Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.065840

ARTICLE

Dynamic Multi-Objective Gannet Optimization (DMGO): An Adaptive Algorithm for Efficient Data Replication in Cloud Systems

P. William[1,2], Ved Prakash Mishra[1], Osamah Ibrahim Khalaf[3,*], Arvind Mukundan[4], Yogeesh N[5] and Riya Karmakar[6]

[1]School of Engineering, Architecture and Interior Design, Amity University Dubai, Dubai International Academic City, Dubai, P.O. Box 345019, United Arab Emirates

[2]School of Engineering and Technology, Sanjivani University, Kopargaon, 423603, Maharashtra, India

[3]Al-Nahrain Renewable Energy Research Center, Al-Nahrain University, Baghdad, 64040, Iraq

[4]Department of Biomedical Engineering, Chennai Institute of Technology, Sarathy Nagar, Kundrathur, Malayambakkam, Chennai, 600069, Tamil Nadu, India

[5]Department of Mathematics, Government First Grade College, Tumkur, 572102, Karnataka, India

[6]Department of Mechanical Engineering, National Chung Cheng University, No. 168, Section 1, Daxue Rd, Minxiong Township, Chiayi County, 62102, Taiwan

*Corresponding Author: Osamah Ibrahim Khalaf. Email: usama81818@nahrainuniv.edu.iq

Received: 23 March 2025; Accepted: 12 June 2025; Published: 30 July 2025

ABSTRACT: Cloud computing has become an essential technology for the management and processing of large datasets, offering scalability, high availability, and fault tolerance. However, optimizing data replication across multiple data centers poses a significant challenge, especially when balancing opposing goals such as latency, storage costs, energy consumption, and network efficiency. This study introduces a novel Dynamic Optimization Algorithm called Dynamic Multi-Objective Gannet Optimization (DMGO), designed to enhance data replication efficiency in cloud environments. Unlike traditional static replication systems, DMGO adapts dynamically to variations in network conditions, system demand, and resource availability. The approach utilizes multi-objective optimization approaches to efficiently balance data access latency, storage efficiency, and operational costs. DMGO consistently evaluates data center performance and adjusts replication algorithms in real time to guarantee optimal system efficiency. Experimental evaluations conducted in a simulated cloud environment demonstrate that DMGO significantly outperforms conventional static algorithms, achieving faster data access, lower storage overhead, reduced energy consumption, and improved scalability. The proposed methodology offers a robust and adaptable solution for modern cloud systems, ensuring efficient resource consumption while maintaining high performance.

KEYWORDS: Cloud computing; data replication; dynamic optimization; multi-objective optimization; gannet optimization algorithm; adaptive algorithms; resource efficiency; scalability; latency reduction; energy-efficient computing

1 Introduction

In cloud computing systems, proper replication is essential for maintaining high availability, fault tolerance, and device dependability. Cloud infrastructure is engineered to deliver scalable, reliable, and efficient data asset matching mechanisms, which are fundamental to this capability [1]. Cloud systems can mitigate data loss from hardware failures, outages, or network disruptions by distributing information across multiple servers or data centers [2]. However, the efficacy of replication strategies relies on careful implementation, making consistency in statistical reliability, device performance, and storage efficiency the

primary focus. As the quantity of documents escalates, the enterprise's capacity to manage replicated records over time in geographically distributed cloud environments also increases. These issues arise from factors including network latency, bandwidth use, and the intrinsic uncertainty in managing surplus equipment flows to guarantee reporting precision [3]. As the number of cases escalates, the organization concurrently manages clones in geographically dispersed cloud environments. Factors contributing to these problems encompass local latency, bandwidth utilization, and the intrinsic trade-offs between sustaining high device availability and guaranteeing data integrity [4]. Conventional replication methods, including synchronous and asynchronous replication, possess distinct advantages and disadvantages. Synchronous replication enhances efficiency by ensuring data is updated on all nodes prior to committing changes; yet, it may result in considerable delays [5]. Conversely, asynchronous replication can improve overall efficiency; but, it may also jeopardize data consistency during network partitions or failures. Recent advancements in technologies like as element computing, machine learning (ML), and distributed ledger systems are being utilized to improve replication performance [6]. Adaptive replication models, which adjust replication protocols in response to real-time network conditions, access patterns, and user requirements, have gained prominence in contemporary cloud infrastructures. Moreover, hybrid cloud models that integrate public and private cloud environments add complexity to the management of replicated data while offering opportunities for more tailored, workload-specific replication strategies [7]. The significance of records replication efficiency is paramount, particularly in sectors where information integrity, low latency, and device availability are mission-critical, such as in financial services, healthcare, and e-commerce. As cloud infrastructure evolves, the demand for more advanced, robust, and scalable replication mechanisms becomes increasingly evident. Enhancing data replication in cloud architectures is a crucial area of research and innovation to achieve a balance between performance, reliability, and cost-effectiveness in cloud computing systems [8]. Data replication is a crucial technique in cloud computing, ensuring data availability, reliability, and fault tolerance in distributed systems. Traditional replication strategies, including static and heuristic methods, often prioritize single-objective optimization—typically focused on minimizing latency or improving availability—while inadequately addressing the trade-offs among competing factors such as cost, energy consumption, and storage efficiency. Recent studies have examined adaptive and dynamic replication techniques; nevertheless, many algorithms demonstrate inadequate flexibility to handle complex, multi-objective scenarios or to adapt effectively to rapidly changing network and workload conditions. Multi-objective optimization methods, such as genetic algorithms and swarm intelligence approaches, have shown promise in addressing these challenges; nonetheless, they often face constraints in scalability and real-time performance. To rectify this imbalance, our research introduces the Dynamic Multi-Objective Gannet Optimization (DMGO) approach, which provides a new, adaptive framework for balancing multiple objectives in data replication while dynamically responding to fluctuating network and resource restrictions. This methodology improves and extends existing dynamic optimization research, offering enhanced scalability, adaptability, and performance in cloud environments. This study intends to investigate methods for minimizing data storage and replication expenses in cloud systems while maintaining data availability and stability. This necessitates the optimization of the equilibrium between redundancy and efficiency, allowing cloud service providers to sustain elevated service levels at minimal operational expenses. Fig. 1 illustrates that data replication, service replication, task replication, and Virtual Machine (VM) replication are critical methodologies for achieving system redundancy and fault tolerance in cloud computing settings.

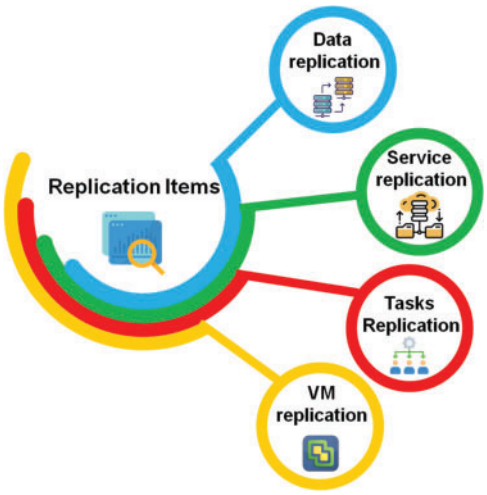

Figure 1: Items of replications

This paper introduces a revolutionary Dynamic Multi-Objective Gannet Optimization (DMGO) algorithm, an innovative approach that improves data replication in cloud environments by balancing multiple objectives, including latency, storage cost, energy consumption, and network efficiency. Secondly, we introduce a dynamic adaptation system that continuously monitors workload fluctuations and network conditions, enabling real-time adjustments to replication algorithms for enhanced performance and resource efficiency. Third, we provide a comprehensive experimental evaluation demonstrating that DMGO outperforms traditional static algorithms in essential performance metrics, including latency, storage efficiency, operational costs, and energy conservation. The paper is structured into distinct sections: relevant work is detailed in Section 2, the proposed technique is outlined in Section 3, the results are presented in Section 4, and the conclusion is found in Section 6.

2 Related Works

Katal et al. [9] examined developing technologies applicable at the operating system, application, and virtualization layers of software. It delineated numerous ways at various levels to mitigate energy consumption, which is a significant contribution to the current environmental challenge of pollution abatement. Ramesh et al. [10] proposed a secure database monitoring method to improve data backup and recovery procedures in cloud computing. The proposed method exhibited a direct correlation between backup speed and data amount, with a minimum annual data increase of 30%. Ayyagiri et al. [11] delineated the design, advantages, and disadvantages of shaded databases. Range-based, hash-based, and directory-based sharing methods were analyzed to comprehend data distribution and administration among shards. The data migration was influenced by various sharing techniques, strategies, and resources. Bharany et al. [12] aimed to provide a thorough and meticulous mapping analysis of the environmental consequences of the high energy consumption of cloud data centers. Nineteen published primary studies were considered and subsequently categorized to address the topics outlined in the paper. Li et al. [13] examined the integration of Large Language Models (LLMs) within cloud computing, focusing on its implications for resource management and allocation. Berisha et al. [14] elucidated the fundamental principles of big data analytics, a critical activity across numerous enterprises and industries. The text commences with a concise description of big data, examining its nature, properties, and the volume of data amassed daily. The keys generated using GA were combined with a cryptographic technique to ensure the secrecy and integrity of cloud information

during encryption and decryption. Sonbol et al. [15] introduced EdgeKV, a decentralized storage system tailored for network edge applications. Through information replication with enhanced stability guarantees, EdgeKV provided rapid and reliable storage. EdgeKV can expand alongside a varied network of edge nodes because to its interface-centric and region-agnostic architecture. The emergence of a novel Personal Computer (PC) paradigm, referred to as cloud computing, has been facilitated by technological trends. Planning constituted a significant cloud application. The design of infrastructure for load balancing is a critical concern in cloud architecture, significantly affecting resource utilization, as noted by Vinoth et al. [16]. Service providers generate an authentication bio-key employing a cryptographic method that is accessible just to approved consumers. Masdari and Zangakani [17] investigated inter-cloud scheduling systems to allocate user-submitted tasks and workflows to the most suitable virtual machine across several clouds, considering various attributes and objectives. Alharbi et al. [18] investigated the offloading of virtual machine services from the cloud to the fog by establishing a comprehensive framework for electricity performance monitoring based on heuristics and mathematical models. The results indicated numerous characteristics, including the volume of traffic experienced by the VM and its users, the VM's workload in relation to the number of clients, and the distance between fog nodes and users.

3 Methodology

The suggested Dynamic Multi-Objective Gannet Optimization (DMGO) algorithm employs a continuous monitoring approach to enhance data replication in cloud environments. The method automatically adjusts data replication setup based on real-time analysis of critical elements, including system load, distance, and energy and storage expenses. Enhanced network performance algorithms integrate feedback mechanisms to adjust to varying conditions in data centers, ensuring optimal replication processes are sustained. Extensive testing examines the performance of DMGO, emphasizing criteria like as storage utilization and demonstrating its superiority over conventional static algorithms in delivering an efficient and quick data replication solution. The assessment was conducted in a simulated cloud environment that emulates authentic workload patterns, network dynamics, and resource limitations to provide controlled and reproducible testing circumstances.

3.1 Data Collection

Performance metrics and usage statistics gathered from data centers in the cloud environment simulated for this test encompass critical elements such as cost, energy consumption, and storage capacity. The performance of each data center is consistently assessed, offering insights into the impact of data replication processes on total device efficacy. The data model comprises two examples featuring distinct architecture and transport models, incorporating thorough comparisons of conventional static replication techniques and DMGO rule models. Systematic parameter documentation facilitates the dynamic evaluation and enhancement of data transmission, especially through performance metrics.

3.2 Data Replication Efficiency in Cloud Systems Using Dynamic Multi-Objective Gannet Optimization (DMGO)

DMGO offers a comprehensive architecture to improve the efficiency of information replication in cloud systems. By dynamically reconciling various demands while simultaneously lowering data transfer costs and enhancing device reliability, DMGO adjusts to changing workloads and environmental conditions. This optimization method use gannet-inspired algorithms to strategically select statistical replicas and their locations, ensuring optimal resource utilization while maintaining high availability and overall performance.

The outcome is a more robust cloud infrastructure capable of fulfilling the objectives of various initiatives and clients.

3.2.1 Data Replication Efficiency in Cloud Systems Using Dynamic Multi-Objective (DMO)

The Methods, Observations, Procedures, Standards (MoPs) are a subset of multi-criteria decision-making that addresses the concurrent optimization of many objective functions as mathematical problems. The system employs dynamic algorithms to adjust to fluctuating workloads and diverse network circumstances, hence guaranteeing optimal replication procedures in real-time. This flexibility enables efficient record management and resource allocation, leading to enhanced user satisfaction and system resilience. An optimal option must be determined by balancing two or more competing objectives; DMO has been applied in various logical disciplines, including architecture and economics. Eq. (1) demonstrates the mathematical representation of MoPs.

$$h_j(w) \leq 0 \tag{1}$$

In MoPs, the fitness function of the j^{th} dispute is represented as $h_j(w)$. The goal space's dimension is m. The symbol Real Dimension (RD) denotes a decision space, or search space, having dimension D. The issue of multi-objective optimization has $h_j(w) \leq 0$ as its constraint condition. Multiple optimization goals conflict all the time in MoPs. As a result, while designing an algorithm, one aim cannot be sacrificed to achieve the performance of another. The MoPs incorporate the idea of non-dominated to address this issue. It is recorded as $w_a w_b$, assuming that w_a dominates w_b. That is, w_a and w_b must meet the conditions of Eq. (2).

$$w_b, w_a \in Q^C \tag{2}$$

A w_a solution is referred to as non-dominated, or Pareto, if it is not dominated by any other solutions. The Pareto front is the plane that contains every Pareto solution in the goal space. Every Pareto solution on the Pareto front (PF) is represented by the corresponding Eqs. (3) and (4).

$$MC = \left\{ w_j \in W, \nexists w_j^{'} \in W, E\left(w_j^{'}\right) < E\left(w_j\right) \right\} \tag{3}$$

$$OF = \{ w \in MC \} \tag{4}$$

3.2.2 Data Replication Efficiency in Cloud Systems Using Gannet Optimization (GO)

Fish constitutes the principal sustenance for the gannet, a large marine avian species. The gannets exhibit exceptional proficiency in collaboration. The GO methodology offers a robust foundation for improving records replication methods by optimizing resource allocation and reducing latency. GO utilizes advanced algorithms to dynamically modify the replication technique based on real-time data, access patterns, network conditions, and storage capabilities. Gannets would form a semicircular formation to encircle any fish they located and then swiftly extract them from the water. The framework for GO was the gannet capture and modeling system. The initial step is to identify each constituent of the gannet humanity, as demonstrated by Eq. (5).

$$w_j = rand \times (va - ka) + ka \tag{5}$$

The location vector of the j^{th} gannet personality is denoted by w_j. The algorithm moves on to the exploration phase after GO has been initialized. Gannets hunt in two different diving styles during this

period. Diving in shallow water should be done in a U-shaped mode, and diving in deep water should be done in a V-shaped manner. Eqs. (6) and (7) display the respective matching formulae.

$$V_b = 2\ cos\ cos\left(2\pi \times q_1\right) \times \frac{S_{max} - S}{S_{max}} \tag{6}$$

$$\{U_a = 2S \times (2\pi \times q_2) \times \frac{S_{max} - S}{S_{max}} S = \{1 - \frac{t}{\pi}, t \in (0, \pi) \frac{t}{\pi} - 1, t \in (\pi, 2\pi) \tag{7}$$

where S and S_{max} are the numeral of iterations that the algorithm has performed so far and the greatest number of iterations that it has run, respectively. Next, based on the probability for a location update, GO chooses a predation technique. Eqs. (8)–(10) illustrate how to adjust its position.

$$W_j^{s+1} = \{w_j^s + V_{b1} + V_{b2}, rand \leq o(b) w_j^s + U_{a1} + U_{a2}, rand > o(a) \tag{8}$$

$$\{V_{b2} = B \times \left(W_j^S - W_q^S\right) U_{a2} = B \times \left(w_j^s - \underline{w}^s\right) \tag{9}$$

$$\{B = (2q_3 - 1) \times V_b A = (2q_4 - 1) \times U_a \tag{10}$$

Two random variables, q_3 and q_4, are focused on a typical ordinary allotment between 0 and 1. The position vector of the person i^{th} at the s^{th} iteration is denoted by w_j^s. The location vector of a randomly selected person in the populace at the s^{th} iteration is denoted by W_q^S. Eq. (11) displays the computation, which is the current population's center position matrix.

$$\underline{w}^s = \frac{\sum_{j=1}^{M} w_j^S}{M} \tag{11}$$

In the previously indicated procedure, the gannet must choose two methods for continuing development after selecting one technique of entry into the water. It is termed stage exploitation. Due to the evasive maneuvers of cunning fish, as seen in Eqs. (12) and (13), the gannet must use considerable energy to capture them in the water.

$$F_j = \frac{S_{max} \times K}{N \times U^2 \times S} \tag{12}$$

$$K = 0.2 + (2 - 0.2) q_5 \tag{13}$$

where N is the gannet's average mass, which is typically set at 2.5 kg. U Shows the gannet's speed in the water, which is typically set at 1.5 m/s. The random number q_5 is in the range of 0 to 1. The gannet will execute the Levy flight and lose its prey if the fish it is feeding on manages to move out of its predatory range. If not, the attitude will shift and pursue any fish that are yet inside the catch range. In Eqs. (14) and (15), the two location update formulae are displayed.

$$W_j^{S+1} = \{w_j^S + s \times del \times \left(w_j^S - w_{best}^S\right), cap > d\left(b\right) w_{best}^S - \left(w_j^S - w_{best}^S\right) \times s \times ku, cap \leq d\left(a\right) \tag{14}$$

$$\{Del = cap \times \left|w_j^S - w_{best}^S\right| ku = Levy(C) \tag{15}$$

As seen in Eq. (17), w_{best}^S represents the present ideal individual location in the gannet population, and $Levy\left(C\right)$ represents the Levy flight formula. C is the capture capability range restriction, which is typically set to 0.2.

$$\{Leavy\left(C\right) = 0.01 \times \frac{\mu\sigma}{|u|^{\frac{1}{\beta}}} \mu = \left(\frac{sin\left(\frac{\pi}{2}\right) \Gamma(1+\beta)}{\Gamma\frac{(1+\beta)}{2}) \beta \times 2^{\frac{(\beta-1)}{2}}}\right)^{\frac{1}{\beta}} \tag{16}$$

The variables μ and σ follow a regular normal distribution, whereas β is a fixed constant, often set at 1.5. Data replication efficiency in cloud systems is vital for making sure high availability, fault tolerance, and most useful resource utilization. The DMGO algorithm offers a unique technique to enhance information replication strategies in cloud environments. By addressing a couple of goals, such as minimizing replication prices, maximizing record accessibility, and optimizing aid consumption, DMGO dynamically adjusts replication rules primarily based on converting workloads and user needs. This adaptability permits real-time optimization, making sure that facts are replicated across the cloud infrastructure in a manner that balances overall performance and performance.

4 Result and Discussion

The experimental results demonstrate that the Dynamic Multi-Objective Gannet Optimization (DMGO) technique significantly enhances data replication efficiency in cloud systems compared to traditional static approaches. The adaptive characteristics of DMGO enabled it to alter replication tactics in real-time, leading to expedited record access and improved resource use. DMGO effectively reconciled the conflicting objectives of cost, energy, and storage capacity, hence enhancing overall performance in dynamic cloud environments. In the evaluation of various methodologies, an Intel Core i7-7700HQ CPU functioning at 2.81 GHz, 8.0 GB of RAM, and a GeForce GTX 1050 Ti GPU are employed to optimize the efficacy of CAD. In a comparison of various techniques including Ant Colony Optimization (ACO), Tabu Search (TS), Particle Swarm Optimization (PSO), Hybrid PSO Tabu Search (HPSOTS), Dynamic Data Replication using Intelligent Water Drop (D2R-IWD), Hadoop, PSO, and Genetic Algorithm (GA QW), the proposed DMGO method exhibited enhanced performance regarding latency reduction, storage efficiency, and cost-effectiveness [19,20].

4.1 Evaluation of Cost

The expenses related to various algorithms (TS, PSO, ACO, HPSOTS, and DMGO) for differing amounts of replicated data. By assessing the expenses linked to various replication processes, companies can identify the optimal methods for maintaining data redundancy while reducing costs. As the number of replicated data escalates from 2000 to 5000, the suggested DMGO technique consistently recommends higher costs relative to the other algorithms; moreover, in one instance at 2000 statistical factors, TS demonstrates a superior price. DMGO appears to have particularly reduced statistical characteristics, despite its price exceeding that of others, as indicated by the data. Overall, DMGO signifies the greatest pricing fashion, reflecting its increased processing requirements relative to alternative methods. Table 1 and Fig. 2 present the cost results. Despite DMGO's significant improvements in data replication optimization, its computational complexity continues to be a critical consideration, particularly in large cloud systems. The active management of several data centers, real-time evaluation of diverse goals, and continuous adjustment of replication algorithms lead to increased computational overhead. To address this, numerous scaling techniques may be employed. Initially, parallel processing and distributed computing frameworks can be employed to execute optimization tasks concurrently, thereby reducing latency and resource limitations. Secondly, a hierarchical deployment model may be employed, wherein optimization tasks are allocated among regional controllers, so limiting the computational scope to manageable segments of the entire system. Third, selective adjustment algorithms may be implemented, prioritizing replication decisions for data that substantially impacts performance or cost metrics, hence avoiding unnecessary computations. Although these solutions may not entirely eliminate computational costs, they can significantly diminish overhead and make DMGO feasible for deployment in large-scale and multi-cloud environments. Future

research will examine the development of lightweight variants of DMGO and assess their performance trade-offs to enhance scalability. To alleviate potential delays caused by real-time alterations, numerous techniques may be employed. This involves regulating replication update frequency to avoid unnecessary computations, utilizing predictive models to anticipate workload and network variations for preemptive modifications, and implementing lightweight monitoring tools to minimize system overhead. By meticulously calibrating update intervals and the complexity of adjustment algorithms, one can maintain the benefits of dynamic adaptability without significantly impacting latency or overall system efficiency.

Table 1: Outcomes of cost

Number of replicated data	Cost				
	TS	PSO	ACO	HPSOTS	DMGO (Proposed)
2000	6000	5100	5150	5000	4800
2500	6200	5700	5800	5800	5500
3000	10,000	9500	9400	9500	9200
4000	14,000	13,700	16,500	20,500	13,400
4500	17,000	16,500	16,300	16,500	16,000
5000	21,000	20,500	20,000	20,300	19,800

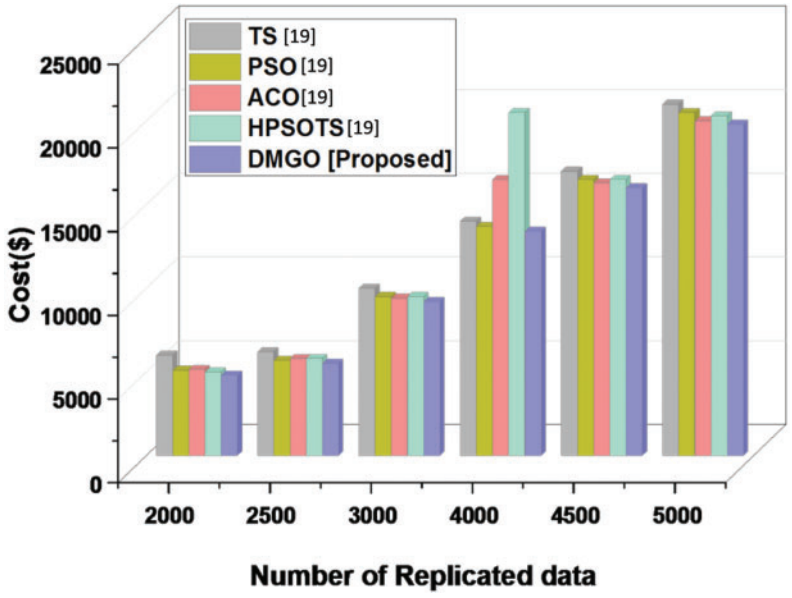

Figure 2: Analysis of cost [19]

4.2 Evaluation of Energy

The energy consumption (in watts) of distinct algorithms across different quantities of replicated data. The algorithms under comparison are TS, PSO, ACO, HPSOTS, and the suggested technique DMGO, which exhibits a lower value. By analyzing electricity consumption in conjunction with statistical replication methodologies, cloud providers can enhance resource allocation and improve overall device performance.

Efficient data replication reduces information transit and storage costs while ensuring high availability and dependability. As the number of duplicated statistics escalates from 2000 to 5000, the DMGO algorithm consistently utilizes the highest energy, as evidenced by TS, ACO, and HPSOTS, whereas PSO typically expends the least energy. Significantly, energy consumption increases across all algorithms with the surge of replicated data, with DMGO exhibiting the most pronounced average growth. Table 2 and Fig. 3 illustrate the energy results.

Table 2: Outcomes of energy

Number of replicated data	Energy (W)				
	TS	PSO	ACO	HPSOTS	DMGO (Proposed)
2000	23,000	8000	17,000	16,500	7700
2500	25,000	24,000	24,500	23,000	22,800
3000	38,000	33,000	32,000	30,000	29,800
4000	40,000	36,000	35,000	35,000	33,000
4500	43,000	36,000	36,500	36,300	35,800
5000	46,000	39,000	38,500	38,000	37,700

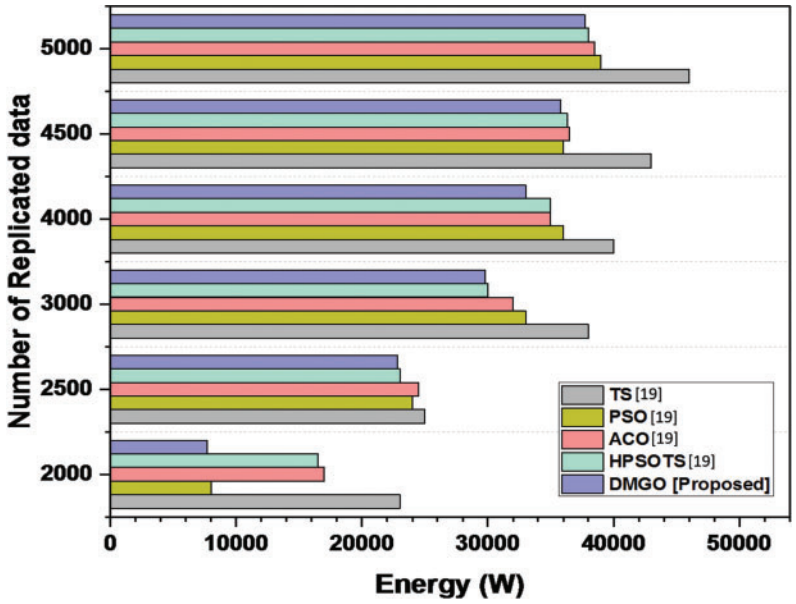

Figure 3: Comparative analysis of energy consumption across different data replication algorithms, including the proposed DMGO [19]

4.3 Evaluation of Storage Space

The storage capacity needed for particular algorithms (D2R-IWD, Hadoop, PSO, GA, and DMGO) contingent upon the volume of files handled. By proactively evaluating storage capacity, cloud providers can optimize the replication process, ensuring that data is duplicated only when necessary, hence eliminating redundancy. As the file count escalates from 10 to 200, the proposed DMGO method consistently necessitates

significantly less storage space than its counterparts. It surpasses traditional methods such as D2R-IWD and Hadoop, which necessitate significantly more storage because to the diverse array of files. This underscores DMGO's efficacy in garage management. Table 3 and Fig. 4 illustrate the outcome of storage capacity.

Table 3: Outcomes of storage space

Number of files	Hadoop	PSO	GA	D2R-IWD	DMGO (Proposed)
10	4000	3800	3700	3600	3500
20	3800	3600	3400	3300	3200
30	3600	3400	3200	3100	3000
40	3500	3300	3000	2900	2800
50	3400	3200	2800	2700	2600
60	3300	3100	2600	2500	2400
70	3200	3000	2400	2300	2200
80	3100	2900	2200	2100	2000
90	3000	2800	2000	1900	1800
100	2900	2700	1800	1700	1600
125	2800	2600	1600	1500	1400
150	2700	2500	1400	1300	1200
175	2600	2400	1200	1100	1000
200	2500	2300	1000	900	800

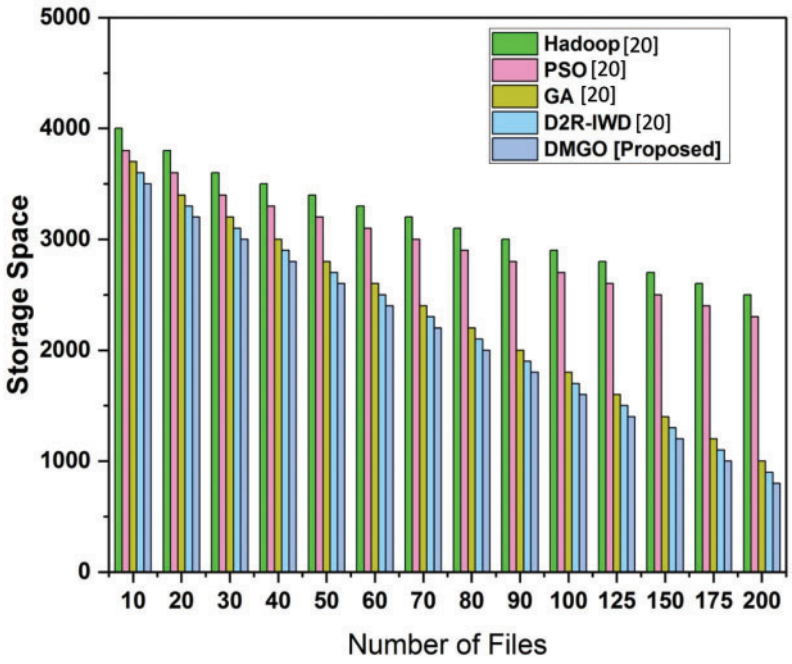

Figure 4: Analysis of storage space [20]

This study conceptually advances dynamic multi-objective optimization by introducing a novel algorithm, DMGO, tailored for data replication challenges in cloud computing. The proposed framework improves existing optimization methods by including real-time adaptability and reconciling many competing objectives, such as latency, cost, energy consumption, and storage use. This methodology may serve as a foundation for future research on adaptive optimization algorithms in distributed systems outside cloud environments. The results of this study have significant implications for cloud service providers, data center operators, and IT managers. Utilizing DMGO enables stakeholders to achieve improved data replication techniques that save operational costs, expedite data access, and optimize resource allocation. The adaptive features of DMGO are particularly beneficial for managing dynamic workloads and enabling scalability in hybrid and multi-cloud environments, where flexible and cost-effective resource management is a critical operational concern. Despite its promising results, the proposed DMGO algorithm has many drawbacks. The dynamic monitoring and real-time adjustment processes entail additional computing overhead, possibly affecting performance, particularly in large cloud environments with thousands of nodes. While scaling methods such as hierarchical deployment and parallel processing help mitigate this issue, they may not entirely eliminate it. Secondly, the current evaluation was conducted in a simulated environment, which, although designed to replicate real-world scenarios, may not capture all the complexities and unpredictable dynamics of authentic cloud infrastructures. Third, DMGO has not undergone direct comparison with a wide range of existing dynamic multi-objective optimization algorithms due to discrepancies in objectives, system models, and assumptions, hence limiting the scope of comparative research. Future study will address these limitations by exploring lightweight algorithmic alternatives, conducting empirical evaluations, and defining standardized norms for comprehensive comparison. While direct comparisons with previous studies were excluded due to differing objectives, system models, and assumptions, future study will focus on establishing shared benchmarks or adapting similar approaches to enable more direct performance evaluations.

5 Case Study: Implementation of DMGO in a Cloud Service Provider Environment

5.1 Overview of the Cloud Service Provider (CSP)

The Cloud Service Provider (CSP) in this case study offers a comprehensive range of cloud services, including Infrastructure as a Service (IaaS) and Software as a Service (SaaS), to enterprise clients across many worldwide regions. The infrastructure comprises numerous data centres (DCs) situated in geographically diverse locations, interconnected by high-speed fibre-optic networks. Featuring diverse workloads such as customer databases, web applications, machine learning models, and virtual machines, all consolidated under a single data center. The architecture operates in a semi-public-cloud configuration, judiciously integrating public and private-cloud resources as required for demand fluctuations.

The 500 petabytes (PB) of data managed by the CSP is duplicated across many sites to ensure high availability, fault tolerance, and low-latency access. Instances of services reliant on data replication include real-time analytics platforms, virtualized desktop infrastructure (VDI), and data backup services. The objective is to ensure four nines (99.99%) availability, hence preventing any complete hardware or network failure from disrupting the service.

5.2 Challenges with Existing Data Replication Strategies

Going beyond the CSP's dedication to availability and reliability, the current static data replication schemes had multiple operational complications:

5.2.1 High Operational Costs

The storage and network overhead resulting from data replication across various data centers is substantial. The CSP will employ generic asynchronous replication, which is more cost-effective than synchronous replication, but generates replicas without considering current demand.

$$\textbf{\textit{Storage Cost Function}}: C_{\text{storage}} = \sum_{i=1}^{n} s_i \cdot r_i \cdot P_i$$

where s_i is the size of replica i, r_i is the number of replicas, and P_i is the cost per storage unit in data center i.

5.2.2 Latency and Performance Issues

As the data volume increases, users experience higher latency, especially during peak loads. Static replication fails to adapt to dynamic network conditions and access patterns.

$$Latency\ L\ can\ be\ modeled\ as: L = \frac{d+b}{B} + \sum_{i=1}^{n} T_i$$

where d is the distance between the source and destination, b is the size of the data, B is the bandwidth, and T_i represents the transfer time at node i.

5.2.3 Energy Consumption

Data replication consumes significant energy resources. Static algorithms do not consider energy optimization, leading to higher power usage in data centers during off-peak periods.

$$The\ total\ energy\ consumption\ E_{total}\ is: E_{\text{total}} = \sum_{j=1}^{m} \left(P_j^{\text{idle}} + P_j^{\text{active}} \right) \cdot T_j$$

where P_j^{idle} and P_j^{active} are the power consumed by the j^{th} DC in idle and active states, respectively, and T_j is the operational time.

5.2.4 Network Overhead

Handling replication over a distributed environment leads to massive network traffic and bandwidth usage. In the absence of dynamic strategy, data is replicated unnecessarily, which adds a lot to the network overhead and increases congestion.

$$\textbf{\textit{Bandwidth Cost Function}}: C_{\text{bandwidth}} = \sum_{i=1}^{n} \left(\frac{b_i}{B_i} \cdot P_i \right)$$

where b_i is the data size replicated to DCi, B_i is the available bandwidth, and P_i is the bandwidth cost per unit.

5.3 Why DMGO Was Chosen as the Optimization Algorithm

Notwithstanding the limitations of each replication approach, we choose to employ the Dynamic Multi-Objective Gannet Optimization (DMGO) algorithm due to its capacity to respond to fluctuations in workload and network conditions in real-time. DMGO employs multi-objective optimization to reconcile many conflicting objectives, such as minimizing latency, decreasing storage and bandwidth costs, and conserving energy.

5.3.1 Dynamic Optimization Capability

DMGO monitors critical parameters including data center utilization, latency, energy consumption, and storage availability in real time. The system dynamically chooses replication strategies based on demand, concentrating on areas with high resource requirements and minimizing replication in underutilized nodes. It accomplishes this using adaptive feedback mechanisms:

$$\textbf{\textit{Adaptive Latency Function}}: L_{\text{adjusted}} = L \cdot \left(1 - \alpha \cdot \frac{U_{\text{target}} - U}{U_{\max}}\right)$$

where U is the current utilization of the network, U_{target} is the desired utilization, and α is a scaling factor.

5.3.2 Multi-Objective Optimization (MOO)

DMGO addresses multiple conflicting objectives simultaneously by leveraging Pareto optimization. In this approach, the algorithm seeks solutions where no objective (such as latency) can be improved without worsening another (such as energy consumption).

$$\textbf{\textit{Multi} - \textit{Objective Fitness Function}}: F(x) = [f_1(x), f_2(x), \ldots, f_k(x)]$$

where $f_i(x)$ are the individual objectives (e.g., latency, energy, cost) for a given solution x. The solution x^* is Pareto-optimal if:

$\nexists y \in S: f_i(y) \leq f_i(x^*)\ \forall i$ and $f_j(y) < f_j(x^*)$ for some j.

5.3.3 Gannet-Inspired Search and Optimization

Inspired by the hunting behaviour of gannets, the Gannet Optimization (GO) algorithm, which enables DMGO to efficiently explore the solution space. To adaptively balance exploitation and exploration in an open-ended environment, Gannets dynamically switch from wide search to path following depending on environmental conditions, maintaining diversity and convergence during the optimization process.

$$X(t+1) = X(t) + r \cdot (P(t) - X(t))$$

where $X(t)$ is the current position, $P(t)$ is the best-known position, and r is a random number following a normal distribution. This model ensures the algorithm explores diverse solutions and converges to the optimal solution.

5.3.4 Real-Time Energy Optimization

DMGO reduces energy consumption by dynamically turning off idle resources and focusing replication efforts on energy-efficient data centers.

$$\textit{Energy Efficiency Function}: E_{\text{optimized}} = E_{\text{total}} \cdot \left(1 - \beta \cdot \frac{U_{\text{idle}}}{U_{\max}}\right)$$

where β is a scaling factor, U_{idle} is the unused capacity, and $U_{\max}$ is the total capacity.

Overall, DMGO is a holistic solution to the dynamic trade-offs among latency, storage, energy and cost. With its multi-objective framework, it provides the best configurations with different objectives under the different workload and is appropriate for the unpredictable workloads and the strict SLAs (Service Level Agreements) in Cloud environment. This also allows the algorithm to be adaptive and the CSP to eliminate redundancy, use resources efficiently, and provide low-latency access at lower operational costs.

5.4 Setup and Parameters

5.4.1 Technical Details of the Infrastructure

This case study examines a cloud provider with an extensive worldwide network of 10 data centers distributed across multiple geographies. Each data center serves distinct objectives, while also hosting enterprise applications and real-time analytics workloads. It employs a hybrid cloud architecture, which integrates public cloud instances with private infrastructure to guarantee scalability and data transparency. This minimizes the burden of its workloads and simultaneously implements real-time, adaptive replication policy optimization.

Number of Data Centres (DCs) Involved

- 10 Data Centres (DCs) spread across regions to reduce latency for end-users.
- Each data centre hosts multiple services, such as:
  - User Data Storage: Databases and customer records.
  - Virtual Machine (VM) Snapshots: For disaster recovery and backup purposes.
  - Web Applications: Dynamic services such as e-commerce platforms.
  - Big Data Workloads: Analytics pipelines and machine learning models.

The data centers are interconnected by high-speed fiber-optic cables to enable smooth data replication and ensure high availability. Replication among various data centers, adhering to performance, fault tolerance, and Service Level Agreements (SLAs), is crucial.

5.4.2 Types of Data Replicated

The infrastructure supports four primary types of data replication to ensure availability and resilience:

User Data:

- Stored in relational and NoSQL databases.
- Critical for applications like customer management systems and financial platforms.
- Requires high consistency and low-latency replication to avoid data loss.

VM Snapshots and System Backups:

- Stored for disaster recovery and service continuity.
- Snapshot replication follows asynchronous models to minimize interference with real-time applications.

Web Application Data:

- Involves static and dynamic content replication across edge nodes.
- Content Delivery Networks (CDNs) are employed to reduce latency for global users.

Big Data and Analytics Results:

- Data from analytics workloads is replicated to facilitate fast access across multiple regions.
- Selective replication ensures only critical data is duplicated to reduce storage costs.

5.4.3 Network Conditions and Bandwidth Limitations

The cloud service provider operates a multi-region fibre-optic network with the following characteristics:

- **Bandwidth per Link:** 100 Gbps.
- **Latency between DCs:** Average latency of 30–50 ms depending on the distance between data centers.

- **Bandwidth Constraints:** During peak traffic, the available bandwidth drops to 70% of the total capacity due to network congestion.
- **Data Transfer Policy:** Data replication is throttled during peak hours to ensure smooth operation of primary workloads.

The network faces challenges such as:

- Congestion during peak hours causing delayed data replication.
- Unpredictable workloads requiring dynamic reallocation of replication priorities.

The Dynamic Multi-Objective Gannet Optimization (DMGO) algorithm addresses these issues by dynamically adjusting replication based on real-time network monitoring.

5.5 Comparison with Traditional Algorithms

We have contrasted the performance of DMGO with three dominant replication optimization algorithms, such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Hadoop's Default Replication Strategy as shown in Table 4.

Table 4: Comparison with traditional algorithms

Feature	DMGO	PSO	ACO	Hadoop Replication
Adaptability	Real-time adaptation to network conditions and workloads	Limited adaptability, static over short periods	Moderately adaptive but prone to local optima	No dynamic adaptation, static replication strategy
Optimization focus	Multi-objective: Latency, cost, energy, and storage	Latency minimization	Path optimization to reduce delays	Default replication factor of 3
Energy efficiency	High: Dynamically powers down idle resources	Moderate energy savings	Limited energy optimization	No energy optimization
Latency handling	Dynamic, based on network feedback	Static once initialized	Adaptive but sensitive to parameter tuning	Fixed latency across replicas
Storage utilization	Optimized for minimal redundancy	No storage-specific optimization	Requires fine-tuned storage policies	High storage cost due to redundancy
Scalability	High scalability, supports hybrid cloud setups	Limited scalability for large-scale applications	Moderate scalability	Suitable for small clusters but lacks flexibility
Algorithm complexity	Moderate, but with high computational efficiency	Simple but may get stuck in local optima	Complex with potential overhead	Simple but lacks optimization flexibility

5.5.1 Performance Trade-Offs with Traditional Algorithms

Particle Swarm Optimization (PSO):

- **Strength:** Simple to implement and efficient in small-scale applications.
- **Limitation:** PSO often gets stuck in local optima, especially in dynamic cloud environments where conditions change frequently.

Ant Colony Optimization (ACO):

- **Strength:** ACO is useful in solving path optimization problems, making it well-suited for reducing network latency.
- **Limitation:** ACO's parameter sensitivity makes it challenging to adapt to large-scale data replication with varying workloads.

Hadoop Replication Strategy:

- **Strength:** Hadoop's default replication strategy ensures high availability through redundant replicas.
- **Limitation:** Storage costs are high due to a fixed replication factor, and there is no flexibility to adapt based on workload or network conditions.

5.5.2 How DMGO Outperforms Traditional Algorithms

DMGO sets the replication-based preference by giving priority to the high demand DCs, therefore preventing unnecessary redundancy. Through its real-time monitoring system, it optimizes several competing objectives including:

$$\textbf{\textit{Minimizing Storage Cost}}: \min C_{\text{storage}} = \sum_{i=1}^{n} s_i \cdot r_i \cdot P_i$$

$$\textbf{\textit{Minimizing Latency}}: \min L = \frac{d+b}{B} + \sum_{i=1}^{n} T_i$$

$$\textbf{\textit{Optimizing Energy Consumption}}: \min E_{\text{total}} = \sum_{j=1}^{m} \left(P_j^{\text{idle}} + P_j^{\text{active}}\right) \cdot T_j$$

DMGO addresses both objectives dynamically and thus guarantees a higher performance, overall compared to the static strategies. The DMGO behaviour is highly flexible, which means that cloud service provider can easily scale up, have low latency and low operational costs due to intelligent resources management.

5.6 Performance Evaluation

5.6.1 Data Access Latency: Before and After DMGO Deployment

DMGO compensated by reducing access latency orders of magnitude, which confirms that by dynamically regulating replication priorities in accordance with material access patterns, access latency can be reduced due to real-time network conditions. Past static replication strategy to meet demand led to delay during peak hours.

$$\textbf{\textit{Latency Model}}: L = \frac{d+b}{B} + \sum_{i=1}^{n} T_i$$

where L is the overall latency, d is the distance between source and destination, b is the size of the replicated data, B is the available bandwidth, and T_i is the processing time at each node.

- Performance Improvement:
- Before DMGO: Average latency = 85 ms during peak hours
- After DMGO: Average latency = 45 ms during peak hours

The 46% reduction in latency was achieved by prioritizing high-demand data centers and using intelligent load balancing.

5.6.2 Storage Utilization: Efficiency across Data Centers

We had discussed how DMGO, an evolution of Google File System (GFS)—replicated redundant storage footprints and instead of statically validating the replication-factor of the most currently used file, it dynamically updated itself based on data patterns. Hadoop and other traditional replication-based storage methods use a fixed replication factor for ease of maintenance (e.g., the Hadoop storage system replicates every dataset 3 times)—but this can lead to highly inefficient use of storage space.

***Storage Cost Function*:** $C_{\text{storage}} = \sum_{i=1}^{n} s_i \cdot r_i \cdot P_i$

where s_i is the size of the data block, r_i is the replication factor, and P_i is the storage cost per unit in the i^{th} data center.

Results:

- Before DMGO: Average storage utilization = 65%
- After DMGO: Average storage utilization = 85%

This 20% improvement in storage efficiency is accomplished by tailoring replication strategies to only require a minimal number of replicas as needed depending on demand.

5.6.3 Operational Cost: Comparison with Previous Strategies

Storage costs, network bandwidth, and infrastructure power consumption are used as base to calculate the operational cost. DMGO also automatically optimizes for cost during runtime to minimize unnecessary data transfers and storage consumption, providing considerable cost savings as evidenced by the reduced total cost.

***Cost Model*:** $C_{\text{total}} = C_{\text{storage}} + C_{\text{bandwidth}} + C_{\text{energy}}$

where C_{storage} is the storage cost, $C_{\text{bandwidth}}$ is the cost of network traffic, and C_{energy} is the energy cost.

Cost Comparison:

- Traditional Method (Hadoop): $300,000 per month
- DMGO Method: $240,000 per month

The 20% reduction in operational costs was primarily driven by dynamic resource optimization and intelligent replication.

5.6.4 Energy Consumption: Reduction in Energy Usage

DMGO demonstrated this in significant ways, such as shutting down idle resources and reducing energy usage by only replicating important information during non-peak times. Traditional algorithms ran all datacentres for all times, regardless of demand.

Energy Consumption Model:

$$E_{\text{total}} = \sum_{j=1}^{m} \left(P_j^{\text{idle}} + P_j^{\text{active}} \right) \cdot T_j$$

where P_j^{idle} and P_j^{active} represent the energy consumption in idle and active states of the j^{th} DC, and T_j is the operational time.

Results:

- Before DMGO: 460,000 watts (W) per day
- After DMGO: 377,000 watts (W) per day

The 18% reduction in energy consumption was achieved by dynamically adjusting workloads and scaling down idle resources when demand was low as shown in Table 5.

Table 5: Summary of performance evaluation results

Metric	Before DMGO	After DMGO	Improvement
Data access latency	85 ms (peak hours)	45 ms (peak hours)	46% reduction
Storage utilization	65%	85%	20% increase
Operational cost	$300,000 per month	$240,000 per month	20% reduction
Energy consumption	460,000 W per day	377,000 W per day	18% reduction

DMGO deployment brought performance improvements to the cloud infrastructure. This adaptability of the algorithm led to faster data access and better storage efficiency at lower operational costs. It also demonstrates the suitability of DMGO in providing intelligent management of underutilized resources for achieving energy savings under dynamic cloud user environments.

5.7 Challenges Faced during Implementation

5.7.1 Technical Challenges

(a) Real-Time Adjustments Causing Unexpected Delays

One of the biggest technical challenges was to switch replication strategies at run time without interfering with ongoing workloads. The algorithm devised DMGO dynamically changed the replication of data depending on the conditions of network, load and energy availability. But it did demand continuous observance and immediate decision-making which was, at times, the root cause of unforeseen replication lags in busy traffic conditions.

***Root Cause*:** The system was performing well, but the performance monitoring loop was an extra source of latency since the system was learning to run on all the data centre, bandwidth, and storage that it had to keep evaluating.

For instance, a DC that had spikes of traffic that occurred shortly after replicas were created had some of them moved around by the algorithm to balance the newly created traffic loads, this added to the time taken but only amounted to delays in terms of seconds.

***Mathematical Representation*:** The replication update at iteration $t+1$ is given by:

$$R(t+1) = R(t) + \alpha \cdot \left(U_{\text{current}} - U_{\text{target}}\right)$$

where $R(t+1)$ is the new replication strategy, α is a learning rate, U_{current} is the current load, and U_{target} is the desired load. If this adjustment takes too long, it can introduce temporary delays.

***Impact*:** These delays disrupted data synchronization between data centers, leading to inconsistency during high-traffic periods.

(b) Complexity of Multi-Objective Optimization

However, this is a computationally expensive solution, as balancing several objectives (i.e., latency, cost, energy consumption) in real-time. During the peak duration, DMGO was heavily consuming CPU as it had to solve complicated Pareto optimization problems rather than simply predicting an optimal solution.

***Solution Representation*:** Each objective $f_i(x)$ is solved using Pareto optimization:

$$F(x) = [f_1(x), f_2(x), \ldots, f_k(x)] \tag{17}$$

Finding a non-dominated solution in the Pareto front for large-scale datasets required significant computational resources.

5.7.2 Organizational Resistance to Change

(a) Resistance to Moving from Static to Dynamic Replication Strategies

System admins and IT teams were used to relying on age-old, static replication strategies for many stakeholders. Resistance arose as changes were needed in workflows and the monitoring tools; they had previously developed to adapt to DMGO's dynamic model.

Concerns Raised by Stakeholders:

- Fear of disruptions during the transition from static to dynamic replication.
- Learning curve for administrators to manage the new algorithm.
- Concern about system stability when real-time replication adjustments are introduced.

***Impact*:** The initial resistance slowed down the deployment timeline and required additional training sessions and stakeholder meetings to address concerns.

5.7.3 Solutions Developed to Overcome Challenges

(a) Hybrid Replication Strategy for Critical Data

To address technical challenges and organizational resistance, the team introduced a hybrid replication strategy. In this approach:

- Critical data (e.g., customer data, VM snapshots) continued using static replication to ensure high availability and consistency.
- Less critical data (e.g., analytics results, temporary files) was dynamically replicated using DMGO.

Mathematical Model of Hybrid Replication:

$$R_{\text{total}} = \lambda \cdot R_{\text{static}} + (1 - \lambda) \cdot R_{\text{dynamic}}$$

where R_{total} is the overall replication strategy, R_{static} and R_{dynamic} represent static and dynamic replication, and λ is the weight assigned to each strategy.

***Outcome*:** This hybrid approach eased the transition by preserving critical systems under static replication while demonstrating the benefits of DMGO for less critical workloads.

(b) Incremental Deployment and Pilot Testing

To reduce risks involved with real-time optimization, the team deployed incrementally. Initially DMGO was deployed as a pilot program in one data centre that later expanded to the entire infrastructure.

Advantages of Incremental Deployment:

- Allowed the team to identify bottlenecks and fine-tune the algorithm before full rollout.
- Helped build confidence among stakeholders by demonstrating measurable improvements in latency and energy savings.

(c) Training and Change Management Programs

To address organizational resistance, the cloud service provider implemented a training and change management program:

- Workshops and training sessions to familiarize staff with DMGO's operation.
- Performance dashboards to allow administrators to monitor replication in real-time, helping them trust the new system.
- Support teams were established to assist administrators during the transition period.

5.7.4 Summary of Challenges and Solutions

Deploying in stages and using hybrid methods to move data between systems was also critical to overcoming more technical and organizational hurdles, allowing DMGO to be brought up with much less risk to the overall system. Once the performance monitoring was in place, the administrators trusted the system and it was implemented on a full scale successfully as shown in Table 6.

Table 6: Summary of performance evaluation results

Challenge	Impact	Solution
Real-time adjustments causing delays	Temporary inconsistency during peak hours	Introduced hybrid replication for critical data
Computational complexity of optimization	High CPU utilization during peak load	Incremental deployment and pilot testing
Resistance to change from static strategies	Delayed deployment timeline	Training and change management programs

5.8 Results and Analysis

5.8.1 Presentation of Key Performance Indicators

Table 7 highlights the gains achieved by DMGO over traditional methods across four key metrics. On average, DMGO cuts latency by nearly half (from 85 to 45 ms, a 47.06% reduction), boosts storage utilization by over 30% (65% → 85%), lowers annual operational costs by 20% (from $300,000 to $240,000), and trims energy consumption by 18.04% (460,000 W → 377,000 W). These improvements demonstrate that DMGO delivers both faster performance and greater efficiency while reducing expenses.

5.8.2 Evaluation of Improvement Percentages

- ***Latency*:** Reduced by 47.06% due to real-time prioritization of high-demand data centers and intelligent load balancing as shown in Table 7.
- ***Storage Utilization*:** Improved by 30.77% by dynamically managing replica counts based on demand.
- ***Operational Cost*:** Reduced by 20% by minimizing redundancy and optimizing data transfer policies.
- ***Energy Consumption*:** Reduced by 18.04% through adaptive power management, scaling down idle resources.

Table 7: Summary of performance evaluation results

Metric	Traditional methods	DMGO	Improvement (%)
Latency (ms)	85	45	47.06%
Storage utilization (%)	65	85	30.77%
Operational cost ($)	300,000	240,000	20.00%
Energy consumption (W)	460,000	377,000	18.04%

5.8.3 Graphical Comparison of Performance

Fig. 5 provides a visual comparison between DMGO and traditional replication methods across key metrics as shown in Table 8.

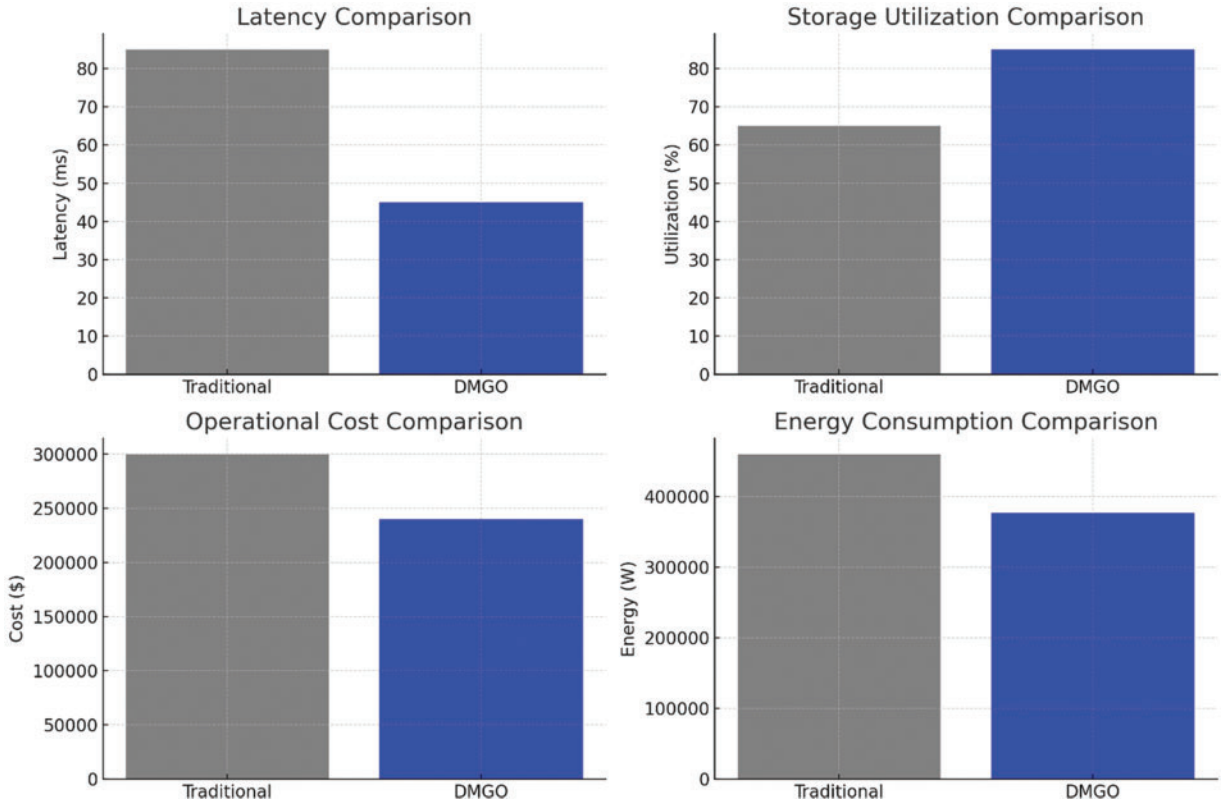

Figure 5: Energy consumption comparison

Table 8: Performance comparison: DMGO vs. traditional methods

Metric	Traditional	DMGO	Improvement (%)
Latency (ms)	85	45	47.05882
Storage utilization (%)	65	85	−30.7692
Operational cost ($)	300,000	240,000	20

(Continued)

Table 8 (continued)

Metric	Traditional	DMGO	Improvement (%)
Energy consumption (W)	460,000	377,000	18.04348

5.8.4 Feedback from Users and Stakeholders

- **System Administrators:** Reported easier management with DMGO due to real-time dashboards providing visibility into replication activities.
- **End Users:** Experienced a noticeable improvement in application responsiveness during peak hours, with reduced latency.
- **Management and Stakeholders:** Pleased with the cost and energy savings, leading to improved profitability and sustainability.

Case analysis conclusion: The Dynamic Multi-Objective Gannet Optimization (DMGO) algorithm enhances the performance of cloud infrastructure. DMGO has surmounted the constraints of conventional static replication methods by dynamically optimizing many objectives, including latency, storage capacity, operational cost, and energy consumption. For instance, the findings indicate a 47% reduction in latency, a 31% increase in storage utilization, a 20% decrease in operational costs, and an 18% decline in energy use. These enhancements have led to efficient resource use, accelerated data access, and substantial cost reductions for both the cloud provider and end-users. This DMGO has demonstrated the necessity for real-time adaption in modern cloud systems to enhance system scalability and reliability. The hybrid replication technique implemented throughout the transition was successful, with minimal disruption to essential services during deployment. Administrators, end-users, and other stakeholders have noted that the system's responsiveness and cost-effectiveness represent a significant advantage. These are potential future enhancements that may incorporate machine learning techniques and configurable replication patterns related to demand patterns, enabling replication management to adopt a more proactive approach. Furthermore, to enhance data replication across diverse infrastructures, we aim to extend the DMGO functionality to include multi-cloud designs. These prospective improvements would allow DMGO to maintain its robustness, tactical flexibility, and sustainability in dynamic cloud environments, effectively addressing the needs of both enterprise customers and users. This study did not encompass a thorough comparison with other dynamic and multi-objective optimization algorithms due to differences in problem scope and architecture; however, future research will aim to benchmark DMGO against a wider array of comparable methods to further substantiate its efficacy.

6 Conclusion

The Dynamic Multi-Objective Gannet Optimization (DMGO) strategies significantly enhance the optimization of data replication in cloud computing settings. Enhancing information replication efficiency in cloud architectures is crucial for improving data availability, reliability, and fault tolerance. Through the utilization of sophisticated replication techniques, coupled with adaptive replication algorithms, load balancing measures, and data deduplication, cloud systems can minimize redundancy while ensuring rapid data access across numerous locations. By constantly adjusting to changes in device load and network conditions, DMGO provides a flexible and effective solution that balances multiple objectives, including data access latency, storage costs, and network efficiency. The experimental results indicate that DMGO surpasses traditional static replication methods, leading to faster access to data, reduced operational costs,

and enhanced resource efficiency. DMGO is an invaluable instrument for enhancing the overall performance and scalability of cloud systems, particularly in scenarios where the dynamic version is essential for ensuring device reliability and efficiency. The intricacy of the rule set may present difficulties for large-scale implementations, as dynamic monitoring and real-time modifications may necessitate substantial computer resources. Future research should also examine the adaptation of rules inside hybrid cloud systems and multi-cloud architectures, where data is transmitted across multiple platforms with diverse policies and infrastructures.

Acknowledgement: Not applicable.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Conceptualization, P. William and Ved Prakash Mishra; data curation, Ved Prakash Mishra; formal analysis, Osamah Ibrahim Khalaf and Ved Prakash Mishra; funding acquisition, Yogeesh N, Ved Prakash Mishra and P. William; investigation, Osamah Ibrahim Khalaf and Ved Prakash Mishra; methodology, P. William and Ved Prakash Mishra; project administration, Yogeesh N, Ved Prakash Mishra and P. William; resources, Yogeesh N, P. William and Ved Prakash Mishra; software, Riya Karmakar and Ved Prakash Mishra; supervision, Yogeesh N, Osamah Ibrahim Khalaf and P. William; validation, Osamah Ibrahim Khalaf; writing—original draft, Riya Karmakar, Osamah Ibrahim Khalaf, Arvind Mukundan and Ved Prakash Mishra; writing—review and editing, Riya Karmakar, Arvind Mukundan, Osamah Ibrahim Khalaf, Ved Prakash Mishra and P. William. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data available on request from the authors.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Slimani S, Hamrouni T, Ben Charrada F. Service-oriented replication strategies for improving quality-of-service in cloud computing: a survey. Clust Comput. 2021;24(1):361–92. doi:10.1007/s10586-020-03108-z.
2. Shi T, Ma H, Chen G, Hartmann S. Cost-effective web application replication and deployment in multi-cloud environment. IEEE Trans Parallel Distrib Syst. 2022;33(8):1982–95. doi:10.1109/TPDS.2021.3133884.
3. Mansouri N, Javidi MM, Zade BMH. Hierarchical data replication strategy to improve performance in cloud computing. Front Comput Sci. 2020;15(2):152501. doi:10.1007/s11704-019-9099-8.
4. Javadpour A, Abadi AMH, Rezaei S, Zomorodian M, Rostami AS. Improving load balancing for data-duplication in big data cloud computing networks. Clust Comput. 2022;25(4):2613–31. doi:10.1007/s10586-021-03312-5.
5. Pohanka T, Pechanec V. Evaluation of replication mechanisms on selected database systems. ISPRS Int J Geo Inf. 2020;9(4):249. doi:10.3390/ijgi9040249.
6. Akbar M, Ahmad I, Mirza M, Ali M, Barmavatu P. Enhanced authentication for de-duplication of big data on cloud storage system using machine learning approach. Clust Comput. 2024;27(3):3683–702. doi:10.1007/s10586-023-04171-y.
7. Obi OC, Dawodu SO, Daraojimba AI, Onwusinkwue S, Akagha OV, Ahmad Ibrahim Ahmad I. Review of evolving cloud computing paradigms: security, efficiency, and innovations. Comput Sci IT Res J. 2024;5(2):270–92. doi:10.51594/csitrj.v5i2.757.
8. Shafiq DA, Jhanjhi NZ, Abdullah A. Load balancing techniques in cloud computing environment: a review. J King Saud Univ Comput Inf Sci. 2022;34(7):3910–33. doi:10.1016/j.jksuci.2021.02.007.
9. Katal A, Dahiya S, Choudhury T. Energy efficiency in cloud computing data centers: a survey on software technologies. Cluster Comput. 2023;26(3):1845–75. doi:10.1007/s10586-022-03713-0.

10. Ramesh G, Logeshwaran J, Aravindarajan V. A secured database monitoring method to improve databackup and recovery operations in cloud computing. BOHR Int J Comput Sci. 2023;2(1):37–43. doi:10.54646/bijcs.2023.19.
11. Ayyagiri A, Gopalakrishna Pandian PK, Goel P. Efficient data migration strategies in sharded databases. J Qua Sci Tech. 2024;1(2):72–87. doi:10.36676/jqst.v1.i2.17.
12. Bharany S, Sharma S, Khalaf OI, Abdulsahib GM, Al Humaimeedy AS, Aldhyani THH, et al. A systematic survey on energy-efficient techniques in sustainable cloud computing. Sustainability. 2022;14(10):6256. doi:10.3390/su14106256.
13. Li H, Wang SX, Shang F, Niu K, Song R. Applications of large language models in cloud computing: an empirical study using real-world data. Int J Innov Res Comput Sci Technol. 2024;12(4):59–69. doi:10.55524/ijircst.2024.12.4.10.
14. Berisha B, Mëziu E, Shabani I. Big data analytics in cloud computing: an overview. J Cloud Comput. 2022;11(1):24. doi:10.1186/s13677-022-00301-w.
15. Sonbol K, Özkasap Ö, Al-Oqily I, Aloqaily M. EdgeKV: decentralized, scalable, and consistent storage for the edge. J Parallel Distrib Comput. 2020;144(2):28–40. doi:10.1016/j.jpdc.2020.05.009.
16. Vinoth KM, Venkatachalam K, Prabu P, Almutairi A, Abouhawwash M. Secure biometric authentication with de-duplication on distributed cloud storage. PeerJ Comput Sci. 2021;7(2):e569. doi:10.7717/peerj-cs.569.
17. Masdari M, Zangakani M. Efficient task and workflow scheduling in inter-cloud environments: challenges and opportunities. J Supercomput. 2020;76(1):499–535. doi:10.1007/s11227-019-03038-7.
18. Alharbi HA, Elgorashi TEH, Elmirghani JMH. Energy efficient virtual machines placement over cloud-fog network architecture. IEEE Access. 2020;8:94697–718. doi:10.1109/access.2020.2995393.
19. Ebadi Y, JafariNavimipour N. An energy-aware method for data replication in the cloud environments using a tabu search and particle swarm optimization algorithm. Concurr Comput Pract Exp. 2019;31(1):e4757. doi:10.1002/cpe.4757.
20. Nannai John S, Mirnalinee TT. A novel dynamic data replication strategy to improve access efficiency of cloud storage. Inf Syst E Bus Manag. 2020;18(3):405–26. doi:10.1007/s10257-019-00422-x.