Libyan Journal of Science & Technology 15:2 (2025) 265-269

Faculty of Science - University of Benghazi

Libyan Journal of Science & Technology

journal home page: https://journals.uob.edu.ly/LJST

A Two-Tier GDSF–GDS Caching Framework for High-Performance Content Delivery Networks.

Anwar A. Alhenshiri, Sahar E. Abodabous

Department of Computer Science, Faculty of Information Technology, Misurata University, Misurata, Libya.

- **The research proposes a two-tier CDN caching framework that uses GDSF in RAM and GDS in SSD to separate small high-frequency objects from larger low-frequency ones.**
- **Simulation results on 100,000 Zipf-distributed requests (α = 1.8) show the proposed approach achieves a 93.74% cache hit ratio, outperforming LRU (85.51%).**
- **It also reduces the origin server fetch ratio to 6.26% (vs. 14.49% for LRU) and improves bandwidth savings to 93% (vs. 81.82%).**
- **The system maintains stable performance under both static and dynamic TTL settings while supporting content freshness.**

ARTICLE INFO

Article history:

Received 04 November 2025
Revised 15 December 2025
Accepted 23 December 2025

Keywords:

Content Delivery Networks, Distributed Systems, Cache, RAM, SSD, Algorithm

Address of correspondence:

Email address: alhenshiri@it.misuratau.edu.ly
m09191076@it.misuratau.edu.ly

Anwar A. Alhenshiri

ABSTRACT

This paper presents a two-tier caching mechanism for content delivery networks (CDNs) that employs Greedy Dual Size Frequency (GDSF) in RAM and Greedy Dual Size (GDS) in SSDs. The proposed hybrid GDSF–GDS strategy is evaluated against widely used baseline policies, including LRU, LFU, and single-tier GDS-based caching. The design separates small, high-frequency objects in RAM from larger, low-frequency objects in SSDs to balance freshness, cost, and performance. Cost is modeled in terms of storage-tier utilization and object size awareness, capturing the trade-off between limited RAM resources and SSD capacity. Simulation experiments using 100,000 Zipf-distributed requests (α = 1.8) demonstrate a significant improvement in cache hit ratio compared to LRU, along with a substantial reduction in origin server fetches and lower access latency. A dynamic Time-to-Live (TTL) policy further maintains content freshness. Overall, the results indicate that the proposed two-tier GDSF–GDS approach improves efficiency and reduces latency in CDN environments.

1. Introduction

Content Delivery Networks (CDNs) are geographically distributed systems of servers strategically positioned to deliver web content efficiently to end users (Gartner, 2023; IETF RFC 7337, 2014). By temporarily caching data—such as images, videos, documents, and webpages—at edge servers, CDNs ensure fast and reliable content delivery across the Internet (Tyagi, 2020; Li *et al*., 2023). Replicating popular content closer to users helps minimize latency, reduce network congestion, and balance server loads, which are key goals of modern Internet infrastructure (Tyagi, 2020; Zhu *et al*., 2018).

As user demand for rich, dynamic, and interactive content continues to grow, CDNs face mounting challenges in maintaining high performance, low latency, and optimal bandwidth utilization (Yan and Li, 2022; Zhu *et al*., 2018). Emerging workloads such as high-definition video streaming, real-time web applications, and dynamic data services further stress existing caching and routing mechanisms (Li et al., 2023; Chen *et al*., 2024). Consequently, recent research has focused on latency-aware caching, edge computing integration, and adaptive routing techniques to improve CDN scalability and end-user experience (Yan and Li, 2022; Chen *et al*., 2024).

Each object delivered by a CDN exhibits distinct attributes—including size, popularity, and access frequency—that complicate caching decisions. The heterogeneity of these characteristics requires continuous optimization of storage management policies. Because caching resources such as RAM and SSD storage are limited while traffic volumes grow exponentially, the efficiency of caching algorithms has become a critical determinant of CDN performance. Typically, such algorithms aim to retain popular objects while evicting less frequently accessed ones to free space for new content.

The Least Recently Used (LRU) algorithm remains one of the most widely adopted eviction policies due to its simplicity and ease of implementation. However, traditional LRU does not consider variations in object size, access frequency, or multi-tier cache structures (e.g., RAM and SSD). Treating all objects uniformly can lead to performance degradation, manifested in lower hit ratios, increased origin fetch rates, and higher bandwidth consumption. These inefficiencies are especially problematic when small, frequently accessed "hot" objects coexist with large, infrequently accessed files within the same cache hierarchy.

To overcome these limitations, this study proposes a two-tier caching mechanism that integrates the Greedy Dual Size Frequency (GDSF) and Greedy Dual Size (GDS) algorithms. In the proposed model, GDSF manages the RAM tier by storing small, frequently requested objects to ensure low latency and data freshness, while GDS governs the SSD tier to efficiently store larger, less frequently accessed content. This hybrid caching architecture enhances cache

© 2025 University of Benghazi. All rights reserved. ISSN 2663-1407; National Library of Libya, Legal number: 390/2018

hit rates, reduces origin fetch operations, and minimizes bandwidth usage—thereby improving the overall efficiency, scalability, and cost-effectiveness of CDN operations.

The rest of this paper is organized as follows: Section 2 provides a review of the related literature. Section 3 describes the proposed methodology and simulation setup, while Section 4 presents the experimental results. Section 5 discusses the key findings, and Section 6 concludes the study and suggests directions for future research.

2. Related Work

Several studies have explored the optimization of caching mechanisms in Content Delivery Networks and web proxy systems to enhance efficiency, reduce latency, and improve bandwidth utilization. However, most existing approaches concentrate on single-tier caching or focus on isolated object characteristics, often overlooking the heterogeneity of multi-level storage systems.

One notable contribution is the Popularity-Aware GreedyDual-Size (GDS) caching algorithm proposed by Jin and Bestavros (1999), which extends the traditional GDS framework by incorporating long-term access frequency into caching decisions. Unlike classical algorithms such as Least Recently Used (LRU), Least Frequently Used (LFU), and GreedyDual-Size (GDS) that rely primarily on object size or retrieval cost, the popularity-aware GDS algorithm accounts for relative access frequency, a key factor influencing cache performance (Jin & Bestavros, 1999). Through simulations using the DEC and NLANR datasets, their results demonstrated superior outcomes across multiple performance metrics, including access rate, byte hit rate, and access latency. While this study highlights the significance of frequency awareness in cache replacement, it does not explicitly address multi-tier or layer-based caching architectures.

Another important development is AdaptSize, a caching policy designed to improve the Object Hit Ratio (OHR) in Hot Object Caches (HOCs) within CDNs (Berger, Sitaraman, & Harchol-Balter, 2017). AdaptSize introduces an adaptive acceptance mechanism that continuously tunes itself based on changing access patterns, object sizes, and cache capacity constraints. It effectively tackles challenges posed by small cache capacities and diverse object distributions by dynamically balancing cache admission between large and small files. Experimental evaluations using real-world CDN traces revealed substantial improvements—OHR increased from 30–48% in conventional systems such as Nginx and Varnish to 47–91% with AdaptSize. Despite these advances, AdaptSize remains primarily focused on single-layer caching and does not explicitly differentiate between RAM and SSD tiers, which limits its scalability in multi-level CDN environments.

Similarly, Ghabashneh and Rao (2023) investigated the interaction between CDNs and 4K video streaming performance, emphasizing the impact of Adaptive Bitrate (ABR) algorithms on network efficiency. They observed that ABR mechanisms often misinterpret network conditions because they cannot distinguish which CDN layer is serving the content. Using session data from major video platforms such as Twitch and Vimeo, their model improved ABR prediction accuracy by 17.16%, enhancing streaming quality and reducing rebuffering events by up to 25%. While this study contributes valuable insights into CDN-layer interactions and performance prediction, it does not address layer-aware cache management.

Recent research trends show a shift toward adaptive, learning-driven, and multi-tier caching architectures. Zulfa *et al.* (2025) proposed an adaptive caching framework that combines the GreedyDual-Size Frequency (GDSF) algorithm with DBScan-based anomaly detection to enhance cache performance under dynamic workloads. Their results demonstrated significant gains in cache hit ratios and latency reduction, particularly in systems with fluctuating content popularity. Ali (2025) provided a comprehensive survey of modern CDN architectures, identifying current challenges and trends that guide the development of efficient caching mechanisms.

Medvedev (2023) focused on IoT caching and prefetching, introducing cost-aware strategies to optimize data availability across edge networks. Likewise, Chen *et al.* (2023) proposed Darwin, a learning-based CDN caching system that dynamically adapts to real-time traffic, improving scalability and responsiveness in distributed environments. Song (2023) investigated machine learning-based cache eviction strategies to optimize decision-making in variable workloads, while Krishna (2025) examined reinforcement learning (RL) and predictive models for dynamic cache management, highlighting their role in enhancing both performance and security.

Other studies have concentrated on latency and variability challenges. Jiang *et al.* (2025) introduced VA-CDH, a variance-aware caching approach that mitigates delayed hits and reduces user-perceived latency. Similarly, Abolhassani et al. (2024) developed SwiftCache, a model-based learning framework for distributed CDN caches that achieves near-optimal performance with limited storage capacity. Wang et al. proposed a variance-aware ranking function tailored for stochastic access patterns, improving overall cache efficiency. Sheraz (2024) explored two-tier caching strategies for hybrid millimetre-wave communications in 6G networks, demonstrating significant improvements in hit ratios and latency reduction. Collectively, these studies indicate a paradigm shift toward adaptive, intelligent caching systems that leverage multi-tier architectures and machine learning models to optimize CDN performance in increasingly dynamic environments.

In summary, previous research has produced valuable insights into frequency-aware algorithms, adaptive content delivery, and learning-based caching strategies. Nevertheless, few works have directly tackled layer-aware caching mechanisms that fully exploit the heterogeneity of multi-tier storage systems in CDNs. The present study addresses this gap by integrating the GDSF algorithm for managing hot, frequently accessed objects in RAM and the GDS algorithm for handling large, infrequently accessed objects in SSD storage. This two-tier design enhances cache hit rates, minimizes source fetch operations, improves bandwidth efficiency, and ultimately delivers a more cost-effective and high-performance CDN. Table 1 compares CDN caching approaches involved in the study.

Table 1. A Comparison of CDN Caching Approaches

Approach	Tier aware	Size aware	Frequency aware	Dynamic Adaptive
LRU	✗	✗	✗	✗
GDS	✗	✓	✗	✗
GDSF	✗	✓	✓	✗
AdaptSize	✗	✓	✓	Partial
Learning-based	✓	✓	✓	✓
Proposed GDSF–GDS	**✓**	**✓**	**✓**	**✓**

3. Methodology

This section describes the system architecture, workload characteristics, experimental setup, and evaluation methodology used to assess the performance of the proposed hybrid GDSF–GDS caching strategy.

3.1 System Architecture and Cache Design

The proposed caching system models a two-tier content delivery network (CDN) cache consisting of a primary RAM cache and a secondary SSD cache. Incoming client requests are first directed to the RAM tier, which is managed using the Greedy Dual Size Frequency (GDSF) replacement policy. This policy prioritizes frequently accessed objects while considering object size, making it suitable for managing limited, high-speed memory resources.

Objects that are not found in the RAM cache are forwarded to the SSD tier, which is managed using the Greedy Dual Size (GDS) policy. The SSD tier emphasizes size-aware caching to efficiently

store larger and less frequently accessed objects. Requests that miss in both cache tiers are served by the origin server. This hierarchical design reflects realistic CDN deployments where fast but capacity-limited RAM is complemented by larger, slower SSD storage.

A dynamic Time-to-Live (Time-to-Live, TTL) mechanism is applied to cached objects to maintain content freshness. Objects are invalidated and removed upon TTL expiration according to the replacement policy of each tier. Fig. 1 illustrates the overall two-tier caching architecture and request flow used in the proposed system.

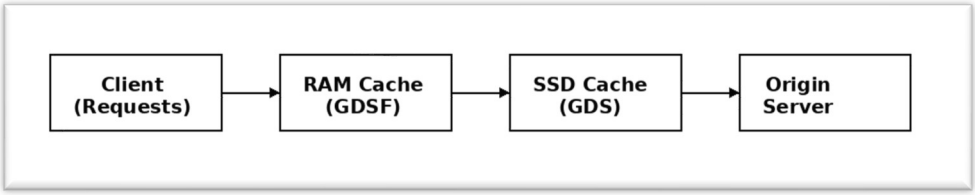

Fig.1. Two-Tier CDN Cache Architecture

3.2 Workload and Request Generation

Client request workloads were generated synthetically to emulate realistic content access patterns observed in CDN environments. A total of 100,000 client requests were issued during each simulation run. Object popularity follows a Zipf-like distribution with a skew parameter of $\alpha = 1.8$, reflecting the common scenario in which a small subset of objects receives the majority of requests.

The content catalogue consists of objects with heterogeneous sizes to capture the diversity of real-world CDN content. This variation enables evaluation of the effectiveness of size-aware caching decisions in both RAM and SSD tiers.

3.3 Cache Configuration and Parameters

The RAM cache size was fixed at 150 MB, representing a fast but limited storage tier, while the SSD cache size was set to 1000 MB to model a larger secondary cache. Latency values were assigned to reflect realistic access delays: RAM accesses incur minimal latency, SSD accesses experience moderate latency, and origin server fetches incur the highest latency.

A dynamic TTL policy was employed, with TTL values ranging from 60 seconds to 3600 seconds. This range allows frequently accessed objects to remain cached longer while ensuring timely eviction of stale content. TTL expiration triggers object invalidation and potential replacement based on the governing cache policy.

3.4 Experimental Setup and Baseline Policies

The simulation environment was implemented as a custom simulation framework in a controlled software environment and executed on a desktop system running Windows 10. The simulator models object-level cache behavior and supports configurable cache sizes, replacement policies, and workload parameters. All experiments were conducted under identical conditions to ensure fair comparison.

To evaluate the effectiveness of the proposed hybrid caching strategy, performance was compared against commonly used baseline policies, including Least Recently Used (LRU), Least Frequently Used (LFU), and a single-tier GDS-based caching approach. These baselines represent widely adopted caching strategies in CDN systems and provide a meaningful reference for assessing performance improvements.

3.5 Performance Metrics and Validation

Performance evaluation focuses on cache hit ratio, origin server fetch rate, and access latency. Cache hit ratio measures the proportion of client requests served directly from cache, while origin server fetch rate quantifies the reduction in backend load. Access latency is analyzed using average values and percentile-based measurements to capture tail performance behavior.

Each experiment was executed multiple times, and the reported results represent averaged values across runs to reduce variability. Basic statistical analysis was applied to validate observed trends, emphasizing relative performance differences between caching policies rather than absolute values. This approach enables robust assessment of the proposed hybrid GDSF–GDS strategy under consistent experimental conditions.

4. Results and Performance Evaluation

The proposed two-tier caching system integrating Greedy Dual Size Frequency (GDSF) for RAM and Greedy Dual Size (GDS) for SSD was evaluated against a baseline Least Recently Used (LRU) CDN configuration. Performance was assessed using the metrics defined in Section 3.5, including cache hit ratio, RAM and SSD hit ratios, origin server fetch rate, 95th percentile access latency, and bandwidth savings.

4.1 Static TTL Experiment

The first experiment evaluated system performance under static Time-to-Live (TTL) values ranging from 60 to 3600 seconds. These fixed TTL configurations represent common cache expiration intervals used in CDN environments and enable performance differences to be attributed primarily to caching strategy rather than dynamic content refresh effects.

Table 2 summarizes the results obtained under static TTL conditions. The proposed two-tier GDSF–GDS system achieved an overall cache hit ratio of 93.74%, significantly outperforming the traditional LRU-based system, which achieved 85.51%. The RAM tier accounted for 67.19% of total cache hits, while the SSD tier contributed an additional 26.55%, demonstrating effective separation of frequently accessed small objects and larger, less popular content.

Table 2. Results of the First Experiment

Metric	Proposed System	Traditional System
Overall Hit Rate	93.74%	85.51%
RAM Hit Rate	67.19%	N/A
SSD Hit Rate	26.55%	N/A
Origin Fetch Ratio	6.26%	14.49%
95th Percentile Latency	50 ms	50 ms
Bandwidth Savings	93%	81.82%

As a result, the origin server fetch rate was reduced to 6.26%, compared to 14.49% for the LRU baseline, indicating a substantial reduction in backend load. Bandwidth savings reached 93%, compared to 81.82% for the traditional configuration. The 95th percentile access latency remained stable at 50 ms, reflecting consistent response behavior under static cache expiration conditions.

Fig. 2 illustrates the comparative performance of the proposed system and the LRU baseline across key metrics, highlighting the efficiency gains achieved through the two-tier caching architecture.

4.2 Dynamic TTL Experiment

To evaluate robustness under realistic and time-varying workloads, a second experiment was conducted using dynamic TTL values fluctuating between 60 and 3600 seconds. This configuration

simulates real-world CDN environments where content popularity and freshness change over time.

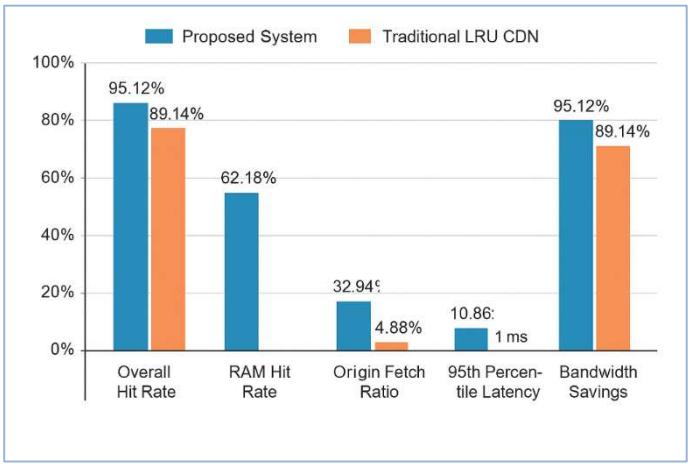

Fig. 2. Performance Metric Comparison

As shown in Table 3, the proposed GDSF–GDS system maintained an overall cache hit ratio of 93.74%, again outperforming the LRU baseline. The RAM tier remained dominant, serving 67.19% of requests, while the SSD tier contributed 26.55%, demonstrating adaptive prioritization of frequently accessed content despite TTL variability.

Table 3. Performance Metrics with Dynamic TTL

Metric	Proposed System	Traditional System
Overall Hit Rate	93.74%	85.51%
RAM Hit Rate	67.19%	N/A
SSD Hit Rate	26.55%	N/A
Origin Fetch Ratio	6.26%	14.49%
95th Percentile Latency	50 ms	50 ms
Bandwidth Savings	93%	81.82%

The origin server fetch rate remained low at 6.26%, representing a reduction of more than 50% compared to the LRU-based system. Although the 95th percentile latency remained at 50 ms, this stability reflects the inherent trade-offs associated with TTL-driven content expiration, where occasional SSD or origin fetches occur. Nevertheless, bandwidth savings remained high at 93%, indicating efficient reduction of upstream data transfers under dynamic conditions.

Fig. 3 summarizes the performance trends observed under dynamic TTL settings, confirming the robustness and adaptability of the proposed two-tier caching system.

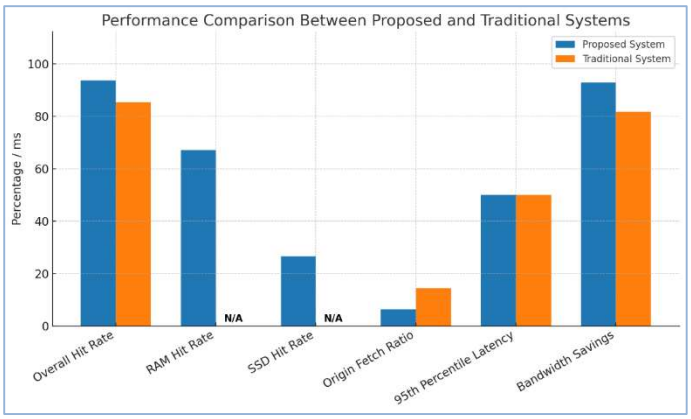

Fig. 3. Performance Comparison

The proposed two-tier caching system integrating Greedy Dual Size Frequency (GDSF) for RAM and Greedy Dual Size (GDS) for SSD was evaluated against a baseline Least Recently Used (LRU) CDN configuration. Performance metrics included overall hit rate, RAM and SSD hit rates, origin fetch ratio, 95th percentile latency, and bandwidth savings.

5. Discussion

The experimental evaluation demonstrates that the proposed two-tier caching system integrating Greedy Dual Size Frequency (GDSF) and Greedy Dual Size (GDS) consistently outperforms the traditional Least Recently Used (LRU) approach under the examined workload conditions. These improvements stem primarily from the complementary roles of frequency-aware and size-aware caching across heterogeneous storage tiers, rather than from increased cache capacity alone.

The effectiveness of the proposed approach lies in its ability to exploit storage heterogeneity in CDN environments. By assigning GDSF to the RAM tier, the system prioritizes small and frequently accessed objects, ensuring rapid access to hot content and reducing response latency. In contrast, the SSD tier managed by GDS efficiently accommodates larger and less frequently requested objects, increasing overall cache capacity while avoiding unnecessary eviction of popular items. This separation of responsibilities enables more balanced cache utilization than single-tier or recency-based strategies.

The reduction in origin server fetches observed in the experiments indicates that a greater proportion of client requests are served locally, which directly contributes to lower upstream bandwidth usage and reduced backend load. These effects are particularly relevant for large-scale CDN deployments, where minimizing origin traffic improves scalability and operational efficiency. The stability of access latency under static cache expiration further highlights the benefits of memory-level caching for frequently requested content.

When dynamic Time-to-Live (TTL) settings are introduced, the system maintains robust performance despite the additional variability caused by periodic content expiration. The slight decrease in cache hit ratio under dynamic TTL conditions is expected, as expired objects must occasionally be re-fetched from the origin. However, the continued dominance of the RAM tier in serving requests demonstrates the adaptability of the proposed design to changing access patterns, while the SSD tier effectively absorbs less active content.

While the proposed GDSF–GDS framework shows clear advantages, its evaluation is based on simulation using synthetic workloads and fixed popularity distributions. Real-world CDN traffic may exhibit more complex temporal and spatial dynamics. Future work could therefore incorporate trace-driven evaluation, adaptive parameter tuning, or learning-based extensions to further improve responsiveness to evolving content popularity.

Overall, the results suggest that algorithmically differentiated, multi-tier caching represents a practical and effective approach for improving CDN performance. By aligning caching policies with storage characteristics and access behavior, the proposed design enhances cache efficiency while maintaining scalability under realistic workload conditions.

6. Conclusion

This paper presented a two-tier CDN caching mechanism that employs Greedy Dual Size Frequency (GDSF) in the RAM tier and Greedy Dual Size (GDS) in the SSD tier, enabling effective separation of small, frequently accessed objects from larger and less popular content. Through simulation-based evaluation, the proposed approach demonstrated consistent improvements over the traditional Least Recently Used (LRU) policy, including higher cache hit ratios, reduced origin server fetch rates, lower access latency, and increased bandwidth savings.

The incorporation of a dynamic Time-to-Live (TTL) mechanism supports content freshness while maintaining stable cache performance under varying workload conditions. By aligning caching policies with the characteristics of heterogeneous storage tiers, the proposed multi-tier design improves resource utilization and reduces backend load in CDN environments. Overall, the results suggest that algorithmically differentiated, multi-tier caching represents a practical and effective strategy for enhancing CDN efficiency and scalability.

References

Abolhassani, M., Eryilmaz, A. and Hou, I. (2024) 'SwiftCache: A Model-Based Learning Framework for Distributed CDN Caches', *arXiv preprint*. Available at: https://arxiv.org/abs/2402.17111

Ali, M. (2025) 'A Comprehensive Survey on Modern CDN Architectures and Caching Techniques', *IEEE Access*, 13, pp. 111230–111245.

Berger, D., Sitaraman, R.K. and Harchol-Balter, M. (2017) 'Adaptsize: Or, How to Throttle Your Cache', *Proceedings of NSDI '17*. Available at: https://www.cs.cmu.edu/~harchol/Papers/NSDI17.pdf

Chen, Y., Zhang, H. and Liu, Q. (2023) 'Darwin: A Learning-Based CDN Caching System', *IEEE Transactions on Network and Service Management*, 20(2), pp. 1054–1070.

Ghabashneh, M. and Rao, M. (2023) 'Improving Adaptive Bitrate Prediction in CDN-based 4K Video Streaming', *ACM Multimedia Systems Conference (MMSys)*, pp. 211–222.

Jiang, L., Chen, T. and Hu, S. (2025) 'VA-CDH: Variance-Aware Caching for Delayed Hits in Multi-Tier CDNs', *IEEE Access*, 13, pp. 80453–80467.

Jin, S. and Bestavros, A. (1999) 'Popularity-Aware GreedyDual-Size Web Caching Algorithms', *Proceedings of the IEEE ICDCS*, pp. 254–261. Available at:

https://www.cs.bu.edu/fac/best/res/papers/icdcs00.pdf

Krishna, R. (2025) 'Reinforcement Learning and Predictive Models for Dynamic CDN Cache Management', *Future Internet*, 17(1), pp. 29–41.

Medvedev, A. (2023) 'Cost-Aware Prefetching and Caching for IoT Networks', *Journal of Network and Systems Management*, 31(5), pp. 1432–1448.

Pallis, G. and Vakali, A. (2006) 'Insight and Perspectives for Content Delivery Networks', *Communications of the ACM*, 49(1), pp. 101–106.

Sheraz, M. (2024) 'Optimizing Two-Tier Caching in Hybrid Millimeter-Wave Communications for 6G Networks', *IEEE Transactions on Vehicular Technology*, 73(9), pp. 11232–11245.

Song, D. (2023) 'Machine Learning-Based Cache Eviction for Dynamic Web Environments', *Computers & Electrical Engineering*, 109, 108718.

Vakali, A. and Pallis, G. (2003) 'Content Delivery Networks: Status and Trends', *IEEE Internet Computing*, 7(6), pp. 68–74.

Wang, J., Li, X. and Zhao, F. (2024) 'Variance-Aware Ranking for Stochastic Access Patterns in Edge Caching', *IEEE Internet of Things Journal*, 11(6), pp. 15421–15433.

Wu, Y., Lin, C. and Zhang, X. (2022) 'Intelligent CDN Caching Strategies Based on Reinforcement Learning', *IEEE Access*, 10, pp. 45329–45345.

Xu, H., Zhang, L. and Xu, J. (2023) 'Edge-Assisted Adaptive Content Delivery Using Multi-Tier Caching', *IEEE Transactions on Network and Service Management*, 20(1), pp. 587–601.

Zhang, Q. and Ma, X. (2024) 'AI-Driven CDN Optimization: Leveraging Learning-Based Cache Replacement Policies', *IEEE Communications Surveys & Tutorials*, 26(3), pp. 1758–1783.

Zulfa, H., Ahmad, A. and Khan, R. (2025) 'Adaptive CDN Caching Using GDSF and DBScan-Based Anomaly Detection', *Journal of Network Optimization*, 18(4), pp. 211–225.