


## Systematic Approaches to Data Placement, Replication and Migration in Heterogeneous Edge-Cloud Computing Systems: A Comprehensive Literature Review



Rohini Thimmapura Venkatesh<sup>1\*</sup>, Dimbachamanahalli Krishnappa Chandrashekar<sup>2</sup>, Pavitra Bai Srinivas Rao<sup>3</sup>,  
Rajashree Sridhar<sup>4</sup>, Sunitha Rajanna<sup>4</sup>

<sup>1</sup> Department of Computer Science and Engineering, Dayananda Sagar College of Engineering Affiliated to Visvesvaraya Technological University, Bangalore, Karnataka 560078, India

<sup>2</sup> Department of Computer Science and Engineering, Cambridge Institute Affiliated to VTU Bangalore, Karnataka 560036, India

<sup>3</sup> Department of ISE, SJB Institute of Technology Affiliated Visvesvaraya Technological University Bangalore, Karnataka 560060, India

<sup>4</sup> Department of Computer Science and Engineering/ Artificial Intelligence & Machine Learning, BNMIT Affiliated VTU Bangalore, Karnataka 560070, India

Corresponding Author Email: [rohinitv@gmail.com](mailto:rohinitv@gmail.com)

<https://doi.org/10.18280/isi.280326>

### ABSTRACT

**Received:** 30 March 2023

**Accepted:** 21 May 2023

#### Keywords:

*data placement, data migration data replica, edge computing replication workload balancing*

The advent of Online Social Networks (OSNs) and the Internet-of-Things (IoT) has catalyzed an unprecedented surge in data generation at smart device endpoints. This phenomenon necessitates robust strategies for efficient data distribution and processing on data servers. Furthermore, the burgeoning volume of data intensifies challenges associated with data placement, replication, and migration in edge-cloud computing paradigms. Considerations such as access delay, cost implications, workload balance, and data security become critical parameters in the storage and processing of data from OSNs and IoT devices. Researchers have proposed various strategies to optimize data placement costs, access latency, migration costs, and load balancing constraints. This paper presents an extensive survey on the existing strategies for data placement, data replication, and data migration. The future research directions in edge-cloud computing informed by this survey are also delineated herein.

## 1. INTRODUCTION

Edge-cloud computing systems are comprised of numerous geographically dispersed data centers connected to a large user node network via the internet and potentially private communication channels, as illustrated in Figure 1 [1]. This system architecture, which evolved over time and through various upgrades, exhibits a high degree of heterogeneity in processor capabilities and storage capacities. Owing to the technical and economic impracticalities of earlier hardware-centric, massive parallel processing approaches, these distributed heterogeneous systems have become the primary processing technique for contemporary data-intensive applications.

These applications involve the storage and processing of vast amounts of data, frequently accessed by a global user community. For instance, Google fields over 60,000 search requests per second, translating to more than 5.5 billion daily search requests and over 2 trillion annually as of 2021 [2]. Other companies face similar data volumes, prompting them to adopt non-traditional, horizontal scaling approaches where large numbers of commodity machines are used for data processing and storage.

In these applications, data is stored across geo-distributed data centers. To optimize data storage, data managers must determine the locations for data placement and replication, ensuring that user data requests are satisfied efficiently and

reliably. Maintaining consistency among data replicas remains a critical requirement.

A data processing task may necessitate access to different data items located across various data centers. The nearest location to the processing center is typically selected for data access, underscoring the importance of data placement decisions for system efficiency. These decisions must also take into account the storage capacity limits of the data centers [3].

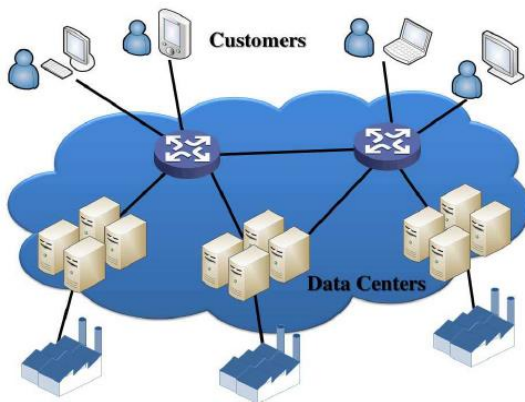
The primary challenge lies in managing the vast volume of data and the rapid processing required to meet user requests. Earlier methods reliant on faster, more powerful hardware with massive parallel processing capabilities were found to be economically unfeasible. Thus, the optimization of data placement in a distributed environment has emerged as a critical concern.

To minimize access delay, maximize data reliability and availability, and balance the load among all nodes of the distributed systems, novel data placement policies, data replication, and data migration strategies are essential [4]. The remainder of this paper surveys these strategies, highlighting their advantages and limitations in an edge-cloud computing context.

In this paper, we have done a review on various methodologies and techniques applied to address the issues such as data placement, data replication and data migration in large Geo-distributed systems. The main research contribution of this review article is as follows:

1. We have broadly categorized the different methodologies applied and various solutions of data placement problem in heterogeneous distributed data processing environment.
2. We briefed about the various approaches for data replication and data migration techniques which maximizes the availability and minimizes the access delay to the users and to balance the load among the nodes of the Geo-distributed systems.
3. Finally, we addressed the various challenges and future research direction for upcoming researchers to perform research in this area and to enhance the performance of the data storage management in distributed systems.

The remainder of the paper is presented as follows. Section 2 describes the review of the data placement problem strategies. Section 3 presents the data replication strategies and methodology applied to solve the data replication in distributed systems. Section 4 addresses the challenges and future direction to conduct research. The last section presents conclusions briefing the future direction of the research.



**Figure 1.** Overall architecture [1]

## 2. RELATED WORK

Data intensive applications in large distributed environment need to deal with common techniques such as data placement, data replication and data migration. Data placement deals with placing the data objects in the proper storage location to maximize the availability and to minimize the latency, network resource utilization.

As the growth of the data from various sectors such as social networks, e-commerce and other IT business etc. increased then data need to stored, processed and made available to the user. The service providers in distributed system are responsible for storing and processing the large volume of data. They have the responsibility to serve user with minimum response time and greater availability.

To address these issues the data must be placed in proper location, and/or same copies must place at multiple locations or data is transferred from one location to another location to minimize the cost incurred in serving the data. Traditionally various techniques have been made attempts to provide the solutions to these problems. Methods such as Distributed Hash Table (DHT), data mining and machine learning techniques exist in the literature.

In this next Section 3, we described data placement methods using clustering techniques, hashing methods and other solutions for cloud data centers.

## 3. STATE-OF-THE-ART DATA PLACEMENT TECHNIQUES

In this section, we discuss the data placement methods that have been described by various authors.

Qin et al. [5] described a data placement scheme in distributed data centers. In this method, authors used Bcube data center topology. The method combines both sequential and random strategies. The three aspects to store the data in node is data access, load balancing and recovery time. The method ensures load balancing and failure repair in the distributed system.

Zhang et al. [6] introduced a data placement scheme based on the location. Scientific applications are producing large volume of data in terms of terabytes to peta bytes. The main issues are data access cost, remote data, associated data access and storage capacity. They described a data placement policy depends on Lagrangian heuristic algorithm. In this method, the data items are placed in data center depends on the user access patterns. These user IO access patterns are depicted on user access pattern space. The data items have storage cost. If all the data items of request pattern are placed in the same data center, then accessing is faster. If the associated data items are stored in the remote node, data access cost includes communication cost. Authors formulated an optimization problem to minimize the data access delay. The objective function is to minimize the local and remote access cost.

Saha and Sharma [7] described a dynamic data placement scheme for heterogeneous nodes with different processing and storage capacity. In this work, a block of data is partitioned into k-partitions by analyzing the history of access frequency of the data blocks. The processing cost and storage cost of each storage node is calculated. The average weight value is calculated at each node. In the first phase, processing cost of each node is considered then average weight value is calculated. In the second phase, storage capacity of all the nodes is considered after which average storage weight is calculated. Finally, average weight is calculated from the average of processing and storage cost and this value is used for distributing blocks across the nodes evenly. In this work, authors solved the issue of load balancing. They ignored data transfer time between processing nodes and the network bandwidth.

### 3.1 Data placement using clustering techniques

Data placement using various clustering methods is discussed here. Li et al. [8] described data placement scheme using k-means clustering. The data item is clustered and placed in the nearest data storage node and data objects are clustered using item-based and user base interest [9]. Collaborative Filtering (CF) method groups the data objects based on user interest and Case Based Reasoning (CBR) groups the data items based on associated data objects. A threshold distance is used, if the distance is less than threshold distance, then data objects are placed in the nearest data center or a new data center cluster is created to store the data objects.

Kchaou et al. [10] have described a data placement using fuzzy C Means clustering technique. In this method, placement of data with two-stage data flow and is modeled as directed a-cyclic graph. The processing of huge data is costly in terms of data movement, execution delay and bandwidth cost. The main objective of this work is to minimize the data migration or transmission between the data centers. The three

types of data transmission are: transmission of requested data, transmission of associated data and re-transmission of data. In this work, authors described a method for data placement scheme containing 2 steps: First is the offline data placement stage and online data placement. In offline data placement, the initial data objects are placed and distributed among data centers. The data objects are placed and dependency of data objects is calculated. This is represented using dependency matrix. Dependency matrix is represented between the data objects  $d_i$  and  $d_j$  and dependency between the data object  $d_i$  and data center. In the second stage, the fuzzy C-Mean algorithm is applied to place the data-sets into data centers. Once the few data centers are overloaded, data placement is adjusted. In this work described the algorithm for minimizing the data movement and the main objective is to place offline and online data among the data centers by using the dependency of the data items.

Atrey et al. [11] described a data placement in geo-distributed cloud data center using spectral clustering-technique. Authors introduced a new framework using hypergraph partition of data items into geo-distributed clouds. They described two algorithms to compute the spectra of hypergraph.i.e SpectralApprox and SpectralDist [12]. The SpeCH is efficient and scalable method for data placement [13]. The SpeCH method is based on the spectral clustering algorithm [14]. The method does not adapt to dynamic changes in the system. The data placement method described in this paper is not applicable for data replication [15].

### 3.2 Data placement in cloud data centers

To match authors and their own affiliations, please insert numerical Data placement in cloud environment by various researchers is illustrated in detail. The state-of-the-art survey and research work by researchers are described [16-18].

Paiva et al. [19] described Autoplacer which automatically optimizes the data replica storage based on the locality pattern. In this work apart from finding the technique to place the data in proper location and also fast data accessing method. In the first challenge, data placement optimization and in the second challenge, a data structure which combines the placement of data is integrated with key-value store. In general, the auto placer combines the data placement and efficient lookup [20]. Auto placer works in sequence of rounds, in each round top-k data item are relocated. In the first step, information regarding the hotspot data items is collected by each node. In the second step, the hotspot information is exchanged by the nodes. In the next step, Probabilistic Associative Array (PAA) is computed by each node and distributed to all the nodes. Finally, the relocation of data items to the new derived location is done.

Sivakumar et al. illustrated a data replication in geo-distributed systems to satisfy the requirements such as access latency, availability and scalability [21]. They used quorum based protocols Dynamo [22] and Cassandra [23]. To achieve low latency with respect to read and write request in maintaining the data consistency of the replicas, latency-aware optimized data replication is presented. They used real traces such as Gowalla, Twitter, Wikipedia and performed experiment on Amazon EC2 using Cassandra cloud and resolves the issues such latency, data consistency and availability.

Li et al. [24] introduced an approach for data placement using workflow for scientific applications for cloud data centres. In this work, a novel workflow-based approach

includes the data placement in cloud data centres with the aim to minimize the transmission cost as compared to the traditional workflow in cluster and grid systems. Author designed a Discrete Particle Swarm Optimization algorithm based on the workflow of the scientific applications. In this method, all the datasets are pre-fetched at build time and during the run time the datasets transferred. In this case the cost of upload and download is taken in to account. The data transfer cost is the bandwidth and migration cost from remote data centre to initial data centre multiplied with the size of the data sets.

Zhou et al. [25] introduced a data placement strategy to improve the performance of the cloud nodes. In the initial phase, clusters are formed based on the dependency of the data items. They build a tree structures system design. In this work, authors stored the data items together, if the data items have dependency. Some of data objects have fixed storage location because of its ownership of the data in the storage node. In this work, smaller size data items are migrated from one node to other, whereas the large size data items are placed at fixed position to reduce the amount data transfer between the nodes. The frequent data transfer can be allowed between the nodes in network with high bandwidth. A heuristic algorithm is designed for data placement in cloud environment [26].

Zhang et al. [27] described optimal data storage in cloud environment. In this work, they considered two types of data: original and generated data. The basic types of costs are included to model resource cost as, bandwidth, storage and computation cost. They proposed a GT-CSB (Generic best Trade-off among computation, storage, bandwidth) algorithm, to modify minimum storage cost to a shortest distance problem using data dependency graph (DDG) [28]. They introduced an approach Provenance elimination strategy (PCE) to general DDG. This method is evaluated and proved that, running time is reduced as compared to other existing method.

Erradi and Mansouri [29] described the cost optimization for data placement and data movement between hot and cold tiers in cloud storage systems. The objects may be tweet or photos send by the Twitter or FaceBook. They proposed two online algorithms for data placement problem. The first cost optimization algorithm uses no replication and stores the data objects in the hot tier. Later, based on the access pattern of get and put request, the data objects are moved to cool tier to optimize the cost of access. In the second algorithm, initially data objects are placed in cool tier and the algorithm considers the replication. In this algorithm data objects are then replicated in the hot tier based on the get and put request of the users. Further they compared the online and offline algorithms with other state-of-the-art algorithm techniques. They found that algorithm with replication performs better than the without replication.

### 3.3 Data placement using various hashing techniques

Consistent hashing is used in many existing systems for data distribution in large distributed system. Some of the existing system which uses consistent hashing for data placement are: Ceph [30], Dynamo [22], Cassandra [23], GlusterFS [31]. Consistent hashing is a key value storage management which increases the scalability and availability. Another important advantage of consistent hashing is that, on adding new nodes or deleting a node less data movement is required and automatically reorganize the data objects. Consistent hashing uses a virtual ring and all the storage nodes are organized on a

ring. The Id numbers of the storage server are aligned to form a virtual ring. The data objects are keys and hashed to position on the ring in a clockwise direction.

In distributed system, there are multiple copies of the same data. Figure 2 illustrates how the consistent hashing works; let us consider three storage nodes A, B and C on the ring. Consistent hashing uses virtual nodes to distribute the data even more uniform and balance the load. Each storage nodes are associated with two or three virtual nodes. The storage node A includes A.1, A.2, A.3 and storage node B includes B.1, B.2 and B.3 as virtual nodes. Virtual nodes are hashed at multiple positions on the hash ring.

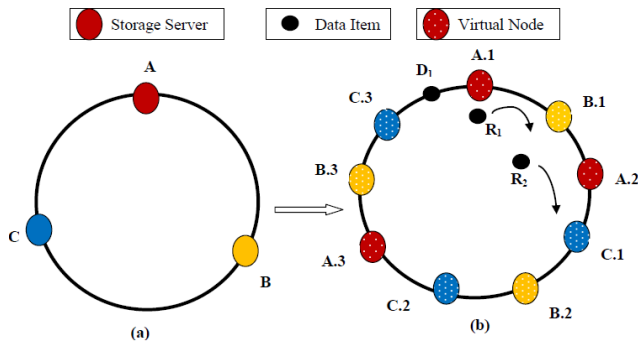


Figure 2. Example consistent hashing

In consistent hashing, if the data item is already hashed in one physical node then skip the virtual nodes of the corresponding physical node. The replica is placed in another virtual node of storage node. For example, let Data item D1 is hashed to physical node A, then the replica of the D1 is placed in the other virtual nodes of storage node let say, B Or C node.

A data placement scheme is described by Qiang et al. is the combination of two algorithms [8]. i.e., K-Means clustering and consistent hashing. In this data placement strategy, to minimize the access latency, a user-based and item-based data is clustered. Case based Reasoning (CBR) and co-ordination filtering technique is used to cluster the data. A k-Mean clustering algorithm is applied. Clusters are stored in the nodes by applying consistent hashing algorithm. Cluster centers are updated dynamically by updating threshold value. Clustering algorithm is combined with consistent hashing to place the data on the nodes.

Consistent hashing algorithm places, hash value space into a virtual counter clockwise. Hash function H is used and hash value h is calculated using the data key to place data on storage node. With this strategy, authors solved the issues such as load balancing, scalability and fault-tolerance. In this work, authors described consistent hashing with elasticity [32, 33]. An elastic consistent hashing adapts to the existing consistent hashing method and in addition to it provides the optimized power and resource utilization. This method resizes the large and small data storage requirement. In this work authors proposed two algorithms: The first is data placement and the second is data migration or data re-integration. Consistent hashing algorithm distributes data objects and balances the load among storage nodes [33]. Data placement is done using consistent hashing. Based on the current load of the nodes, nodes are removed and added to the system depending on the workload requirements.

When the load is less compared to the data storage nodes then they are removed. When workload requirement increases, then data storage nodes are scaled-up to satisfy the demand

and to achieve high performance. If the workload decreases then data storage nodes are removed to reduce the power and resource utilization requirement. In this work, the authors proposed algorithm for data migration. When a node is added, data objects are migrated to the newly added nodes. Data selective migration strategy is applied to data to minimize rate of data migration between nodes.

Using this method, we can increase the performance. In this work, the system uses 2-way replication. Exactly one copy of data object is placed in primary data storage node. If the data item is already is stored in primary data storage node, then other copy should be placed in secondary storage node.

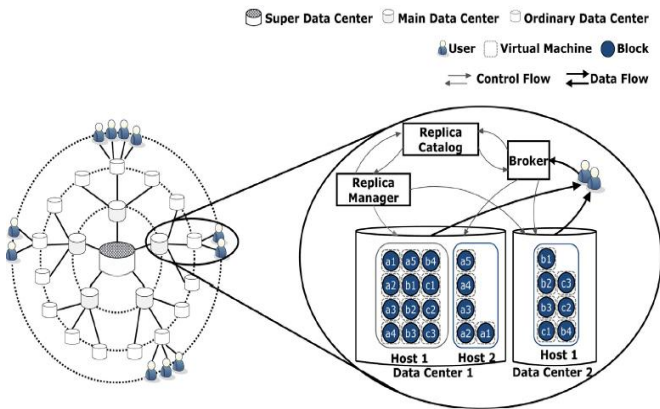
Zhou et al. [34] introduced a data distribution using hierarchical consistent hashing (HiCH). In this method, data nodes are divided into buckets and these buckets are formed as separate consistent hashing ring for hotness, access pattern and latency.

Zhou et al. [35] introduced a consistent hashing called attributedCH. To characterize the distinct node features, each node maintains an attributeCH that distributes the data in heterogeneous nodes on a consistent hashing ring. In this work, authors illustrated a concept called attributedCH, which maintains capacity, network bandwidth and location at each node. This information is being used in data distribution. Before data distribution is done, attributedCH divides the hash ring into sectors. In this method, data selection with small data movement increases the performance, load balance and resource utilization.

#### 4. STATE-OF-THE-ART DATA REPLICATION TECHNIQUES

Data replication is the crucial issue, which improves the data availability and reliability. Data replication is classified into static and dynamic replication [36, 37]. Gill and Singh [38] described a cost-aware replication for cloud data centers in heterogeneous environment as shown in the Figure 3. In this work, availability of data replica is stored at multiple locations of the data center. The probability of the data block is different at different region of the data center. Data center consists of set of data center, users, scheduler/Broker, and replica catalog and replica manager [38]. User request data items and request is given to broker. Broker sends a request to replica catalog. Replica catalog holds information such list of files with location of each file. Broker schedules the nearest data center to the user after receiving the response from replica catalog. Replica manager maintains the replica creation, deletion and is present in the entire data center.

The cost at super data center is high and reliable ordinary data center. Replica catalog receives request from Replica Manager. The request includes location of data replicas and probability of the availability of the replicas at multiple locations. Replica Manager gets the reply from the replica catalog. The Replica manager finds the probability of replica unavailability of all the replicas. With this information, Replica Manager finds the file availability. The high probability of available files has high data access rate, indicates that, these files are more popular files. It means that, the system provides high availability. If there are files with high probability of available but have less popular, it means that, the overall system availability is not effectively increasing.



**Figure 3.** A data replication in cloud data centers

The cost of replication at any data center is given by processing speed, performance and probability of replica availability. The total cost of replication at all the data centers is obtained and compared with the total budget. They proposed a strategy called DCR2S algorithm which consists of three steps. In the first step, finding which file and when to replicate, second step is to finding the number of replicas to create and third step is to locating the new replicas. They calculated the cost of replication, and file availability as the number of replicas are increased.

Mansouri et al. [39] described the data replication strategy based on the popularity of the data. They proposed the algorithm called Dynamic Popularity aware Replication Strategy (DPRS). This algorithm selects the location to place the data in the cloud by considering the free storage availability, number of requests and data access behavior. They considered data distribution and number of data request as the parameter to find the popularity of the data. If the number of accesses is more, then file is considered as popular. Apart from this criterion, two other important issues considered are: data item life time and data distribution over time to differentiate the old and new data requests. DPRS scheme includes five steps as follows: aggregation, popularity of file, replication finding files for replication and replica placement. This algorithm comprises the parallel download scheme. In this approach files are divided into equal sized bytes and equal to the number of servers. Each server will download the file for transfer parallel. If  $x$  is the number of servers, then  $(\text{size of the file}/x)$  bytes will be downloaded parallel by all the  $x$  servers. In this work, they solved the issues such as mean access time, network resource utilization, data storage and replica frequency and hit ratio. In this work, authors have not discussed network security and replica consistency.

Mansouri et al. [40] described storage and processing management in cloud data centers. The various methods for data replication policy in cloud, grid and hybrid environment are described in detail [36, 39, 41]. Mansouri and Javidi [42] proposed a data replication strategy called pre-fetching data replication algorithm based on the prediction. In this work, the popular files are pre-replicated and stored at different location. The strategy finds the co-relation between files access pattern and it pre-fetches the files which are associated. So, when the next time request arrives for the files, it will be locally available. PDR algorithm works as follows: In the first step, it builds the dependency matrix and stores the dependency between the files. In the second step, strategy finds the popular files based on the average file request rate. In the third step,

the unwanted files are replaced with most popular files. PDR strategy uses fuzzy replacement strategy [39].

Cavalcante et al. [43] introduced a replica placement scheme called popRing. This replica placement scheme is called, Key Value-Store for distributed data intensive applications. They described the problem as multi-objective optimization. A popRing method is applied to place the replicas and is depends on genetic algorithm. They used OpenStack-Swift as the benchmark to evaluate the popRing algorithm. In this work, the placement of every data item is mapped to data nodes. These data nodes are called virtual nodes. The mapping function used in the popRing approach is consistent hashing. popRing method is used to minimize load im-balancing and eliminate redundancy of the data in the storage node. The popRing [44] scheme is used to minimize the replica storing and maintenance cost. They formulated the multi-objective function and solution to this optimization minimizes the objective function value.

Azari et al. [45] described a strategy for data replication in grid systems. The algorithm is called PGFR This algorithm is based on accessing the dependent files. All the dependent files such as image, video, audio files are made available locally. The algorithm creates connectivity graph to identify the group of related files at grid location. The users accessing at one grid site have group of interesting dependent files. Placing these dependent files reduces the access time and improves performance. The algorithm consists of three steps: The first step is to construct dependency graph: dependency graph is constructed based on access pattern and file access sequence. When the file is requested at one grid site, if the file is not present in the requested grid site, the file is stored at grid site and number of file access is set to one. This information is stored in database. If the requested file is present in the grid site, then the number of file access is incremented by one. The second step is to identify popular files, each time when the file is accessed the vertex value incremented and this represents the number of file accesses. In third step, when the data request arrives at grid site and requested data is not present then replicate and store replicas at each grid region.

Mseddi et al. [46] described a data migration scheme called, CRANE: an efficient data migration scheme for cloud storage. This scheme complements replica creation and replica storage management is performed effectively. The main objective of this method is to reduce the response time required for data replica placement. Replica placement includes replica creation, locating/migrating data replicas between or within data centers. This task consumes network traffic between those nodes which are involved in data migration. Second contribution is that algorithm tries minimize the network traffic and the availability of data. OpenStack project presents a data replica placement strategy with data replica migration. In this OpenStack, the data is 3-way replicated and stored across the data nodes to improve the availability. Swift finds the optimal replica placement and calls replica migration scheme. CRANE, considers network bandwidth and experimental results are compared with OpenStack Swift with the CRANE method.

Zhou et al. [25] designed an algorithm for data placement as an efficient way to minimize the access latency and storage cost. In this work, a OSN storage system is designed, which includes four different categories of cost: cost of inter data center, cost of data storage, traffic cost of data center and user data migration. In this work, a data placement cost minimizing system model is designed. Data placement with replica is done by analyzing the latency cost. The replication strategy applied

to all the nodes is same and all the nodes are peer nodes. Migration of data from source node to destination includes cost moving data objects from one data center to another and calculated the cost before and after data migration. System is modeled as the interaction between users and data centers as a graph. They developed a cost minimizing strategy called LRP for placement of replicas and data migration. The strategy adapts to the dynamic changes in the distributed system and minimizes the cost.

Hassanzadeh-Nazarabadi et al. [47] introduced a decentralized dynamic data replication of skip graph data based on locality for P2P cloud. They designed locality-aware and decentralized dynamic replication algorithm for skip graphs. They used average delay between the pair-wise latency of data request.

Bok et al. [48] illustrated a workload-balancing scheme in geo-distributed systems using data migration and data replication of data objects. A balancer is present in node acting as a central server. The load balancer distributes the data among all the nodes. The load of the node is calculated and if the node is overloaded, then node sends the current load information to the balancer. The balancer distributes the data among all the nodes and balances the load. In this work, they considered the hash based scheme to store data objects. When node is included and removed in distributed system client maintains the metadata information and periodic load synchronization using load balancer. This reduces the accesses to the load balancer.

Mansouri and Buyya [49] specified the difference between the accessing cost of hot-spot objects and cold-spot objects. The monetary cost consists of creation of replica, read, write and migration cost. They achieved the optimal cost with linear and dynamic programming. They obtained a heuristic solution for set covering problem for replica placement, get and put requests and replica migration. In this work, all the users are assigned to closest DC based on the get and put requests and data is replicated and migrated in other DC's based on the get and put request. These replicas are called slave replicas.

The objects present in hot spot receives more get, put requests compare to object present in cold spot. The objective of the optimization problem is to minimize the get/put and replica migration cost and the location of replicas. They developed RPCLV algorithm for replica placement and migration based on the get, put and replica migration cost. The designed RPCLV algorithm reduces the data storage cost and data transfer latency of the between the data centres.

Mokadem and Hameurlain [50], Limam et al. [51] described a replication scheme, to satisfy the tenant requirement. In this approach data replication is included only if calculated approximate value of response time is greater than the threshold response time or query response time exceeds the number of times of threshold response time. They introduced a DRAPP replication technique. The method deals with: when to replicate? What and how many copies to replicate? Where to place the replicas and which replica to delete? Replica degree is adjusted dynamically to reduce the resource utilization [50]. They described the effect of response time and bandwidth consumption as the impact of varying number of cloudlets and data centre.

## 5. STATE-OF-THE-ART OF DATA MIGRATION SCHEMES

Briefly and descriptively title each table and caption each

figure. In this section, state-of-the-art data migration strategies by various researchers are discussed.

Mseddi et al. [52] described a replica migration scheme called CRANE. It described a system with replica placement and migration sequence from source to destination. The main objective of this work is to reduce the time required to migrate replicas from source storage server to target storage server. In this method, a new replica location is used to create a minimal time and to copy the data to new location. The method maintains network traffic and high availability. The implemented CRANE replica migration is compared with most comprehensive method and integrate in to openstack [53, 54]. The method minimizes the network traffic and with moderate level of availability. In this work, the method reduces the replica migration time while meeting minimum availability and bandwidth capacity of the network link. In this work, cost optimization for minimizing the replica migration time is established with well defined constraints. The replicas are migrated from source node to destination node. The selection of source replica from multiple replicas and network path is important. In openStack swift at any point of time, only one replica partition is participating in replica migration and during this period other replicas are not available [46]. OpenStack Swift does not account the network bandwidth usage when partitions are migrating. When a new data center is added, the partitions are relocated using as-unique-as-possible algorithm.

Teli et al. [55] described an algorithm to reduce the cost of data movement and data aggregation. In this work, the geographic data centers are modeled as graph system. The nodes represent the vertices and communication between the vertices represents the edges of the graph. Weights on the edges represent bandwidth cost. The proposed algorithms are: cost optimization of data movement and aggregation. After building the graph model, corresponding bandwidth cost matrix is calculated. They found the cost of data transfer is by multiplying size of the data object with the bandwidth cost.

Mansouri et al. [56] illustrated a dynamic data migration and data replication scheme. The strategy is based on the two main costs: first one is residential cost which includes get and put cost and next is data movement cost which includes the cost of the network bandwidth. They introduced two algorithms such as optimal offline algorithm and two online algorithms. The scheme dynamically select storage objects and finds the read/write residential cost and data movement cost. They formulated a cost optimization problem with constraints. Offline algorithm requires high time complexity, hence they introduced a new online algorithm which includes residential and migration cost. In this algorithm, data migration happens only when there is a cost saving and total migration cost is equal to or less than without migration of the data.

Zhang et al. [57] introduced a data placement and replication strategy using genetic algorithm. In this work, to optimize the cost of access delay and storage cost replicas are placed at multiple proper locations. They modeled the OSN system as graphs which optimize the cost inter-node traffic and storage cost. They build a method, to optimize the storage cost and minimize the latency a minimum number of replicas are stored at different locations. They constructed a social graph of OSN using Facebook dataset and designed a GA based algorithm. The cost model is built to minimize the latency and inter-server traffic. In this work, load balancing among server is not considered.

Yang et al. [58], Rohini and Ramakrishna [59] introduced a cost optimization for data placement and load-balancing strategy in OSN. In this approach, the data objects required by the users are placed in the same node. The system is modeled as graph and objective of the work is to minimize the cost of storage and transfer of data. A cost-effective load-balancing algorithm based on the graph partitioning (BGPA) is developed for OSN. The BGPA algorithm optimizes the storage cost and transfer cost. The distributed system used in this work is based on static and not with respect to the dynamic changes in the system.

## 6. CONCLUSION

Data placement, data replication and data migration are very important issues in edge-cloud computing system. Data placement in heterogeneous distributed systems stores user required data items nearest to the user. Data replication strategy enhances the availability and reliability of the data to the users by placing replicas in the various data centers. Data migration in distributed system is relocates the data objects dynamically to improve the performance of the system. In this work, we discussed various state-of-the art survey on data placement policies such as data placement using hashing techniques, clustering techniques and genetic algorithms. State-of-the-art data replication strategies and data migration techniques. From the comprehensive survey work, we have identified research gaps and future directions for further research in the data placement edge-cloud computing system.

## REFERENCES

- [1] Jing, C., Zhu, Y., Li, M. (2013). Customer satisfaction-aware scheduling for utility maximization on geo-distributed cloud data centers. In 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, Zhangjiajie, China, pp. 218-225. <https://doi.org/10.1109/HPCC.and.EUC.2013.40>
- [2] Trieu, T.T., Ngo, D.N. (2017). Towards building a platform for e-social on-demand learning and development. *International Journal of Information and Education Technology*, 7(11): 814-817. <https://doi.org/10.18178/ijiet.2017.7.11.978>
- [3] Yu, B., Pan, J. (2015). Location-aware associated data placement for geo-distributed data-intensive applications. In 2015 IEEE Conference on Computer Communications (INFOCOM), Hong Kong, China, pp. 603-611. <https://doi.org/10.1109/INFOCOM.2015.7218428>
- [4] Wang, M., Zhang, J., Dong, F., Luo, J. (2014). Data placement and task scheduling optimization for data intensive scientific workflow in multiple data centers environment. In 2014 Second International Conference on Advanced Cloud and Big Data, Huangshan, China, pp. 77-84. <https://doi.org/10.1109/CBD.2014.19>
- [5] Qin, Y., Ai, X., Chen, L., Yang, W. (2016). Data placement strategy in data center distributed storage systems. In 2016 IEEE International Conference on Communication Systems (ICCS), Shenzhen, China, pp. 1-6. <https://doi.org/10.1109/ICCS.2016.7833566>
- [6] Zhang, J., Chen, J., Luo, J., Song, A. (2016). Efficient location-aware data placement for data-intensive applications in geo-distributed scientific data centers. *Tsinghua Science and Technology*, 21(5): 471-481. <https://doi.org/10.1109/TST.2016.7590316>
- [7] Saha, T.K., Sharma, T. (2017). A dynamic data placement policy for heterogeneous hadoop cluster. In 2017 4th International Conference on Advances in Electrical Engineering (ICAEE), Dhaka, Bangladesh, pp. 302-307. <https://doi.org/10.1109/ICAEE.2017.8255371>
- [8] Li, Q., Wang, K., Wei, S., Han, X., Xu, L., Gao, M. (2014). A data placement strategy based on clustering and consistent hashing algorithm in Cloud Computing. In 9th International Conference on Communications and Networking in China, Maoming, China, pp. 478-483. <https://doi.org/10.1109/CHINACOM.2014.7054342>
- [9] Chedrawy, Z., Abidi, S.S.R. (2005). An intelligent knowledge sharing strategy featuring item-based collaborative filtering and case based reasoning. In 5th International Conference on Intelligent Systems Design and Applications (ISDA'05), Warsaw, Poland, pp. 67-72. <https://doi.org/10.1109/ISDA.2005.22>
- [10] Kchaou, H., Kechaou, Z., Alimi, A.M. (2018). A two-stage fuzzy c-means data placement strategy for scientific cloud workflows. In 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Rio de Janeiro, Brazil, pp. 1-8. <https://doi.org/10.1109/FUZZ-IEEE.2018.8491530>
- [11] Atrey, A., Van Seghbroeck, G., Volckaert, B., De Turck, F. (2018). Scalable data placement of data-intensive services in geo-distributed clouds. In CLOSER2018, the 8th International Conference on Cloud Computing and Services Science, pp. 497-508. SCITEPRESS-Science and Technology Publications.
- [12] Atrey, A., Van Seghbroeck, G., Mora, H., De Turck, F., Volckaert, B. (2019). SpeCH: A scalable framework for data placement of data-intensive services in geo-distributed clouds. *Journal of Network and Computer Applications*, 142: 1-14. <https://doi.org/10.1016/j.jnca.2019.05.012>
- [13] Atrey, A., Van Seghbroeck, G., Mora, H., De Turck, F., Volckaert, B. (2019). Unifying data and replica placement for data-intensive services in geographically distributed clouds. In 9th International Conference on Cloud Computing and Services Science (CLOSER), pp. 25-36. <https://doi.org/10.5220/0007613400250036>
- [14] Rohini, T., Ramakrishna, M. (2019). Spectral clustering and bounded-load consistent hashing for data placement in heterogeneous geo-distributed systems. *Journal of Advanced Research in Dynamic and Control Systems*, 11(06): 306-315.
- [15] Vengadeswaran, S., Balasundaram, S. R. (2020). Clust: grouping aware data placement for improving the performance of large-scale data management system. In Proceedings of the 7th ACM IKDD CoDS and 25th COMAD, pp. 1-9. <https://doi.org/10.1145/3371158.3371159>
- [16] Mazumdar, S., Seybold, F., Kritikos, K., Verginadis, Y. (2019). A survey on data storage and placement methodologies for Cloud-Big Data ecosystem. *Journal of Big Data*, 6(1): 15. <https://doi.org/10.1186/s40537-019-0178-3>
- [17] He, S., Li, Z., Zhou, J., Yin, Y., Xu, X., Chen, Y., Sun, X.H. (2019). A holistic heterogeneity-aware data placement scheme for hybrid parallel I/O systems. *IEEE*

- Transactions on Parallel and Distributed Systems, 31(4): 830-842. <https://doi.org/10.1109/TPDS.2019.2948901>
- [18] Kaur, A., Gupta, P., Singh, M. (2020). A data placement strategy based on crow search algorithm in cloud computing. *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science)*, 13(1): 43-52. <https://doi.org/10.2174/2213275912666181127123431>
- [19] Paiva, J., Ruivo, P., Romano, P., Rodrigues, L. (2014). Autoplacer: Scalable self-tuning data placement in distributed key-value stores. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 9(4): 1-30.
- [20] Paiva, J., Ruivo, P., Romano, P., Rodrigues, L. (2014). Autoplacer: Scalable self-tuning data placement in distributed key-value stores. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 9(4): 1-30. <https://doi.org/10.1145/2641573>
- [21] Shankaranarayanan, P.N., Sivakumar, A., Rao, S., Tawarmalani, M. (2014). Performance sensitive replication in geo-distributed cloud datastores. In 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Atlanta, GA, USA, pp. 240-251. <https://doi.org/10.1109/DSN.2014.34>
- [22] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, A., Vogels, W. (2007). Dynamo: Amazon's highly available key-value store. *ACM SIGOPS Operating Systems Review*, 41(6): 205-220. <https://doi.org/10.1145/1323293.1294281>
- [23] Lakshman, A., Malik, P. (2010). Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 44(2): 35-40. <https://doi.org/10.1145/1773912.1773922>
- [24] Li, X., Zhang, L., Wu, Y., Liu, X., Zhu, E., Yi, H., Wang, F., Zhang, C., Yang, Y. (2016). A novel workflow-level data placement strategy for data-sharing scientific cloud workflows. *IEEE Transactions on Services Computing*, 12(3): 370-383. <https://doi.org/10.1109/TSC.2016.2625247>
- [25] Zhou, J., Fan, J., Jia, J., Cheng, B., Liu, Z. (2018). Optimizing cost for geo-distributed storage systems in online social networks. *Journal of Computational Science*, 26: 363-374. <https://doi.org/10.1016/j.jocs.2017.08.001>
- [26] Zhao, Q., Xiong, C., Wang, P. (2016). Heuristic data placement for data-intensive applications in heterogeneous cloud. *Journal of Electrical and Computer Engineering*, 2016: 3516358. <https://doi.org/10.1155/2016/3516358>
- [27] Zhang, J., Yuan, D., Cui, L., Zhou, B.B. (2019). A highly efficient algorithm towards optimal data storage and regeneration cost in multiple clouds. *Future Generation Computer Systems*, 99: 459-472. <https://doi.org/10.1016/j.future.2019.04.002>
- [28] Yuan, D., Cui, L., Li, W., Liu, X., Yang, Y. (2015). An algorithm for finding the minimum cost of storing and regenerating datasets in multiple clouds. *IEEE Transactions on Cloud Computing*, 6(2): 519-531. <https://doi.org/10.1109/TCC.2015.2491920>
- [29] Erradi, A., Mansouri, Y. (2020). Online cost optimization algorithms for tiered cloud storage services. *Journal of Systems and Software*, 160: 110457. <https://doi.org/10.1016/j.jss.2019.110457>
- [30] Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D., Maltzahn, C. (2006). Ceph: A scalable, high-performance distributed file system. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, pp. 307-320.
- [31] Davies, A., Orsaria, A. (2013). Scale out with GlusterFS. *Linux Journal*, 2013(235): 1.
- [32] Xie, W., Zhou, J., Reyes, M., Noble, J., Chen, Y. (2015). Two-mode data distribution scheme for heterogeneous storage in data centers. In 2015 IEEE International Conference on Big Data (Big Data), Santa Clara, CA, USA, pp. 327-332. <https://doi.org/10.1109/BigData.2015.7363772>
- [33] Xie, W., Chen, Y. (2017). Elastic consistent hashing for distributed storage systems. In 2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Orlando, FL, USA, pp. 876-885. <https://doi.org/10.1109/IPDPS.2017.88>
- [34] Zhou, J., Xie, W., Gu, Q., Chen, Y. (2016). Hierarchical consistent hashing for heterogeneous object-based storage. In 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, pp. 1597-1604. <https://doi.org/10.1109/TrustCom.2016.0247>
- [35] Zhou, J., Chen, Y., Wang, W. (2018). Attributed consistent hashing for heterogeneous storage systems. In *Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques*, pp. 1-12. <https://doi.org/10.1145/3243176.3243202>
- [36] Mansouri, N., Javidi, M.M. (2020). A review of data replication based on meta-heuristics approach in cloud computing and data grid. *Soft Computing*, 24: 14503-14530. <https://doi.org/10.1007/s00500-020-04802-1>
- [37] Rohini T., Ramakrishna, M.V. (2020). Adaptive dynamic data replication with load-balancing in distributed systems. *Journal of Advanced Research in Dynamical and Control Systems*, 12(3): 1034-1043. <http://doi.org/10.5373/JARDCS/V12SP3/20201349>
- [38] Gill, N.K., Singh, S. (2016). A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers. *Future Generation Computer Systems*, 65: 10-32. <https://doi.org/10.1016/j.future.2016.05.016>
- [39] Mansouri, N., Rafsanjani, M.K., Javidi, M.M. (2017). DPRS: A dynamic popularity aware replication strategy with parallel download scheme in cloud environments. *Simulation Modelling Practice and Theory*, 77: 177-196. <https://doi.org/10.1016/j.simpat.2017.06.001>
- [40] Mansouri, Y., Toosi, A.N., Buyya, R. (2017). Data storage management in cloud environments: Taxonomy, survey, and future directions. *ACM Computing Surveys (CSUR)*, 50(6): 1-51. <https://doi.org/10.1145/3136623>
- [41] Mansouri, N., Javidi, M.M. (2018). A hybrid data replication strategy with fuzzy-based deletion for heterogeneous cloud data centers. *The Journal of Supercomputing*, 74: 5349-5372. <https://doi.org/10.1007/s11227-018-2427-1>
- [42] Mansouri, N., Javidi, M.M. (2018). A new prefetching-aware data replication to decrease access latency in cloud environment. *Journal of Systems and Software*, 144: 197-215. <https://doi.org/10.1016/j.jss.2018.05.027>
- [43] Cavalcante, D.M., de Farias, V.A., Sousa, F.R., Paula, M.R.P., Machado, J.C., de Souza, J.N. (2018). PopRing: A popularity-aware replica placement for distributed key-value store. In *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)*, pp. 440-447.



- [44] Cavalcante, D.M., Farias, V.A., Sousa, F.R.C., Paula, M.R.P., Machado, J.C., Souza, N. (2018). PopRing: A popularity-aware replica placement for distributed key-value store. In Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018), pp. 440-447.
- [45] Azari, L., Rahmani, A.M., Daniel, H.A., Qader, N.N. (2018). A data replication algorithm for groups of files in data grids. *Journal of Parallel and Distributed Computing*, 113: 115-126. <https://doi.org/10.1016/j.jpdc.2017.10.008>
- [46] Mseddi, A., Salahuddin, M.A., Zhani, M.F., Elbiaze, H., Glietho, R.H. (2018). Efficient replica migration scheme for distributed cloud storage systems. *IEEE Transactions on Cloud Computing*, 9(1): 155-167. <https://doi.org/10.1109/TCC.2018.2858792>
- [47] Hassanzadeh-Nazarabadi, Y., Küpçü, A., Özkasap, Ö. (2018). Decentralized and locality aware replication method for DHT-based P2P storage systems. *Future Generation Computer Systems*, 84: 32-46. <https://doi.org/10.1016/j.future.2018.02.007>
- [48] Bok, K., Choi, K., Choi, D., Lim, J., Yoo, J. (2019). Load balancing scheme for effectively supporting distributed in-memory based computing. *Electronics*, 8(5): 546. <https://doi.org/10.3390/electronics8050546>
- [49] Mansouri, Y., Buyya, R. (2019). Dynamic replication and migration of data objects with hot-spot and cold-spot statuses across storage data centers. *Journal of Parallel and Distributed Computing*, 126: 121-133. <https://doi.org/10.1016/j.jpdc.2018.12.003>
- [50] Mokadem, R., Hameurlain, A. (2020). A data replication strategy with tenant performance and provider economic profit guarantees in Cloud data centers. *Journal of Systems and Software*, 159: 110447. <https://doi.org/10.1016/j.jss.2019.110447>
- [51] Limam, S., Mokadem, R., Belalem, G. (2019). Data replication strategy with satisfaction of availability, performance and tenant budget requirements. *Cluster Computing*, 22: 1199-1210. <https://doi.org/10.1007/s10586-018-02899-6>
- [52] Mseddi, A., Salahuddin, M.A., Zhani, M.F., Elbiaze, H., Glietho, R.H. (2015). On optimizing replica migration in distributed cloud storage systems. In 2015 IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON, Canada, pp. 191-197. <https://doi.org/10.1109/CloudNet.2015.7335304>
- [53] Kumar, K.A., Quamar, A., Deshpande, A., Khuller, S. (2014). SWORD: Workload-aware data placement and replica selection for cloud data management systems. *The VLDB Journal*, 23: 845-870. <https://doi.org/10.1007/s00778-014-0362-1>
- [54] Ayache, M., Erradi, M., Freisleben, B. (2015). Access control policies enforcement in a cloud environment: Openstack. In 2015 11th International Conference on Information Assurance and Security (IAS), Marrakech, Morocco, pp. 26-31. <https://doi.org/10.1109/ISIAS.2015.7492740>
- [55] Teli, P., Thomas, M.V., Chandrasekaran, K. (2016). Big data migration between data centers in online cloud environment. *Procedia Technology*, 24: 1558-1565. <https://doi.org/10.1016/j.protcy.2016.05.135>
- [56] Mansouri, Y., Toosi, A.N., Buyya, R. (2017). Cost optimization for dynamic replication and migration of data in cloud data centers. *IEEE Transactions on Cloud Computing*, 7(3): 705-718. <https://doi.org/10.1109/TCC.2017.2659728>
- [57] Zhang, L., Li, X., Khalajzadeh, H., Yang, Y., Zhu, R., Ji, X., Ju, C., Yang, Y. (2018). Cost-effective and traffic-optimal data placement strategy for cloud-based online social networks. In 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD)), Nanjing, China, pp. 110-115. <https://doi.org/10.1109/CSCWD.2018.8465343>
- [58] Yang, Y., Li, X., Khalajzadeh, H., Liu, X., Ji, X., Qian, F. (2019). Data placement cost optimization and load balancing for online social networks. In 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD), Suzhou, China, pp. 162-167. <https://doi.org/10.1109/CBD.2019.00038>
- [59] Rohini, T.V., Ramakrishna, M.V. (2022). Cost optimization for dynamic data migration and replacement with load-balancing in geo-distributed systems. In: Kumar, A., Senatore, S., Gunjan, V.K. (eds) ICDSMLA 2020. Lecture Notes in Electrical Engineering, vol 783. Springer, Singapore. [https://doi.org/10.1007/978-981-16-3690-5\\_143](https://doi.org/10.1007/978-981-16-3690-5_143)