IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 18, 2021, accepted August 5, 2021, date of publication August 31, 2021, date of current version September 9, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3108765

Vigorous Replication Strategy With Balanced Quorum for Minimizing the Storage Consumption and Response Time in Cloud Environments

FAZLINA MOHD ALI[1], ROHAYA LATIP[1], (Member, IEEE), MOHAMED A. ALRSHAH[1], (Senior Member, IEEE), AZIZOL ABDULLAH[1], AND HAMIDAH IBRAHIM[2]

[1]Department of Communication Technology and Network, Faculty of Computer Science and IT, Universiti Putra Malaysia, Seri Kembangan, Selangor 43400, Malaysia

[2]Department of Computer Science, Faculty of Computer Science and IT, Universiti Putra Malaysia, Seri Kembangan, Selangor 43400, Malaysia

Corresponding authors: Fazlina Mohd Ali (sygfaz5@gmail.com), Rohaya Latip (rohayalt@upm.edu.my), and Mohamed A. Alrshah (mohamed.asnd@gmail.com)

This work was supported in part by the Ministry of Education (MOE), Malaysia, and in part by Universiti Putra Malaysia.

ABSTRACT Digital data is establishing enormously across the globe, which is considered highly valuable due to its significant contributions to facilitate different individuals' lives. Even though technology evolving rapidly, data availability remains a major challenge due to the heterogeneity of the data with different types, platforms, and technologies. Additionally, the acquired data must be always available and accessible for users regardless of time and location, which appears to be another crucial issue due to many inefficient system implementations. The most essential option to solve this issue is by providing the best replication strategies that are able to afford business continuity without interruption. Essentially, cloud replication must be able to secure huge data by enabling comprehensive replication strategies, optimal data availability, fast data retrieval, and cost-effective data management and maintenance. Therefore, this paper mainly introduces the Vigorous Replication Strategy with Balanced Quorum (VRS-BQ) replica placement technique for minimizing storage consumption, response time and replication process time in cloud environments. Comprehensive experiments have been conducted using the well-known CloudSim simulation tool, in which the results reveal that the proposed VRS-BQ algorithm attains 25% lower average response time, 22% lower replication time, and 20% lower storage consumption compared to the existing DPRS algorithm while maintaining a high level of data availability.

INDEX TERMS Replication strategies, cloud environment, cloud computing, replica placement, replication algorithms.

I. INTRODUCTION

Data exchange volume among heterogeneous technologies is immeasurable and overwhelming the entire universe [1]–[3]. The massive use of smart devices yields a huge global data medium, in which 2.5 quintillion bytes of new data are generated daily through various digital platforms, as estimated by researchers [4], [5].

International Data Corporation (IDC), stated that digital data capacity produced and copied across the globe will grow twice in size every two years and projected to be 40 trillion gigabytes by 2020 [6], [7]. According to [8], this high-volume of data similarly considered as big data since it includes both structured and unstructured data (videos, images, audio and etc.). The enormous challenge encountered in the big data field is always how to manage, process, store and make use of the data as valuable assets accessible all the time [9]. As big data adopt cloud computing services like data storage platform, subsequently, it should provide data access with various efficient services [10].

Cloud services became a competitive technology to provide a massive storage pool to any size of data [11], [12]. The pay-as-you-go paradigm offered by cloud computing is a feasible and cost-effective deal devoted to everyone [13], [14].

The associate editor coordinating the review of this manuscript and approving it for publication was Fan-Hsun Tseng.

Through few and various services as well as virtual resources, including Platform as a Service (PaaS), Software as a Service (SaaS), Consistency as a Services (CaaS), and Infrastructure as a Service (IaaS), cloud providers stand vibrantly, resiliently, and the most preferable for users [15]–[17]. Nevertheless, cloud providers encounter challenges to accommodate adequate resources with high availability without neglecting the sensitivity of data stored in their storage. Moreover, fault tolerance and node failures are also major concerns for cloud service providers [18].

Therefore, to address arising issues such as data replication, which is one of the popular strategies used for over decades and well-known in the cloud environment [19]–[21]. Data replication techniques are heuristic multi-dimensional performance accelerators for all system users in the cloud [22]. Replication is all about saving more than one copy of data in different distributed storages or clouds [2]. Despite the key function of replication as a data enabler, it has an occasional tendency to breed performance degradation issues [23], [24]. To avoid these negative impacts, an established and competent replication strategies should be introduced to solve such performance complications. Hence, cloud services working based on replication will provide higher data availability, faster response time, better fault tolerance, and higher cost-efficiency for both cloud providers and tenants if compared to non-replicated clouds [25], [26].

In order to overcome the drawbacks in existing solutions, this research proposes a new replication technique to improve the performance with a low vulnerability that would satisfy replication cloud users. The main contributions of this work are: (1) to develop a novel file popularity-based selection algorithm, namely Vigorous Replication Strategy with Balanced Quorum (VRS-BQ), that improves the response time for file download in a replication cloud environment. (2) to introduce an intelligent and comprehensive replica placement technique based on BQ to reduce storage consumption with faster replication time. (3) to carry out a comprehensive performance evaluation for the proposed VRS-BQ and the existing DPRS algorithm.

The remainder of this paper has been structured as follows: Section II discusses the related works on replication strategies and data placement in cloud environment. Section III presents the proposed solution and the configuration of the system implementation on the CloudSim. Section IV presents and discusses the simulation results. Finally, Section V concludes the work and Section VI presents the future directions.

II. RELATED WORKS

According to systematic research by Castro-Medina *et al.* [27], replication is the most commonly discussed topic among all 83 articles in the last decade. This phenomenon proves data replication in the cloud environment is emerging world-widely as an explicit strategy for data management in the cloud environment.

A. STATIC AND DYNAMIC REPLICATION

There are two categories of replication in cloud environments; static and dynamic replication [2]. A static approach works based on defining the required strategies in advance prior to the replication strategy. The predetermined method rather easier to deploy but it is hardly adapted to the changes of replication environments due to its inflexible behavior [28]. Regardless of the inadequacy of static approaches, [29] developed Raw Data Server (RDS) and Meta Data Servers (MDS) are using replication strategy adapting static method. Similarly, [30] employed a static approach in their proposed Google File System (GFS) algorithms to achieve load balancing among fixed replicas. Both [30] and [29] were able to reduce the response time, however, both approaches suffered from some limitations on the replica placement phase. Specifically, replication files were restricted to use the same number of replicas even if the file access pattern changed. This led to storage and bandwidth consumption increase and also required high monetary cost to maintain many replica copies. Reference [31] also developed another static algorithm, namely the MinCopySet algorithm, to accomplish data durability and to reduce the response time based on a random selection of fixed replicas. However, this algorithm suffered from highly cost-expensive resource allocation for the environment, where many resources were either under-utilized or overused. Consequently, this behavior led to high energy consumption and severe performance degradation.

As for the dynamic replication, it is an efficient way for the grid, fog, edge, and cloud computing environments, which are capable to intelligently manage data replication needs depending on user access patterns [22], [32]. The process to select data files, replica placement or replica deletion can be done flexibly through adapting dynamic methods [33]. Earlier in 2012, [34] proposed a Dynamic Data Replication Strategy (D2RS), which is capable to determine whether the file is required to be copied and to find out how many necessary replicas and where are their optimal places in the replication environment. The algorithm achieved low response time and high load balancing but it suffered from poor bandwidth utilization, especially during the selection of node to place replica. In 2016, PRCR algorithm [35] also used a dynamic approach to choose necessary files and to replicate them into other sites. This algorithm showed a low storage consumption due to replicating the important files into only two sites and the unimportant files into only one site. However, it caused a high processing time due to the complexity of its data classification phase.

Most recently, research by Zhang *et al.*, 2021 [36] proposed a dynamic replication approach that addresses the issue of massive data movement among data centers in the cloud. The authors proposed an inter-data center data replication system with a dynamic bandwidth separation algorithm called BDS+. The algorithm aims to speed up data transfer by adopting a dynamic bandwidth separation that ensures bandwidth is allocated for online traffic through estimating traffic demand and rescheduling bulk-data transfers for offline

data services. It uses an application-level multi-cast designed for centralized architecture networks that appoint a central controller to manage data transmission among intermediate servers. This approach effectively improved bandwidth utilization, but it omitted the significant increase of replication time caused by the proposed sorting process of the scheduled traffic.

B. FILE SELECTION BASED ON POPULARITY

The substantial of choosing appropriate data files for replication purposes is crucial across any cloud-based environment. In grid environments, researchers adhere not to copy all data in replication nodes to secure storage capacity. This precisely demonstrates that copying all data files is not viable and inconsequential [37]–[39]. As for cloud environments, some researchers focused on implementing dynamic algorithms to identify suitable data files to be replicated in order to reduce storage consumption and minimize network congestion.

Reference [40] is one of the earlier research that introduced a lightweight data replication technique for cloud data centers. Through targeting better quality of service, this study developed a smart algorithm capable to select only crucial data to be replicated, but unfortunately, it introduced a high response time due to its high processing overhead. Reference [34] proposed a similar method to select important data files based on a certain predetermined threshold, thus, files would be only replicated if the threshold were attained, which indeed created another issue making the algorithm more dependent on the value of the threshold rather than the pattern of the files' demand.

DPRS algorithm [41] is another technique proposed to select the most popular data, which dynamically counts the frequency of requested files then the top 20% of most accessed files are nominated as replication candidates. Moreover, DPRS adapted a parallel download method to enhance the download response time. That is why it achieved low storage and bandwidth consumption compared to few other research works published in [30], [34], [42], [43]. However, the reduction of download time is still unsatisfactory due to neglecting the replication time.

Reference [21] proposed the Dynamic Replica Creation Algorithm (DRCA) and Data Replica Scheduling Algorithm (DRSA) based on a meta-heuristic approach, in which file selection is done based on its popularity. Specifically, common files are copied based on cumulative time periods and the number of files-access times. The authors claimed that better results were obtained in terms of data availability and replication time, however, their approach suffered from high response time and high bandwidth consumption due to implementing complex mathematical models.

Recently, Mansouri *et al.*, 2021 [44] proposed a dynamic replication algorithm namely Hierarchical Data Replication Strategy (HDRS). It identifies the popularity of a file based on a predicted number of file-access times then it replicates that popular file into the best site using network level locality. HDRS uses the labeling concept at the sites with specific names such as parents, siblings, and ancestors. The authors claimed that their proposed strategy has successfully reduced the response time, bandwidth, and latency. However, it introduces high replication time due to multiple checking procedures at master, parent, siblings, ancestor nodes that contribute to the replication process overheads.

John *et al.*, 2020 [45] proposed another algorithm, namely RSPC, that introduced a replication strategy that satisfies the tenant objectives and the provider profit. It proposed a popularity-based file selection strategy through analyzing data access by users in a determined period of time. Further, the authors raised a concern on insufficient storage space and they recommended compressing unpopular files prior to storing them for future access needs. Besides that, a time threshold was also implied to decide when unpopular files could be deleted, in which unpopular files would be deleted permanently if the time threshold reached. Although the aim of this algorithm was achieved, it suffers from high processing time due to its highly complex decision-making process.

Mansouri *et al.*, 2021 [46] proposed the Secure Data Replication (SDR) algorithm as an extension of their predecessor research introduced in 2017 [41], which identifies popular files as replication candidates based on the number of file-access times, where the higher the number of file-access times, the higher the popularity. Thereafter, only 20% of the candidates will be selected for replication. In fact, the selection of popular files is not new but it was mixed with fuzzy inferences to select the best data center to store newly generated replica copies. The authors claimed that SDR could achieve a low response time compared to other algorithms, however, it showed a dramatically high replication time due to the complexity of the fuzzy process.

C. DATA PLACEMENT

The data placement method mainly aims to accomplish a cost-effective performance with high data availability, minimal storage consumption, effective load balancing, and fast response time. In order to achieve that, file replicas must be placed close to the user's nodes that requested those files [47]. Similarly, Reference [48] proposed the Replica Placement Based on Load Balance (RPBLB) technique to reduce remote users' access time. RPBLB suggested duplicating the most used data to new storage without neglecting the storage load balancing. Even though the response time was reduced, energy consumption was slightly high due to the extra processing during the data center load verification stage.

Reference [49] proposed a Data Replication Algorithm Profit of Provider (DRAPP) as a data placement method, which was introduced to satisfy both tenant and cloud replication providers. This method dynamically determines popular data based on the number of file-access times, then, it activates replication processes once SLA criteria are reached. Then, it places replicas depending on the tenant's available budget taking into account storage node load balance. Additionally, it estimates expenditure and revenue intended

for both tenants and cloud replication providers in order to determine the number of replicas. Results show that DRAPP can achieve high availability, low network traffic, and low response time, as this method is measured after the necessary conditions in SLA are verified. However, the entire process is consuming high processing power and time due to the tedious procedure of getting approval from users before the respective phases start.

In 2018, a Hybrid Data Replication Strategy (HRS) with fuzzy-based deletion for heterogeneous cloud data centers was proposed by [50]. HRS focused on data placement to ensure that the storage consumption is preserved. HRS ascertains new replica placement by considering few conditions such as the frequency of replica requested, the data failure probability, and the storage centrality factors. HRS gathers input on replica size and storage space prior to placing the selected replica in the storage. If the storage capacity was not sufficient for new replica placement, it would consider the existing replica deletion process. It also considers the last access of the replica, prediction of future access possibilities, and the availability of the same replica in other nodes. Thus, replica deletion would be done only if the conditions of replacement were fulfilled. This behavior guarantees low storage imbalance, low network traffic, and low response time. However, it does not address some issues on high storage consumption and process overheads that may occur due to the complexity of fuzzy inference.

In contrast, Reference [51] proposed a cost-effective Hybrid PSO TS (HPSOTS) algorithm that places limited data copies in storage nodes to reduce energy consumption as the main goal. It resolves optimization problems by mixing metaheuristics with a combination of Particle Swarm Optimization (PSO) and Tabu Search (TS). It was designed to search the essential number of replicas before replication starts, where six particles (Data Centers) are chosen based on pre-determined conditions. Then, it effectively resolves a few cloud replication matters by reducing energy consumption and cost. Despite this achievement, HPSOTS suffers from some limitations such as low data availability. Since the number of replicas should be set manually as a preset constant for the algorithm, some decisive data will not be able to get extra copies, which leads to poor data availability.

On the other perspective, Reference [52] developed a cost-effective dynamic redundant replica strategy based on the security level (namely SL-DRM) to flexibly adjust the number of replicas and to construct the data cache strategy using the Location Correlation of Cache (LC-Cache) to improve data read speed. This study was conducted based on a case study of video surveillance cameras in hotel buildings. This strategy succeeded to produce duplicated copies of video footage based on the cruciality of camera locations while guaranteeing the ability to influence the security level according to certain factors for the owner of the surveillance system. This strategy attained low storage consumption with a controlled number of video copies, however, it still suffers from low data availability.

Storing valuable data in unknown storage location is inducing untrustworthiness of data integrity among cloud users. Bowers *et al.*, 2021 [53] focused their study on reliability issues of data placement by cloud providers. They proposed an integrated Location-Aware Storage Technique (LAST) into the open-source Hadoop Distributed File System (HDFS) called LAST-HDFS algorithm. This technique works as monitoring manager to detect any illegal data transfers in cloud and also it enables file storage location to track files during migration and replication process. Results show high-level of security and privacy on the placement of migrated and replicated data in the cloud, however, it exhibited high cost due to applying highly sophisticated security features. Moreover, it generates high network traffic due to the location monitoring and detection functions that require huge data exchange.

Improving Clustering Based Critical Parent (CbCP) with Replication (ICR) algorithm has been proposed by Nik *et al.*, 2021 [54] to introduce some performance enhancements on replica placement scheduling. The ICR consists of three sub-algorithms which are: (1) a scheduling algorithm that predicts any probability of server failure on cluster and determines suitable cluster taking resources reliability into account. (2) a fix-up algorithm that checks the necessity of starting replication task earlier without incurring additional cost. (3) a task replication algorithm that identifies available resources and time slots to proceed with replication task until placement of replica. This work shows high data reliability with low execution cost, however, it shows high network traffic due to imposing high number of tasks whenever resources are available, which increases energy consumption.

Li *et al.*, 2020 [55] proposed a dynamic replica placement strategy targets financial cost reduction, high data consistency, and low overhead to satisfying the user experience. The authors applied the concept of node renting to fulfil users needs, which could exhibit low financial cost and good user experience, and they adapted an eventual consistency updates that updates data changes on distributed nodes in certain time intervals. However, these updates could lead to low data reliability due to allowing data download even before the update time. Moreover, the data availability has been neglected in this work as targeting low storage space has been the main aim of this research.

Khalajzadeh *et al.* [56] proposed a Cost-Effective Dynamic Data Placement consisting a dynamic greedy algorithm which is used to find the optimal number of appropriate data centers to replicate the most accessed user data. It aims to deliver dynamic and optimized data placement with tolerable latency and low service cost, especially for social networks such as Facebook. It works based on collecting data access times and durations along with connections of friends to determine the necessity of replica creation in the nearest data centers for every user to guarantee minimum latency. However, this solution introduced significantly high complexity and high network traffic.

D. PERFORMANCE METRICS

Literature insights reveal that replication strategies are the most preferable and widespread trending in the cloud replication environment. By viewing the common goals of the literature, we found that replication strategies are used to fulfill the demands of cloud users to obtain speedy data access, high data availability, fast replication process, low storage consumption, and certainly cost-effective data maintenance. Countless research works progressing tremendously in those areas, however, vulnerability still persisted [29], [57], [58]. Figure 1 demonstrates performance metrics measured in most research works as the key issues in the replication environment.

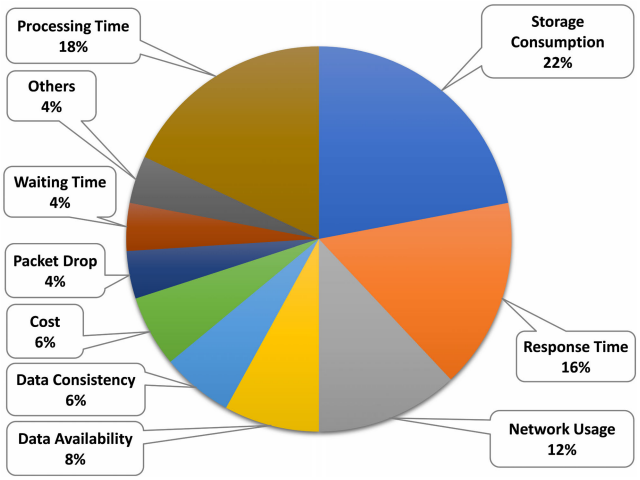


FIGURE 1. Popular issues in cloud replication.

Considering the above-discussed limitations, issues, and gaps as in Figure 1, this research proposes a novel replication strategy namely VRS-BQ, which introduces:

1) Low response time for file download requests achieved via an exclusive computation formula to guarantee ideal selection of popular files to be replicated.
2) Low storage consumption with faster replication time via implementing an innovative replica placement technique based on users' request.

Since the DPRS algorithm by Mansouri *et al.* [41] was recognized to accomplish various goals on replication performance compared to other research works, our work focuses on multiple enhancements towards replication performance on the DPRS algorithm, using an identical experimental environment as per [41] to provide fair benchmarking. Our paper reveals the drawbacks in the DPRS algorithm and successfully introduces a novel VRS-BQ algorithm which outperforms the DPRS in terms of storage consumption and response time.

III. THE PROPOSED VIGOROUS REPLICATION STRATEGY WITH BALANCED QUORUM (VRS-BQ)

This paper proposes the VRS-BQ, which is tailored for cloud replication environments, based on four main Modules:

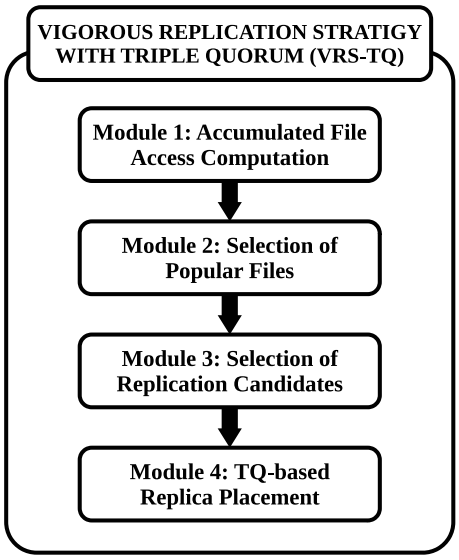


FIGURE 2. VRS-BQ components.

(1) Accumulated File Access Computation, (2) Selection of Popular Files, (3) Selection of Replication Candidates, and (4) BQ-based Replica Placement, as shown in Figure 2. These four components will be explained in the next subsections.

A. ACCUMULATED FILE ACCESS COMPUTATION

File-access requests are counted by the Local Replica Manager (LRM_j) for the requested files available locally at a respective cluster j. However, if a requested file is not existing locally, the LRM_j will extend the file request to the Global Replica Manager (GRM) to get the needed file from other clusters.

The VRS-BQ algorithm calculates the number of file-access times ($\hat{F}_i$) when a user requests to download a file (F_i), where the file index $i \in \{1, 2, \ldots, m\}$ and m is the total number of available files at the LRM_j. A set of files information F_{list} have to be saved at the LRM_j and another set of accumulated file-access counts for these files are saved as $\hat{F}_{list}$ in a descending order.

For example, if we have 3 files $F_{list} = \{F_1, F_2, F_3\}$ accessed in a certain LRM_j for 10, 20, and 15 times, respectively, these access times will be saved in a list as $\hat{F}_{list} = \{\hat{F}_1, \hat{F}_2, \hat{F}_3\}$. Then, the $\hat{F}_{list}$ will be descendingly reordered based on these accumulated file-access counts as $\hat{F}_{list} = \{10, 15, 20\}$. Thereafter, a list of popularity factors (P_{list}) will be used to save popularity factors for all files at LRM_j, as explained in the next subsection.

B. FILE POPULARITY FACTOR

File popularity factor (P_i) identifies the times of requesting or accessing a certain file in a respective LRM_j, which is calculated as in Equation (1) below:

$$P_i = \begin{cases} P_i^{old} + \hat{F}_i \times p, & \hat{F}_i > 0 \\ P_i^{old} - q, & \text{otherwise,} \end{cases} \quad (1)$$

where p and q are preset values representing the best response times without neglecting the file popularity in the replication environment, where p and q have been set for this work to 0.1 and 0.15, respectively, as recommended in [41] where always $p < q$. Additionally, the old popularity factor P_i^{old} of the first round is set to 0.005 based on 200 initial master files that was generated earlier.

In order to determine the most popular file candidates for the replication process, VRS-BQ introduces Δ_i, which represents the frequency of file popularity for file i, as in Equation (2) below:

$$\Delta_i = P_i \times (\rho_{min} + \rho_{max}), \tag{2}$$

where $(\rho_{min} + \rho_{max})$ denotes the range of replication probability which reflects the best value for replica copies, while ρ_{min} and ρ_{max} are the minimum and maximum probability of replication for the entire system, which are calculated as in Equation (3) and (4), respectively:

$$\rho_{min} = \frac{1}{n}, \tag{3}$$

$$\rho_{max} = \frac{W_{max}}{n}, \tag{4}$$

where n is the total number of clusters used in the entire system, W_{max} is the maximum number of replicas, and $W_{max} = \lfloor \frac{n}{2} \rfloor$. Subsequently, the frequencies of files popularity will be calculated at LRM_j, where j is the cluster index and $j \in \{1 \rightarrow n\}$, and then it will be saved in a popularity vector annotated as $\vec{\Delta}_j$, which will be descending ordered and passed to the *GRM*.

Based on the above two equations (3) and (4), at least one replica copy will be available for an individual file with the least popular value while the maximum possible replica copies for a single file in the entire system is limited by the half of existing number of clusters to guarantee an adequate data availability, cost-effectiveness and to meet both user and system resource requirements [59]. Moreover, ρ_{min} and ρ_{max} will play the role of minimum and maximum dynamic thresholds to avoid any possible unnecessary waste of storage. Nevertheless, the proposed algorithm is flexible enough in allocating new replica copies based on users' pattern in accessing popular files and at the same time it could prevent excessive storage consumption.

C. FILE REPLICATION CANDIDATE

Indeed, the *GRM* is the central unit that is responsible for handling all file replicas at all *LRMs*. Once an $\vec{\Delta}_j$ is received by the *GRM*, the file candidates for replication in the received $\vec{\Delta}_j$ will be selected based on the highest 20% of Δ_i values to be saved in the $\vec{\Delta}_j$ and the remaining 80% of the values will be discarded.

In order for the *GRM* to keep track of all files at all *LRMs*, all $\vec{\Delta}_j$ vectors will be saved in $\vec{\Delta}_{list}$, as $\vec{\Delta}_{list} = \{\vec{\Delta}_j, \vec{\Delta}_{j+1}, \ldots, \vec{\Delta}_n\}$, where j is the *LRM* index and n is the number of *LRMs* in the entire system.

D. BALANCED QUORUM (Q) BASED REPLICA PLACEMENT TECHNIQUE

The key objective of the Q-based technique is to provide competent data availability in replication environment and improvise replication time and storage consumption. The detailed steps of the Q-based technique are presented in the coming paragraphs.

The *GRM* applies the Q-based technique to check the presence of a requested file F_i (from an $\vec{\Delta}_j$) in the closest neighboring clusters of the requesting cluster on both sides; left and right. This technique decides the left quorum Q_L and the right quorum Q_R to determine the number of clusters on the left and right sides that needs to be checked for the existence of a requested file while including the requesting cluster C_{req} as shown in Figure 3. Based on that, the total integer number of searched clusters Q will be calculated as:

$$Q = Q_L + Q_R + C_{req}, \tag{5}$$

where C_{req} is always equal to 1. Since, the proposed technique uses a balanced quorum, in which $Q_L = Q_R = Q'$, where the total number of searched clusters including the requesting cluster is calculated as:

$$Q = 2Q' + 1, \tag{6}$$

where Q' is an integer number representing the prime quorum used for both directions of search plus 1 to consider the requesting cluster. Subject to $Q_{min} \leq Q \leq \lfloor \frac{n}{2} \rfloor$, where n is the total number of clusters in the system, Q_{min} is the minimum number of clusters needed to be searched for the existence of a certain file, and $Q_{min} = 3$ representing one cluster on the left and one on the right plus the requesting cluster, which obligates $1 \leq Q' \leq \lfloor \frac{n}{4} \rfloor$ excluding the requesting cluster and its counterpart from the other half, thus, Q' must obligate $1 \leq Q' \leq \lfloor \frac{n-2}{4} \rfloor$ to ensure the searched clusters are always less than or equal to the half of available clusters in the entire system, as shown in Figure 3.

In order to make the proposed technique tune-able in the runtime, we introduced the searching weight β to balance between the storage consumption and file availability against the response delay. Specifically, the number of searched clusters could reach the half of the total clusters in the system when the $\beta = 1$, while it would reach the minimum (2 neighboring clusters and the requesting cluster) if $\beta = 0$, where always $0 \leq \beta \leq 1$. Moreover, the number of file replicas could reach the maximum $\frac{n}{2}$ only when $\beta = 0$. Furthermore, the response delay could reach the minimum while storage consumption and file availability will reach the maximum when $\beta = 0$ and vice versa. Thus, the proposed balanced quorum Q will be calculated based on Equation (6) while considering β and all above-mentioned conditions, as in Equation (7) below:

$$Q = 2 \times max\left(1, \left\lfloor \frac{\beta(n-2)}{4} \right\rfloor\right) + 1, \tag{7}$$

where the *max* function is used to always ensure that $Q \geq 3$ as in Equation (5).

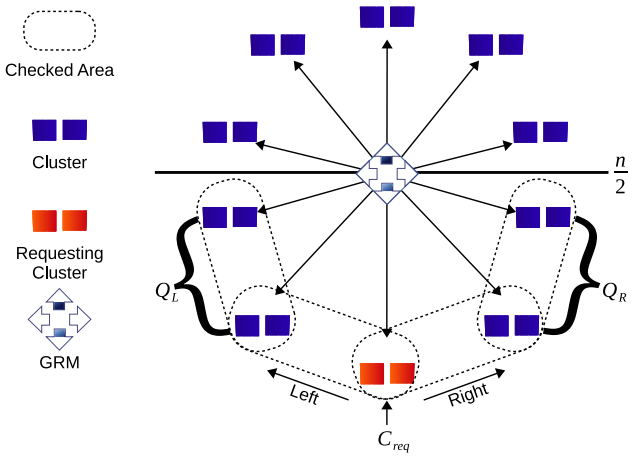


FIGURE 3. The proposed Q-based technique.

Since the key characteristic of cloud computing is its scalability, the Q-based replica placement technique enables an intelligent method to regulate storage verification needs conferring to the size of replication architecture. Specifically, it allows user access patterns to control the necessity of replica based on file popularity rather than manually predetermining the number of replica copies to be placed in the storage.

Further, an F_i copy has to be replicated in the requesting cluster only if the Q-based technique verifies the absence of F_i in the searched quorum of clusters. Then, the Q-based technique will need to determine the best data center(s) in the requesting cluster to admit that file. Moreover, if the parallel download feature is desired, the replicated file will be segmented into σ portions that have to be admitted into σ data centers with the highest merit M considering the data centers with the highest file popularity, the greatest free storage space, and the best central location in the cluster, taking into account that σ is a system parameter which needs to be preset by the system admin to decide the number of segments for a file to enable parallel downloads while $\sigma = 1$ can be used to disable parallel download feature, using Equation (8) adopted from [41]:

$$M_i^{j,k} = w_r \frac{r_i^{j,k}}{R_i^j} + w_s \frac{s^{j,k}}{S^{j,k}} + w_d \frac{D^j - d^{j,k}}{D^j}, \text{ for file } F_i \quad (8)$$

where $M_i^{j,k}$ is the merit of the k^{th} data center in the j^{th} cluster, which is counted for each data center, where the highest value of $M_i^{j,k}$ represents the best data center to replicate the requested file i. Specifically, $M_i^{j,k}$ is computed based on three important properties:

1) The highest popularity Δ_i of the requested file F_i among all data centers, which is calculated as the total number of requests $r_i^{j,k}$ for the most popular file Δ_i in data center k divide by the highest number of requests R_i^j in cluster j.
2) The greatest free storage space available among all data centers, which is calculated as the free storage space $s^{j,k}$ divided by the total storage space $S^{j,k}$ in data center k at cluster j.
3) The best central data center location calculated as the complement of one for the summation of distances $d^{j,k}$ from the k^{th} data center to all other data centers in the same cluster j divided by D^j, where D^j is the highest $d^{j,k}$ in the j^{th} cluster.

In order to allow administrators to tune their systems at the run time; w_r, w_s, and w_d are used to represent the weights of the above-mentioned three properties; the file popularity, the free storage space, and the central location; where $w_r + w_s + w_d = 1$.

During the selection of the best data centers, the calculated $M_i^{j,k}$ values will be descendingly saved in $\vec{M}_i^j$, where the best σ data centers with the highest items in the $\vec{M}_i^j$ will be kept in the list while the other items will be deleted. Subsequently, the $\vec{M}_i^j$ items will be used to admit the segments of the requested file, where the segments will be calculated using the segmentation formula adopted from [41]:

$$\vec{\sigma}_i[t] = \frac{\vec{M}_i^j[t]}{\sum_{t=1}^{\sigma} \vec{M}_i^j[t]} \text{ for each item in } \vec{M}_i^j, \quad (9)$$

where $\vec{\sigma}_i$ represents the vector that includes the segmentation percentages of F_i to be admitted into the σ data centers and t denotes the item index in the $\vec{\sigma}_i$.

For better understanding, algorithms 1 and 2 present all needed details for the proposed VRS-BQ replication strategy components at the *LRM* and the *GRM*, respectively.

E. EXPERIMENTAL SETUP

Overall system architecture of this work consist number of clusters that includes multiple data centers and one *LRM* for each cluster, where all *LRMs* are centrally controlled by one *GRM* for the entire system, as shown in Figure 4.

Specifically, the *GRM* as the root of topology is placed at the center of the cloud environment connected to the other clusters through their *LRMs* via multiple routers and links, where each cluster is connected to a local storage. Each *LRM* residing in a cluster is responsible to keep the local replica information such as logical and physical file names, file access counts, file popularity values, file locations, and master files data. Moreover, each *LRM* passes its local information to the *GRM* when replication process starts. Thus, the *GRM* will be able to decide on the location selection of file replication based on replica placement conditions.

The proposed VRS-BQ has been developed and tested using the Java-based CloudSim version 3.0.3, which is considered as one of the best simulators widely used in this domain [19]. The performance evaluation of this study adheres to few assumptions as; (1) No cluster nodes and network components failure throughout the experiment. (2) Replication files are restricted for download and no

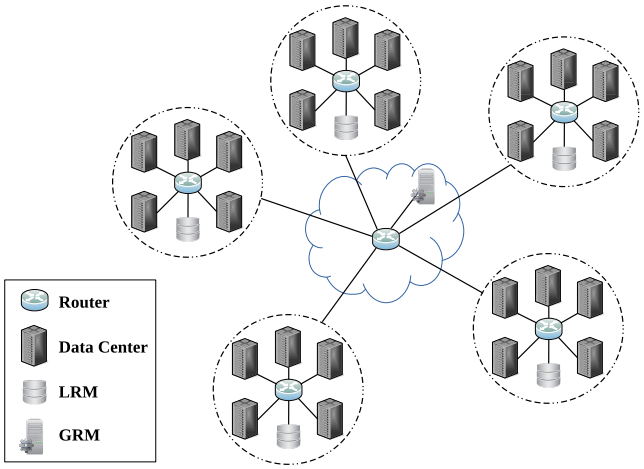


FIGURE 4. System architecture.

Algorithm 1 VRS-BQ Components at the LRM_i

Event *OnInitialization()* **do**
  $j \leftarrow$ *LRM* index assigned by the *GRM*;
  $n \leftarrow$ Number of clusters in the system;
  $m \leftarrow$ Number of files available at an LRM_j;
  $p = 0.1, q = 0.15, P_{init} = 0.005$;
  $W_{max} = \lfloor \frac{n}{2} \rfloor, \rho_{min} = \frac{1}{n}, \rho_{max} = \frac{W_{max}}{n}$;
  $F_{list} \leftarrow \{F_1, \ldots, F_m\}$;
  $\hat{F}_{list} \leftarrow \{\hat{F}_1, \ldots, \hat{F}_m\}$;
  $P_{list} \leftarrow \{P_1, \ldots, P_m\}$;
  $\vec{\Delta} \leftarrow \{\Delta_1, \ldots, \Delta_m\}$;
end
Event *OnDownloadRequest(file)* **do**
  if *file* $\notin LRM_i$ **then**
    $m = 1 + +$;
    $F_{list}[m] =$ GetFileFromGRM(*file*);
    $\hat{F}_{list}[m] = 1$);
    $P_{list}[m] = P_{init} - q$;
  else
    $\hat{F}_{list}[file] = 1 + +$;
    $P_{list}[file] = P_{list}[file] + \hat{F}_{list}[file] \times p$;
  end
  /*Calculate Popular Frequency for *file*/
  $\vec{\Delta}[file] = P_{list}[file] \times (\rho_{min} + \rho_{max})$;
  /*Save $\vec{\Delta}$ items in descending order*/
  DescendingSort($\vec{\Delta}$);
  Download(*file*, $\vec{\Delta}_j$);
end

Algorithm 2 VRS-BQ Components at the *GRM*

Event *OnInitialization()* **do**
  $n \leftarrow$ Number of clusters in the system;
  $\beta \leftarrow$ System parameter value;
  $\sigma \leftarrow$ Number of allowed file segments for parallel download;
  $\vec{\Delta}_{list} \leftarrow \{\vec{\Delta}_j, \vec{\Delta}_{j+1}, \ldots, \vec{\Delta}_n\}$;
end
Function *Download(file,* $\vec{\Delta}_j$*)*
  Keep only the top 20% of the $\vec{\Delta}_j$;
  Balanced_Quorum($\vec{\Delta}_j$);
  Save $\vec{\Delta}_j$ into $\vec{\Delta}_{list}$;
end
Function *Balanced_Quorum(file,* $\vec{\Delta}_j$*)*
  $Q = 2 \times max\left(1, \left\lfloor \frac{\beta(n-2)}{4} \right\rfloor\right) + 1$;
  /*Check availability of the *file* at the requesting cluster and its neighboring clusters subject to the Q value*/
  if *file is available at neighboring cluster* **then**
    /**file* riplica is *NOT* needed*/
    Send the requested file from the neighboring cluster;
  else
    /**file* riplica is needed*/
    /*find the best σ data centers in the requesting cluster using the data center merit M*/
    $M_i^{j,k} = w_r \frac{r_i^{j,k}}{R_i^j} + w_s \frac{s^{j,k}}{S^{j,k}} + w_d \frac{D^j - d^{j,k}}{D^j}$, for file F_i
    Save the Ms in $\vec{M}_i^j$
    Descendingly order the $\vec{M}_i^j$
    Keep only the top σ merits in the $\vec{M}_i^j$
    /*Calculate the percentages of the file that needs to be segmented into the σ data centers*/
    $\vec{\sigma}_i[t] = \frac{\vec{M}_i^j[t]}{\sum_{t=1}^{\sigma} \vec{M}_i^j[t]}$ for each item in $\vec{M}_i^j$
    Save the file segments into the relevant data centers;
    Establish parallel download of *file*;
  end
end

updates or changes of files are considered. Based on [41], we presume that weights are equally important ($w_r = w_s = w_d = \frac{1}{3}$) as configured in Table 1.

At the initial stage of replication, the master files are generated using a Zipf' distribution, while a uniform distribution is used for the file download requests. Additionally, the master files are fixed to 200 files, where every master file has one replica copy distributed randomly through the clusters prior to the experiment's start. Moreover, the simulation has been repeated 100 times with different numbers of jobs per round to get an average value close to reality.

As for the environmental setup, there are 10 clusters, each one consists of 10 data centers with a local storage size of 60 GB per cluster. Thousands of virtual machines (VMs) have been considered to ensure that at least 1 VM is allocated for each data center. All cloudlets and resources in this architecture are managed by the broker, where 200 master

files were generated using the Zipf' distribution along with one replica copy for each file. The file sizes used for this experiment are in the range of 100MB to 20GB. Moreover, jobs sending times, recognized as cloudlets 'SendTime' in CloudSim, are subject to Poisson distribution, while requests of file downloads are subject to Uniform distribution. Table 1 shows the parameters setup of the simulation:

TABLE 1. Simulation parameters.

PARAMETER	VALUE(S)
Total number of clusters	10
Total number of nodes	100
Number of nodes within the same clusters	10
Number of different files	200
Range of random file sizes	From 1 to 20GB)
Constant file sizes	1, 5 and 10GB
Number of files accessed by a Job	3-10
Round length	100
Number of intermediate nodes between 2 nodes in the same Cluster	1
Number of intermediate nodes between 2 successive Clusters	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)
w_r, w_s, w_d	1/3
Searching weight β	0 and 1
No. of segments σ	2
Storage size for every cluster nodes	60 (GB)

IV. RESULTS AND DISCUSSION

In order to evaluate the performance of the proposed solution, several experiments have been conducted as described in the previous section, where the results of these experiments are explained in the following subsections.

A. AVERAGE RESPONSE TIME (ART)

The Response Time (RT) of a certain job is defined as the time difference between sending that job and its processing-start, while the Average Response Time (ART) is defined as the summation of RT for all jobs of all users divided by the count of all jobs in the entire system, which is measured based on the formula shown in Eq. (10):

$$\text{ART} = \frac{\sum_{i=1}^{I}\sum_{k=1}^{K_i}(S_k^i - R_k^i)}{\sum_{i=1}^{I} K_i}, \tag{10}$$

where I is the total number of users, K_i is the total number of jobs belongs to the i^{th} user, k is the job index, while S_k^i and R_k^i are the sending time and the processing-start time of the k^{th} job for the i^{th} user, respectively.

Figure 5 presents results on the ART specifically measured for file download activities through the proposed VRS-BQ algorithm compared to its predecessor, the DPRS algorithm,

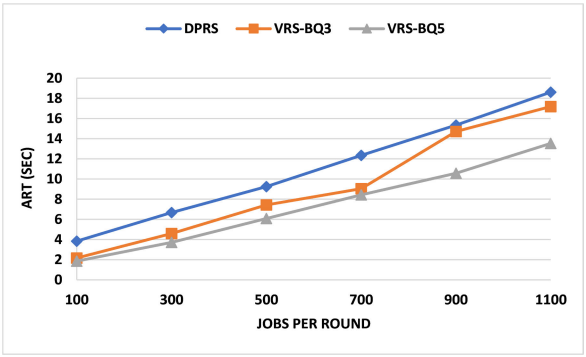


FIGURE 5. ART for random file sizes download activities.

which has been chosen as a benchmark due to the claim of outperforming other standards as in Reference [41].

For a better analytical view, results in every graph in this section include both outcomes when the administrator chooses either weightage $\beta = 0$ or $\beta = 1$. As in this research, 10 clusters were deployed, therefore the checking quorums in the environment are obligated to 3 quorums if the administrator chooses $\beta = 0$ and 5 quorums when $\beta = 1$ is selected. Hereafter, in the subsequent results, weightages of $\beta = 0$ and $\beta = 1$ will be presented as VRS-BQ3 and VRS-BQ5, respectively.

In Figure 5, the simulation was completed using *Zipf'* distribution for 200 master files generation with file size range of 1GB to 20GB and each job interval was simulated for 100 rounds. Figure 5 shows that VRS-BQ has a faster response time than DPRS for respective job intervals per round in all cases. The total average response time for file downloads shows exponential relation with the increase of jobs per round. Based on the same figure, it is obvious that VRS-BQ3 and VRS-BQ5 show about 17% and 33% faster average response time, respectively, compared to the DPRS. This significant improvement is due to the fact that DPRS includes constant criteria for computation, especially to retrieve the average popularity value in all clusters to find relationships for popular files selection. This constant criterion causes irrelevant processing operations that significantly increase the response time.

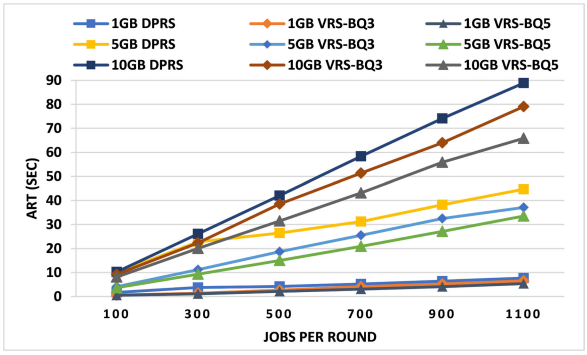
Since the file size is an important coefficient that affects job completion time, another experiment has been accomplished with constant file sizes for 100 rounds. For each time, constant file size is tested with multiple jobs per rounds iterations; 100, 300, 500, 700, 900, and 1100 jobs; in separated simulations. Although in the previous experiment, the file range was scaled from 1GB to 20GB, the measurement for constant file size is not inclusive 10Gb due to large file size in every experiment causing storage full and the results attained are not enough for the comparison between both proposed VRS-BQ and DPRS algorithm. Therefore, constant file sizes have been measured in every experiment, where the constant file sizes are limited to 1GB, 5GB, and 10GB, which was tested with multiple jobs per round to ensure fair and accurate

results obtained to verify the competitiveness of the VRS-BQ algorithm.

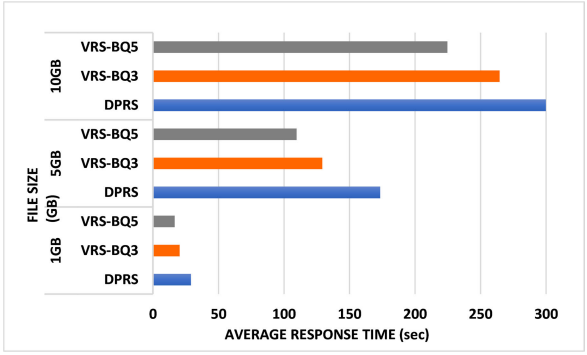
As seen in Figure 6, consistent improvements were attained in overall ART performance by the proposed VRS-BQ3 and VRS-BQ5. Specifically, the total ART improvements obtained for VRS-BQ3 and VRS-BQ5 inclusive all number of jobs per rounds, for 1Gb file size are about 30% and 43%, respectively, compared to the DPRS. As for the 5Gb file size, the response time for file download was accelerated by about 25% and 37% for VRS-BQ3 and VRS-BQ5, respectively. As for the 10Gb file size, the enhancement obtained was about 12% and 25% for VRS-BQ3 and VRS-BQ5, respectively compared to the DPRS.

This achievement is due to the proposed file selection criteria introduced by the proposed VRS-BQ, which works as a dynamic and comprehensive mechanism to find the most popular files and place them in the proper locations based on the characteristics of the underlying architecture of the data center, user behavior, and demand on files. Therefore, it provides high file availability in local sites that helps users to get their files retrieved faster than using the DPRS. Moreover, the VRS-BQ algorithm shows an efficient performance even when the number of jobs grows regardless of the file size.

Overall, VRS-BQ is able to shorten the response time due to its comprehensive strategy that determines the most popular files to be replicated. VRS-BQ mainly relies on crucial popularity factors, which made it able to ensure local availability of crucial files for faster file concurrent downloads. Contrarily, DPRS relies on irrelevant criteria that depend on complex iterative processes based on some unimportant information to find popular files, which yields unnecessary computations leading to higher response time.



(a) ART vs. Jobs/Round for Constant File Sizes



(b) Total ART vs. Constant File Sizes

FIGURE 6. ART and total ART for constant file sizes.

1) POPULARITY-BASED FILE SELECTION

Unlike other research, this experiment has done a further investigation on the impact of the proposed VRS-BQ algorithm on popular files selection to show the difference in behavior between those studied algorithms. This is because the VRS-BQ strategy has an influence on the selection of popular files, therefore, subsequent experiments have been conducted to observe the effect of the proposed algorithm on popular files selection. To achieve that, two (2) simulation experiments have been run for 100 rounds with different job intervals vary from 100 to 1100 jobs per round, first experiment applied a random range of file sizes while the second experiment applied constant file sizes, as 1Gb, 5Gb, and 10Gb. Figure 7 shows the total popular files that were not selected in the entire replication process.

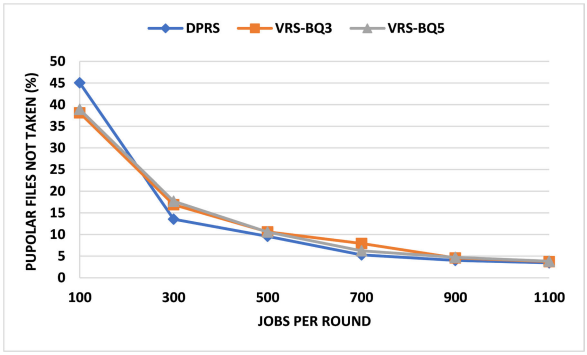
As seen in Figure 7(a), all algorithms are very close to each other in terms of the average selected popular files. However, the proposed algorithm shows a significant difference in the cases of huge files such as 5GB and 10GB. As seen in Figure 7(b), the proposed algorithm shows less selection of large files for replication which significantly saves the storage capacities of data centers.

From Figure 7(b), it is very clear that the proposed algorithm shows an opposite behavior of the DPRS, in which the proposed algorithm selects less percentage of popular files for replication when file sizes are large, while the DPRS selects a high percentage of popular files for replication when file sizes are large. This makes the DPRS more greedy for storage and introduces huge network traffic due to the frequent exchanging of large files between data centers which negatively affects the response time as well.

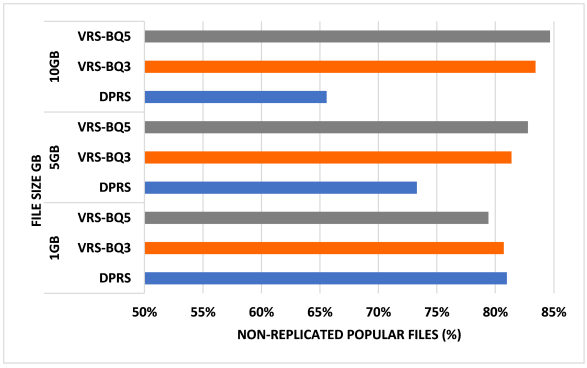
Hence, it can be concluded that the proposed VRS-BQ is able to achieve better response time with a slight difference in the selection of popular files in the entire replication process. Precisely, the proposed popularity-based file selection criteria of the VRS-BQ, based on Eq. (1) and Eq. (2) is able to deliver a significant impact by dynamically selecting candidate files based on dynamic user access pattern, which efficiently improves the response time and storage consumption while maintaining the other performance aspects as it is clearly presented in the coming graphs.

2) FILE REPLICATION

The following experimental results are to confirm that the VRS-BQ technique is not devoted only to reducing the average response time for file downloads, but can improve the overall replication time as well. In Figure 8(b), an evaluation for total replication performance has been measured for three (3) constant file size classes; 1GB, 5GB, and 10GB. Each experiment for each file size class has been run for

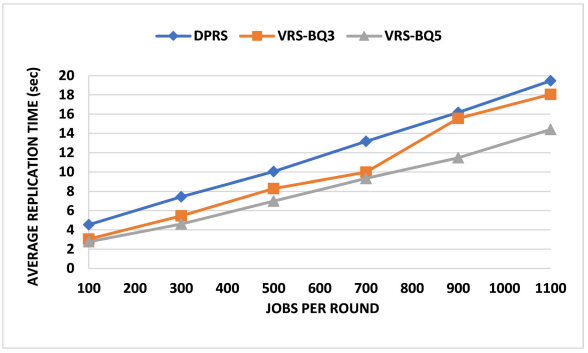


(a) Non-Replicated Popular Files per job/Round for Random File Sizes

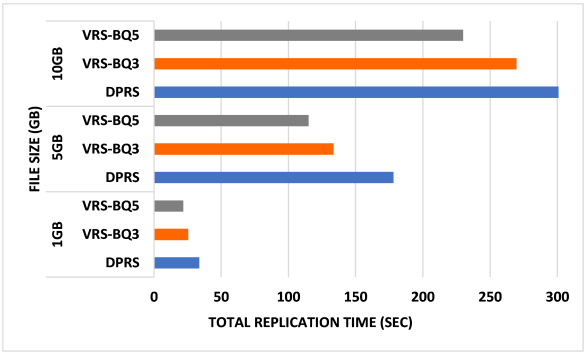


(b) Non-Replicated Popular Files per Constant File Sizes

FIGURE 7. Non-replicated popular files per random and constant file sizes.



(a) Average Replication Time per Jobs/Round for Random File Sizes



(b) Total Average Replication Time for Constant File Sizes

FIGURE 8. Average replication time per random and constant file sizes.

100 simulation times with incremental jobs; as 100, 300, 500, 700, 900, 1100 jobs per round.

Figure 8(b) proves that the proposed VRS-BQ has outperformed the DPRS with significant improvements, where it shows about 24% and 35% improvement for VRS-BQ3 and VRS-BQ5, respectively, compared to the DPRS for 1GB file size class. As for the 5GB file size class, approximately 25% and 35% of acceleration have been achieved by the proposed BQ3 and BQ5, respectively, compared to the DPRS. As for the 10GB file size class, 11% and 24% of replication time reduction have been attained by BQ3 and BQ5, respectively, compared to the DPRS algorithm.

Even though the proposed VRS-BQ technique has introduced additional conditions on the file replication and verification process to elect the best cluster to place the candidate replicas, its time overhead contributed to the replication process is very negligible. As shown in Figure 8, the proposed VRS-BQ algorithm can efficiently place the popular files at the most appropriate location that significantly reduces the average response time and average replication time for file downloads in all tested cases. This improvement has been achieved by taking the user-file demand pattern into account in the proposed VRS-BQ algorithm.

B. STORAGE CONSUMPTIONS

As well-known, storage consumption is a key metric for cloud systems, which is usually measured to observe how much storage is consumed for the entire replication process in the cloud environment. The proposed VRS-BQ implements a novel and distinctive way to place a file replica in storage nodes based on three (3) main criteria including the available storage in the targeted nodes, as explained in III-D, hence, it disseminates replicas efficiently through cluster nodes. Eq. 11 has been used to measure the storage consumption of replication storage in the entire system, where SC is the total storage consumption of the entire system, sc_j^k and S_j^k denote the storage consumption and the total available storage of the k^{th} data center in the j^{th} cluster, respectively.

$$SC = \sum_{j=1}^{n} \sum_{k=1}^{K} \frac{sc_j^k}{S_j^k}, \tag{11}$$

where K is the number of data centers in every cluster j while n is the total number of clusters in the entire system.

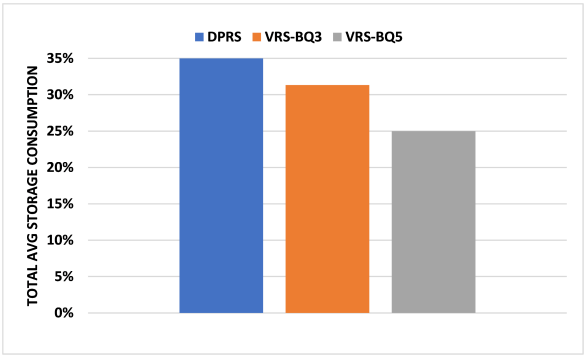
Figure 9(a), shows a significant reduction in storage consumption due to the proposed VRS-BQ algorithm that is capable to reduce storage consumption in replication environments based on user download patterns. The bar chart depicts that the DPRS storage consumption is 35%, while it shows 31% and 25% for VRS-BQ3 and VRS-BQ5, respectively. More specifically, the proposed VRS-BQ saves about 10% of storage on average compared to the DPRS algorithm, which is considered a significant contribution to help cloud tenants efficiently utilize their resources.

BQ5 creates a smaller number of replica copies, thus, it contributes more storage-saving while BQ3 performs a lesser number of quorum-checking that results in a greater number of replica copies which leads to consuming more storage space. However, the dynamicity of the VRS-BQ technique to check and choose appropriate replica placement is mainly contributing to the space-saving in storage nodes compared to DPRS. The main issue with DPRS is that it consumes more storage space due to its replica placements in storage nodes without relying on an efficient technique.

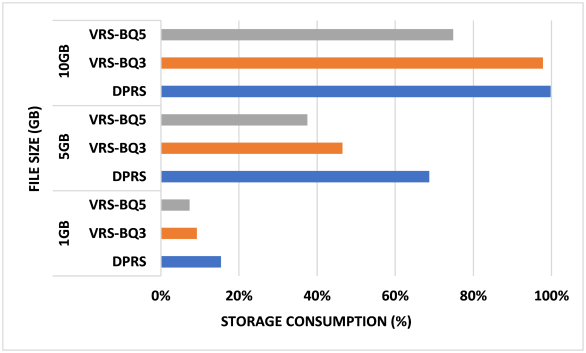
Moreover, Figure 9(b) explores more specifically the storage consumption based on different file sizes, through conducting simulation using constant file size classes to ensure that the obtained results are accurate and reliable. As it is clear in Figure 9(b), the proposed algorithm reduces the storage consumption up to 25% compared to DPRS.

Figure 9(b), demonstrates experimental results for three (3) constant file size classes in different simulation runs to attain and prove precise outcomes to avoid any possible unseen impact of dissimilar file sizes. The proposed VRS-BQ shows significant improvement in terms of storage-saving, in which, it saves up to 35% of the storage compared to the DPRS algorithm.

Unlike DPRS, the proposed VRS-BQ algorithm with its unique placement technique intelligently interacts with the closest clusters before deciding to allocate a new replica. Despite the extra condition verification process, VRS-BQ persistently reduces storage consumption with better replication time. Whereas, the DPRS shows higher storage consumption due to its behavior that stores popular files in all clusters every time the files are identified as popular.



(a) Storage Consumption for Random File Sizes



(b) Storage Consumption for Constant File Sizes

FIGURE 9. Storage consumption in cases of random and constant file sizes.

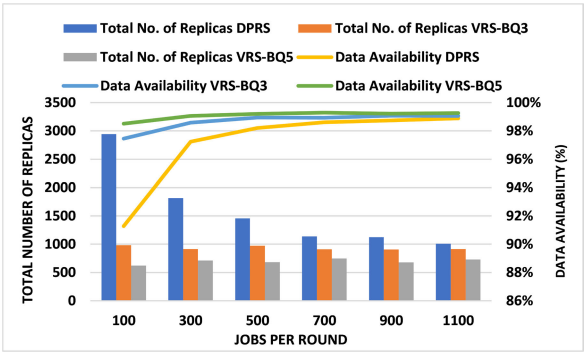


FIGURE 10. Data availability vs. Number of replicas.

C. DATA AVAILABILITY

Experimental results prove that the proposed VRS-BQ algorithm is efficient to minimize the number of needed replicas that keep the data availability as high as possible. Based on the scenario of random files ranging from 1GB to 20GB, Figure 10 shows the average percentage of data availability versus the total number of replica copies for each job per round. As it is clear, the proposed VRS-BQ algorithm achieved significantly high data availability ranging from 97% to 99% in most cases, while the benchmark DPRS ranged from 91% to 99%. Moreover, the data availability average of the DPRS decreases significantly whenever the number of jobs per round is getting lower, while the proposed VRS-BQ decreases its data availability smoothly due to its dependency on file downloads patterns. More interestingly, the proposed VRS-BQ is able to maintain a high level of data availability with less number of replicas because the VRS-BQ technique is able to dynamically decide the essential number of replicas based on user access patterns and it properly selects the best location to place the replica copies. Based on Figure 10, it is very clear that the proposed VRS-BQ could significantly reduce the storage consumption while maintaining a high level of data availability.

Ultimately, this research developed a novel VRS-BQ algorithm that improves multiple performance perspectives in cloud environments compared to the existing DPRS algorithm. Based on the discussed results in this section, it is very clear that the proposed VRS-BQ could significantly outperform the existing DPRS with faster response time for file downloads and minimum storage consumption which by its role reduces the entire replication time due to replicating the most popular files only to maximize the data availability.

V. CONCLUSION

Data Replication is one of the popular strategies used for over a decade that has become well-known in cloud environments.

Cloud as a data management platform enables multiple services for numerous organizations including data replication strategies. As data growing exponentially, researchers are proactively developing new techniques to advance these replication strategies that still suffer from some drawbacks despite their contributions to achieving the cloud objectives.

This work proposes a dynamic replication strategy with a unique replica placement technique, namely VRS-BQ, which mitigates the most negative consequences in replication performance. The main contributions of this work are three folds:

- First, it proposes an efficient popularity-based files selection strategy that dynamically adjusts its process based on the frequency of requested files and literally optimizes the average response time for file downloads, where VRS-BQ achieves about 25% optimization in terms of average response time and accelerated 25% of replication time compared to the DPRS.
- Second, it introduces a unique replica placement technique, where unnecessary replications are avoided and the best replica locations are intelligently determined to save storage space and accelerate the entire replication process. On average, VRS-BQ reduces approximately 20% of space in cloud storage while maintaining high level of data availability in most cases.
- Third, a comprehensive performance evaluation has been carried out, where experimental results proved that the proposed VRS-BQ could significantly outperform and replace the existing DPRS algorithm.

These achievements are due to the unique popularity factor calculation technique of the proposed VRS-BQ that efficiently selects the most important files and places them near to where they were mostly requested, which leads to faster user download with faster response time. Additionally, the VRS-BQ succeeded to avoid creating unnecessary replica copies due to its unique approach which dramatically reduces storage consumption and replication time.

Even though energy optimization has not been directly considered in this paper, it has been implicitly considered since power consumption is a product of data processing and exchange. Thus, building an algorithm such as the proposed VRS-BQ, which minimizes unnecessary data replication and ensures placement of file replicas near to where it was mostly requested, contributes explicitly to reducing processing overhead and network traffic that implicitly minimize the power consumption.

In nutshell, this paper has successfully addressed some prominent issues and provided an efficient solution for them, namely VRS-BQ, which accelerates response time, reduces replication time, and minimizes storage consumption while maintaining a high level of data availability due to its unique popularity-based technique. Moreover, the VRS-BQ algorithm is worthwhile to be implemented in production cloud environments which would apparently contribute to better performance for industrial practices to guarantee business continuity with uninterrupted data retrievals.

VI. FUTURE WORK

Despite the VRS-BQ contribution to minimizing the response time, increasing data availability, and minimizing storage consumption, there is still room for improvement. Specifically, the VRS-BQ should consider calculating the 20% of the $\vec{\Delta}_j$ locally at the cluster itself to minimize the network traffic among the clusters and their *GRM* that would relieve the *GRM* by reducing its processing load, which by its role will reduce the response time of the entire system.

REFERENCES

[1] S. Luo, G. Zhang, C. Wu, S. U. Khan, and K. Li, "Boafft: Distributed deduplication for big data storage in the cloud," *IEEE Trans. Cloud Comput.*, vol. 8, no. 4, pp. 1199–1211, Oct. 2020.

[2] S. George and E. B. Edwin, "A review on data replication strategy in cloud computing," in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res. (ICCIC)*, Dec. 2017, pp. 1–4.

[3] M. Khan, X. Wu, X. Xu, and W. Dou, "Big data challenges and opportunities in the hype of industry 4.0," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[4] Q. Xia, W. Liang, and Z. Xu, "QoS-aware data replications and placements for query evaluation of big data analytics," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–7.

[5] A. Sheth, "Internet of Things to smart IoT through semantic, cognitive, and perceptual computing," *IEEE Intell. Syst.*, vol. 31, no. 2, pp. 108–112, Mar. 2016.

[6] R. Kaur, I. Chana, and J. Bhattacharya, "Data deduplication techniques for efficient cloud storage management: A systematic review," *J. Supercomput.*, vol. 74, no. 5, pp. 2035–2085, May 2018.

[7] H. Yuan, X. Chen, T. Jiang, X. Zhang, Z. Yan, and Y. Xiang, "DedupDUM: Secure and scalable data deduplication with dynamic user management," *Inf. Sci.*, vol. 456, pp. 159–173, Aug. 2018.

[8] C. Yang, Q. Huang, Z. Li, K. Liu, and F. Hu, "Big Data and cloud computing: Innovation opportunities and challenges," *Int. J. Digit. Earth*, vol. 10, no. 1, pp. 13–53, 2017.

[9] H. Cai, B. Xu, L. Jiang, and A. V. Vasilakos, "IoT-based big data storage systems in cloud computing: Perspectives and challenges," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 75–87, Jan. 2017.

[10] Z. Yan, L. Zhang, W. Ding, and Q. Zheng, "Heterogeneous data storage management with deduplication in cloud computing," *IEEE Trans. Big Data*, vol. 5, no. 3, pp. 393–407, Sep. 2019.

[11] L. Singh and J. Malhotra, "A survey on data placement strategies for cloud based scientific workflows," *Int. J. Comput. Appl.*, vol. 141, no. 6, pp. 30–33, May 2016.

[12] M. Lauer, "Data security in the cloud why cloud computing?" *Seminar*, vol. 7, no. 4, pp. 61–64, 2011.

[13] Z. Yan, W. Ding, X. Yu, H. Zhu, and R. H. Deng, "Deduplication on encrypted big data in cloud," *IEEE Trans. Big Data*, vol. 2, no. 2, pp. 138–150, Jun. 2016.

[14] J. Ma, G. Wang, and X. Liu, "DedupeSwift: Object-oriented storage system based on data deduplication," in *Proc. 15th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun., 10th IEEE Int. Conf. Big Data Sci. Eng., 14th IEEE Int. Symp. Parallel Distrib.*, Aug. 2016, pp. 1069–1076.

[15] C. B. Tan, M. H. A. Hijazi, Y. Lim, and A. Gani, "A survey on proof of retrievability for cloud data integrity and availability: Cloud storage state-of-the-art, issues, solutions and future trends," *J. Netw. Comput. Appl.*, vol. 110, pp. 75–86, May 2018.

[16] N. K. Nivetha and D. Vijayakumar, "Modeling fuzzy based replication strategy to improve data availabiity in cloud datacenter," in *Proc. Int. Conf. Comput. Technol. Intell. Data Eng. (ICCTIDE)*, Jan. 2016, pp. 1–6.

[17] Q. Liu, G. Wang, and J. Wu, "Consistency as a service: Auditing cloud consistency," *IEEE Trans. Netw. Service Manage.*, vol. 11, no. 1, pp. 25–35, Mar. 2014.

[18] J. Wang, H. Wu, and R. Wang, "A new reliability model in replication-based big data storage systems," *J. Parallel Distrib. Comput.*, vol. 108, pp. 14–27, Oct. 2017.

[19] M. R. Djebbara and H. Belbachir, "Cost function based on analytic hierarchy process for data replication strategy in cloud environment," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 9, pp. 2638–2648, 2018.

[20] F. Xie, J. Yan, and J. Shen, "Towards cost reduction in cloud-based workflow management through data replication," in *Proc. 5th Int. Conf. Adv. Cloud Big Data (CBD)*, Aug. 2017, pp. 94–99.
[21] Y. Shao, C. Li, Z. Fu, L. Jia, and Y. Luo, "Cost-effective replication management and scheduling in edge computing," *J. Netw. Comput. Appl.*, vol. 129, pp. 46–61, Mar. 2019.
[22] B. Spinnewyn, J. F. Botero, and S. Latre, "Cost-effective replica management in fault-tolerant cloud environments," in *Proc. 13th Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2017, pp. 1–9.
[23] H. E. Ciritoglu, T. Saber, T. S. Buda, J. Murphy, and C. Thorpe, "Towards a better replica management for Hadoop distributed file system," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jul. 2018, pp. 104–111.
[24] F. Hanandeh, M. Khazaaleh, H. Ibrahim, and R. Latip, "CFS: A new dynamic replication strategy for data grids," *Int. Arab J. Inf. Technol.*, vol. 9, no. 1, pp. 94–99, 2012.
[25] B. A. Milani and N. J. Navimipour, "A systematic literature review of the data replication techniques in the cloud environments," *Big Data Res.*, vol. 10, pp. 1–7, Dec. 2017.
[26] S. U. R. Malik, S. U. Khan, S. J. Ewen, N. Tziritas, J. Kolodziej, A. Y. Zomaya, S. A. Madani, N. Min-Allah, L. Wang, C.-Z. Xu, Q. M. Malluhi, J. E. Pecero, P. Balaji, A. Vishnu, R. Ranjan, S. Zeadally, and H. Li, "Performance analysis of data intensive cloud systems based on data management and replication: A survey," *Distrib. Parallel Databases*, vol. 34, no. 2, pp. 179–215, Jun. 2016.
[27] F. Castro-Medina, L. Rodriguez-Mazahua, M. A. Abud-Figueroa, C. Romero-Torres, L. A. Reyes-Hernandez, and G. Alor-Hernandez, "Application of data fragmentation and replication methods in the cloud: A review," in *Proc. Int. Conf. Electron., Commun. Comput. (CONIELECOMP)*, Feb. 2019, pp. 47–54.
[28] N. K. Gill and S. Singh, "A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers," *Future Gener. Comput. Syst.*, vol. 65, pp. 10–32, Dec. 2016.
[29] Z. Zeng and B. Veeravalli, "Optimal metadata replications and request balancing strategy on cloud data centers," *J. Parallel Distrib. Comput.*, vol. 74, no. 10, pp. 2934–2940, Oct. 2014.
[30] S.-Q. Long, Y.-L. Zhao, and W. Chen, "MORM: A multi-objective optimized replication management strategy for cloud storage cluster," *J. Syst. Archit.*, vol. 60, no. 2, pp. 234–244, Feb. 2014.
[31] A. Cidon, R. Stutsman, S. Rumble, S. Katti, J. Ousterhout, and M. Rosenblum, "MinCopysets: Derandomizing replication in cloud storage," in *Proc. Netw. Syst. Design Impementation (NSDI)*, 2013, pp. 1–5.
[32] R. Latip, M. Othman, A. Abdullah, H. Ibrahim, and M. N. Sulaiman, "Quorum-based data replication in grid environment," *Int. J. Comput. Intell. Syst.*, vol. 2, no. 4, pp. 386–397, Dec. 2009.
[33] B. A. Milani and N. J. Navimipour, "A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions," *J. Netw. Comput. Appl.*, vol. 64, pp. 229–238, Apr. 2016.
[34] D.-W. Sun, G.-R. Chang, S. Gao, L.-Z. Jin, and X.-W. Wang, "Modeling a dynamic data replication strategy to increase system availability in cloud computing environments," *J. Comput. Sci. Technol.*, vol. 27, no. 2, pp. 256–272, Mar. 2012.
[35] W. Li, Y. Yang, and D. Yuan, "Ensuring cloud data reliability with minimum replication by proactive replica checking," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1494–1506, May 2016.
[36] Y. Zhang, X. Nie, J. Jiang, W. Wang, K. Xu, Y. Zhao, M. J. Reed, K. Chen, H. Wang, and G. Yao, "BDS+: An inter-datacenter data replication system with dynamic bandwidth separation," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 918–934, Apr. 2021.
[37] R. Latip, T. Mizan, N. F. Ghazali, R. A. Kadir, and F. A. Hanandeh, "Replica control protocol: Triple quorum replication (TQR) in data grid," in *Proc. 5th Int. Conf. Comput. Sci. Inf. Technol.*, Mar. 2013, pp. 303–307.
[38] M. A. Fazlina, R. Latip, H. Ibrahim, and A. Abdullah, "A review: Replication strategies for big data in cloud environment," *Int. J. Eng. Technol.*, vol. 7, no. 4.31, pp. 357–362, 2018.
[39] S. Souravlas and A. Sifaleras, "Binary-tree based estimation of file requests for efficient data replication," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 7, pp. 1839–1852, Jul. 2017.
[40] M. Hussein and M. Mousa, "A light-weight data replication for cloud data centers environment," *Int. J. Eng. Innov. Technol.*, vol. 1, no. 6, pp. 169–175, 2012.
[41] N. Mansouri, M. K. Rafsanjani, and M. M. Javidi, "DPRS: A dynamic popularity aware replication strategy with parallel download scheme in cloud environments," *Simul. Model. Pract. Theory*, vol. 77, pp. 177–196, Sep. 2017.
[42] K. A. Kumar, A. Quamar, A. Deshpande, and S. Khuller, "SWORD: Workload-aware data placement and replica selection for cloud data management systems," *VLDB J.*, vol. 23, no. 6, pp. 845–870, Dec. 2014.
[43] N. Mansouri, "Adaptive data replication strategy in cloud computing for performance improvement," *Frontiers Comput. Sci.*, vol. 10, no. 5, pp. 925–935, Oct. 2016.
[44] N. Mansouri, M. M. Javidi, and B. M. H. Zade, "Hierarchical data replication strategy to improve performance in cloud computing," *Frontiers Comput. Sci.*, vol. 15, no. 2, pp. 1–17, Apr. 2021.
[45] S. N. John and T. T. Mirnalinee, "A novel dynamic data replication strategy to improve access efficiency of cloud storage," *Inf. Syst. e-Bus. Manage.*, vol. 18, no. 3, pp. 405–426, Sep. 2020.
[46] N. Mansouri, M. M. Javidi, and B. M. H. Zade, "A CSO-based approach for secure data replication in cloud computing environment," *J. Supercomput.*, vol. 77, no. 6, pp. 5882–5933, Jun. 2021.
[47] R. Salem, M. A. Salam, H. Abdelkader, and A. A. Mohamed, "An artificial bee colony algorithm for data replication optimization in cloud environments," *IEEE Access*, vol. 8, pp. 51841–51852, 2020.
[48] X. Fu, J. Li, W. Liu, S. Deng, and J. Wang, "Data replica placement policy based on load balance in cloud storage system," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Autom. Control Conf. (ITNEC)*, Mar. 2019, pp. 682–685.
[49] S. Limam, R. Mokadem, and G. Belalem, "Data replication strategy with satisfaction of availability, performance and tenant budget requirements," *Cluster Comput.*, vol. 22, no. 4, pp. 1199–1210, 2019.
[50] N. Mansouri and M. M. Javidi, "A hybrid data replication strategy with fuzzy-based deletion for heterogeneous cloud data centers," *J. Supercomput.*, vol. 74, no. 10, pp. 5349–5372, Oct. 2018.
[51] Y. Ebadi and N. J. Navimipour, "An energy-aware method for data replication in the cloud environments using a Tabu search and particle swarm optimization algorithm," *Concurrency Comput., Pract. Exper.*, vol. 31, no. 1, p. e4757, Jan. 2019.
[52] R. Li, J. Zhang, and W. Shen, "Replicas strategy and cache optimization of video surveillance systems based on cloud storage," *Future Internet*, vol. 10, no. 4, p. 34, Apr. 2018.
[53] A. Bowers, C. Liao, D. Steiert, D. Lin, A. Squicciarini, and A. Hurson, "Detecting suspicious file migration or replication in the cloud," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 1, pp. 296–309, Jan. 2021.
[54] S. S. M. Nik, M. Naghibzadeh, and Y. Sedaghat, "Task replication to improve the reliability of running workflows on the cloud," *Cluster Comput.*, vol. 24, no. 1, pp. 343–359, Mar. 2021.
[55] C. Li, J. Bai, Y. Chen, and Y. Luo, "Resource and replica management strategy for optimizing financial cost and user experience in edge cloud computing system," *Inf. Sci.*, vol. 516, pp. 33–55, Apr. 2020.
[56] H. Khalajzadeh, D. Yuan, B. B. Zhou, J. Grundy, and Y. Yang, "Cost effective dynamic data placement for efficient access of social networks," *J. Parallel Distrib. Comput.*, vol. 141, pp. 82–98, Jul. 2020.
[57] P. J. Kumar, V. University, and P. Ilango, "Data replication in current generation computing environment," *Int. J. Eng. Trends Technol.*, vol. 45, no. 10, pp. 488–492, Mar. 2017.
[58] M. Yi, J. Wei, and L. Song, "Efficient integrity verification of replicated data in cloud computing system," *Comput. Secur.*, vol. 65, pp. 202–212, Mar. 2017.
[59] J. Li, P. Zhang, Y. Li, W. Chen, Y. Liu, and L. Wang, "A data-check based distributed storage model for storing hot temporary data," *Future Gener. Comput. Syst.*, vol. 73, pp. 13–21, Aug. 2017.

FAZLINA MOHD ALI received the bachelor's degree in computer science (networking) from Universiti Putra Malaysia, in 2006, where she is currently pursuing the Ph.D. degree with the Department of Communication Technology and Networks, and the master's degree in science (information technology) from Universiti Teknologi MARA, Shah Alam, in 2012. She is attached to the Government of Malaysia for more than 13 years, as a Senior Officer of Information Technology. Her research interests include parallel and distributed computing, data replication, cloud computing, big data, and deduplication.

ROHAYA LATIP (Member, IEEE) received the Bachelor of Computer Science degree from Universiti Teknologi Malaysia, in 1999, and the M.Sc. degree in distributed systems and the Ph.D. degree in distributed database from Universiti Putra Malaysia. She was also the Head of the HPC Section at Universiti Putra Malaysia, from 2011 to 2012, and consulted the Campus Grid Project and also the Wireless for Hostel in Campus UPM Project. She is currently an Associate Professor with the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. She is also currently the Head of the Department of Communication Technology and Networks. She is a Co-Researcher at the Institute for Mathematic Research (INSPEM). Her research interests include big data, cloud and grid computing, network management, and distributed database.

MOHAMED A. ALRSHAH (Senior Member, IEEE) received the B.Sc. degree in computer science from Naser University, Libya, in 2000, and the M.Sc. and Ph.D. degrees in communication technology and networks from Universiti Putra Malaysia (UPM), in May 2009 and February 2017, respectively. Currently, he is a Senior Lecturer with the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, UPM. He has published a number of articles in high impact factor scientific journals. His research interests include high-speed TCP protocols, high-speed wired and wireless networks, parallel and distributed algorithms, WSN, the IoT, and cloud computing.

AZIZOL ABDULLAH received the M.Sc. degree in engineering (telematics) from The University of Sheffield, U.K., in 1996, and the Ph.D. degree in distributed system from Universiti Putra Malaysia, in 2010. He is currently a Senior Lecturer with the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. He has been appointed as a Fellow Researcher at the ITU-UUM Asia Pacific Centre of Excellence for Rural ICT Development (ITU-UUM). His main research interests include cloud and grid computing, network security, wireless and mobile computing, and computer networks.

HAMIDAH IBRAHIM received the Ph.D. degree in computer science from the University of Wales, Cardiff, U.K., in 1998. She is currently a Full Professor with the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia (UPM). Her current research interests include databases (distributed, parallel, mobile, biomedical, and XML) focusing on issues related to integrity maintenance/checking, ontology/schema/data integration, ontology/schema/data mapping, cache management, access control, data security, transaction processing, query optimization, query reformulation, preference evaluation—context-aware, information extraction, concurrency control, data management in grid, and knowledge-based systems.