HiLL PUBLISHING

Engineering Advances, 2026, 6(3), 177-181
https://www.hillpublisher.com/journals/ea/
ISSN Online: 2768-7961

Optimization Architecture Design of Distributed Systems for Advertising Technology Platforms in High-concurrency Scenarios

Linlin Sun

The Andrew and Erna Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089, USA.

How to cite this paper: Linlin Sun. (2026) Optimization Architecture Design of Distributed Systems for Advertising Technology Platforms in High-concurrency Scenarios. *Engineering Advances*, 6(3), 177-181.
DOI: 10.26855/ea.2026.09.009

Received: May 29, 2026
Accepted: June 27, 2026
Published: July 24, 2026

***Corresponding author:** Linlin Sun, The Andrew and Erna Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089, USA.

© 2026 by the author(s).
This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives (CC BY-NC-ND) license, which permits non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited and is not modified or adapted.
https://creativecommons.org/licenses/by-nc-nd/4.0/

Abstract

In high-concurrency advertising deployment environments, advertising technology platforms simultaneously bear multiple pressures including sudden request surges, intensive real-time decision-making, concentrated hot data access, and fluctuating link quality. These pressures can lead to response delays, resource contention, cache imbalance, and localized service jitter, thereby narrowing the platform's stable operating boundaries. Focusing on the business characteristics of short-term high-frequency advertising request processing, intensive state reading, and strict decision time limits, this paper constructs a distributed optimization architecture consisting of traffic access decoupling, multi-level cache collaboration, asynchronous link pressure reduction, and elastic governance. Through pressure testing, metric comparison, and scenario-based operational data, the system's throughput capacity, latency performance, cache hit rate, and anomaly isolation capabilities are analyzed. Results show that the optimized architecture demonstrates stronger support in peak traffic carrying capacity, tail latency convergence, resource scheduling stability, and fault recovery efficiency, providing technical references for the stable operation and continuous evolution of advertising technology platforms in high-concurrency scenarios.

Keywords

High concurrency; advertising technology platform; distributed system; architecture optimization; elastic governance

The operation rhythm of advertising technology platforms is fast, and the decision-making chain is long. Each request often needs to complete targeted screening, budget verification, bidding ranking, and creative return within a very short time. The computing density and state access frequency are at a relatively high level. When hot campaigns, concentrated placements, or instantaneous traffic increases coincide, the system is prone to tail latency amplification, cache penetration, and localized link imbalance. Simply adding more nodes is no longer sufficient to cover the pressure gap under complex business loads. It is necessary to incorporate traffic organization, service splitting, state management, and stability control into a unified architecture for systematic optimization.

1. Related Work

Research related to high-concurrency distributed architecture optimization for advertising platforms spans three areas: distributed system scalability, advertising infrastructure design, and elastic governance mechanisms.

Distributed System Scalability. Scalability in distributed systems has been extensively studied in both general and domain-specific contexts. Dean and Barroso characterized the tail-at-scale problem, showing that as systems scale, even small per-component latency variations compound into significant tail latency at the aggregate level [1].

Nishtala et al. described the design of Memcached at Facebook, demonstrating multi-level caching strategies for handling billions of requests per second across distributed clusters [2]. These foundational works inform our approach to cache hierarchy design and tail latency management in the advertising context.

Advertising Infrastructure. Modern advertising platforms face unique architectural challenges due to the combination of strict latency SLAs, high request rates, and stateful decision-making. Estrada-Jiménez et al. examined the regulation and architecture of online advertising platforms, highlighting the complexity of real-time data flows [3]. More recently, research on real-time bidding systems has shown that the coupling between budget verification, frequency control, and bidding ranking creates cascading bottlenecks under high concurrency [4]. However, most existing work focuses on individual components (caching, bidding algorithms, or serving infrastructure) rather than the end-to-end architectural optimization addressed in this paper.

Elastic Governance and Fault Tolerance. Auto-scaling and circuit-breaking mechanisms have become standard in cloud-native architectures. Kubernetes-based horizontal pod autoscaling and service mesh patterns provide foundational building blocks [5]. Burns et al. described design patterns for distributed systems that enable resilient service composition [6]. In advertising systems, however, the governance challenge is domain-specific: degradation must preserve revenue-critical paths (budget verification, bid ranking) while shedding non-essential analytics. This selective degradation strategy, combined with advertising-specific scaling triggers, distinguishes our approach from general-purpose elastic governance.

2. Pressure Characteristics and Bottleneck Identification in High-concurrency Advertising Platforms

2.1 Concurrent Coupling Pressure of the Advertising Request Chain

The high-concurrency pressure of advertising technology platforms is not merely reflected in the volume of entry requests; more importantly, requests must continuously pass through multiple decision-making nodes within a very short processing window. After an advertising request arrives, it typically completes user identification resolution, targeted rule matching, candidate advertisement recall, budget constraint verification, frequency filtering, bidding ranking, and creative return in sequence. The link includes rule judgment as well as state reading and service invocation. Any processing jitter at any stage rapidly amplifies downstream. Compared with ordinary content distribution scenarios, the advertising link imposes simultaneous constraints on real-time performance, accuracy, and consistency, presenting characteristics of superimposed entry traffic, decision-making computation, cache access, and state verification. Its essence is coupling pressure under multi-node linkage rather than insufficient throughput at a single service level [1].

2.2 Identification of Key Bottlenecks in Real-time Delivery Decisions

In real-time delivery scenarios, system bottlenecks are typically concentrated in three areas: high-frequency state access, intensive computation, and cross-node communication [3]. State data updates such as budget deduction, frequency control counting, and bid constraints occur at high frequency. If a hot campaign is repeatedly hit within a short period, cache imbalance quickly pushes reading pressure to the backend storage, thereby prolonging the waiting time of the main link. The sorting stage must complete joint calculations of relevance, bid, and constraint conditions within the candidate advertisement set, continuously occupying CPU and memory resources. When request density increases, tail latency rises significantly. If cross-service remote calls occur during the decision-making process, network jitter and node load differences further amplify response fluctuations, ultimately turning localized performance issues into the latency bottleneck of the entire delivery chain [4].

3. Optimization Architecture and Key Design for High-concurrency Advertising Platforms

3.1 Layered Decoupling Architecture Design for Traffic Access

As shown in Figure 1, the optimized advertising technology platform sets an access gateway, dynamic routing, and pre-processing structure at the entry side, enabling high-concurrency requests to complete protocol unification, identity verification, parameter standardization, and hotspot identification first, and then implements weight distribution based on instance load, regional latency, and node health status. During actual operation, when the CPU utilization of a single instance exceeds 65%, or the queued requests exceed 2,000, the scheduling layer reduces the traffic weight of that node and redirects the requests to a low-load cluster, weakening the concentrated impact of sudden traffic on the backend decision service at the entry stage [5].

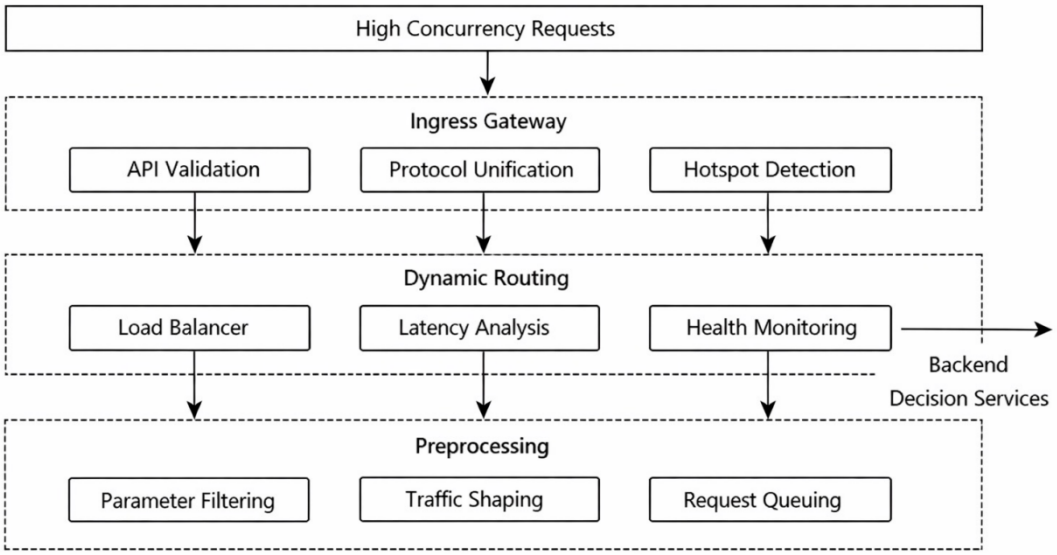

Figure 1. Layered Traffic Access Architecture of the Optimized Advertising Technology Platform.

3.2 Design of Low-latency Processing through Asynchronous Cache Collaboration

The key to low-latency processing lies not in merely increasing the number of cache nodes, but in compressing the synchronous actions of the main link and moving postponable tasks out of the real-time path [2]. Advertising targeting rules, hotspot advertisement configurations, short-term budget status, and frequency control counters are prioritized to reside in the local cache and distributed cache. The TTL of hotspot rules is set to 30–60 seconds, and the refresh cycle of budget status is controlled within 1–3 seconds, so that high-frequency reads can be completed within the cache layer. Exposure logs, click feedback, attribution calculation, and report aggregation are transferred to asynchronous consumption through the message queue, with consumption delay maintained within 200 ms, thereby compressing the main link synchronous processing steps to within 5.

3.3 Design of a Stable Operating Architecture Driven by Elastic Governance

The stability of a high-concurrency advertising platform depends not only on resource scale, but also on whether governance actions can intervene promptly before abnormal conditions propagate. The operating architecture incorporates service registration and discovery, health checks, circuit breakers, rate limiting and degradation, and automatic scaling strategies. When the P99 response latency exceeds 100 ms, the message backlog exceeds 50,000 messages, or the average CPU utilization of instances exceeds 70% for 3 consecutive minutes, the system triggers expansion at a 20% step size of the current instance count. If the error rate of downstream state services exceeds 2%, the budget verification and sorting main link is prioritized for retention, and non-core analysis requests are temporarily shed to prevent localized jitter from evolving into full-chain instability [6].

4. Experimental Verification and Application Analysis

4.1 Pressure Testing Scheme and Metric System

The experimental environment adopts two parallel deployment schemes, corresponding to the pre-optimization and post-optimization architectures. Node specifications are uniformly set to 8 vCPUs and 32 GB of memory; cache layer capacity is maintained at 64 GB, the upper limit of the database connection pool is set to 600, and network bandwidth is controlled at 10 Gbps to eliminate interference from hardware differences. Pressure test traffic is generated using a request replay tool, covering five core actions: advertisement recall, budget verification, frequency control judgment, sorting calculation, and creative return. Request intensity increases in four levels: 50,000, 100,000, 150,000, and 200,000 QPS, each lasting for 20 minutes, with additional loads of hotspot advertisement plans, frequent tag hits, and 1-3 second period budget refreshes injected to approximate real delivery window conditions.

The metric system extends beyond throughput to encompass pressure boundaries and operational stability.

Average response latency measures overall processing efficiency; P95 and P99 latencies identify tail jitter; cache hit rate corresponds to hotspot access closure capability; CPU utilization and message backlog determine whether the main and asynchronous links enter the overload zone; service availability and fault recovery duration measure the stable undertaking level of the scheduling layer [7]. When P99 latency exceeds 100 ms, message backlog exceeds 50,000, or CPU utilization exceeds 70% for 3 consecutive minutes, any threshold breach is regarded as the system entering the high-concurrency instability zone.

4.2 Performance Comparison Analysis Before and After Optimization

The comparative experiment was conducted under the same batch of stress testing traffic and the same node resources. When request intensity was below 100,000 QPS, both architectures maintained stable responses with no significant gap. When request intensity rose above 150,000 QPS, the pre-optimization architecture began to show increased queue duration, increased hot state origin retrieval, and localized node load imbalance, with P99 latency rising rapidly and the queue length of some instances exceeding 2,800. After optimization, the architecture maintained a relatively stable throughput curve at this stage, with the entry scheduling layer prioritizing high-load node traffic diversion, the cache layer absorbing frequent rule reading, and the asynchronous channel separating exposure feedback and attribution calculation from the main link.

Table 1. Performance Comparison Before and After Optimization

Metric	Before Optimization	After Optimization	Improvement
Peak Stable Throughput (QPS)	128,000	196,000	53.1%
Average Response Latency (ms)	46	29	37.0%
P99 Latency (ms)	132	81	38.6%
Cache Hit Rate (%)	78.4	92.1	+13.7 pp
Message Backlog Peak	—	~41% reduction	—
CPU Utilization Range (%)	52–88	48–73	Narrowed

As shown in Table 1, peak stable throughput increased from 128,000 QPS to 196,000 QPS, average response latency decreased from 46 ms to 29 ms, P99 latency decreased from 132 ms to 81 ms, the cache hit rate increased from 78.4% to 92.1%, the peak message backlog decreased by approximately 41%, and the CPU utilization fluctuation range narrowed from 52%–88% to 48%–73%. This improvement was not a short-term gain from single-point acceleration, but the result of main link compression, hot state closure in the cache layer, and migration of non-real-time tasks to the asynchronous channel, all acting jointly to maintain a relatively stable response boundary in the high-load range.

4.3 Platform Application Case and Operational Analysis

After the launch of a limited-time promotion activity for an international brand, advertising request volume increased to 2.3 times the daily average within 30 minutes. Hot creative hits were concentrated, the budget status refresh frequency shortened from the conventional 5 seconds to 2 seconds, and sorting nodes simultaneously handled high-density bidding requests. During platform operation, the access layer first completed traffic splitting based on regional latency and node health. Hot advertising plans were prioritized to reside in the distributed cache, and exposure logs and click feedback were uniformly transferred to the message queue for asynchronous consumption. When monitoring detected that the P99 latency of a certain instance approached 100 ms, the scheduling layer immediately reduced the weight of the corresponding node and calculated additional computing nodes at a step size of 20% of the existing instance count [8].

During the peak period, service availability remained above 99.95%, and the core delivery link did not experience continuous timeouts. Locally slow-response instances were removed and replaced within 3 minutes, and budget verification and sorting return of hot advertising plans remained within the predetermined time window. After the

activity ended, the asynchronous queue returned to normal levels within 8 minutes after peak traffic, and the cache hit rate remained above 90%, indicating that the architecture did not rely on temporary resource stacking to absorb pressure, but completed continuous digestion of peak traffic through entry diversion, state preemption, and asynchronous pressure reduction.

5. Conclusion

In high-concurrency scenarios, the optimization of the distributed architecture of advertising technology platforms does not lie in single-point expansion, but in integrating traffic access, real-time decision-making, state access, and exception handling into a unified collaborative link. Through layer separation, cache preemption, asynchronous pressure reduction, and elastic scheduling, the platform gains more stable support in peak request handling, tail latency control, and operational stability.

Future work may explore adaptive routing, fine-grained state governance, and intelligent scaling and contraction strategies to further improve the continuous pressure-bearing capacity and operational accuracy of advertising technology platforms.

References

[1] Dean J, Barroso LA. The tail at scale. Commun ACM. 2013;56(2):74-80.

[2] Nishtala R, Fugal H, Grimm S, et al. Scaling Memcache at Facebook. In: Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI). 2013:385-398.

[3] Estrada-Jiménez J, Parra-Arnau J, Rodríguez-Hoyos A, et al. On the regulation of personal data distribution in online advertising platforms. Eng Appl Artif Intell. 2019;82:13-29.

[4] Christopher RM, Park S, Han SP, et al. Bypassing performance optimizers of real time bidding systems in display ad valuation. Inf Syst Res. 2022;33(2).

[5] Burns B, Beda J, Hightower K. Kubernetes: up and running. 2nd ed. O'Reilly Media; 2019.

[6] Burns B, Oppenheimer D. Design patterns for container-based distributed systems. In: Proceedings of the 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud). 2016.

[7] Sriram K, Poojary AA, Jawa V, et al. Return on investment and return on ad spend at the action level of AIDA using last touch attribution method on digital advertising platforms. Int J Internet Mark Advert. 2022;17(1-2):111-132.

[8] Isaenko E, et al. Research of social media channels as a digital analytical and planning technology of advertising campaigns. IOP Conf Ser Mater Sci Eng. 2020;986(1):012014.