيا بيردانا ، المركز الإداري للحكومة الفيدرالية ، 62502 بوتراجايا ، ماليزيا

الخلاصة:
الحوسبة السحابية هي عبارة عن منصة ضخمة لتقديم بيانات كبيرة الحجم من أجهزة متعددة وتقنيات مختلفه. هناك طلب كبير من قبل مستأجري السحابة للوصول إلى بياناتهم بشكل أسرع دون أي انقطاع. يبذل مقدمو الخدمات السحابية كل جهدهم لضمان تأمين كل البيانات الفردية وإمكانية الوصول إليها دائمًا. ومن الملاحظ بإن استراتيجية النسخ المتماثل المناسبة القادرة على اختيار البيانات الأساسية مطلوبة في بيئات النسخ السحابي كأحد الحلول. اقترحت هذه الورقة استراتيجية اختيار الملفات الحاسمة (CFSS) لمعالجة وقت الاستجابة الضعيف في بيئة النسخ المتماثل السحابي. يتم استخدام محاكي سحابة يسمى CloudSim لإجراء التجارب اللازمة ، ويتم تقديم النتائج لإثبات التحسن في أداء النسخ المتماثل. تمت مناقشة الرسوم البيانية التحليلية التي تم الحصول عليها بدقة ، وأظهرت النتائج تفوق خوارزمية CFSS المقترحة على خوارزمية أخرى موجودة مع تحسن بنسبة 10.47 ٪ في متوسط وقت الاستجابة لوظائف متعددة في كل جولة.

الكلمات المفتاحيه: الحوسبة السحابية، اختيار ملف حاسم، نسخ البيانات، خوارزمية النسخ المتماثل.