Hindawi
Scientific Programming
Volume 2022, Article ID 2496269, 13 pages
https://doi.org/10.1155/2022/2496269

Research Article

Fast Data Access through Nearest Location-Based Replica Placement

Umar Shoaib,[1] Muhammad Junaid Arshad,[1] Hasan Ali Khattak,[2] Maryam Ezat Ullah,[2] Ahmad Almogren,[3] and Sikandar Ali[4]

[1] *Department of Computer Science, University of Gujrat, Gujrat 15213, Pakistan*
[2] *School of Electrical Engineering & Computer Science, National University of Sciences & Technology (NUST), Islamabad 44000, Pakistan*
[3] *Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia*
[4] *Department of Information Technology, University of Haripur, Haripur 22620, Pakistan*

Correspondence should be addressed to Hasan Ali Khattak; hasan.alikhattak@seecs.edu.pk and Sikandar Ali; sikandar@cup.edu.cn

Received 10 November 2021; Accepted 22 February 2022; Published 16 March 2022

Academic Editor: Rahman Ali

Nowadays, millions of telecommunication devices, IoT sensors, and web services, especially social media sites, are producing big data every second. Such applications with massive data-generating capabilities need to access these data quickly. Among other approaches, cloud computing provides content delivery networks, which utilize data replication for better latency to such real-time applications. Fast processing, storage, and timely analysis of these data are the challenge for most of these future Internet applications. However, cloud computing is the ultimate storage and processing paradigm to resolve such issues and to deal with big data, with the same speed in which data are being produced. Moreover, cloud computing technology has been evolved with the tag line of "Everything as A Service" now, and for powering all these services, provision of data is a compulsory and significant task. To provide end user with easy and fast access to data, the cloud maintains backup and replicates the copies in multiple data centers. The geographical locations of these data centers where data are being placed have a profound impact on data access time. To deal with the challenge of effective data replication and to reduce data access time to a minimum, we propose a genetic algorithm (GA)-based technique to suggest and store data on nearly located data centers. The proposed algorithm improves the access time, and thus, the efficiency of cloud servers by providing quality of service (QoS) to end user.

1. Introduction

Cloud computing provides dynamic and scalable virtualization resources to users that can be either on-demand or pay-per-use. These resources can be accessed through Internet from anywhere at any time [1]. The resources include software platforms, storage, hardware, networks, and applications as on-demand services. Data are an important part of these sharable resources because big data is being produced every second by heterogeneous communication devices and IoT sensors. Applications of cloud computing and big data range from social networks, healthcare informatics, smart city applications, and urban planning to name a few [2, 3, 4].

Cloud platform provides infinite storage, processing power, and information services for both individual and enterprises [5]. Virtualization is the most important perspective of cloud computing, which plays an important role to access resource pool in pay-per-use fashion. Capital investment can be removed through virtualization [6]. Cloud computing will not be affordable without deployment of virtualization techniques.

That is why virtualization is on the topmost layer of cloud infrastructure. Resources like hardware, software, network, storage, and operating system can be provided to end users through these techniques where they can utilize the resources regardless of their geographic placement [7]. The main components of cloud computing are users, data centers, and distributed servers. Distributed servers are located on dispersed geographical regions. Multitenancy is the ability for users to access anywhere at any time anything through the Internet.

Hypervisors are a combination of hardware, software, or firmware, which defines virtual partitioning that runs on hardware. It is a virtual machine manager that enhances the capabilities to run multiple operating systems on a system and allocating resources to each operating system without any interaction. Likewise, hypervisors are basically groups of hardware that hosts multiple virtual machines and possess all necessary information to make those virtual machines work [1]. CPU, RAM, and disk drives are a set of hardware shared among multiple operating systems. Hypervisors have capability to control all the systems. In addition, when the number of operating systems increases, the risk is also increased.

The protection techniques like abstraction isolation, state restore, transience, and external monitoring are being used for data abstraction. For this purpose, virtual machines are allocated in their own bound resources for protection. This abstraction provides extra security by restricting hardware resource access. Operating system runs on different machines with alternate configurations. Isolation allows each guest OS to run without any dependency on host machine.

The isolation also prevents attacks on one VM, which may affect the other VM running on server or host OS. State restore can easily recover or restore previous state of the VM in OS.

As contents of VM are stored on virtual disks, backups are maintained after committing each change in server or host OS. In the case of attack, VM can restore to its previous state [8]. Transience is the ability to remotely turn on or off a system whenever required. Minimizing the timing of the server OS can prevent malicious attacks. For example, if a malicious virus affects a computer, the online VM will be affected as well, but the online systems are affected more than offline ones. External monitoring is required to observe VM and to detect attacks occurring outside the VM. Research community is actively working on advanced protection techniques to monitor the activities of guest systems [9].

Data replication on different geographical locations provides easy access to data and prevents data loss but also cause data access latency. To deal such issues, a general mechanism is placed to predetermine a number of replicas in different distributed and dispersed clouds in order to decrease response time for users [10, 11]. Deployment of private data center is costly that is why virtualization techniques can provide cheap, secure, and reliable services. As the number of devices and users increases, centralized approach may not help in such scenario and create several issues due to heavy traffic, high bandwidth usage, latency, and delayed response issues.

Delay and latency issues increase with a growing number of devices and users [12]. These issues might affect the performance of cloud. The centralized approach might fail due to high traffic or increasing number of users, and devices might effect on efficiency of cloud in specialized applications such as modern vehicular cloud [13, 14].

In order to increase file availability and minimize latency across geographically distributed cloud systems and to ease resource sharing, replica placement and selection strategy must be incorporated into such systems. Increasing performance of data access in distributed systems is using replication [15]. Replicating multiple copies of files in different places increases the performance of data access and minimizing response time [16].

Replicas in data replication are identical copies of data that are kept at different sites that are geographically distributed and data replication manages huge data by producing replicas. Most important aspect of replica placement is to choose location where to place replica so that access time could be improved. This work mainly contributes by giving out a detailed outline for the data replication techniques along with proposal of an enhanced genetic algorithm in order to improve the quality of service by efficiently placing the replica placement in cloud data centers.

The remaining paper is structured as follows: Section 2 explains the domain with the help of state-of-the-art related works. Section 3 details the working of replica selection mechanisms in the cloud environments. Section 4 gives information about the replication algorithms and their inner workings when taking into context the middle-ware software. Moreover, our proposed genetic algorithm-based replication algorithm is proposed in Section 5. This is further followed by experimental results and discussion based on the initial proof of concept in Section 6, whereas Section 7 concludes the paper and indicates limitations of the approach with directions for future work.

2. Related Work

Finding the best suitable location for replica placement is most crucial and important in replica placement because this decision can minimize latency issues [17]. It is not cost-effective way for all users to access a file from one data center. Consequently, it will lead toward increase of data access latency. These issues get even worse when we want to share large volume of data with limited storage and network bandwidth. Since the number of users is in huge amount, it will cause latency if all users access data or slice from a single center.

Replication is a strategy that confirms the efficient access with less bandwidth consumption and latency. Creating replicas minimizes latency issue by diverting traffic to other data centers, thus minimizing waiting and response time, and overall, overhead is distributed among several data centers instead of overloading one data center. Data replication is necessary for resolving issues of delay in data access [12].

It is most important in distributed environment where overhead of managing replicas is a challenging issue. The gossip algorithm is a communication-based replication algorithm in which participants have same values or common state. Moreover, the transmission of duplicate information leads to enormous wastage of network capacity, computing, and bandwidth resources [18]. A suitable number of replicas are stored on each node as processing and computation power of every node is different [19]. Chen et al. [20] developed a cooperative replication scheme, weighted dynamic data replication policy, proposes a system in which data are replicated by categorizing it as either hot (currently in use) or cold (stale or currently unused) data by assigning a weight based on its access popularity.

WDDRP reduces the space consumption issue [21]. Hybrid replication strategy is proposed for replica selection, placement, and replacement steps. It contains three steps: first, it selects best site; second, it chooses best replica node to place it in best site; and third, its replacement is done to improve response time. HRS comprehends best replica site for best replica [12]. Stochastic diffusion search (SDS) proposes an algorithm for efficient integrity of data replication, and it uses technique of multi-agent global optimization that mimics behavior of communication among ants and agents for minimizing the replication cost. RRSD proposes a file replication method in order to reduce the number of replicas. This method uses replica placement and redundant replica deletion multiple times for achieving its goal [22].

Replication strategy is bounded by two factors: the first one is storage available at different sites because storage is a scarce, important, and costly resource that should be utilized effectively and intelligently and second is the bandwidth available within data centers that affect the data access performance in cloud distributed computing. Mostly, the files in geographical distributed locations are of large size. Placing them in suitable location is important, and choosing location and proper data center where replicas are placed is also important in this aspect. The data center storage comes into play. In addition, the storage is matched to check whether it meets the requirement of the recent file. Then, the suitable location is selected to limit or minimize the latency issue [16].

Because of extreme growth in data usage and data access, replication strategies are used and deployed in cloud centers. Nowadays, most of the companies replicate data offsite, so in case of data corruption or loss, it could easily be recovered. Data replication offers faster backup and recovery option with minimum latency and access time, so overall performance increases [29].

The emergence of Grid and cloud gave data replication a special place in research. Data replication techniques include static and dynamic replication mechanisms, which may be helpful in real-time applications [30]. In static replication technique, a number of nodes are well defined and predetermined. The number of replicas to be created, and nodes where replicas should be placed are decided on cloud design or setup time [8]. These techniques are simple to implement but do not vary with conditions or requirements [31].

Some of the most important static replication techniques include Google File System, MinCopysets, and MORM (Multi-objective Optimized Replication Management) [29]. In GFS, replication of data chunk is done by inserting replicas on different chunk servers having a small amount of utilization of disk space and by placing these replicas in form of chunks on racks, but the drawback of algorithm is a fixed number of replicas. MinCopysets technique is based upon scalable replication technique where random node is selected for data distribution for parallelization and load balancing.

Servers are partitioned into replication groups, and chunk is replicated by random primary node selection. It improves data durability, but latency and write operation delay also increase. MORM (Multi-objective Optimized Replication Management) technique is an offline artificial immune-based replication algorithm. Artificial immune-based algorithm is similar to human's immune system, which can produce antibodies by reacting to antigen. Based on particular objectives, an appropriate number of replicas are chosen and placed among nodes. This is done for each file to get optimal objective value [23].

Dynamic replication strategy varies with requirements and manages replicas based on bandwidth and capacity. It makes decision intelligently about location and current situation. But it has drawback of facing difficulty in collecting information of all data nodes at runtime, and consistency of data is hard to achieve in dynamic replication. Dynamic strategy involves intelligent decision on runtime, and it decides where to replicate the data and what data to replicate and decides on runtime all the requirements for data replication [8]. This strategy is most efficient in service-oriented environment where data about a number of locations and user access pattern are decided in a dynamic manner. It can optimize the use of resources and storage and considers other important factors efficiently with introducing effectiveness.

The involvement of intelligent algorithms in dynamic strategy enhances its selection and placement capabilities, and, consequently, higher intelligent cognitive and deciding the replica location according to requirements and needs. Considering other factors can also make dynamic replication effective, higher, and more efficient than static techniques [32]. Some of the most important techniques introduced in dynamic strategies are D2RS (dynamic data replication strategy) that is multitier hierarchical cloud system. It is based on temporal locality, and its outcome comes in the form of increased data availability and decreased bandwidth consumption [33].

CDRM (cost-effective dynamic replication management) is an efficient scheme, which is based on Hadoop Distributed File System. LRM (locality replication manager) is also Hadoop architecture based on performance, and it ensures the improvement of data block's physical locality and QoS. It is energy and resource-efficient [32]. QADR (QOS-Aware Data Replication) is based on minimum cost and maximum flow-based principle, which increases the average recovery time. Data loss and deciding the access pattern of user on runtime are major challenges that dynamic strategy is facing. The replica creation must reduce latency and be fast enough to improve availability [26].

3. Replication Selection on Cloud

In most scenarios, data are stored or retrieved from different geographical locations. In other ways, we can say that data are either dispersed or distributed on different geographical regions of cloud data center nodes. Geo-distributed data can create different access latency, consistency, and security issues [34]. Accessing and storing geo-distributed or dispersed data effectively and efficiently are important in such scenarios. One of the most important measures to effectively access geo-distributed data is replication. Replication is a technique to store different copies of data at different geo-distributed data center nodes [35].

By storing replicas of data at multiple sites, if one site fails, then data can be accessed from another site. Also, the request finds the closer site to access data that improve the access latency issue and fault tolerance. Figure 1 shows a graphical user interface designed in form of a Web interface that simulates genetic algorithm on 200 points. These points can be placed on random location on-screen. The placement of these algorithms is performed via coordinates on-screen.

Simulation starts and begins the process by finding the shortest path with the help of genetic algorithm by giving best value and a number of generations to produce a shortest path with mutation rate.

Figure 2 shows the path with best value and mutation rate. It involves 200 data centers depicted by points and evolves through 607 generations with mutation rate.

4. Middle-Ware in Replication Selection

When it comes to cloud, geographical location of the data center plays a vital role in data center selection. The location of any data center is a rather important factor when you choose cloud service providers because of speed. The end users will demand high data transfer speed and high site performance; even the slightest delays can turn away visitors from a website, and they might not return to the site again.

Figure 3 shows example of replicated environment. Site 1, Site 2, Site 3, and Site 4 are 4 different data center locations connected through middle-ware infrastructure. Site 2 contains data stored in File X, which is further replicated on Site 1, Site 3, and Site 4. We assume that User 1 tries to access File X and let distance be proportional to access cost of file for simplicity. We can access File X at much cheaper cost from Site 1 and Site 3 because Site 1 and Site 3 are closer to User 1 as compared to other sites. File is accessible and will not be lost even if 3 of 4 sites are down [36].

In cloud computing reliability, effectiveness and efficiency are major (QoS) parameters in resource utilization. These parameters control overall efficiency or performance of cloud systems. Reducing response time, latency, and optimizing CPU utilization increases the performance of the system. Replicas are created to speed up access and lower response time [37].

The storage systems are essential parts of cloud systems. The storage servers used in cloud computing are high performance. Increased demand in cloud services can result in storage server failure. Therefore, to tackle this issue, an open source storage system that is very efficient cloud system such as Hadoop Distributed File System (HDFS) is used. HDFS is designed in such a way that it could be deployed in hardware having low cost [37].

Reliability, effectiveness, and efficiency are quality of service (QOS) parameters in resource utilization in cloud computing. These parameters control overall efficiency or performance of cloud systems. Reducing response time, latency, and optimizing CPU utilization increases the performance of the systems. Replicas are created to speed up access and lower response time [37]. The storage systems are essential part of cloud systems. The storage servers used in cloud computing are high performance. Increased demand in cloud services can result in storage server failure. Therefore, to tackle this issue, efficient cloud system like Hadoop Distributed File System (HDFS) is used. It is open source storage for cloud and designed in such a way that it could be deployed in hardware having low cost [37].

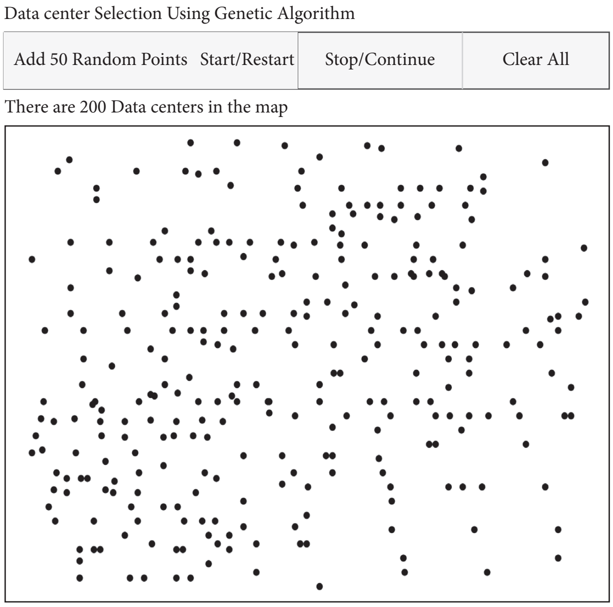
5. Replication Architecture

There are three main parts of replication architecture, namely, scheduling broker, replica broker, and data center. The main broker that controls, manages, and schedules the tasks and data relocation is scheduling broker. Replica manager contains basic information about location and logs of replica. The features of replication architecture are explained in Figure 4.

A replica selection and placement technique are proposed in which popular data at a popular location can be selected in a dynamic way. First, when data are accessed, algorithm will be triggered and start counting the number of times a replica is accessed in a cloud center. When data chunk or replica access reaches a threshold frequency, it will be considered popular and its replica will be created. Popular data are accessible decided on the basis of popular location and the nearest center where data could be stored. This is called dynamic strategy because we must take decision of selection and placement of replica on runtime basis. A well-known data replication technique in distributed systems is used for minimizing user waiting time, higher opportunities of file availability, and lessens the bandwidth consumption of cloud system [39].

The genetic algorithm (GA) is a heuristic search-based optimization algorithm often employed to achieve approximate and optimized solutions for search problems. The proposed solution of the problem is represented with string known as a genotype or chromosome. A basic GA with little probability, tournament selection, and uniform crossover is used to find parameters. In GA approach, the variables or chromosomes decoded to create population.

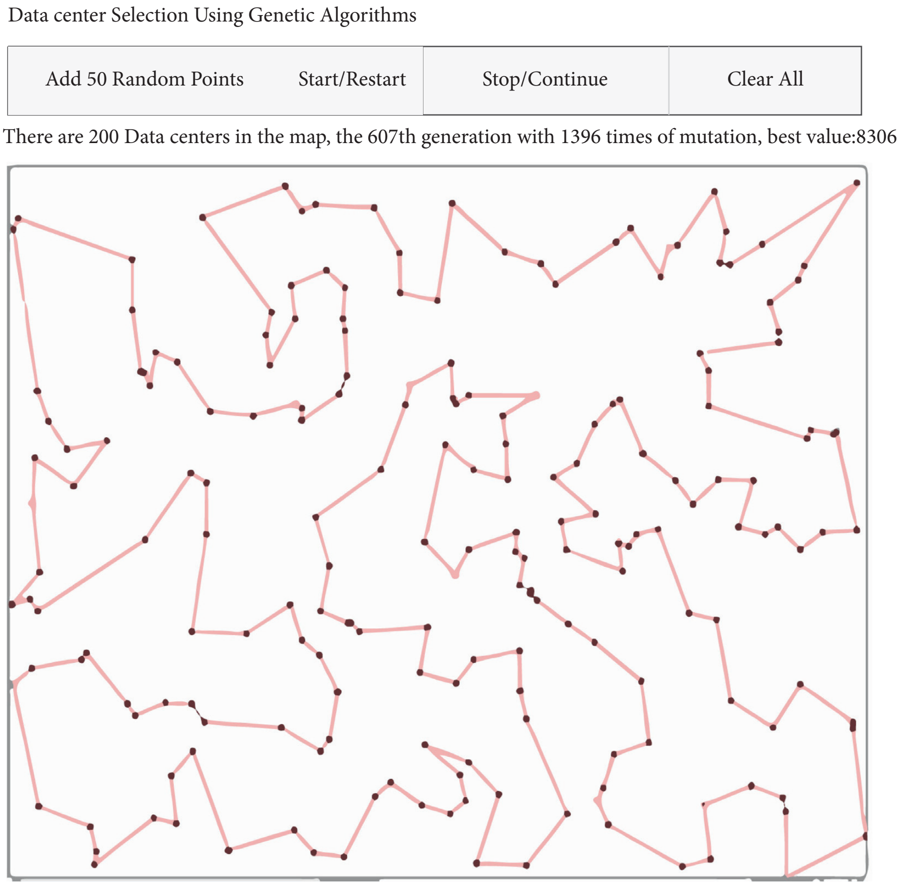
These chromosomes are then converted to real numbers using specified lower and upper limits [39]. Then, fitness of new population is calculated. And GA begins its search from randomly generated new population converging to provide an optimal solution. The GA uses three operators for passing population from generation to generation: the first one is selection, which selects good chromosomes in a generation and forms the crossover population; second one is crossover, which transmits best features of current population to next population, and its rate is 70% and 90% of total population; and lastly, the mutation operator allows further diversity in features as seen in Algorithm 1.



Data center Selection Using Genetic Algorithm

There are 200 Data centers in the map

Figure 1: Data center selection using Web-based interface.



Data center Selection Using Genetic Algorithms

There are 200 Data centers in the map, the 607th generation with 1396 times of mutation, best value:8306

Figure 2: Cloud selection.

5.1. Data Center Search Using Genetic Algorithm. Figure 5 represents data center search using GA, generating the population of regions and data centers, which determines the data transfer cost of each user in different regions. Basically, regions are defined previously as the areas in which continents are divided. When a user is in a region,

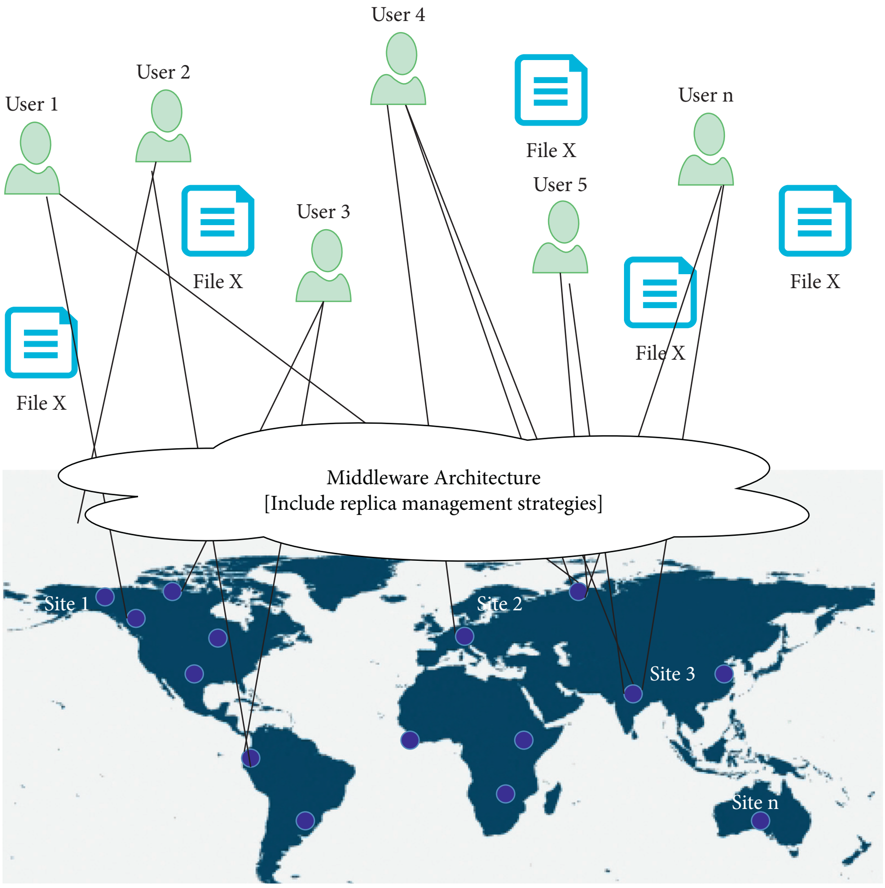


FIGURE 3: Replication middle-ware architecture.

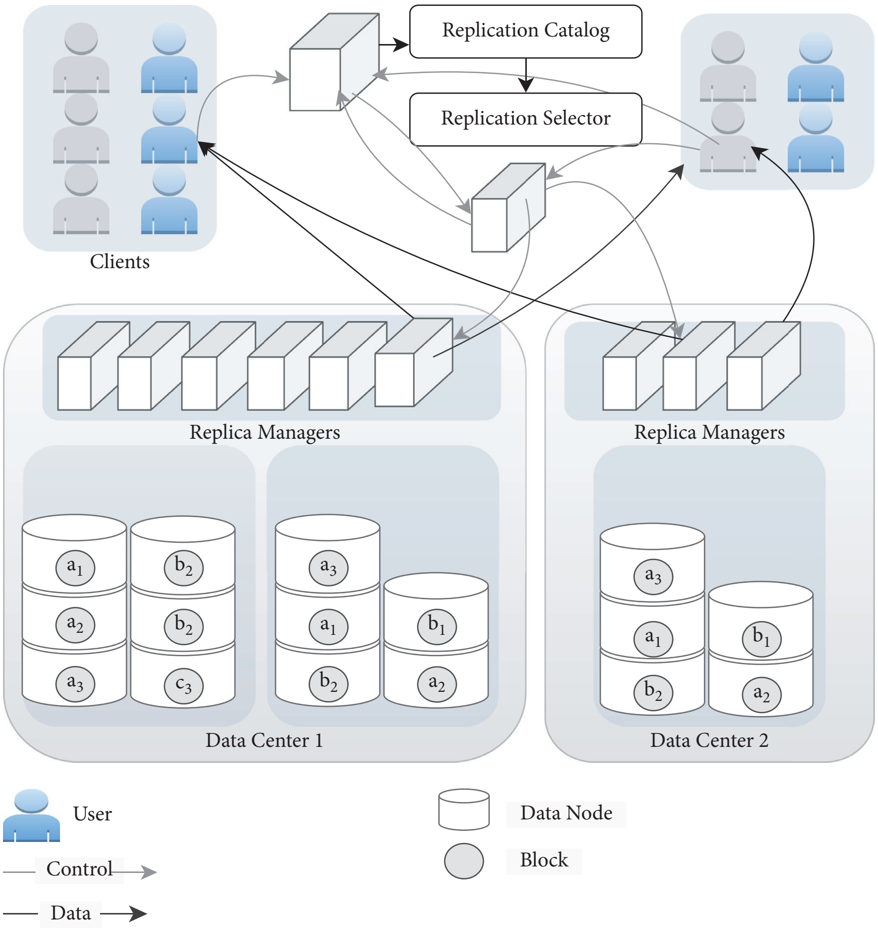


FIGURE 4: Replication architecture (Sun, Chang, and Gao, 2012) [38].

```
Initialize population with data centers of Index table
Initialize user base
Initialize broker
Initialize index table
Initialize population of all data centers
Initialize index
Initialize data center controller
Initialize Internet characteristics with latency order by region
Initialize broker with Internet characteristic values
Save the best solution $X^*$ with its fitness
while best data center is not found do
    SELECT parents
    RECOMBINE pairs of regions
    MUTATE the resulting regions
    EVALUATE data centers based on latencies
    SELECT individuals for the new generation
end
Function select lowest latency data center
Initialize characteristic lists in lowest latency first calculated from regions
if datacenter has a fitness value better than other data centers then
    select best first data center located at first top two positions in the proximity list
else if more than two data centers are located with similar latencies then
then
    select two centers randomly and place data at both of them.
```

Algorithm 1: Nearest location-based replica selection and placement algorithm.

user sends data at different locations. The data transfer cost is the cost of communication between two users located in different regions.

When a user base transfers data, Internet and user base are initialized and maintained an index of all data centers. When Internet get initialized with the start of communication, it receives a message that is from a user base. Then, broker queries to find the specific destination data center controller on which request is to be sent and broker retrieves the region. Finally, a list of all data centers with latencies ordered in low latency first is initialized and population is prepared. Then, it is mutated, and crossover operation takes place until a fitness is achieved to order a list of data centers with lowest latencies. Then, the top two lowest latencies of data centers are selected, and files are placed among those data centers.

The algorithm is basically implemented in cloud analyst, which is an open source cloud computing simulation tool [40]. It is a graphical user interface implemented in Java. It consists of several different components and classes. But the components of main classes consist of regions, Internet, service broker, user base, Internet cloudlet, and data center controller, VM load balancer, and GUI as seen in Figure 6.

5.2. Using Routing Request Selection. The component region divides the world into 6 regions. These regions basically coincide with 6 continents in the world. Regions are implemented so that it keeps the virtual reality that centers basically exist within a region that exists within a continent. It depends on user where he wants to place data center. This geographical distribution is necessary to maintain realistic and simple environment for simulation.

Figure 6 shows and enlists the procedure of routing requests. The Internet is second component that shows the Internet communication among regions and users. Actually, it maintains a matrix of transmission latencies and data transfer delays. So, the transmission latency and bandwidth are configurable.

Data center broker takes the decision about fulfilling the request of any user. In our case, the user traffic is routed by the broker to the top two data centers based on lowest latency. User base generates traffic, and it is configurable and depends on user whether someone configure it for a single user or a group of users. Internet cloudlets are groups of user-based requests where the number of requests that are grouped into a single cloudlet can be configured according to requirement. It maintains information about the originator of traffic, input-output files, number of requests, and application id used to deliver it to specific user.

Data center controller is the main entity that is responsible for whole data center managements including virtual machine creation and annihilates it and routing user requests from user bases with the help of Internet to virtual machines. Cloud analyst functionality depends on it. Virtual machine load balancer decides which VM is to be assigned to cloudlet to processing user requests. Currently, there are three load balancing policies: round Robin: simple round Robin algorithm to allocate virtual machines; active monitoring determines the active tasks and allocates VM to each; and throttled for load balancing by allocating predefined number of Internet cloudlets at any given time to single VM. If a number of available requests are less than the number of groups, some requests will be queued until the availability of VM.

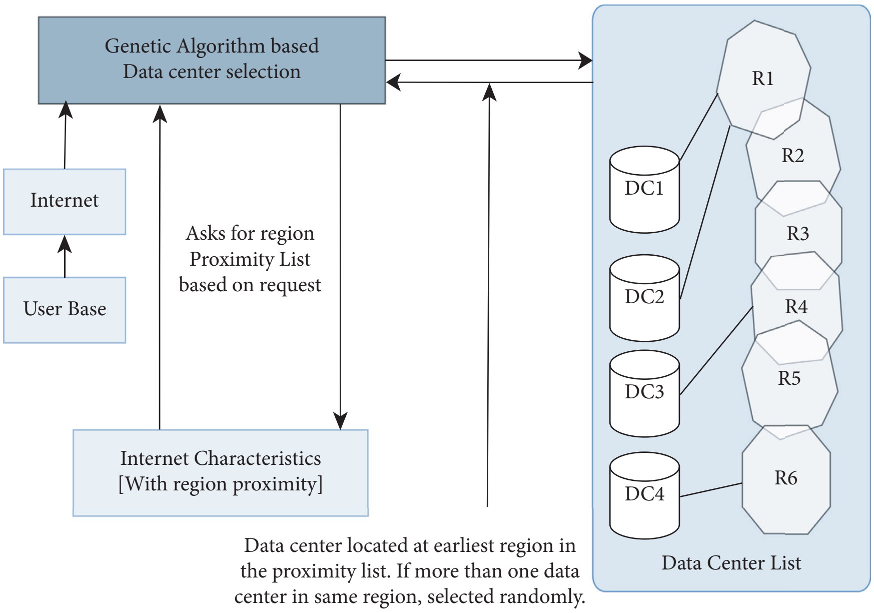


Figure 5: Proposed system data center search using GA.

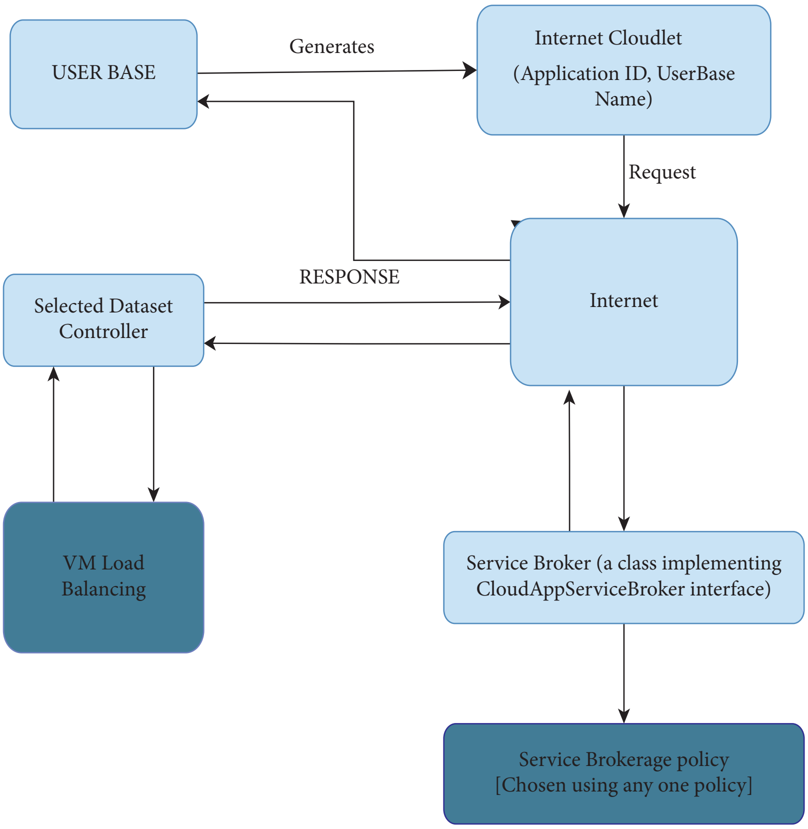


Figure 6: User routing requests.

Finally, GUI is the graphical user interface in which we set simulation parameter and save configurations afterward and start simulating with option of stopping it. And we can save and view results.

5.3. Latency in Data Transmission. The data transmission latency is calculated using

$$T_{\text{total}} = T_{\text{latency}} + T_{\text{transfer}}. \quad (1)$$

T_{latency} is network latency, and T_{transfer} is the time taken to transfer size of data of single request. T_{latency} is calculated from latency matrix in Internet characteristics as shown in

$$T_{\text{transfer}} = \frac{D}{BW_{\text{total}}}, \quad (2)$$

where $\text{BW} = \text{BW}_{\text{total}}/\text{Nr}$, BW_{total} is the total bandwidth, and Nr is the number of user requests during transmission between two users located in two regions. When we use cloud analyst simulator, then it uses these formulas to calculate latency. The cloud analyst gives advance feature to input latency values in textboxes in matrix form. The bandwidth is also given in the matrix, where user can input different values for different regions. When the simulation is started, the cloud analyst simulator generates their own results in graphs and charts.

6. Results and Discussion

Delay and bandwidth matrices contain static values that contain values for delay between regions in cloud analyst. The cloud analyst contains continents and regions where data centers are located. The delay matrix contains the delay between regions. And bandwidth is the available bandwidth between regions.

In Table 1, we took 50 data centers with 25 virtual machines and users with varying number of peak users. The graph in Figure 7 shows that proposed genetic algorithm improves overall response time and result came out to be 248.94 ms, while that of another genetic algorithm was 249.19 ms. This experiment is done by taking 50 data centers and 25 virtual machines and 100 user bases that generates traffic for data transmission across data centers. So, there came optimized results in case of response time.

In Table 2, user base configuration is given. UB1 is the name of user base that generates traffic. And it is placed in Region 2 and each user creates 60 requests/hr. Data size per request is 100 bytes. The peak timing is 3 to 9, and at peak time, peak users are 1000 and off time users are 100.

In Table 3, data center configuration is given in which five data centers are placed in Region 0 with five virtual machines per data center at varying cost.

Table 4 shows that when these settings are configured, then the output for data processing is 0.86 for proposed algorithm, and that of other is 0.95. Overall cost of data center processing task with proposed algorithm is 117.70 much lower than old proposed algorithm.

The results are shown in Figure 8. The graph shows comparison between the two algorithms for the data processing time. And it can be seen clearly that the data processing time is lower for algorithm with nearest location replica selection using genetic algorithm. Our proposed algorithm processes more data in less time, whereas other algorithms take more time to process 60 requests per user with 100 bytes of data. Throughput is defined as the data transmitted per unit time. The graph in Figure 9 shows that the throughput of our proposed algorithm is 1000 Mb data per hour. 60 requests are generated per hour. The throughput came out to be 1000 Mb per hour.

Table 1: 50 data centers with 25 VM and 100 user bases in 1000 s.

Proposed genetic algorithm	Genetic algorithm without threshold frequency
248.94	249.19

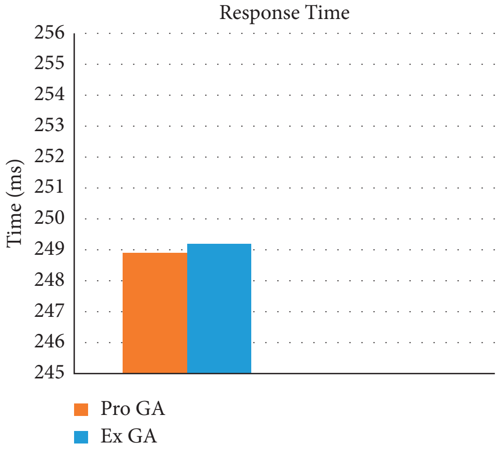


Figure 7: Response time graph for the genetic algorithm against the proposed genetic algorithm.

The experiment is performed by taking 100 user bases that generate traffic and upload data into the cloud centers. 50 cloud centers are taken that store data and user's access data from these data centers. The data centers are configured by placing them in different regions. As cloud analyst consists of 7 continents, these continents are named as regions in the cloud analyst. The architecture of cloud center consists of operating system, which is Linux, and 32-bit operating system is incorporated in the cloud centers with XEN as virtual machine monitor that actually manages the whole virtual network and resources. The number of hardware units is 3. Similarly, user bases are placed at 7 different regions with request per user/hour are 60. So that user only creates 60 requests per hour not more than 60. When simulation is performed the simulation, time is configured to run for 5 min.

When experiment is performed, the genetic algorithm selects the nearest data centers in the nearest locations located to the users, and then, user bases generate traffic to send or access data from this location, Simulation Running Window 2. As a result, the output generated is a minimized latency 0.83, which were previously 0.95 through old GA proposed algorithm that did not incorporated two nearest locations and threshold frequency.

The graph in Figure 10 shows the results of configuration setup in cloud analyst as shown in Tables 5 and 1. The output values are given that shows the overall values as compared to the other algorithms without genetic algorithm

Table 2: User base configuration.

User base name	Region	Request per user (hr)	Data size per request (bytes)	Peak hour GMT	Avg. peak users	Off peak users
UB1	2	60	100	3.00-9.00	1000	100

Table 3: Data center configuration.

DC name	DC 1	DC 2	DC 3	DC 4	DC 5
Region	0	0	0	0	0
VM per DC	5	5	5	5	5
Cost ($) per VM/hour	0.4	0.2	0.3	0.1	0.15

Table 4: Output 1.

	Proposed algorithm	Old GA proposed algorithm
Overall cost ($)	117.70	121.30
Data processing time	0.86	0.95

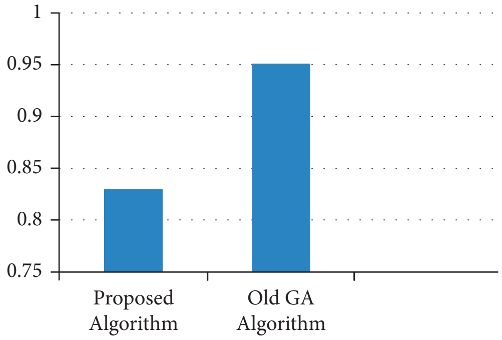


Figure 8: Data processing time in milliseconds for proposed genetic algorithm against the generic GA.

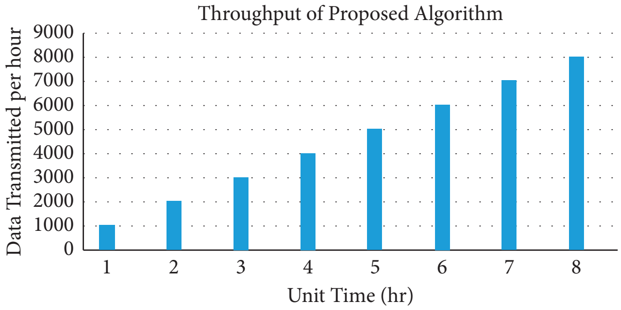


Figure 9: Throughput of our proposed mechanism showing high throughput against the time.

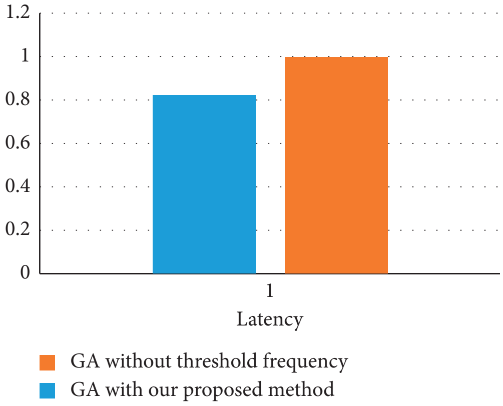


Figure 10: Latency graph comparing the proposed method against threshold frequency-based genetic algorithm.

Table 5: Comparison of different techniques.

Mechanism	Main idea	Advantages	Disadvantages
[23]	Three factors are considered as follows: to insert new replicas whose disk space utilization is below average, to insert the new replicas with below-average disk space utilization, to limit the number of recent replica creations on each server, and to spread the replicas across the racks.	High reliability Low response time Medium load balancing High availability	High energy High replication cost High storage cost
[24]	The benefits of randomized load balancing are used to improve the data durability.	High reliability Low response time High availability Medium load balancing	High energy High replication cost High storage cost
[23]	The trade-off is balanced among different parameters such as mean service time, mean access latency, load variance, energy consumption, and mean file availability. This balancing is done to get near optimal solution.	High availability Low bandwidth consumption Low replication cost	High bandwidth consumption
[25]	It is the greedy algorithm approach with different start points to find replication node.	High scalability High performance Low access latency Low execution time	High energy consumption High replication cost High bandwidth consumption
[26]	This method uses dynamic replica management method, which is based on response time.	High performance Low response time High rapid data download Low energy consumption	Low reliability Low load balancing High replication cost
[27]	It is the greedy algorithm-based approach to check whether the application requires higher QoS.	High availability High scalability Low replication cost	High time complexity High bandwidth consumption
[28]	It considers cost-effective data replication management as a purpose.	High reliability High availability Low replication cost	High response time Low load balancing

implementation. So, the final result shows that latency and data availability have improved as related to the other algorithm.

7. Conclusion

Ineffective replica selection and placement results issues like latency, delay, and effective bandwidth utilization. The placement and selection of suitable replica involve not only finding best site or node for storing data but also deciding a suitable number of replicas to minimize latency rate. In this work, we proposed a technique that effectively choose replica and place them, in a way that resource access from cloud is optimized by minimizing cloud overhead by placing copies of data in best two nearest node. In order to minimize replication cost, this strategy is proposed using genetic algorithm to search best cloud data center based on latency and selection of best data center. So, overall replication cost, latency, and response time are reduced and data copy is placed in two centers; if data are corrupted or lost in one data center, then it can be found in other cloud data centers, so data availability is increased. And because of selection of nearest cloud center selection, the latency issue will be minimized. Genetic algorithm is the most efficient, heuristic evolutionary algorithm that works on the principle of natural selection. And it adopts an intuitive or intelligent approach in finding a best solution. It is widely adopted in the optimization of searching tasks. Now, the recent advancement in genetic algorithm is the parallel genetic algorithm that can be applied in future to further minimize replication cost and latency and searching can be optimized. It further reduces the latency issue by optimizing searching of best solution from a given search space.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by https://doi.org/10.13039/501100002383King Saud University, Saudi Arabia, through research supporting project number RSP-2021/184, and in part by https://doi.org/10.13039/501100012226Fundamental Research Funds for Central Universities (No. 2462020YJRC001).

References

[1] N. K. Nivetha and D. Vijayakumar, “Modeling fuzzy based replication strategy to improve data availabiity in cloud datacenter,” in *Proceedings of the 2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE’16)*, pp. 1–6, IEEE, Kovilpatti, India, January 2016.

[2] A. Siddiqa, M. A. Shah, H. A. Khattak et al., “Social internet of vehicles: Complexity, adaptivity, Issues and beyond,” *IEEE Access*, vol. 6, pp. 62089–62106, 2018.

[3] H. A. Khattak, H. Arshad, S. U. Islam et al., “Utilization and load balancing in fog servers for health applications,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, p. 91, 2019.

[4] K. S. Awaisi, A. Abbas, M. Zareei et al., “Towards a fog enabled efficient car parking architecture,” *IEEE Access*, vol. 7, pp. 159100–159111, 2019.

[5] Qi. Han, M. Shiraz, A. Gani, Md. Whaiduzzaman, and S. Khan, “Sierpinski triangle based data center architecture in cloud computing,” *The Journal of Supercomputing*, vol. 69, no. 2, pp. 887–907, 2014.

[6] B. Varghese and R. Buyya, “Next generation cloud computing: New trends and research directions,” *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[7] W. Wei, K. Wang, K. Wang, H. Gu, and H. Shen, “Multiresource balance optimization for virtual machine placement in cloud data centers,” *Computers & Electrical Engineering*, vol. 88, Article ID 106866, 2020.

[8] D. C. Birkestrand, P. J. Heyrman, and E. C. Prosser, “Virtual machine placement in a cloud computing environment based on factors including optimized processor-memory affinity,” US Patent10055,258, 2018.

[9] Y. Li, J. Liu, B. Cao, and C. Wang, “Joint optimization of radio and virtual machine resources with uncertain user demands in mobile cloud computing,” *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2427–2438, 2018.

[10] B. Alami Milani and N. Jafari Navimipou, “A comprehensive review of the data replication techniques in the cloud environments: major trends and future directions,” *Journal of Network and Computer Applications*, vol. 64, pp. 229–238, 2016.

[11] S. Iqbal, M. L. M. Kiah, N. B. Anuar, B. Daghighi, A. W. A. Wahab, and S. Khan, “Service delivery models of cloud computing: Security issues and open challenges,” *Security and Communication Networks*, vol. 9, no. 17, pp. 4726–4750, 2016.

[12] N. Mansouri and M. M. Javidi, “A hybrid data replication strategy with fuzzy-based deletion for heterogeneous cloud data centers,” *The Journal of Supercomputing*, vol. 74, no. 10, pp. 5349–5372, 2018.

[13] S. A. A. Shah, E. Ahmed, J. J. P. C. Rodrigues, I. Ali, and R. Md Noor, “Shapely value perspective on adapting transmit power for periodic vehicular communications,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 977–986, 2018.

[14] A. S. Sadiq, B. Alkazemi, S. Mirjalili et al., “An efficient ids using hybrid magnetic swarm optimization in wanets,” *IEEE Access*, vol. 6, pp. 29041–29053, 2018.

[15] A. Akbari, A. Khonsari, and S. M. Ghoreyshi, “Thermal-aware virtual machine allocation for heterogeneous cloud data centers,” *Energies*, vol. 13, no. 11, p. 2880, 2020.

[16] S. Yang, P. Wieder, M. Aziz, R. Yahyapour, X. Fu, and X. Chen, “Latency-sensitive data allocation and workload consolidation for cloud storage,” *IEEE Access*, vol. 6, pp. 76098–76110, 2018.

[17] N. Xiong, A. V Vasilakos, L. T. Yang et al., “Comparative analysis of quality of service and memory usage for adaptive failure detectors in healthcare systems,” *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 495–509, 2009.

[18] R. Azimi and H. Sajedi, “A decentralized gossip based approach for data clustering in peer-to-peer networks,” *Journal of Parallel and Distributed Computing*, vol. 119, pp. 64–80, 2018.

[19] X. Fan, C. Huang, J. Zhu, and B. Fu, “R-dra: A replication-based distributed randomized algorithm for data dissemination in connected vehicular networks,” *Wireless Networks*, vol. 25, no. 7, pp. 3767–3782, 2019.

[20] F. Chen, D. Zhang, J. Zhang et al., “Distribution-aware cache replication for cooperative road side units in vanets,” *Peer-to-Peer Networking and Applications*, vol. 11, no. 5, pp. 1075–1084, 2018.

[21] S. Gopinath and E. Sherly, “A dynamic replica factor calculator for weighted dynamic replication management in cloud storage systems,” *Procedia Computer Science*, vol. 132, pp. 1771–1780, 2018.

[22] S. Sun, W. Yao, B. Qiao, M. Zong, X. He, and X. Li, “Rrsd: A file replication method for ensuring data reliability and reducing storage consumption in a dynamic cloud-p2p environment,” *Future Generation Computer Systems*, vol. 100, pp. 844–858, 2019.

[23] S. Q. Long, Y. L. Zhao, and W. Chen, “Morm: A multi-objective optimized replication management strategy for cloud storage cluster,” *Journal of Systems Architecture*, vol. 60, no. 2, pp. 234–244, 2014.

[24] A. Cidon, S. Ryan, S. Rumble, S. Katti, J. Ousterhout, and M. Rosenblum, “Mincopysets: Derandomizing replication in cloud storage,” in *Proceedings of the 10th USENIX Symposium NSDI*, pp. 1–5, Lombard, IL, USA, April 2013.

[25] R. W. Davies, K. Morgan, and O. Hassan, “A high order hybrid finite element method applied to the solution of electromagnetic wave scattering problems in the time domain,” *Computational Mechanics*, vol. 44, no. 3, pp. 321–331, 2009.

[26] X. Bai, H. Jin, X. Liao, X. Shi, and Z. Shao, “Rtrm: A response time-based replica management strategy for cloud storage system, Grid and Pervasive Computing,” in *Proceedings of the International Conference on Grid and Pervasive Computing*, pp. 124–133, Springer, Seoul, Korea, May 2013.

[27] J. W. Lin, C. H. Chen, and J. M. Chang, “Qos-aware data replication for data-intensive applications in cloud computing systems,” *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 101–115, 2013.

[28] W. Li, Y. Yang, and Y. Dong, “A novel cost-effective dynamic data replication strategy for reliability in cloud data centres,” in *Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, pp. 496–502, IEEE, Sydney, Australia, December 2011.

[29] Z. Liao, R. Zhang, S. He, D. Zeng, J. Wang, and H. J. Kim, “Deep learning-based data storage for low latency in data center networks,” *IEEE Access*, vol. 7, pp. 26411–26417, 2019.

[30] A. Siddiqua, M. A. Shah, H. A. Khattak, I. Ud Din, and M. Guizani, “Icafe: intelligent congestion avoidance and fast emergency services,” *Future Generation Computer Systems*, vol. 99, pp. 365–375, 2019.

[31] M. A. Judge, A. Khan, A. Manzoor, and H. A. Khattak, “Overview of smart grid implementation: Frameworks,

impact, performance and challenges," *Journal of Energy Storage*, vol. 49, Article ID 104056, 2022.

[32] H. Khalajzadeh, Y. Dong, J. Grundy, and Y. Yang, "Improving cloud-based online social network data placement and replication," in *Proceedings of the 2016 IEEE 9th International Conference on Cloud Computing (CLOUD)*, pp. 678–685, IEEE, San Francisco, CA, USA, July 2016.

[33] M. Shiraz, A. Gani, A. Shamim, S. Khan, and R. W. Ahmad, "Energy efficient computational offloading framework for mobile cloud computing," *Journal of Grid Computing*, vol. 13, no. 1, pp. 1–18, 2015.

[34] H. A. Khattak, M. A. Shah, S. Khan, I. Ali, and M. Imran, "Perception layer security in internet of things," *Future Generation Computer Systems*, vol. 100, pp. 144–164, 2019.

[35] M. Díaz, C. Martín, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing," *Journal of Network and Computer Applications*, vol. 67, pp. 99–117, 2016.

[36] A. Gani, G. M. Nayeem, M. Shiraz, M. Sookhak, M. Whaiduzzaman, and S. Khan, "A review on interworking and mobility techniques for seamless connectivity in mobile cloud computing," *Journal of Network and Computer Applications*, vol. 43, pp. 84–102, 2014.

[37] N. Khan, N. Ahmad, T. Herawan, and Z. Inayat, "Cloud computing Locally sub-clouds instead of globally one cloud," *International Journal of Cloud Applications and Computing*, vol. 2, no. 3, pp. 68–85, 2012.

[38] 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," *Journal of Computer Science and Technology*, vol. 27, no. 2, pp. 256–272, 2012.

[39] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT press, Cambridge, MA, USA, 1998.

[40] U. Ullah, A. Khan, M. Zareei, I. Ali, H. A. Khattak, and I. U. Din, "Energy-effective cooperative and reliable delivery routing protocols for underwater wireless sensor networks," *Energies*, vol. 12, no. 13, p. 2630, 2019.